

Computing with Equations

Chris Preston

Fakultät für Mathematik

Universität Bielefeld

for Martina

Contents

Preface

1. Introduction	1
1.1 Computing with equations: An example	1
1.2 A procedure for interpreting equations	9
1.3 Equations as replacement rules	15
1.4 Notes	20
2. Some universal algebra	21
2.1 Typed sets	23
2.2 Algebras and homomorphisms	28
2.3 Initial and free algebras	36
2.4 Term algebras	43
2.5 Extensions of signatures and algebras	49
2.6 Tree algebras	52
2.7 Notes	57
3. Data objects	58
3.1 The basic data objects and the ground term algebra	58
3.2 Bottomed algebras and bottomed extensions	60
3.3 Regular extensions and the trace	67
3.4 Monotone extensions	72
3.5 Cored extensions	78
3.6 Notes	82
4. Completion of bottomed extensions	83
4.1 Associated orderings for regular extensions	85
4.2 Complete partially ordered sets	91
4.3 Initial completions	97
4.4 Initial completion of monotone regular extensions	105
4.5 Notes	109

5. Functions	110
5.1 Functional types and functional extensions	112
5.2 Functional algebras	119
5.3 Functionally free algebras	124
5.4 Primitive functions	133
5.5 Notes	138
6. Equations	139
6.1 The basic term algebras	141
6.2 Equations and their solutions	152
6.3 Replacement rules	161
6.4 Computing values	170
6.5 An algorithm for computing values	174
6.6 Notes	185
7. Completeness of the algorithm	186
7.1 Solutions as fixed-points	186
7.2 Logical relations and functionally free algebras	194
7.3 An application of the method of logical relations	198
7.4 Uniform values	204
7.5 Notes	210
Bibliography	211
Index	214

Preface

About ten years ago the Mathematics Department at which I am employed was forced to get involved in teaching a first year course in Computer Science. One half of the course was based on the book *Introduction to Functional Programming*¹ by Richard Bird and Philip Wadler and used David Turner's *Miranda*² as programming language. I taught this course several times, the first time at least with an appalling lack of knowledge of what Computer Science is all about. The present study has grown out of what I have learnt from this undertaking. Its intention is to give a mathematical account of how I believe students could be taught to think about functional programming languages and to explain how such languages work.

For those who understand something of the terminology used to describe the semantics of programming languages, this intention can be stated more precisely as follows: I present a certain denotational semantics for a rudimentary functional programming language and show that this is equivalent to the operational semantics defined in terms of an explicit algorithm.

This account is certainly not meant as a text for the kind of course from which it arose, and in fact it has nothing to say about the art of functional programming. It endeavours rather to provide a mathematical basis for such a course, and is aimed at those who would regard such a basis as being important, i.e. at an audience of mathematicians.

The kind of functional programming languages to be dealt with are those like *Miranda*, *Haskell* and *ML*, in which objects, usually functions, are defined by equations. The meaning of such equations will be given directly without resorting to the lambda calculus (and so the book is not really suitable for dealing with a language such as *Lisp*). In fact, this avoidance of the lambda calculus should not pass without comment. It seems that almost all the accounts offering a mathematical explanation of functional programming are written either by computer scientists or logicians. They are therefore written in a language foreign to the very large majority of 'ordinary' mathematicians. I hope that the present study is at least written in a language with which this group of people is familiar.

The mathematical framework which will be employed is to some extent determined by the languages the study seeks to explain. For example, data types in *Miranda*, *Haskell* and *ML* are defined by specifying a signature, and the equations in these languages are built up out of terms taken from corresponding term algebras. This means that some elementary concepts from universal algebra can hardly be avoided. Moreover, one is then more-or-less forced to use the fact that term algebras are initial as a basis for giving equations a meaning. This can be seen as an instance of the 'initial algebra semantics' philosophy propagated by the ADJ group³.

¹ Prentice Hall, Hemel Hempstead (1988)

² *Miranda* is a trademark of Research Software Limited.

³ consisting of J.A. Goguen, J.W. Thatcher, E.G. Wagner and J.B. Wright, for example in *Initial algebra semantics and continuous algebras*, Journal ACM, 24, 68–95, 1977

The aim is to interpret equations over sets without an order structure, although each such set will contain a ‘completely undefined’ element and can also contain ‘partially defined’ elements. In general there need not be a unique least solution, and in fact, there need not be a solution at all. However, this approach makes sense provided it is known the equations have a solution and only those terms are considered whose values are independent of which solution is used. One reason for taking such an approach is to avoid explanations involving pulling ‘magic’ partial orders out of a hat, which are based on mathematics way beyond the comprehension of a typical student in a first year Computer Science course. Not surprisingly, in order to prove results about these solutions it is necessary to introduce the usual machinery of complete partially ordered sets, continuous functions and least fixed-points, but these objects do not always have to occur in the foreground.

There are two topics which are not treated here, and whose omission should at least be commented upon. The first concerns local definitions. These have not been considered in order to keep the presentation as simple as possible, but in principle their inclusion would not cause any difficulty and requires no additional mathematical concepts. The second topic is that of polymorphism. It could be claimed that this has also been omitted to simplify the presentation. However, its inclusion would require a new, and much less elementary, mathematical framework.

As already indicated, the main result presented here is a statement about the equivalence of operational and denotational semantics. The proof of the hard part of this result (the completeness of the operational semantics) has been strongly influenced by the proofs of similar statements in Glynn Winskel’s book *The Formal Semantics of Programming Languages*⁴. I highly recommend Winskel’s book.

I have tried to keep the account self-contained, and have thus included all the standard results from universal algebra and domain theory (together with their proofs) which are needed. However, although the formal mathematical prerequisites are minimal and regardless of what my original intention was, it is probably realistic to suppose that the presentation has ended up at the graduate level. It is not assumed that the reader knows anything about functional programming, but some experience of this topic would, of course, not be amiss.

The usual thanks to secretaries and the suchlike for typing the manuscript are in the present case redundant: I typed it all myself. This means that all errors occurring in this study, regardless of what kind, are entirely due to me.

Chris Preston
Bielefeld
March 2000

⁴ The MIT Press, Cambridge, Massachusetts (1993)

Chapter 1 Introduction

1.1 Computing with equations: An example

This study is concerned with explaining how and why functional programming languages work and in particular with what it means for functions to be defined by equations. The analysis of the following simple example should help give a rough idea of what this is all about.

The sequence $\{f_n\}_{n \geq 0}$ of Fibonacci numbers $0, 1, 1, 2, 3, 5, 8, 13, \dots$ is generated by putting $f_0 = 0$, $f_1 = 1$ and for each $n \geq 2$ letting $f_n = f_{n-1} + f_{n-2}$. A possible method of obtaining this sequence is via the function $\mathbf{pfb} : \mathbb{N} \times \mathbb{N}^2 \rightarrow \mathbb{N}^2$ defined as the unique function satisfying the equation

$$\mathbf{pfb}(n, p) = \begin{cases} p & \text{if } n = 0, \\ \mathbf{pfb}(n-1, \mathbf{it}(p)) & \text{otherwise,} \end{cases}$$

where $\mathbf{it} : \mathbb{N}^2 \rightarrow \mathbb{N}^2$ is given by $\mathbf{it}(m, n) = (n, m + n)$ for all $m, n \in \mathbb{N}$. (Both the existence and uniqueness of \mathbf{pfb} can easily be verified using induction on n .) Note that here \mathbb{N} is the set $\{0, 1, 2, \dots\}$, i.e., 0 is considered to be a natural number. The fact that \mathbf{pfb} satisfies this equation is by itself sufficient to conclude, again using induction on n , that $\mathbf{pfb}(n, (f_m, f_{m+1})) = (f_{n+m}, f_{n+m+1})$ for all $m, n \in \mathbb{N}$. In particular,

$$\mathbf{pfb}(n, (0, 1)) = (f_n, f_{n+1})$$

for all $n \in \mathbb{N}$. Thus if $\mathbf{fst} : \mathbb{N}^2 \rightarrow \mathbb{N}$ is given by $\mathbf{fst}(m, n) = m$ for each $(m, n) \in \mathbb{N}^2$ and a function $\mathbf{fib} : \mathbb{N} \rightarrow \mathbb{N}$ is defined by

$$\mathbf{fib}(n) = \mathbf{fst}(\mathbf{pfb}(n, (0, 1)))$$

then $\mathbf{fib}(n) = f_n$ for each $n \in \mathbb{N}$.

The functions $\mathbf{it} : \mathbb{N}^2 \rightarrow \mathbb{N}^2$, $\mathbf{fst} : \mathbb{N}^2 \rightarrow \mathbb{N}$, $\mathbf{pfb} : \mathbb{N} \times \mathbb{N}^2 \rightarrow \mathbb{N}^2$ and $\mathbf{fib} : \mathbb{N} \rightarrow \mathbb{N}$ occurring above can be considered to be defined by the equations

$$\begin{aligned} \mathbf{it}(m, n) &= (n, m + n), \\ \mathbf{fst}(m, n) &= m, \\ \mathbf{pfb}(n, p) &= \begin{cases} p & \text{if } n = 0, \\ \mathbf{pfb}(n-1, \mathbf{it}(p)) & \text{otherwise,} \end{cases} \\ \mathbf{fib}(n) &= \mathbf{fst}(\mathbf{pfb}(n, (0, 1))), \end{aligned}$$

in as much as they are their unique solution, i.e., they are the unique set of functions satisfying these equations. (Saying that the functions satisfy the equations of course means that they satisfy them for all $m, n \in \mathbb{N}$ and all $p \in \mathbb{N}^2$.)

But there is another, more practical, sense in which \mathbf{it} , \mathbf{fst} , \mathbf{pfb} and \mathbf{fib} are defined by the equations: The equations can be seen as replacement (or substitution) rules for

actually computing the values of the functions: For example, to compute $\text{fib}(4)$ first deduce that $\text{pfb}(4, (0, 1)) = (3, 5)$ via the steps

$$\begin{aligned} \text{pfb}(4, (0, 1)) &= \text{pfb}(3, \text{it}(0, 1)) = \text{pfb}(2, \text{it}(\text{it}(0, 1))) \\ &= \text{pfb}(1, \text{it}(\text{it}(\text{it}(0, 1)))) = \text{pfb}(0, \text{it}(\text{it}(\text{it}(\text{it}(0, 1)))) = \text{it}(\text{it}(\text{it}(\text{it}(0, 1)))) \\ &= \text{it}(\text{it}(\text{it}(1, 1))) = \text{it}(\text{it}(1, 2)) = \text{it}(2, 3) = (3, 5) \end{aligned}$$

and then obtain $\text{fib}(4) = \text{fst}(\text{pfb}(4, (0, 1))) = \text{fst}(3, 5) = 3$. In fact, it is easy to see that the value of $\text{fib}(n)$ can be computed in this way for any $n \in \mathbb{N}$.

In most modern functional programming languages the equations for the functions it , fst , pfb and fib can be written more-or-less exactly as they appear above. For example, in the programming language *Haskell* they can be written as

```

it (m,n) = (n,m+n)
fst (m,n) = m
pfb n p
  | n == 0      = p
  | otherwise   = pfb (n-1) (it p)
fib n = fst (pfb n (0,1))

```

(the *Haskell* form of the equation for pfb being just a syntactic variation of the form

$$\begin{aligned} \text{pfb } n \text{ } p &= p, && \text{if } n = 0 \\ &= \text{pfb } (n-1) \text{ } (\text{it } p), && \text{otherwise} \end{aligned}$$

which is how this equation would be written in *Miranda*).

The evaluation mechanism in *Haskell* (or in *Miranda*) would use these equations as replacement rules in essentially the same way as above to compute values. For example, it would reduce the term $\text{fib } 4$ to the constant term 3 via the same steps which were used to deduce that $\text{fib}(4) = 3$.

Now it is important to realize that the *Haskell* equations for it , fst , pfb and fib encode only a part of the information contained in the original equations. To be more specific, if only the *Haskell* equations are given then it is not possible, without additional information, to reconstruct the set-up started with above. For example, they could just as well be describing functions $\text{it} : \mathbb{Z}^2 \rightarrow \mathbb{Z}^2$, $\text{fst} : \mathbb{Z}^2 \rightarrow \mathbb{Z}$, $\text{pfb} : \mathbb{Z} \times \mathbb{Z}^2 \rightarrow \mathbb{Z}^2$ and $\text{fib} : \mathbb{Z} \rightarrow \mathbb{Z}$ satisfying the equations

$$\begin{aligned} \text{it}(m, n) &= (n, m + n), \\ \text{fst}(m, n) &= m, \\ \text{pfb}(n, p) &= \begin{cases} p & \text{if } n = 0, \\ \text{pfb}(n - 1, \text{it}(p)) & \text{otherwise,} \end{cases} \\ \text{fib}(n) &= \text{fst}(\text{pfb}(n, (0, 1))) \end{aligned}$$

for all $m, n \in \mathbb{Z}$, $p \in \mathbb{Z}^2$. If this is the case then there is a problem, since the new

equations no longer have a unique solution, and so it does not make sense to speak of *the* functions defined by the equations. In fact, for each $h : \mathbb{Z}^2 \rightarrow \mathbb{Z}^2$ there is a (unique) function $\mathbf{pfb}_h : \mathbb{Z} \times \mathbb{Z}^2 \rightarrow \mathbb{Z}^2$ satisfying the equation for \mathbf{pfb} and such that $\mathbf{pfb}_h(-1, p) = h(p)$ for each $p \in \mathbb{Z}^2$. There is then also the corresponding function $\mathbf{fib}_h : \mathbb{Z} \rightarrow \mathbb{Z}$ satisfying the equation for \mathbf{fib} given by $\mathbf{fib}_h(n) = \mathbf{fst}(\mathbf{pfb}_h(n, (0, 1)))$ for each $n \in \mathbb{Z}$. However, the values of interest *are* uniquely determined by the new equations: That fact that \mathbf{pfb}_h satisfies the equation for \mathbf{pfb} still suffices to conclude

$$\mathbf{pfb}_h(n, (0, 1)) = (f_n, f_{n+1})$$

for all $n \in \mathbb{N}$, and thus $\mathbf{fib}_h(n) = f_n$ for all $n \in \mathbb{N}$, independent of h . Moreover, the new equations, considered as replacement rules, behave exactly like the original ones in the way they would be used to compute the value of $\mathbf{fib}(n)$ for any $n \in \mathbb{N}$.

A bit more generally, the following holds: Let $n \in \mathbb{Z}$ and $p \in \mathbb{Z}^2$; then $\mathbf{pfb}_h(n, p)$ does not depend on h if and only if $n \in \mathbb{N}$. Thus for $n \in \mathbb{N}$ it makes sense to speak of the value of $\mathbf{pfb}(n, p)$. Furthermore, it is easy to see that this value can be computed using the equations as replacement rules. A corresponding statement then holds for \mathbf{fib} , namely that $\mathbf{fib}_h(n)$ does not depend on h if and only if $n \in \mathbb{N}$, and if $n \in \mathbb{N}$ then, as already claimed above, the value of $\mathbf{fib}(n)$ can be computed exactly as in the original formulation.

Now it is a fact that in all functional programming languages values are computed by using the equations as replacement rules. The reason that such languages actually work is thus that some form of what could be called the *completeness requirement* is satisfied: This requirement is that whenever the value of a function is independent of which solution is being used then this value can be computed using the equations as replacement rules.

The completeness requirement is satisfied by both of the interpretations of the *Haskell* equations looked at above. However, this must have something to do with the form of these equations: Consider the *Haskell* equation

$$h\ n = 2 * (h\ n)$$

Following the first interpretation of the original *Haskell* equations, this then describes a function $h : \mathbb{N} \rightarrow \mathbb{N}$ satisfying the equation $h(n) = 2 \times h(n)$ for all $n \in \mathbb{N}$. In this case the unique solution is with $h(n) = 0$ for all $n \in \mathbb{N}$. But there is clearly no way here of using the equation as a replacement rule in order to compute that the value of $h(n)$ is 0 for any $n \in \mathbb{N}$. Moreover, the situation is not changed if the the second interpretation is followed based on \mathbb{Z} instead of \mathbb{N} .

The aim of this study is to develop a framework within which all equations satisfy the completeness requirement and then to use the framework to ‘explain’ a simple functional programming language. Of course, the example in the previous paragraph implies that such a framework will have to involve something more than that occurring in the two interpretations of the *Haskell* equations for it , \mathbf{fst} , \mathbf{pfb} and \mathbf{fib} considered above.

The second of these interpretations of the *Haskell* equations (using \mathbb{Z} instead of \mathbb{N}) is perhaps closer to the ‘correct’ interpretation, since in *Haskell* the built-in data objects for whole numbers, as in most programming languages, correspond to \mathbb{Z} and not \mathbb{N} .

However, there is still an important ingredient which is missing: The fact that, for example, $\text{fib}_h(-1)$ depends on h can be regarded as saying that $\text{fib}(-1)$ is ‘undefined’, and this suggests that the set-up should be extended to also include ‘undefined’ data objects.

To make this more precise, let $\perp_{\mathbb{Z}}$ be some element not in \mathbb{Z} (to be thought of as an ‘undefined’ element of \mathbb{Z}) and put $\mathbb{Z}^\perp = \mathbb{Z} \cup \{\perp_{\mathbb{Z}}\}$. Moreover, put $\mathbb{P} = \mathbb{Z}^2$, let $\perp_{\mathbb{P}}$ be some element not in \mathbb{P} (to be thought of as an ‘undefined’ element of \mathbb{P}) and let $\mathbb{P}^\perp = \mathbb{P} \cup \{\perp_{\mathbb{P}}\}$. Then a third interpretation of the *Haskell* equations is that they describe functions $\text{it} : \mathbb{P}^\perp \rightarrow \mathbb{P}^\perp$, $\text{fst} : \mathbb{P}^\perp \rightarrow \mathbb{Z}^\perp$, $\text{pfb} : \mathbb{Z}^\perp \times \mathbb{P}^\perp \rightarrow \mathbb{P}^\perp$ and $\text{fib} : \mathbb{Z}^\perp \rightarrow \mathbb{Z}^\perp$ satisfying the equations

$$\begin{aligned} \text{it}(p) &= \begin{cases} (m, \ell + m) & \text{if } p = (\ell, m) \in \mathbb{P}, \\ \perp_{\mathbb{P}} & \text{if } p = \perp_{\mathbb{P}}, \end{cases} \\ \text{fst}(p) &= \begin{cases} \ell & \text{if } p = (\ell, m) \in \mathbb{P}, \\ \perp_{\mathbb{Z}} & \text{if } p = \perp_{\mathbb{P}}, \end{cases} \\ \text{pfb}(n, p) &= \begin{cases} p & \text{if } n = 0, \\ \text{pfb}(n-1, \text{it}(p)) & \text{if } n \in \mathbb{Z} \setminus \{0\}, \\ \perp_{\mathbb{P}} & \text{if } n = \perp_{\mathbb{Z}}, \end{cases} \\ \text{fib}(n) &= \text{fst}(\text{pfb}(n, (0, 1))) \end{aligned}$$

for all $n \in \mathbb{Z}^\perp$, $p \in \mathbb{P}^\perp$. The situation here is analogous to that encountered in the second interpretation: For each mapping $h : \mathbb{P}^\perp \rightarrow \mathbb{P}^\perp$ there is a (unique) function $\text{pfb}_h : \mathbb{Z}^\perp \times \mathbb{P}^\perp \rightarrow \mathbb{P}^\perp$ satisfying the equation for pfb and such that $\text{pfb}_h(-1, p) = h(p)$ for each $p \in \mathbb{P}^\perp$. There is then also the corresponding function $\text{fib}_h : \mathbb{Z}^\perp \rightarrow \mathbb{Z}^\perp$ satisfying the equation for fib given by $\text{fib}_h(n) = \text{fst}(\text{pfb}_h(n, (0, 1)))$ for each $n \in \mathbb{Z}^\perp$. It is again the case that $\text{pfb}_h(n, p)$ and $\text{fib}_h(n)$ do not depend on h if and only if $n \in \mathbb{N}$, and thus the completeness requirement is still met in this new set-up, since if $n \in \mathbb{N}$ then the equations can be used as replacement rules just as before in order to compute the values of $\text{pfb}(n, p)$ and $\text{fib}(n)$.

Consider the mapping $\perp : \mathbb{P}^\perp \rightarrow \mathbb{P}^\perp$ with $\perp(p) = \perp_{\mathbb{P}}$ for each $p \in \mathbb{P}^\perp$. Then the ‘least-defined’ solution of the equations is obtained by taking $h = \perp$: More precisely, $\text{pfb}_\perp(n, p) = \perp_{\mathbb{P}}$ and $\text{fib}_\perp(n) = \perp_{\mathbb{Z}}$ whenever $n \notin \mathbb{N}$. Thus $\text{pfb}_\perp(n, p) = \perp_{\mathbb{P}}$ (resp. $\text{fib}_\perp(n) = \perp_{\mathbb{Z}}$) whenever $\text{pfb}_h(n, p)$ (resp. $\text{fib}_h(n)$) depends on h .

Now in this third interpretation it is not immediately clear what the meaning of an expression such as $\text{fst}(1, \text{fib}_h(-1))$ should be, since fst is a function from \mathbb{P}^\perp to \mathbb{Z}^\perp with $\mathbb{P}^\perp = \mathbb{Z}^2 \cup \{\perp_{\mathbb{P}}\}$, and it is possible that $\text{fib}_h(-1) = \perp_{\mathbb{Z}} \notin \mathbb{Z}$. It is thus necessary to extend the identity mapping from $\mathbb{Z} \times \mathbb{Z}$ to \mathbb{P} to a mapping $(\cdot, \cdot) : \mathbb{Z}^\perp \times \mathbb{Z}^\perp \rightarrow \mathbb{P}^\perp$, and the only reasonable way to do this is to define

$$(\ell, m) = \begin{cases} (\ell, m) & \text{if } \ell, m \in \mathbb{Z}, \\ \perp_{\mathbb{P}} & \text{otherwise.} \end{cases}$$

Note that $\text{fst}(1, \text{fib}_h(-1))$ then depends on h , (since $\text{fst}(1, \text{fib}_h(-1)) = 1$ if $h \neq \perp$ but $\text{fst}(1, \text{fib}_\perp(-1)) = \perp_{\mathbb{Z}}$); in this sense the value of the expression $\text{fst}(1, \text{fib}(-1))$ is not defined.

The third interpretation of the *Haskell* equations can be considered as an extension of the second obtained by adding ‘undefined’ data objects. This is essentially equivalent to extending the second interpretation to allow not only total but also partial functions as solutions of the equations: If X and Y are sets then a function $g : X' \rightarrow Y$ with $X' \subset X$ is called a *partial function* from X to Y ; the set X' is then called the *domain* of g and is denoted by $\text{Dom}(g)$. Of course, an ‘ordinary’ function $g : X \rightarrow Y$ is also a partial function with $\text{Dom}(g) = X$, and the adjective *total* will be used when it is deemed necessary to emphasise the fact that such a function is involved.

For each set X let \perp_X be some element not belonging to X and put $X^\perp = X \cup \{\perp_X\}$. As above, \perp_X should be thought of as representing an ‘undefined’ value from X . Let X_1, \dots, X_n and Y be sets; then for each partial function g from $X_1 \times \dots \times X_n$ to Y a total function $g^\perp : X_1^\perp \times \dots \times X_n^\perp \rightarrow Y^\perp$ can be defined by letting

$$g^\perp(x_1, \dots, x_n) = \begin{cases} g(x_1, \dots, x_n) & \text{if } (x_1, \dots, x_n) \in \text{Dom}(g), \\ \perp_Y & \text{otherwise.} \end{cases}$$

A function $f : X_1^\perp \times \dots \times X_n^\perp \rightarrow Y^\perp$ is said to be *strict* if $f(x_1, \dots, x_n) = \perp_Y$ whenever $x_j = \perp_{X_j}$ for some $j = 1, \dots, n$. Note that the functions occurring as solutions of the equations in the third interpretation are all strict.

Proposition 1.1.1 *The mapping $g \mapsto g^\perp$ maps the set of all partial functions from $X_1 \times \dots \times X_n$ to Y bijectively onto the set of all strict functions from $X_1^\perp \times \dots \times X_n^\perp$ to Y^\perp .*

Proof It is clear that g^\perp is a strict function. Moreover,

$$\text{Dom}(g) = \{(x_1, \dots, x_n) \in X_1 \times \dots \times X_n : g^\perp(x_1, \dots, x_n) \neq \perp_Y\}$$

and $g(x_1, \dots, x_n) = g^\perp(x_1, \dots, x_n)$ for all $(x_1, \dots, x_n) \in \text{Dom}(g)$, and therefore the mapping $g \mapsto g^\perp$ is injective. Finally, if $h : X_1^\perp \times \dots \times X_n^\perp \rightarrow Y^\perp$ is a strict function then $h = g^\perp$, where g is the partial function from $X_1 \times \dots \times X_n$ to Y with

$$\text{Dom}(g) = \{(x_1, \dots, x_n) \in X_1 \times \dots \times X_n : h(x_1, \dots, x_n) \neq \perp_Y\}$$

and $g(x_1, \dots, x_n) = h(x_1, \dots, x_n)$ for all $(x_1, \dots, x_n) \in \text{Dom}(g)$. \square

Using the bijection given in Proposition 1.1.1 the solutions of the equations in the third interpretation can be regarded as being made up of partial functions. Now there are several good reasons for allowing not just total but also partial functions. The main practical reason is that it increases the chances of having a solution. Related to this is the well-known fact that it is in general impossible to decide whether a given set of equations has a solution composed of total functions. Moreover, even if a particular set of equations has such a solution, it is often the case that the ‘most natural’ solution involves partial functions. For instance, in the third interpretation the ‘least-defined’ solution can be considered as the ‘most natural’ one: The strict function \mathbf{pfb}_\perp corresponds to a partial function with domain $\mathbb{N} \times \mathbb{P}$ and \mathbf{fib}_\perp corresponds to a partial function with domain \mathbb{N} .

However, rather than work with partial functions directly it has been decided to work with the alternative formulation involving ‘undefined’ data objects (and then to make

an implicit appeal to Proposition 1.1.1). In fact there is a small but important point in which these two approaches are not completely equivalent, and this can be seen by looking at the definition of a constant function: Consider the *Haskell* equation

$$h\ n = 1$$

As in the third interpretation of the original *Haskell* equations, this can be regarded as describing a function $h : \mathbb{Z}^\perp \rightarrow \mathbb{Z}^\perp$ satisfying the equation $h(n) = 1$ for all $n \in \mathbb{Z}^\perp$, which trivially has a unique solution. But this solution is not a strict function, and so it does not correspond via Proposition 1.1.1 to a partial function from \mathbb{Z} to \mathbb{Z} . Of course, it could simply be stipulated that the solution of the equation has to be the strict function $h' : \mathbb{Z}^\perp \rightarrow \mathbb{Z}^\perp$ given by

$$h'(n) = \begin{cases} 1 & \text{if } n \in \mathbb{Z}, \\ \perp_{\mathbb{Z}} & \text{if } n = \perp_{\mathbb{Z}}, \end{cases}$$

which corresponds to the constant function from \mathbb{Z} to \mathbb{Z} with value 1. However, it will be seen that h and not h' is usually the more appropriate solution.

It turns out that a framework within which the completeness requirement is met for all equations is more likely to be found in the direction of the third rather than that of the second interpretation. For example, again consider the *Haskell* equation

$$h\ n = 2 * (h\ n)$$

If the first or second interpretation of the original *Haskell* equations are followed then there is a unique solution with $h(n) = 0$ for all n , but there is no way of using the equation as a replacement rule in order to compute that the value of $h(n)$ is 0 for any n . In other words, the completeness requirement is not satisfied here. This problem does not occur if the third interpretation is followed: The equation then describes a function $h : \mathbb{Z}^\perp \rightarrow \mathbb{Z}^\perp$ satisfying the equation $h(n) = 2 \times h(n)$ for all $n \in \mathbb{Z}^\perp$, and the only sensible definition for $2 \times \perp_{\mathbb{Z}}$ is that it should be $\perp_{\mathbb{Z}}$. Therefore in this case there is a ‘least-defined’ solution with $h(n) = \perp_{\mathbb{Z}}$ for all $n \in \mathbb{Z}^\perp$ (which corresponds to the trivial partial function with empty domain) as well as the solution with $h(\perp_{\mathbb{Z}}) = \perp_{\mathbb{Z}}$ and $h(n) = 0$ for all $n \in \mathbb{Z}$. Thus the completeness requirement is satisfied trivially, since here there are no values of the function h which need to be computed.

Here is another example which gives a further reason why the third interpretation is to be preferred to the second: Consider the following *Haskell* equations

$$\begin{aligned} g\ n & \\ | \ n == 0 & \quad = \ 0 \\ | \ \text{otherwise} & \quad = \ 1 + g\ n \\ h\ n & = \ g\ 0 \end{aligned}$$

Following the second interpretation of the original *Haskell* equations, these describe functions $g : \mathbb{Z} \rightarrow \mathbb{Z}$ and $h : \mathbb{Z} \rightarrow \mathbb{Z}$ satisfying the equations

$$\mathbf{g}(n) = \begin{cases} 0 & \text{if } n = 0, \\ 1 + \mathbf{g}(n) & \text{otherwise,} \end{cases}$$

$$\mathbf{h}(n) = \mathbf{g}(0)$$

for all $n \in \mathbb{Z}$, and here there is the problem that, although these equations have no solution, they can still be used as replacement rules to compute that the value of $\mathbf{h}(0)$ is 0 for each $n \in \mathbb{Z}$. Once again, this problem does not occur if the third interpretation is followed: The equations then describe functions $\mathbf{g} : \mathbb{Z}^\perp \rightarrow \mathbb{Z}^\perp$ and $\mathbf{h} : \mathbb{Z}^\perp \rightarrow \mathbb{Z}^\perp$ satisfying the equations

$$\mathbf{g}(n) = \begin{cases} 0 & \text{if } n = 0, \\ 1 + \mathbf{g}(n) & \text{if } n \in \mathbb{Z} \setminus \{0\}, \\ \perp_{\mathbb{Z}} & \text{if } n = \perp_{\mathbb{Z}}, \end{cases}$$

$$\mathbf{h}(n) = \mathbf{g}(0)$$

for all $n \in \mathbb{Z}^\perp$. The only sensible definition of $1 + \perp_{\mathbb{Z}}$ is that it should be $\perp_{\mathbb{Z}}$, and hence the equations now have a unique solution which is given by

$$\mathbf{g}(n) = \begin{cases} 0 & \text{if } n = 0, \\ \perp_{\mathbb{Z}} & \text{otherwise,} \end{cases}$$

(so \mathbf{g} corresponds to a partial function with domain $\{0\}$) and $\mathbf{h}(n) = 0$ for all $n \in \mathbb{Z}^\perp$. In particular, since $\mathbf{h}(n) = 0$ for all $n \in \mathbb{Z}$, it is no longer paradoxical that the equations can be used as replacement rules to compute that the value of $\mathbf{h}(n)$ is 0 for each $n \in \mathbb{Z}$.

Now, although the third interpretation is, in itself, completely satisfactory, it is not quite the interpretation to be found in *Haskell*. The difference lies in the choice of \mathbb{P}^\perp as the ‘correct’ extension of the set \mathbb{P} . Suppose that $\perp_{\mathbb{P}} \notin \mathbb{Z}^\perp \times \mathbb{Z}^\perp$ and put $\mathbb{P}^\diamond = \mathbb{Z}^\perp \times \mathbb{Z}^\perp \cup \{\perp_{\mathbb{P}}\}$. Then a fourth interpretation of the *Haskell* equations is that they describe functions $\text{it} : \mathbb{P}^\diamond \rightarrow \mathbb{P}^\diamond$, $\text{fst} : \mathbb{P}^\diamond \rightarrow \mathbb{Z}^\perp$, $\text{pfb} : \mathbb{Z}^\perp \times \mathbb{P}^\diamond \rightarrow \mathbb{P}^\diamond$ and $\text{fib} : \mathbb{Z}^\perp \rightarrow \mathbb{Z}^\perp$ satisfying the equations

$$\text{it}(p) = \begin{cases} (m, \ell + m) & \text{if } p = (\ell, m) \in \mathbb{Z}^\perp \times \mathbb{Z}^\perp, \\ \perp_{\mathbb{P}} & \text{if } p = \perp_{\mathbb{P}}, \end{cases}$$

$$\text{fst}(p) = \begin{cases} \ell & \text{if } p = (\ell, m) \in \mathbb{Z}^\perp \times \mathbb{Z}^\perp, \\ \perp_{\mathbb{Z}} & \text{if } p = \perp_{\mathbb{P}}, \end{cases}$$

$$\text{pfb}(n, p) = \begin{cases} p & \text{if } n = 0, \\ \text{pfb}(n - 1, \text{it}(p)) & \text{if } n \in \mathbb{Z} \setminus \{0\}, \\ \perp_{\mathbb{P}} & \text{if } n = \perp_{\mathbb{Z}}, \end{cases}$$

$$\text{fib}(n) = \text{fst}(\text{pfb}(n, (0, 1)))$$

for all $n \in \mathbb{Z}^\perp$, $p \in \mathbb{P}^\diamond$. Of course, here it is necessary to extend the usual addition on \mathbb{Z} to a binary operation on \mathbb{Z}^\perp which is done by defining $\ell + m$ to be $\perp_{\mathbb{Z}}$ whenever

$(\ell, m) \notin \mathbb{P}$. This interpretation adds not only the ‘undefined’ object $\perp_{\mathbb{P}}$ to the set \mathbb{P} but also the elements in the set

$$\mathbb{P}^{\circ} \setminus \mathbb{P}^{\perp} = \{(\perp_{\mathbb{Z}}, \perp_{\mathbb{Z}})\} \cup \{(m, \perp_{\mathbb{Z}}) : m \in \mathbb{Z}\} \cup \{(\perp_{\mathbb{Z}}, n) : n \in \mathbb{Z}\}$$

which can be considered as ‘partially defined’ objects. It is reasonable to regard these ‘partially defined’ objects as being not completely ‘undefined’. For instance, it makes sense to consider the first component of $(m, \perp_{\mathbb{Z}})$ for each $m \in \mathbb{Z}$.

It is the fourth interpretation which is actually employed by *Haskell*. The situation here is more-or-less the same as that in the third interpretation: Note that if the function $\mathbf{pfb} : \mathbb{Z}^{\perp} \times \mathbb{P}^{\circ} \rightarrow \mathbb{P}^{\circ}$ satisfies the equation for \mathbf{pfb} then $\mathbf{pfb}(-1, p) = \mathbf{pfb}(-1, (\perp_{\mathbb{Z}}, \perp_{\mathbb{Z}}))$ for all $p \in \mathbb{P}^{\circ} \setminus \mathbb{P}^{\perp}$ (since

$$\mathbf{pfb}(-1, p) = \mathbf{pfb}(-3, \text{it}(\text{it}(p))) = \mathbf{pfb}(-3, (\perp_{\mathbb{Z}}, \perp_{\mathbb{Z}})) = \mathbf{pfb}(-1, (\perp_{\mathbb{Z}}, \perp_{\mathbb{Z}}))$$

whenever $p \in \mathbb{P}^{\circ} \setminus \mathbb{P}^{\perp}$). Conversely, if $h : \mathbb{P}^{\circ} \rightarrow \mathbb{P}^{\circ}$ is such that $h(p) = h((\perp_{\mathbb{Z}}, \perp_{\mathbb{Z}}))$ for all $p \in \mathbb{P}^{\circ} \setminus \mathbb{P}^{\perp}$ then there exists a (unique) function $\mathbf{pfb}_h : \mathbb{Z}^{\perp} \times \mathbb{P}^{\circ} \rightarrow \mathbb{P}^{\circ}$ satisfying the equation for \mathbf{pfb} and such that $\mathbf{pfb}_h(-1, p) = h(p)$ for each $p \in \mathbb{P}^{\circ}$. There is then also the corresponding function $\mathbf{fib}_h : \mathbb{Z}^{\perp} \rightarrow \mathbb{Z}^{\perp}$ satisfying the equation for \mathbf{fib} given by $\mathbf{fib}_h(n) = \mathbf{fst}(\mathbf{pfb}_h(n, (0, 1)))$ for each $n \in \mathbb{Z}^{\perp}$. It is not the details of these solutions which are important, but the fact that $\mathbf{pfb}_h(n, p)$ and $\mathbf{fib}_h(n)$ are independent of h if and only if $n \in \mathbb{N}$. Thus the completeness requirement is once again met here, since if $n \in \mathbb{N}$ then the values of $\mathbf{pfb}(n, p)$ and $\mathbf{fib}(n)$ can be computed as before using the equations as replacement rules.

There is again a ‘least-defined’ solution of the equations, which is obtained by taking $h = \perp$, where $\perp : \mathbb{P}^{\circ} \rightarrow \mathbb{P}^{\circ}$ is the mapping given by $\perp(p) = \perp_{\mathbb{P}}$ for each $p \in \mathbb{P}^{\circ}$. Then $\mathbf{pfb}_{\perp}(n, p) = \perp_{\mathbb{P}}$ and $\mathbf{fib}_{\perp}(n) = \perp_{\mathbb{Z}}$ whenever $n \notin \mathbb{N}$. Thus $\mathbf{pfb}_{\perp}(n, p) = \perp_{\mathbb{P}}$ (resp. $\mathbf{fib}_{\perp}(n) = \perp_{\mathbb{Z}}$) whenever $\mathbf{pfb}_h(n, p)$ (resp. $\mathbf{fib}_h(n)$) depends on h .

The identity mapping from $\mathbb{Z} \times \mathbb{Z}$ to \mathbb{P} was extended in the third interpretation to a mapping $(\cdot, \cdot) : \mathbb{Z}^{\perp} \times \mathbb{Z}^{\perp} \rightarrow \mathbb{P}^{\perp}$. In the fourth interpretation the corresponding extension is just the inclusion mapping of $\mathbb{Z}^{\perp} \times \mathbb{Z}^{\perp}$ into \mathbb{P}° . Note that the expression $\mathbf{fst}(1, \mathbf{fib}(-1))$ now has the value 1, since $\mathbf{fst}(1, \mathbf{fib}_h(-1)) = 1$ for all h .

Whether the fourth interpretation is to be preferred to the third is, to some extent, just a matter of taste. However, it is a fact that, whereas the ‘older’ functional programming languages (such as the original ‘eager’ version of *ML*) use the third interpretation, the ‘newer’ languages tend to use the fourth (or a slight modification of it in which the ‘undefined’ element $\perp_{\mathbb{P}}$ is identified with the element $(\perp_{\mathbb{Z}}, \perp_{\mathbb{Z}})$ of $\mathbb{Z}^{\perp} \times \mathbb{Z}^{\perp}$). The framework to be presented in this study includes all of these interpretations.

Of course, (as has been seen above with $\mathbf{fst}(1, \mathbf{fib}(-1))$) the value of an expression can depend on which interpretation is being used. This fact must be borne in mind when the equations are used as replacement rules. For instance, the replacement of $\mathbf{fst}(1, \mathbf{fib}(-1))$ by 1, which would seem to be justified by the definition of \mathbf{fst} , is valid in the fourth interpretation but not in the third.

1.2 A procedure for interpreting equations

In this section a procedure will be outlined which, in particular, can be used to obtain the third and fourth interpretations of the *Haskell* equations for `it`, `fst`, `pfb` and `fib` introduced in the previous section. In its general form the procedure will then constitute the basis of the study.

First, however, it is convenient to rewrite the *Haskell* equations using a uniform prefix notation. More precisely, the infix operator names `+`, `-` and `==` will be replaced by corresponding prefix operator names `add`, `sub` and `eq`, and the built-in constructor name `(.,.)` for pair-building by a prefix constructor name `Pair`. Moreover, `otherwise` will be replaced by `True` (for which it is really just a synonym), and this results in the equations taking on the following guise:

```
it (Pair m n) = Pair n (add m n)
fst (Pair m n) = m
pfb n p
  | eq n 0    = p
  | True      = pfb (sub n 1) (it p)
fib n = fst (pfb n (Pair 0 1))
```

Note the two usages of brackets in the original *Haskell* equations: The first, in the form `(.,.)` to denote the pair-building operation, has now been replaced with the prefix operator name `Pair`. The second (for example in `pfb (sub n 1) (it p)` and `fst (pfb n (Pair 0 1))`) is as a method of determining how a term is constructed out of its subterms. As can be seen, brackets will still be employed for this purpose.

The above equations are essentially built up out of names, which can be classified into the following four groups:

- The names `it`, `fst`, `fpb` and `fib`, which are those of the functions which the equations are supposed to define.
- The names `p`, `m` and `n`, which play the role of ‘local variables’.
- The names `add`, `sub` and `eq`, which are those of ‘primitive’ operators.
- The names `Pair`, `True`, `0` and `1`, which are those of ‘data constructors’.

In order to understand how these names are allowed to fit together each name must be first assigned to a type. There are three basic data types involved here which will be denoted by `int` (for *integer*), `ipr` (for *integer pair*), and `bool` (for *boolean*). Each name which refers to a data object is assigned the corresponding type: The names `m`, `n`, `0` and `1` are assigned the type `int`, `p` is assigned the type `ipr` and `True` is assigned the type `bool`. The remaining names are names of functions, and each of these is assigned a functional type of the form $\beta_1 \cdots \beta_n \rightarrow \beta$, where $\beta_1, \dots, \beta_n, \beta$ are basic data types, i.e., elements of the set `{int, ipr, bool}`. For example, `pfb` is assigned the type `int ipr → ipr`, since it is the name of a function having two arguments, the first argument taking values of type `int`, the second values of type `ipr`, and which

produces values of type `ipr`. Similarly, the name `it` is assigned the type `ipr → ipr`, `fst` the type `ipr → int`, `fib` the type `int → int`, `add` and `sub` are assigned the type `int int → int`, `eq` the type `int int → bool` and, finally, `Pair` is assigned the type `int int → ipr`.

With these assignments it is easily checked that each of the individual components of an equation is a ‘syntactically correct’ term: The ‘syntactically correct’ terms of type `int` occurring in the equations are the four terms

$$\text{fst (Pair } m \ n), \quad m, \quad \text{fib } n, \quad \text{fst (pfb } n \ (\text{Pair } 0 \ 1)),$$

those of type `ipr` are the five terms

$$\text{it (Pair } m \ n), \quad \text{Pair } n \ (\text{add } m \ n), \quad \text{pfb } n \ p, \quad p, \quad \text{pfb (sub } n \ 1) \ (\text{it } p)$$

and those of type `bool` are `eq n 0` and `True`.

Note that in each of the equations for `it`, `fst` and `fib` the left- and right-hand sides are terms of the same type. Similarly, `pfb n p` (which can be seen as the left-hand side of the equation for `pfb`) is a term of type `ipr`, as are the two possible right-hand sides `p` and `pfb (sub n 1) (it p)` (each of which is ‘guarded’ by a term of type `bool`).

Associated with the set $B = \{\text{int}, \text{ipr}, \text{bool}\}$ of basic data types needed in the *Haskell* equations is a corresponding set

$$K = \{\text{Pair}, \text{True}, \text{False}\} \cup \underline{Int}$$

of ‘data constructor’ names, where here \underline{Int} is the set containing for each integer $n \in \mathbb{Z}$ its standard representation \underline{n} as a string of characters. (Of course, the names `Pair`, `True`, `0` and `1` from K already occurred in the equations.) Moreover, each element of K is assigned a type: `True` and `False` are of type `bool`, \underline{n} is of type `int` for each $n \in \mathbb{Z}$ and `Pair` is of type `int int → ipr`.

The pair $\Lambda = (B, K)$, together with the assignment of the appropriate type to each element of K , is a particular instance of what is known as a *signature*.

The general procedure as it would be applied to interpreting the *Haskell* equations for `it`, `fst`, `pfb` and `fib` will now be described. It involves taking the following steps:

1. The first step is to choose what is called a Λ -*algebra*. This is any pair (X_B, p_K) consisting of a family of sets $X_B = \{X_\beta : \beta \in B\}$, elements p_{True} , p_{False} of X_{bool} , for each $n \in \mathbb{Z}$ an element $p_{\underline{n}}$ of X_{int} and a mapping $p_{\text{Pair}} : X_{\text{int}} \times X_{\text{int}} \rightarrow X_{\text{ipr}}$.

In both the third and fourth interpretations of the *Haskell* equations the ‘natural’ choice of the Λ -algebra (X_B, p_K) was made. This is with:

- $X_{\text{int}} = \mathbb{Z}$, $X_{\text{ipr}} = \mathbb{P} = \mathbb{Z} \times \mathbb{Z}$ and $X_{\text{bool}} = \mathbb{B} = \{T, F\}$,
- the mapping $p_{\text{Pair}} : X_{\text{int}} \times X_{\text{int}} \rightarrow X_{\text{ipr}}$ given by $p_{\text{Pair}}(m, n) = (m, n)$,
- $p_{\text{True}} = T$ and $p_{\text{False}} = F$,
- $p_{\underline{n}} = n$ for each $n \in \mathbb{Z}$.

In fact, this particular Λ -algebra (X_B, p_K) has a certain universal property, that of being *initial*, which essentially characterises it in the class of all Λ -algebras.

2. The next step is to fix for each $\beta \in B$ an element \perp_β not in the set X_β (to be thought of as an ‘undefined’ object of type β) and then to extend the Λ -algebra (X_B, p_K) to a new Λ -algebra (D_B, f_K) with $\perp_\beta \in D_\beta$ for each $\beta \in B$. By ‘extend’ is meant that $X_\beta \subset D_\beta$ for each $\beta \in B$, $f_\kappa = p_\kappa$ for each $\kappa \in K \setminus \{\text{Pair}\}$ and that p_{Pair} is the restriction of f_{Pair} to $X_{\text{int}} \times X_{\text{int}}$.

The third interpretation of the *Haskell* equations took $D_{\text{int}} = \mathbb{Z}^\perp$, $D_{\text{ipr}} = \mathbb{P}^\perp$ and $D_{\text{bool}} = \mathbb{B}^\perp = \{T, F, \perp_{\text{bool}}\}$ (with $\perp_{\text{int}} = \perp_{\mathbb{Z}}$ and $\perp_{\text{ipr}} = \perp_{\mathbb{P}}$) and defined the mapping $f_{\text{Pair}} : D_{\text{int}} \times D_{\text{int}} \rightarrow D_{\text{ipr}}$ to be given by

$$f_{\text{Pair}}(m, n) = \begin{cases} (m, n) & \text{if } m, n \in \mathbb{Z}, \\ \perp_{\text{ipr}} & \text{otherwise.} \end{cases}$$

In the fourth interpretation $D_{\text{int}} = \mathbb{Z}^\perp$, $D_{\text{ipr}} = \mathbb{P}^\diamond$ and $D_{\text{bool}} = \mathbb{B}^\perp = \{T, F, \perp_{\text{bool}}\}$ (again with $\perp_{\text{int}} = \perp_{\mathbb{Z}}$ and $\perp_{\text{ipr}} = \perp_{\mathbb{P}}$) and $f_{\text{Pair}} : D_{\text{int}} \times D_{\text{int}} \rightarrow D_{\text{ipr}}$ was taken to be the inclusion mapping of $D_{\text{int}} \times D_{\text{int}} = \mathbb{Z}^\perp \times \mathbb{Z}^\perp$ into $D_{\text{ipr}} = \mathbb{Z}^\perp \times \mathbb{Z}^\perp \cup \{\perp_{\text{ipr}}\}$.

3. The third step is to decide which kinds of functions will be allowed to occur as the solutions of equations. This is done by for each functional type $\sigma = \beta_1 \cdots \beta_n \rightarrow \beta$ choosing an appropriate subset D_σ of the set $D_\beta^{D_{\beta_1} \times \cdots \times D_{\beta_n}}$ of all functions from $D_{\beta_1} \times \cdots \times D_{\beta_n}$ to D_β . The only functions of type σ which will be allowed as solutions are then those lying in the set D_σ . In this way the family of sets D_B is extended to a family $D_S = \{D_\sigma : \sigma \in S\}$, where $S = B \cup F$ and where F is the set of functional types.

The third and fourth interpretations of the *Haskell* equations made no restrictions on the functions. In other words, the choice was to take $D_\sigma = D_\beta^{D_{\beta_1} \times \cdots \times D_{\beta_n}}$ for each functional type $\sigma = \beta_1 \cdots \beta_n \rightarrow \beta$. However, more typical is the situation to be considered later in which the functions are restricted to being, in a certain sense, either ‘monotone’ or ‘continuous’.

4. The next step is to interpret the names **add**, **sub** and **eq** of the ‘primitive’ operators occurring in the equations. This means that mappings **add** and **sub** must be chosen from $D_{\text{int}} \times D_{\text{int}}$ to D_{int} as well as a mapping **eq** : $D_{\text{int}} \times D_{\text{int}} \rightarrow D_{\text{bool}}$. But in practice there is really no choice here: As in the third and fourth interpretations of the *Haskell* equations it will always be the case that $X_{\text{int}} = \mathbb{Z}$ and $X_{\text{bool}} = \mathbb{B}$ and then **add**, **sub** and **eq** more-or-less have to be chosen to be the ‘obvious’ strict extensions of the operators $+$, $-$ and $=$. (Of course, in general the rest of the usual arithmetical and relational operators are allowed to occur, and then these must also be interpreted in the same way.)

Suppose appropriate choices for the Λ -algebras (X_B, p_K) and (D_B, f_K) and for the family D_S have been made. In particular, it is assumed that $X_{\text{int}} = \mathbb{Z}$ with $p_n = n$ for each $n \in \mathbb{Z}$, $X_{\text{bool}} = \mathbb{B}$ with $p_{\text{True}} = T$, $p_{\text{False}} = F$, and that $D_{\text{int}} = \mathbb{Z} \cup \perp_{\text{int}}$ and $D_{\text{bool}} = \mathbb{B} \cup \perp_{\text{bool}}$. Then the interpretation of the *Haskell* equations is that

they now describe elements $it \in D_{\text{ipr} \rightarrow \text{ipr}}$, $fst \in D_{\text{ipr} \rightarrow \text{int}}$, $pfb \in D_{\text{int} \text{ ipr} \rightarrow \text{ipr}}$ and $fib \in D_{\text{int} \rightarrow \text{int}}$ (and so in particular they describe functions $it : D_{\text{ipr}} \rightarrow D_{\text{ipr}}$, $fst : D_{\text{ipr}} \rightarrow D_{\text{int}}$, $pfb : D_{\text{int}} \times D_{\text{ipr}} \rightarrow D_{\text{ipr}}$ and $fib : D_{\text{int}} \rightarrow D_{\text{int}}$) satisfying

$$\begin{aligned}
 it(p) &= \begin{cases} f_{\text{Pair}}(\ell, \text{add}(k, \ell)) & \text{if } p \neq \perp_{\text{ipr}} \text{ and } p = f_{\text{Pair}}(k, \ell) \\ & \text{with } (k, \ell) \in D_{\text{int}} \times D_{\text{int}}, \\ \perp_{\text{ipr}} & \text{if } p = \perp_{\text{ipr}}, \end{cases} \\
 fst(p) &= \begin{cases} k & \text{if } p \neq \perp_{\text{ipr}} \text{ and } p = f_{\text{Pair}}(k, \ell) \\ & \text{with } (k, \ell) \in D_{\text{int}} \times D_{\text{int}}, \\ \perp_{\text{int}} & \text{if } p = \perp_{\text{ipr}}, \end{cases} \\
 pfb(n, p) &= \begin{cases} p & \text{if } \text{eq}(n, 0) = F, \\ pfb(\text{sub}(n, 1), it(p)) & \text{if } \text{eq}(n, 0) = T, \\ \perp_{\text{ipr}} & \text{if } \text{eq}(n, 0) = \perp_{\text{bool}}, \end{cases} \\
 fib(n) &= fst(pfb(n, f_{\text{Pair}}(0, 1)))
 \end{aligned}$$

for all $n \in D_{\text{int}}$, $p \in D_{\text{ipr}}$. Note that this formulation requires the Λ -algebra (D_B, f_K) to satisfy certain conditions: The equations for it and fst really only make sense if for each $p \in D_{\text{ipr}}$ with $p \neq \perp_{\text{ipr}}$ there exists unique elements $k, \ell \in D_{\text{int}}$ such that $p = f_{\text{Pair}}(k, \ell)$. This requirement will later lead to the concept of what will be called a *regular* extension of the Λ -algebra (X_B, p_K) .

The reader is left to check that the third and fourth interpretations of the *Haskell* equations introduced in the previous section can both be obtained as special cases of the more general procedure outlined above.

The preceding discussion indicates a part of what is involved in defining functions by equations, and it corresponds to what will be dealt with in the first part of this study. The choice has been made to work with a language which is essentially based on the ‘case’ construction as it occurs in *Haskell*. Using this instead of the ‘pattern matching’ occurring in the original equations, the equations for the functions it , fst , pfb and fib can be written in *Haskell* as follows:

```

it p = case p of
      Pair m n -> ita m n
ita m n = Pair n (add m n)
fst p = case p of
      Pair m n -> fsta m n
fsta m n = m
pfb n p = case eq n 0 of
          True -> p
          False -> pfb (sub n 1) (it p)
fib n = fst (pfb n (Pair 0 1))

```

Here `ita` and `fsta` are auxiliary functions which, although they are superfluous in the *Haskell* set-up, are needed in the following, which are the equations for `it`, `fst`, `pfb` and `fib` as they will appear in the language to be introduced in this study:

```

it p = case p of {Pair -> ita}
ita m n = Pair n (add m n)
fst p = case p of {Pair -> fsta}
fsta m n = m
pfb n p = case (eq n 0) of {True -> p,
                             False -> pfb (sub n 1) (it p)}
fib n = fst (pfb n (Pair 0 1))

```

These auxiliary functions are needed because, unlike in *Haskell*, in the framework to be employed here anonymous functions cannot be defined using abstraction (which is essentially what `Pair m n -> Pair n (add m n)` and `Pair m n -> m` denote).

The interpretation of these equations corresponding to that of the *Haskell* equations given above is that they should describe elements $it \in D_{\text{ipr} \rightarrow \text{ipr}}$, $ita \in D_{\text{int} \times \text{int} \rightarrow \text{ipr}}$, $\text{fst} \in D_{\text{ipr} \rightarrow \text{int}}$, $\text{fsta} \in D_{\text{int} \times \text{int} \rightarrow \text{int}}$, $\text{pfb} \in D_{\text{int} \times \text{ipr} \rightarrow \text{ipr}}$ and $\text{fib} \in D_{\text{int} \rightarrow \text{int}}$ (and so in particular they describe functions $it : D_{\text{ipr}} \rightarrow D_{\text{ipr}}$, $ita : D_{\text{int}} \times D_{\text{int}} \rightarrow D_{\text{ipr}}$, $\text{fst} : D_{\text{ipr}} \rightarrow D_{\text{int}}$, $\text{fsta} : D_{\text{int}} \times D_{\text{int}} \rightarrow D_{\text{int}}$, $\text{pfb} : D_{\text{int}} \times D_{\text{ipr}} \rightarrow D_{\text{ipr}}$ and $\text{fib} : D_{\text{int}} \rightarrow D_{\text{int}}$) satisfying the equations

$$\begin{aligned}
it(p) &= \begin{cases} ita(k, \ell) & \text{if } p \neq \perp_{\text{ipr}} \text{ and } p = f_{\text{Pair}}(k, \ell) \\ & \text{with } (k, \ell) \in D_{\text{int}} \times D_{\text{int}}, \\ \perp_{\text{ipr}} & \text{if } p = \perp_{\text{ipr}}, \end{cases} \\
ita(m, n) &= f_{\text{Pair}}(n, \text{add}(m, n)), \\
\text{fst}(p) &= \begin{cases} \text{fsta}(k, \ell) & \text{if } p \neq \perp_{\text{ipr}} \text{ and } p = f_{\text{Pair}}(k, \ell) \\ & \text{with } (k, \ell) \in D_{\text{int}} \times D_{\text{int}}, \\ \perp_{\text{ipr}} & \text{if } p = \perp_{\text{ipr}}, \end{cases} \\
\text{fsta}(m, n) &= m, \\
\text{pfb}(n, p) &= \begin{cases} p & \text{if } \text{eq}(n, 0) = T, \\ \text{pfb}(\text{sub}(n, 1), \text{it}(p)) & \text{if } \text{eq}(n, 0) = F, \\ \perp_{\text{ipr}} & \text{if } \text{eq}(n, 0) = \perp_{\text{bool}}, \end{cases} \\
\text{fib}(n) &= \text{fst}(\text{pfb}(n, f_{\text{Pair}}(0, 1)))
\end{aligned}$$

for all $m, n \in D_{\text{int}}$, $p \in D_{\text{ipr}}$. It is clear that for any reasonable choice of the extension (D_B, f_K) and the family D_S the solutions of these new equations will be, ignoring the functions `ita` and `fsta`, the same as the solutions of the original *Haskell* equations.

In general equations will be defined, as in the particular example above, by starting with a suitable signature $\Lambda = (B, K)$, which provides the data types that can occur in the equations and also the names and the types of the corresponding ‘data constructors’. An equation then consists of a left- and a right-hand side, each of which has to be a ‘syntactically correct’ term of the same type. The terms which occur as left-hand sides have to have a particularly simple form and there are various other conditions which have to be satisfied, for example that the only ‘local variables’ which can occur on the right-hand side of an equation are those occurring on the left-hand side.

The procedure for interpreting the equations is in principle the same as that described above: First choose an initial Λ -algebra (X_B, p_K) , the elements in the set X_β being the basic data objects of type β for each $\beta \in B$, then choose a suitable regular extension (D_B, f_K) of (X_B, p_K) to take account of ‘undefined’ and possibly also ‘partially defined’ data objects, and finally choose a family D_S to determine which kinds of functions can occur as solutions. The interpretation of a system of equations is then that they describe functions, each of which is a member of the appropriate set in the family D_S , and which satisfy the equations obtained in the same way as in the particular example above. (Note that in general it will be possible for a system of equations not to have a solution.)

The fact that (X_B, p_K) is taken to be an initial Λ -algebra implies it is essentially uniquely determined by the signature Λ . The only real choice in the interpretation thus lies in that of the extension (D_B, f_K) and that of the family D_S .

Equations and their solutions will be treated in Chapter 6. In the preceding chapters the necessary mathematical machinery will be developed: Chapter 2 deals with some standard topics from universal algebra, Chapter 3 looks at a framework for specifying data objects, Chapter 4 with a selection of elementary results from domain theory and Chapter 5 with the construction of function hierarchies.

1.3 Equations as replacement rules

This section begins by looking at a couple of examples (again based on the *Haskell* equations for `it`, `fst`, `pfib` and `fib`) in order to recapitulate, and make a bit more precise, what is meant by the value of a term. It then discusses how the value of a term can be computed by using the equations as replacement rules.

Consider first the term `fib 4`; this denotes the expression `fib(4)` (assuming that the ‘data constructor’ `4` actually denotes the number `4`) and in both the third and fourth interpretation of the *Haskell* equations `fib(4)` has the value `3`, regardless of which solution of the equations is being used. In both these interpretations it will thus be said that the term `fib 4` has the value `3`.

Consider next the term `fst (Pair 1 (fib -1))`; assuming the ‘data constructors’ again denote what is expected, this term denotes the expression `fst(1, fib(-1))`. In the fourth interpretation of the *Haskell* equations `fst(1, fib(-1))` has the value `1` for any solution of the equations, and so here the term `fst (Pair 1 (fib -1))` is said to have the value `1`. But in the third interpretation the value of `fst(1, fib(-1))` depends on which solution of the equations is being used, and here it is considered that the term `fst (Pair 1 (fib -1))` does not have a value.

More generally, given a system of equations, a term is regarded as having a value if the value of the expression it denotes is completely defined (i.e., is neither ‘undefined’ nor ‘partially defined’) and is independent of which solution of the equations is being used. Of course, this notion only really makes sense if the equations actually have a solution.

Starting in the second half of Chapter 6, equations will be considered as replacement rules and a basic algorithm will be presented for computing the value of a term. The algorithm takes a term and tries to reduce it to a so-called ground term (i.e., a term containing only names of ‘data constructors’). For example, consider the final form of the equations for `it`, `fst`, `pfib` and `fib` in Section 1.2, and assume that the interpretation corresponding to the third interpretation of the *Haskell* equations is being used. Then the term `pfib 2 (Pair 0 1)` would be reduced to the ground term `Pair 1 2` via the intermediate steps listed in Example 1.3.1 on the following page.

Each of the terms in the list in Example 1.3.1 is replaced by the term following it, and this replacement process does not change values: A term in the list has a value if and only if the term following also has the same value. Thus if this replacement process ever arrives at a ground term then, since a ground term always has a value, the original term must also have a value which is equal to the value of the ground term. In particular, the term `pfib 2 (Pair 0 1)` has the value `(1,2)`, because this is the value of the ground term `Pair 1 2`.

The algorithm depends, of course, on the equations started with. Moreover, it must also depend on the interpretation which is being used. For example, with the same equations as before and the interpretation corresponding to the fourth interpretation of the *Haskell* equations the term `fib (Pair 1 (fib -1))` is reduced immediately to the ground term `1`, which confirms the fact that with this interpretation the term `fst (Pair 1 (fib -1))` does have the value `1`.

Example 1.3.1 Assume the interpretation is being used which corresponds to the third interpretation of the *Haskell* equations. Then `pfb 2 (Pair 0 1)` would be reduced to the ground term `Pair 1 2` via the following steps:

```

pfb 2 (Pair 0 1)
case (eq 2 0) of {True -> Pair 0 1,
                  False -> pfb (sub 2 1) (it (Pair 0 1))}
case False of {True -> Pair 0 1,
               False -> pfb (sub 2 1) (it (Pair 0 1))}
pfb (sub 2 1) (it (Pair 0 1))
case (eq (sub 2 1) 0) of {True -> it (Pair 0 1),
                         False -> pfb (sub (sub 2 1) 1) (it (it (Pair 0 1)))}
case (eq 1 0) of {True -> it (Pair 0 1),
                 False -> pfb (sub (sub 2 1) 1) (it (it (Pair 0 1)))}
case False of {True -> it (Pair 0 1),
               False -> pfb (sub (sub 2 1) 1) (it (it (Pair 0 1)))}
pfb (sub (sub 2 1) 1) (it (it (Pair 0 1)))
case (eq (sub (sub 2 1) 1) 0) of {
  True -> it (it (Pair 0 1)),
  False -> pfb (sub (sub (sub 2 1) 1) 1)
              (it (it (it (Pair 0 1))))}
case (eq (sub 1 1) 0) of {True -> it (it (Pair 0 1)),
                         False -> pfb (sub (sub (sub 2 1) 1) 1)
              (it (it (it (Pair 0 1))))}
case (eq 0 0) of {True -> it (it (Pair 0 1)),
                 False -> pfb (sub (sub (sub 2 1) 1) 1)
              (it (it (it (Pair 0 1))))}
case True of {True -> it (it (Pair 0 1)),
              False -> pfb (sub (sub (sub 2 1) 1) 1)
              (it (it (it (Pair 0 1))))}
it (it (Pair 0 1))

```

(Example 1.3.1 is continued on the next page.)

Example 1.3.1 (continued)

```

case (it (Pair 0 1)) of {Pair -> ita}
case (case (Pair 0 1) of {Pair -> ita}) of {Pair -> ita}
case (ita 0 1) of {Pair -> ita}
case (Pair 1 (add 0 1)) of {Pair -> ita}
case (Pair 1 1) of {Pair -> ita}
ita 1 1
Pair 1 (add 1 1)
Pair 1 2

```

On the other hand, if the interpretation corresponding to the third interpretation of the *Haskell* equations is used instead then the algorithm can never reach a ground term: If it did then this would imply that the term `fst (Pair 1 (fib -1))` had a value, which is here not the case. In fact the algorithm produces a non-terminating list of replacements.

If the algorithm terminates with a ground term then this implies that the original term has a value and it also computes what this value is. However, the question still remains concerning the completeness of the algorithm: Does the algorithm terminate with a ground term whenever it is applied to a term which has a value? Of course, if this were the case then in particular the completeness requirement formulated in Section 1.1 would be satisfied.

In Chapter 7 the constructions introduced in Chapter 4 will be applied to present a framework within which the algorithm is complete for each system of equations. A somewhat imprecise description will now be given of how the framework is obtained. This involves first saying something about partially ordered sets and then something about complete partially ordered sets.

A *partially ordered set* (or *poset*) is a pair (Y, \sqsubseteq) consisting of a non-empty set Y and a *partial order* \sqsubseteq on Y , i.e., \sqsubseteq is a binary relation \sqsubseteq on Y satisfying:

- (i) $y \sqsubseteq y$ for all $y \in Y$.
- (ii) If $y_1 \sqsubseteq y_2$ and $y_2 \sqsubseteq y_1$ then $y_1 = y_2$.
- (iii) If $y_1 \sqsubseteq y_2$ and $y_2 \sqsubseteq y_3$ then $y_1 \sqsubseteq y_3$.

(In other words, the relation \sqsubseteq is reflexive, anti-symmetric and transitive.) It is usual, however, to just write Y instead of (Y, \sqsubseteq) and to assume \sqsubseteq can be determined from the context. If Y_1, \dots, Y_n are posets then there is a natural *product partial order* \sqsubseteq

on the cartesian product $Y_1 \times \cdots \times Y_n$ in which $(y_1, \dots, y_n) \sqsubseteq (y'_1, \dots, y'_n)$ if and only if $y_j \sqsubseteq_j y'_j$ for each $j = 1, \dots, n$ (where of course \sqsubseteq_j is the partial order on Y_j for each j). If Y and Y' are posets then a mapping $h : Y \rightarrow Y'$ is said to be *monotone* if $f(y_1) \sqsubseteq' f(y_2)$ whenever $y_1, y_2 \in Y$ with $y_1 \sqsubseteq y_2$.

Posets come into the set-up which has been considered here because (as will be shown in Section 4.1) if (D_B, f_K) is a regular extension of (X_B, p_K) then for each $\beta \in B$ there exists a partial order \sqsubseteq_β on the set D_β such that $z \sqsubseteq_\beta z'$ can be sensibly interpreted as meaning that z is 'less-defined' than z' .

In the third and fourth interpretations of the *Haskell* equations the partial orders \sqsubseteq_{int} and $\sqsubseteq_{\text{bool}}$ are both 'flat', which means that if $\beta \in \{\text{int}, \text{bool}\}$ and $z, z' \in D_\beta$ with $z \neq z'$ then $z \sqsubseteq_\beta z'$ if and only if $z = \perp_\beta$. In the third interpretation \sqsubseteq_{ipr} is also 'flat'; in the fourth interpretation (where $D_{\text{ipr}} = \mathbb{Z}^\perp \times \mathbb{Z}^\perp \cup \{\perp_{\text{ipr}}\}$) the partial order \sqsubseteq_{ipr} is given as follows: $\perp_{\text{ipr}} \sqsubseteq_{\text{ipr}} z$ for all $z \in D_{\text{ipr}}$ and if m, m', n, n' are elements of $\mathbb{Z}^\perp = D_{\text{int}}$ then $(m, n) \sqsubseteq_{\text{ipr}} (m', n')$ if and only if $m \sqsubseteq_{\text{int}} m'$ and $n \sqsubseteq_{\text{int}} n'$.

Note that in both these cases the mapping $f_{\text{pair}} : D_{\text{int}} \times D_{\text{int}} \rightarrow D_{\text{ipr}}$ is then monotone, i.e., $f_{\text{pair}}(z_1, z_2) \sqsubseteq_{\text{ipr}} f_{\text{pair}}(z'_1, z'_2)$ whenever $z_1 \sqsubseteq_{\text{int}} z'_1$ and $z_2 \sqsubseteq_{\text{int}} z'_2$. This fact will play an important role, but for a general regular extension it does not always hold. An additional assumption is thus required on (D_B, f_K) , that of being monotone. (Monotone extensions are introduced in Section 3.4.) In general a minimality condition on (D_B, f_K) will also be imposed in order to ensure that the family of partial orders \sqsubseteq_B is unique. (This condition is already satisfied in the third and fourth interpretations of the *Haskell* equations.)

Given the family of partial orders \sqsubseteq_B , a corresponding choice for the family D_S can then be made: For each functional type $\sigma = \beta_1 \cdots \beta_n \rightarrow \beta$ the set D_σ of functions of type σ can be taken to be the set of all monotone functions from $D_{\beta_1} \times \cdots \times D_{\beta_n}$ (with the product partial order) to D_β .

If the above choice for D_S is made then it turns out that the algorithm is complete for any system of equations involving only the data types `int`, `bool` and `ipr`. However, this is only true because these data types have a rather trivial structure. In general a further construction has to be made and, as will now be explained, it is here that complete posets enter the picture.

A poset Y is *complete* if the least upper bound of A exists for each directed subset A of Y . A subset A of Y is said to be *directed* if it is non-empty and for each $y_1, y_2 \in A$ there exists $y \in A$ such that $y_1 \sqsubseteq y$ and $y_2 \sqsubseteq y$. Moreover, if A is a non-empty subset of Y then an element $y \in Y$ is called an *upper bound* of A if $y' \sqsubseteq y$ for all $y' \in A$; y is called the *least upper bound* of A if y is an upper bound of A and $y \sqsubseteq y'$ for each upper bound y' of A . (It is clear that there can be at most one element $y \in Y$ having these properties.) If the least upper bound of A exists then it will be denoted by $\bigsqcup A$. It is not difficult to see that if Y_1, \dots, Y_n are complete posets then so is $Y_1 \times \cdots \times Y_n$ with the product order. If Y_1 and Y_2 are complete posets then a mapping $h : Y_1 \rightarrow Y_2$ is said to be *continuous* if h is monotone and $h(\bigsqcup A) = \bigsqcup h(A)$ for each directed subset A of Y_1 .

The posets D_{int} , D_{bool} and D_{ipr} coming from the third and fourth interpretations of the *Haskell* equations are all complete. This follows from the fact that each directed subset is finite and thus contains a maximum element which is then the least upper bound. For the same reason the mapping $f_{\text{pair}} : D_{\text{int}} \times D_{\text{int}} \rightarrow D_{\text{ipr}}$ is continuous. Moreover, if $\sigma = \beta_1 \cdots \beta_n \rightarrow \beta$ is a functional type then each function in D_σ (i.e., each monotone function from $D_{\beta_1} \times \cdots \times D_{\beta_n}$ to D_β) is also continuous.

However, if data types more complex than `int`, `bool` and `ipr` are involved then the posets D_β , $\beta \in B$, are typically not complete. Thus in general it is necessary to extend D_β to a complete poset \check{D}_β , and in fact \check{D}_β will be taken to be the so-called initial (or ideal) completion of D_β . (Such completions will be dealt with in Section 4.3.) For each $\kappa \in K$ there is then a unique continuous extension \check{f}_κ of the mapping f_κ , and in this way a new regular monotone extension $(\check{D}_B, \check{f}_K)$ of (X_B, p_K) is obtained. Finally, for each functional type $\sigma = \beta_1 \cdots \beta_n \rightarrow \beta$ the set of functions \check{D}_σ of type σ is now taken to be the set of all continuous functions from $\check{D}_{\beta_1} \times \cdots \times \check{D}_{\beta_n}$ to \check{D}_β .

If $(\check{D}_B, \check{f}_K)$ is used as the extension of (X_B, p_K) and \check{D}_S as the supply of functions then it turns out that each system of equations always has a solution and that the algorithm is complete. (These facts will be proved in Chapter 7.) In particular, this means that there is a framework within which the completeness requirement formulated in Section 1.1 is satisfied.

The problem with this approach is that it not easy to explain to a typical student in a first year Computer Science course. However, the algorithm can still be used to compute values without directly referring to the framework introduced above. Suppose some extension (D_B, f_K) of (X_B, p_K) and some supply of functions D_S are provided (though not necessarily the extension $(\check{D}_B, \check{f}_K)$ and the supply of functions \check{D}_S). The question is then whether the algorithm will compute the value of a given term. But for this to make sense the term must actually have a value, i.e., the value of the expression it denotes must be completely defined and be independent of which solution of the equations is being used. Now in all reasonable cases a proof that the term has a particular value will not depend on the extension and the supply of functions. This means it will show that the term has a value in the special framework described above, and so the algorithm will compute this value, and hence also compute it for the set-up specified by (D_B, f_K) and D_S .

With this modified approach the student does not have to know anything about complete posets and the like. As replacement, though, he or she must learn what it means to prove that a term has a value. This topic is dealt with in Section 7.4.

1.4 Notes

The reader with no previous experience of functional programming is recommended to look at Bird and Wadler (1988) before proceeding any further. Other books offering good introductions to functional programming from various viewpoints are Field and Harrison (1988), Glaser, Henkin and Till (1984), Henderson (1980), Holyer (1993), MacLennan (1990), Paulson (1991), Reade (1989) and Wikström (1987).

An important remark for the knowledgeable: The exposition in the Introduction has been over-simplified in several places. In particular, contrary to the impression given there, higher-order types will be allowed.

The approach described in this chapter has similarities with that taken in Rosen (1973). What was called the completeness requirement corresponds to Rosen's *Validity Theorem* (Theorem 8.4), a result which was conjectured in Morris (1968). Moreover, Rosen points out the connections with Kleene's first recursion theorem (Kleene (1952), §66), which can be seen as the fore-runner of the material to be presented in Chapter 7.

Chapter 2 Some universal algebra

In this chapter those basic facts from universal algebra are presented which will be needed in the subsequent chapters. The emphasis here is on initial algebras in their various forms.

In order to give a foretaste of the kind of universal algebra which will occur, it is perhaps instructive to start with a very familiar situation and serve it up in a somewhat unusual guise. (What follows is, however, certainly not new: It appears in essentially this form in Dedekind's book *Was sind und was sollen die Zahlen?* first published in 1888.)

Consider the set $\mathbb{N} = \{0, 1, 2, \dots\}$ of natural numbers together with its first element 0 and the successor function $s : \mathbb{N} \rightarrow \mathbb{N}$ (defined by $s(n) = n + 1$ for each $n \in \mathbb{N}$). It can then be asked what is special about the triple $(\mathbb{N}, 0, s)$. To make things a bit more precise, define a triple (X, e, p) consisting of a non-empty set X , an element $e \in X$ and a mapping $p : X \rightarrow X$ to be a *natural number triple*, and then ask how $(\mathbb{N}, 0, s)$ can be characterised in the class of all such triples.

To answer this question the appropriate structure-preserving mappings must first be introduced: If (X, e, p) and (X', e', p') are natural number triples then a mapping $\pi : X \rightarrow X'$ is called a *homomorphism* from (X, e, p) to (X', e', p') if $\pi(e) = e'$ and $p' \pi = \pi p$ (i.e., $p'(\pi(x)) = \pi(p(x))$ for all $x \in X$). If π is such a homomorphism and the mapping π is a bijection then it is easy to see that the inverse mapping $\pi^{-1} : X' \rightarrow X$ is a homomorphism from (X', e', p') to (X, e, p) . In this case π is called an *isomorphism*, and (X, e, p) and (X', e', p') are said to be *isomorphic*.

The identity mapping is a homomorphism and the composition of two homomorphisms is again a homomorphism; being isomorphic thus defines an equivalence relation on the class of all natural number triples. Now if the equivalence class containing $(\mathbb{N}, 0, s)$ can be identified in some reasonable way then the question asked above can be considered to have been answered satisfactorily. Two simple characterisations of this equivalence class are given below.

A natural number triple (X, e, p) is said to be *initial* if for each such triple (X', e', p') there is a unique homomorphism from (X, e, p) to (X', e', p') . Then $(\mathbb{N}, 0, s)$ is initial, since given (X', e', p') , a homomorphism from $(\mathbb{N}, 0, s)$ to (X', e', p') can be defined by induction, and its uniqueness also follows by induction. It follows that a triple is isomorphic to $(\mathbb{N}, 0, s)$ if and only if it is initial. In other words, the equivalence class containing $(\mathbb{N}, 0, s)$ consists of exactly all the initial natural number triples. This was the first characterisation.

Here is the second characterisation: A natural number triple (X, e, p) will be called a *Peano triple* if the following three conditions are satisfied:

- (i) The mapping p is injective.
- (ii) $p(x) \neq e$ for all $x \in X$.
- (iii) The only subset X' of X containing e with $p(x) \in X'$ for all $x \in X'$ is the set X itself.

Then $(\mathbb{N}, 0, s)$ is a Peano triple. (This can be seen as one of the possible formulations of the Peano axioms. In particular, the statement that $(\mathbb{N}, 0, s)$ satisfies (iii) is nothing but the principle of mathematical induction.) The reader is left to show that a triple is isomorphic to $(\mathbb{N}, 0, s)$ if and only if it is itself a Peano triple. In other words, the equivalence class containing $(\mathbb{N}, 0, s)$ also consists of exactly all the Peano triples.

Of course, a corollary of these two characterisations is that a natural number triple is initial if and only if it is a Peano triple. This is a special case of an important result presented in Section 2.3 which states that an algebra is initial if and only if (in the terminology employed there) it is unambiguous and minimal. For a general algebra unambiguity corresponds to conditions (i) and (ii) in the definition of a Peano triple, while minimality corresponds to condition (iii). This characterisation is sometimes expressed by saying that initial algebras are exactly those for which there is *no confusion* (unambiguity) and *no junk* (minimality).

Section 2.1, as well as fixing some general notation, introduces the simple notion of a typed set. This takes into account the situation met with in most modern programming languages in which each name occurring in a program is assigned, either explicitly or implicitly, a type. The kind of object a name can refer to is then determined by its type.

In Section 2.2 signatures, algebras and homomorphisms are introduced, and invariant families (i.e., subalgebras) are also looked at. Section 2.3 then deals with initial and free algebras, and in Section 2.4 term algebras are introduced in order to give simple explicit examples of initial algebras. Finally, in Section 2.5 something is said about extensions of signatures and algebras, and Section 2.6 presents a somewhat idiosyncratic approach to tree algebras.

Except for Section 2.6 (which can be omitted) the choice of material in this chapter is determined entirely by what will be needed later. The reader interested in a more balanced account of modern universal algebra should consult the books referenced at the end of the chapter.

2.1 Typed sets

Before beginning in the next section with universal algebra proper the simple notion of a typed set will be introduced. This takes into account the situation met with in most modern programming languages in which each name occurring in a program is assigned, either explicitly or implicitly, a type. The kind of object a name can refer to is then determined by its type.

The first task, however, is to fix some notation. The empty set will be denoted by \emptyset and the set $\{\emptyset\}$ by \mathbb{I} ; thus \mathbb{I} is the ‘standard’ set containing exactly one element. However, to avoid confusion the single element in \mathbb{I} will be denoted by ε (rather than by \emptyset).

The words *function* and *mapping* are considered to be synonyms. Nevertheless, in this chapter almost exclusive use of the word *mapping* will be made. (*Functions* are what the equations are supposed to define.) If X and Y are sets then Y^X will be used to denote the set of all mappings from X to Y . In particular, $Y^\emptyset = \mathbb{I}$ for each set Y .

Let $f : X \rightarrow Y$ be a mapping; then X is called the *domain* of f and will be denoted by $\text{Dom}(f)$ and Y is called the *codomain*. For each $A \subset X$ put

$$f(A) = \{y \in Y : y = f(x) \text{ for some } x \in A\}$$

and for each $B \subset Y$ put $f^{-1}(B) = \{x \in X : f(x) \in B\}$. The subset $f(X)$ of the codomain Y is called the *image* of f and will be denoted by $\text{Im}(f)$.

If X and Y are sets then a *partial mapping* from X to Y is a mapping $f : X' \rightarrow Y$ with X' a subset of X , and as above X' is called the *domain* of f and is denoted by $\text{Dom}(f)$. As already stated in the Introduction, an ‘ordinary’ mapping $g : X \rightarrow Y$ is also a partial mapping with $\text{Dom}(g) = X$, and the adjective *total* will be used when it is deemed necessary to emphasise the fact that such a mapping is involved. However, the word *mapping* used without qualification always refers to a total mapping.

Let S be a set and suppose for each $\sigma \in S$ some object α_σ is given (which will usually be a set or a mapping). Then by the *family* $\{\alpha_\sigma : \sigma \in S\}$ is meant the mapping from S to the set consisting of the objects α_σ , $\sigma \in S$, which assigns the object α_σ to each $\sigma \in S$. Unless something to the contrary is explicitly stated this family will be denoted by α_S . If X_S and Y_S are families of sets then $Y_S \subset X_S$ will mean that $Y_\sigma \subset X_\sigma$ for each $\sigma \in S$. This will also be indicated by saying that X_S *contains* Y_S or that Y_S *is contained in* X_S .

The topic announced in the title of this section will now be considered. A pair $(I, \langle \cdot \rangle)$ consisting of a set I and a mapping $\langle \cdot \rangle : I \rightarrow S$ is called an *S-typed set* with *typing* $\langle \cdot \rangle$. The set S should here be thought of as a set of ‘types’, I as a set of ‘names’ and $\langle \cdot \rangle$ as specifying the type of objects to which the ‘names’ can be assigned. If $\eta \in I$ with $\langle \eta \rangle = \sigma$ then η is said to be *of type* σ .

The *S*-typed set $(I, \langle \cdot \rangle)$ will usually just be denoted by I , it being assumed that the typing $\langle \cdot \rangle$ can be inferred from the context. In fact, except where there is some danger of confusion, $\langle \cdot \rangle$ will always be used to denote the typing on a typed set. The empty set \emptyset will be regarded as an *S*-typed set (with the corresponding unique typing).

Note that an ordinary set can (and will) be considered as an \mathbb{I} -typed set, since for any set I there is a unique typing $\langle \cdot \rangle : I \rightarrow \mathbb{I}$.

If $\alpha_S = \{ \alpha_\sigma : \sigma \in S \}$ is a family of objects and I is an S -typed set then, in order to increase the legibility, it is convenient to omit the brackets and just use α_η to denote the object $\alpha_{\langle \eta \rangle}$ for each $\eta \in I$.

For each family of sets $X_S = \{ X_\sigma : \sigma \in S \}$ put $|X_S| = \bigcup_{\sigma \in S} X_\sigma$. If I is an S -typed set then X_\diamond^I will denote the set of all *typed* mappings from I to $|X_S|$, i.e., the set of mappings $c : I \rightarrow |X_S|$ such that $c(\eta) \in X_\eta$ for each $\eta \in I$. The elements of X_\diamond^I are called *assignments*. An assignment $c \in X_\diamond^I$ thus assigns to each ‘name’ $\eta \in I$ an element $c(\eta) \in X_\eta$ of the appropriate type. Note that $X_\diamond^\emptyset = \mathbb{I}$; moreover, if $Y_S \subset X_S$ then Y_\diamond^I can clearly be regarded as a subset of X_\diamond^I .

The symbol \diamond in the expression X_\diamond^I is a reminder that the X occurring here only has a meaning within the context of the family X_S . It also helps to distinguish X_\diamond^I from the notation X^I used for the set of all mappings from the set I to the set X . (Note that if J and Y are sets, and J is considered as an \mathbb{I} -typed set then in fact $Y^J = Y_\diamond^J$, provided Y is identified with the family $Y_\mathbb{I}$ given by $Y_\epsilon = Y$.)

When dealing with explicit examples, in which case the set I is always finite, an assignment $c \in X_\diamond^I$ will be specified by choosing an enumeration η_1, \dots, η_m of the elements of I and then writing it as $\{ \eta_1 \rightarrow c(\eta_1), \eta_2 \rightarrow c(\eta_2), \dots, \eta_m \rightarrow c(\eta_m) \}$.

Recall that \mathbb{N} is the set $\{0, 1, 2, \dots\}$, i.e., 0 is a natural number. For each $n \in \mathbb{N}$ let $[n] = \{1, 2, \dots, n\}$; in particular $[0] = \emptyset$. Let $n \in \mathbb{N}$ and for each $j = 1, \dots, n$ let X_j be a set; then the *cartesian product* $X_1 \times \dots \times X_n$ is defined to be the set of all mappings ϱ from $[n]$ to $\bigcup_{j=1}^n X_j$ such that $\varrho(j) \in X_j$ for each j . In particular, if $n = 0$ then $X_1 \times \dots \times X_n = \mathbb{I}$. For each j let $x_j \in X_j$; then as usual (x_1, \dots, x_n) denotes the element ϱ of $X_1 \times \dots \times X_n$ such that $\varrho(j) = x_j$ for each j ; clearly each element of $X_1 \times \dots \times X_n$ has a unique representation of this form.

Of course, the cartesian product is just a special case of a set of assignments: Here the family of sets is $X_{[n]} = \{ X_j : j \in [n] \}$ and then $X_1 \times \dots \times X_n = X_\diamond^{[n]}$, where $[n]$ is considered as an $[n]$ -typed set with the identity typing.

The simplest case of a cartesian product is when the sets X_1, \dots, X_n are all the same: Let X be a set; then for each $n \in \mathbb{N}$ the n -fold cartesian product of X with itself will be denoted by X^n , thus X^n is the set of all mappings from $[n]$ to X . In particular, $X^0 = \mathbb{I}$; moreover, it is convenient to identify X^1 with X in the obvious way. Note that each element of X^n can be regarded as a finite X -typed set: More precisely, $(x_1, \dots, x_n) \in X^n$ is the typed set consisting of the set $[n]$ with the typing which assigns to each j the type x_j .

For each set X the set $\bigcup_{n \geq 0} X^n$ will be denoted by X^* , which should be thought of as the set of all finite lists of elements from X . Note that since X^1 is being identified with X , it follows that X is a subset of X^* . (In other words, each element x of X is identified with the list whose single component is equal to x .) If $m \geq 1$ then the element (x_1, \dots, x_m) of $X^m \subset X^*$ will always be denoted simply by $x_1 \cdots x_m$. Following the remark in the previous paragraph each list of elements from X is a finite

X -typed set: The list $x_1 \cdots x_n$ is the X -typed set consisting of the set $[n]$ with the typing which assigns to each j the type x_j .

Let X_S be a family of sets and $\gamma = \sigma_1 \cdots \sigma_n \in S^*$; then, considering γ as an S -typed set, $X_\diamond^\gamma = Y_1 \times \cdots \times Y_n$ with $Y_j = X_{\sigma_j}$ for each $j = 1, \dots, n$, which leads to the usual notation $X_{\sigma_1} \times \cdots \times X_{\sigma_n}$ for the set X_\diamond^γ .

In order to avoid any foundational difficulties it is convenient to fix once and for all some large set \mathbb{M} of names, and then only to consider S -typed sets whose underlying sets are subsets of \mathbb{M} . This means that there is the set (and not just the class) of all S -typed sets. It is assumed that $\mathbb{N} \subset \mathbb{M}$ and (in order to give explicit examples) that $\mathbb{A}^* \setminus \{\varepsilon\} \subset \mathbb{M}$ for some suitable alphabet \mathbb{A} containing the usual *ASCII*-symbols.

The set of all finite S -typed sets will be denoted by \mathcal{F}_S . Since $\mathbb{N} \subset \mathbb{M}$ it follows in particular that $S^n \subset \mathcal{F}_S$ for each $n \in \mathbb{N}$ and also $S^* \subset \mathcal{F}_S$. (Beware though that the list ε with no components is then the S -typed set \emptyset .)

Now let X_S and Y_S be families of sets and $\varphi_S : X_S \rightarrow Y_S$ be a family of mappings, i.e., $\varphi_\sigma : X_\sigma \rightarrow Y_\sigma$ for each $\sigma \in S$. Then for each S -typed set I there is a mapping $\varphi_\diamond^I : X_\diamond^I \rightarrow Y_\diamond^I$ defined for each $c \in X_\diamond^I$, $\eta \in I$ by

$$\varphi_\diamond^I(c)(\eta) = \varphi_\eta(c(\eta))$$

(recalling that $\varphi_\eta = \varphi_{\langle \eta \rangle}$). Note that if $\gamma = \sigma_1 \cdots \sigma_n \in S^*$ then φ_\diamond^γ is just the mapping from $X_\diamond^\gamma = X_{\sigma_1} \times \cdots \times X_{\sigma_n}$ to $Y_\diamond^\gamma = Y_{\sigma_1} \times \cdots \times Y_{\sigma_n}$ defined by

$$\varphi_\diamond^\gamma(x_1, \dots, x_n) = (\varphi_{\sigma_1}(x_1), \dots, \varphi_{\sigma_n}(x_n))$$

for each $(x_1, \dots, x_n) \in X_\diamond^\gamma$.

Lemma 2.1.1 (1) Let X_S be a family of sets and $\text{id}_S : X_S \rightarrow X_S$ be the family of identity mappings (i.e., with $\text{id}_\sigma : X_\sigma \rightarrow X_\sigma$ the identity mapping for each $\sigma \in S$). Then $\text{id}_\diamond^I : X_\diamond^I \rightarrow X_\diamond^I$ is also the identity mapping for each S -typed set I .

(2) Let X_S , Y_S and Z_S be families of sets and let $\varphi_S : X_S \rightarrow Y_S$ and $\psi_S : Y_S \rightarrow Z_S$ be families of mappings. Then for each S -typed set I

$$(\psi\varphi)_\diamond^I = \psi_\diamond^I \varphi_\diamond^I$$

where $(\psi\varphi)_S : X_S \rightarrow Z_S$ is the family given by $(\psi\varphi)_\sigma = \psi_\sigma \varphi_\sigma$ for each $\sigma \in S$.

(3) Let X_S and Y_S be families of sets and $\varphi_S : X_S \rightarrow Y_S$ be a family of mappings. If the mapping $\varphi_\sigma : X_\sigma \rightarrow Y_\sigma$ is injective (resp. surjective) for each $\sigma \in S$ then for each S -typed set I the mapping $\varphi_\diamond^I : X_\diamond^I \rightarrow Y_\diamond^I$ is also injective (resp. surjective). In particular, if $\varphi_\sigma : X_\sigma \rightarrow Y_\sigma$ is a bijection for each $\sigma \in S$ then the mapping $\varphi_\diamond^I : X_\diamond^I \rightarrow Y_\diamond^I$ is a bijection, and in this case

$$(\varphi_\diamond^I)^{-1} = (\varphi_\diamond^{-1})^I.$$

Proof (1) This is clear.

(2) Let $c \in X_\circ^I$ and $\eta \in I$; then

$$\begin{aligned} (\psi\varphi)_\circ^I(c)(\eta) &= (\psi_\eta\varphi_\eta)(c(\eta)) = \psi_\eta(\varphi_\eta(c(\eta))) \\ &= \psi_\eta(\varphi_\circ^I(c))(\eta) = \psi_\circ^I(\varphi_\circ^I(c))(\eta) = (\psi_\circ^I\varphi_\circ^I)(c)(\eta) \end{aligned}$$

and thus $(\psi\varphi)_\circ^I = \psi_\circ^I\varphi_\circ^I$.

(3) If φ_σ is injective for each σ then there exists a family of mappings $\psi_S : Y_S \rightarrow X_S$ such that $\psi_\sigma\varphi_\sigma$ is the identity mapping on X_σ for each $\sigma \in S$. Then by (1) $(\psi\varphi)_\circ^I$ is the identity mapping on X_\circ^I and by (2) $\psi_\circ^I\varphi_\circ^I = (\psi\varphi)_\circ^I$. Hence φ_\circ^I is injective. The other case is almost identical: If φ_σ is surjective for each σ then there exists a family of mappings $\psi_S : Y_S \rightarrow X_S$ such that $\varphi_\sigma\psi_\sigma$ is the identity mapping on Y_σ for each $\sigma \in S$, and as in the first part it then follows that $\varphi_\circ^I\psi_\circ^I$ is the identity mapping on Y_\circ^I . Finally, if φ_σ is a bijection for each $\sigma \in S$ then $(\varphi_\sigma^{-1})^I\varphi_\sigma^I$ is the identity mapping on X_σ^I and $\varphi_\circ^I(\varphi_\circ^{-1})^I$ is the identity mapping on Y_\circ^I , and thus $(\varphi_\circ^I)^{-1} = (\varphi_\circ^{-1})^I$. \square

Some of the mappings which will occur involve the braces \llbracket and \rrbracket , usually ornamented with subscripts and often also superscripts. In this case the value of the mapping is denoted by placing the argument between the braces. Consider, for instance, a family of mappings $\llbracket \cdot \rrbracket_S$, with $\llbracket \cdot \rrbracket_\sigma$ a mapping from X_σ to Y_σ for each $\sigma \in S$; then the element of Y_σ which is obtained by applying $\llbracket \cdot \rrbracket_\sigma$ to $x \in X_\sigma$ is denoted here by $\llbracket x \rrbracket_\sigma$. Now if I is an S -typed set then the corresponding mapping from X_\circ^I to Y_\circ^I will be denoted as above by $\llbracket \cdot \rrbracket_\circ^I$, and its argument will again be placed between the braces. Thus if $c \in X_\circ^I$ then $\llbracket c \rrbracket_\circ^I$ is the element of Y_\circ^I given by $\llbracket c \rrbracket_\circ^I(\eta) = \llbracket c(\eta) \rrbracket_\eta$ for each $\eta \in I$.

A *partial order* \sqsubseteq on a set Y is a binary relation \sqsubseteq on Y satisfying:

- (i) $y \sqsubseteq y$ for all $y \in Y$.
- (ii) If $y_1 \sqsubseteq y_2$ and $y_2 \sqsubseteq y_1$ then $y_1 = y_2$.
- (iii) If $y_1 \sqsubseteq y_2$ and $y_2 \sqsubseteq y_3$ then $y_1 \sqsubseteq y_3$.

In other words, the relation \sqsubseteq is reflexive, anti-symmetric and transitive.

Let X_S be a family of sets and let \sqsubseteq_S be a corresponding family of partial orders, i.e., \sqsubseteq_σ is a partial order on X_σ for each $\sigma \in S$. Then for each S -typed set I there is a natural partial order \sqsubseteq_\circ^I on the set X_\circ^I defined by stipulating that $c \sqsubseteq_\circ^I c'$ if and only if $c(\eta) \sqsubseteq_\eta c'(\eta)$ for each $\eta \in I$.

Let $n \geq 2$ and for each $j = 1, \dots, n$ let (Y_j, \sqsubseteq_j) be a poset. Then the above construction produces the usual *product partial order* \sqsubseteq on $Y_1 \times \dots \times Y_n$ in which $(y_1, \dots, y_n) \sqsubseteq (y'_1, \dots, y'_n)$ if and only if $y_j \sqsubseteq_j y'_j$ for each j .

The final topic in this section is a simple observation which will be applied throughout the study. Suppose that a number of objects from a given family of sets X_S are to be 'defined'. Then this can be organised by first choosing an appropriate S -typed

I consisting of the ‘names’ of these objects, and then ‘constructing’ an assignment $c \in X_{\diamond}^I$ which gives a meaning to each ‘name’ (and thus ‘defines’ the objects $c(\eta)$, $\eta \in I$). In particular, this device will be employed to formulate the equations which appear in Chapter 6 so that their solutions are assignments rather than sets of functions. A reformulation in this spirit of the equations considered in Chapter 1 is given in Example 2.1.1 below.

Example 2.1.1 Let (D_B, f_K) and the family D_S be as in Section 1.2 (for the case of the equations for it , ita , fst , fst_a , pfb and fib). Let

$$I = \{\text{it}, \text{ita}, \text{fst}, \text{fst}_a, \text{pfb}, \text{fib}\}$$

be the S -typed set with

$$\begin{aligned} \text{it} & \text{ of type } \text{ipr} \rightarrow \text{ipr}, \quad \text{ita} \text{ of type } \text{int int} \rightarrow \text{ipr}, \\ \text{fst} & \text{ of type } \text{ipr} \rightarrow \text{int}, \quad \text{fst}_a \text{ of type } \text{int int} \rightarrow \text{int}, \\ \text{pfb} & \text{ of type } \text{int ipr} \rightarrow \text{ipr} \text{ and } \text{fib} \text{ of type } \text{int} \rightarrow \text{int}. \end{aligned}$$

If $c \in D_{\diamond}^I$ and $\eta \in I$ then, since it increases the legibility, $c(\eta)$ will usually be written as c_{η} ; this means that $c_{\text{it}} \in D_{\text{ipr} \rightarrow \text{ipr}}$, $c_{\text{ita}} \in D_{\text{int int} \rightarrow \text{ipr}}$, $c_{\text{fst}} \in D_{\text{ipr} \rightarrow \text{int}}$, $c_{\text{fst}_a} \in D_{\text{int int} \rightarrow \text{int}}$, $c_{\text{pfb}} \in D_{\text{int ipr} \rightarrow \text{ipr}}$ and that $c_{\text{fib}} \in D_{\text{int} \rightarrow \text{int}}$.

The equations for it , ita , fst , fst_a , pfb and fib can now be considered as the following equations for an assignment $c \in D_{\diamond}^I$:

$$c_{\text{it}}(p) = \begin{cases} c_{\text{ita}}(k, \ell) & \text{if } p \neq \perp_{\text{ipr}} \text{ and } p = f_{\text{Pair}}(k, \ell) \\ & \text{with } (k, \ell) \in D_{\text{int}} \times D_{\text{int}}, \\ \perp_{\text{ipr}} & \text{if } p = \perp_{\text{ipr}}, \end{cases}$$

$$c_{\text{ita}}(m, n) = f_{\text{Pair}}(n, \text{add}(m, n)),$$

$$c_{\text{fst}}(p) = \begin{cases} c_{\text{fst}_a}(k, \ell) & \text{if } p \neq \perp_{\text{ipr}} \text{ and } p = f_{\text{Pair}}(k, \ell) \\ & \text{with } (k, \ell) \in D_{\text{int}} \times D_{\text{int}}, \\ \perp_{\text{ipr}} & \text{if } p = \perp_{\text{ipr}}, \end{cases}$$

$$c_{\text{fst}_a}(m, n) = m,$$

$$c_{\text{pfb}}(n, p) = \begin{cases} p & \text{if } \text{eq}(n, 0) = T, \\ c_{\text{pfb}}(\text{sub}(n, 1), c_{\text{it}}(p)) & \text{if } \text{eq}(n, 0) = F, \\ \perp_{\text{ipr}} & \text{if } \text{eq}(n, 0) = \perp_{\text{bool}}, \end{cases}$$

$$c_{\text{fib}}(n) = c_{\text{fst}}(c_{\text{pfb}}(n, f_{\text{Pair}}(0, 1)))$$

for all $m, n \in D_{\text{int}}$, $p \in D_{\text{ipr}}$.

2.2 Algebras and homomorphisms

The structure which plays a fundamental role in all of what follows is that of an algebra associated with a signature. A *signature* is a quadruple $\Sigma = (S, N, \Delta, \delta)$, where S and N are non-empty sets and $\Delta : N \rightarrow \mathcal{F}_S$ and $\delta : N \rightarrow S$ are mappings (recalling that \mathcal{F}_S is the set of all finite S -typed sets). The set S should here be thought of as a set of *types*; N can be regarded as a set of *operator names*.

For each $\sigma \in S$ put $N_\sigma = \{\nu \in N : \delta(\nu) = \sigma\}$. Moreover, if $\nu \in N$ with $\delta(\nu) = \sigma$ and $\Delta(\nu) = L$ then ν is said to be *of type* $L \rightarrow \sigma$. This terminology will be used to help avoid mentioning Δ and δ explicitly.

If $\Sigma = (S, N, \Delta, \delta)$ is a signature then a Σ -*algebra* is any pair (X_S, p_N) consisting of a family of sets $X_S = \{X_\sigma : \sigma \in S\}$ and a family of mappings $p_N = \{p_\nu : \nu \in N\}$ such that if $\nu \in N$ is of type $L \rightarrow \sigma$ then p_ν is a mapping from X_σ^L to X_σ . For each $\sigma \in S$ the set X_σ should be thought of as a set of elements of type σ and for each $\nu \in N$ the mapping p_ν can be thought of as the operator corresponding to the operator name ν .

A signature $\Sigma = (S, N, \Delta, \delta)$ is said to be *enumerated* if $\Delta(\nu) \in S^*$ for each $\nu \in N$ (recalling that $S^* \subset \mathcal{F}_S$). If Σ is an enumerated signature then (X_S, p_N) being a Σ -algebra means that if $\nu \in N$ is of type $\sigma_1 \cdots \sigma_n \rightarrow \sigma$ then p_ν is a mapping from $X_{\sigma_1} \times \cdots \times X_{\sigma_n}$ to X_σ .

A simple enumerated signature $\Lambda = (B, K, \Theta, \vartheta)$ and a ‘natural’ Λ -algebra (X_B, p_K) are given in Example 2.2.1 on the following page. The usual way of representing such a signature is then illustrated in Example 2.2.2.

A general signature can always be replaced by an ‘equivalent’ enumerated signature. (This really just amounts to fixing an enumeration of the elements in the set $\Delta(\nu)$ for each $\nu \in N$.) In fact, what is here called an enumerated signature corresponds to the usual notion of a signature. The reason for working with the more general definition introduced above is that in the long run it turns out to be more natural.

For the remainder of the section let Σ denote a fixed signature with $\Sigma = (S, N, \Delta, \delta)$. Note that the sets N_σ , $\sigma \in S$, form a partition of the set N . (For instance, in the signature Λ in Example 2.2.1 $K_{\text{bool}} = \{\text{True}, \text{False}\}$, $K_{\text{nat}} = \{\text{Zero}, \text{Succ}\}$, $K_{\text{int}} = \underline{\text{Int}}$, $K_{\text{ipr}} = \{\text{Pair}\}$ and $K_{\text{list}} = \{\text{Nil}, \text{Cons}\}$.)

The next task, of course, is to explain what are the structure-preserving mappings between algebras. Let (X_S, p_N) and (Y_S, q_N) be Σ -algebras and let $\pi_S : X_S \rightarrow Y_S$ be a family of mappings, i.e., $\pi_\sigma : X_\sigma \rightarrow Y_\sigma$ for each $\sigma \in S$. Then the family π_S is called a Σ -*homomorphism* from (X_S, p_N) to (Y_S, q_N) if

$$q_\nu \pi_\sigma^L = \pi_\sigma p_\nu$$

whenever $\nu \in N$ is of type $L \rightarrow \sigma$. This fact will also be expressed by saying that $\pi_S : (X_S, p_N) \rightarrow (Y_S, q_N)$ is a Σ -homomorphism.

Example 2.2.1 \underline{Int} always denotes the subset of $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, -\}^*$ containing for each integer n its standard representation \underline{n} as a string of characters. The mapping $n \mapsto \underline{n}$ thus maps \mathbb{Z} bijectively onto \underline{Int} .

Define an enumerated signature $\Lambda = (B, K, \Theta, \vartheta)$ by letting

$$B = \{\text{bool}, \text{nat}, \text{int}, \text{ipr}, \text{ilst}\}$$

(ipr resp. ilst is an abbreviation of *integer pair* resp. *integer list*),

$$K = \{\text{True}, \text{False}, \text{Zero}, \text{Succ}, \text{Pair}, \text{Nil}, \text{Cons}\} \cup \underline{Int},$$

and with $\Theta : K \rightarrow B^*$ and $\vartheta : K \rightarrow B$ given by

$$\begin{aligned} \Theta(\text{True}) &= \Theta(\text{False}) = \varepsilon, & \vartheta(\text{True}) &= \vartheta(\text{False}) = \text{bool}, \\ \Theta(\text{Zero}) &= \varepsilon, & \Theta(\text{Succ}) &= \text{nat}, & \vartheta(\text{Zero}) &= \vartheta(\text{Succ}) = \text{nat}, \\ \Theta(\text{Pair}) &= \text{int int}, & \vartheta(\text{Pair}) &= \text{ipr}, \\ \Theta(\text{Nil}) &= \varepsilon, & \Theta(\text{Cons}) &= \text{int ilst}, & \vartheta(\text{Nil}) &= \vartheta(\text{Cons}) = \text{ilst}, \\ \Theta(\underline{n}) &= \varepsilon \text{ and } \vartheta(\underline{n}) = \text{int} \text{ for each } n \in \mathbb{Z}. \end{aligned}$$

Thus True and False are of type $\varepsilon \rightarrow \text{bool}$, Zero is of type $\varepsilon \rightarrow \text{nat}$, Succ of type $\text{nat} \rightarrow \text{nat}$, Pair of type $\text{int int} \rightarrow \text{ipr}$, Nil of type $\varepsilon \rightarrow \text{ilst}$, Cons of type $\text{int ilst} \rightarrow \text{ilst}$ and \underline{n} is of type $\varepsilon \rightarrow \text{int}$ for each $n \in \mathbb{Z}$.

Now define a Λ -algebra (X_B, p_K) with a family of sets X_B and a family of mappings p_K by letting

$$\begin{aligned} X_{\text{bool}} &= \mathbb{B} = \{T, F\}, & X_{\text{nat}} &= \mathbb{N}, & X_{\text{int}} &= \mathbb{Z}, \\ X_{\text{ipr}} &= \mathbb{Z}^2, & X_{\text{ilst}} &= \mathbb{Z}^*, \\ p_{\text{True}} &: \mathbb{I} \rightarrow X_{\text{bool}} \text{ with } p_{\text{True}}(\varepsilon) = T, \\ p_{\text{False}} &: \mathbb{I} \rightarrow X_{\text{bool}} \text{ with } p_{\text{False}}(\varepsilon) = F, \\ p_{\text{Zero}} &: \mathbb{I} \rightarrow X_{\text{nat}} \text{ with } p_{\text{Zero}}(\varepsilon) = 0, \\ p_{\text{Succ}} &: X_{\text{nat}} \rightarrow X_{\text{nat}} \text{ with } p_{\text{Succ}}(n) = n + 1, \\ p_{\underline{n}} &: \mathbb{I} \rightarrow X_{\text{int}} \text{ with } p_{\underline{n}}(\varepsilon) = n \text{ for each } n \in \mathbb{Z}, \\ p_{\text{Pair}} &: X_{\text{int}} \times X_{\text{int}} \rightarrow X_{\text{ipr}} \text{ with } p_{\text{Pair}}(m, n) = (m, n), \\ p_{\text{Nil}} &: \mathbb{I} \rightarrow X_{\text{ilst}} \text{ with } p_{\text{Nil}}(\varepsilon) = \varepsilon, \\ p_{\text{Cons}} &: X_{\text{int}} \times X_{\text{ilst}} \rightarrow X_{\text{ilst}} \text{ with } p_{\text{Cons}}(m, s) = m \triangleleft s, \end{aligned}$$

where $m \triangleleft s$ is here the element of \mathbb{Z}^* obtained by adding m to the beginning of the list s , i.e.,

$$m \triangleleft s = \begin{cases} m m_1 \cdots m_n & \text{if } s = m_1 \cdots m_n \text{ with } n \geq 1, \\ m & \text{if } s = \varepsilon. \end{cases}$$

Example 2.2.2 An enumerated signature $\Sigma = (S, N, \Delta, \delta)$ with S and N finite can (and in most functional programming languages will) be represented in a form similar to the following, where $\sigma_1, \dots, \sigma_n$ is some enumeration of the elements in the set S , $\nu_{k1}, \dots, \nu_{km_k}$ an enumeration of the elements of N_{σ_k} for each k and where $\gamma_{kj} = \Delta(\nu_{kj})$:

$$\begin{aligned} \sigma_1 &::= \nu_{11} \gamma_{11} \mid \cdots \mid \nu_{1m_1} \gamma_{1m_1} \\ \sigma_2 &::= \nu_{21} \gamma_{21} \mid \cdots \mid \nu_{2m_2} \gamma_{2m_2} \\ &\vdots \\ \sigma_n &::= \nu_{n1} \gamma_{n1} \mid \cdots \mid \nu_{nm_n} \gamma_{nm_n} \end{aligned}$$

The enumerated signature $\Lambda = (B, K, \Theta, \vartheta)$ introduced in Example 2.2.1 can thus be represented in the form

```
bool ::= True | False
nat  ::= Zero | Succ nat
int  ::= ... -2 | -1 | 0 | 1 | 2 ...
ipr  ::= Pair int int
ilst ::= Nil | Cons int ilst
```

Of course, there is a problem here with the type `int`, since K_{int} is infinite, but in all real programming languages this type is ‘built-in’ and so it does not need to be included in the specification of the signature.

If Σ is enumerated then $\pi_S : (X_S, p_N) \rightarrow (Y_S, q_N)$ being a Σ -homomorphism means that if $\nu \in N$ is of type $\sigma_1 \cdots \sigma_n \rightarrow \sigma$ then

$$q_\nu(\pi_{\sigma_1}(x_1), \dots, \pi_{\sigma_n}(x_n)) = \pi_\sigma(p_\nu(x_1, \dots, x_n))$$

must hold for all $(x_1, \dots, x_n) \in X_{\sigma_1} \times \cdots \times X_{\sigma_n}$, this condition being interpreted as $q_\nu(\varepsilon) = \pi_\sigma(p_\nu(\varepsilon))$ when ν is of type $\varepsilon \rightarrow \sigma$.

If (X_S, p_N) is a Σ -algebra then by Lemma 2.1.1 (1) the family of identity mappings $\text{id}_S : X_S \rightarrow X_S$ defines a Σ -homomorphism from (X_S, p_N) to itself. Furthermore, the composition of two Σ -homomorphisms is again a Σ -homomorphism:

Proposition 2.2.1 *Suppose (X_S, p_N) , (Y_S, q_N) and (Z_S, r_N) are Σ -algebras and let $\pi_S : (X_S, p_N) \rightarrow (Y_S, q_N)$ and $\varrho_S : (Y_S, q_N) \rightarrow (Z_S, r_N)$ be Σ -homomorphisms. Then $(\varrho\pi)_S = \{\varrho_\sigma \pi_\sigma : \sigma \in S\}$ is a Σ -homomorphism from (X_S, p_N) to (Z_S, r_N) .*

Proof Let $\nu \in N$ be of type $L \rightarrow \sigma$; then by Lemma 2.1.1 (2)

$$r_\nu (\varrho\pi)_\circ^L = r_\nu \varrho_\circ^L \pi_\circ^L = \varrho_\sigma q_\nu \pi_\circ^L = \varrho_\sigma \pi_\sigma p_\nu = (\varrho\pi)_\sigma p_\nu$$

and hence $(\varrho\pi)_S$ is a Σ -homomorphism from (X_S, p_N) to (Z_S, r_N) . \square

Proposition 2.2.2 *Let $\pi_S : (X_S, p_N) \rightarrow (Y_S, q_N)$ be a Σ -homomorphism. Suppose for each $\sigma \in S$ that $\pi_\sigma : X_\sigma \rightarrow Y_\sigma$ is a bijection and let $\pi_\sigma^{-1} : Y_\sigma \rightarrow X_\sigma$ be the inverse mapping. Then $\pi_S^{-1} : (Y_S, q_N) \rightarrow (X_S, p_N)$ is also a Σ -homomorphism.*

Proof Let $\nu \in N$ be of type $L \rightarrow \sigma$. Then $q_\nu \pi_\circ^L = \pi_\sigma p_\nu$, and therefore by Lemma 2.1.1 (3) it follows that $p_\nu (\pi_\circ^{-1})^L = p_\nu (\pi_\circ^L)^{-1} = \pi_\sigma^{-1} q_\nu$, which implies that π_S^{-1} is a Σ -homomorphism. \square

A Σ -homomorphism $\pi_S : (X_S, p_N) \rightarrow (Y_S, q_N)$ is said to be a Σ -*isomorphism* if the mapping $\pi_\sigma : X_\sigma \rightarrow Y_\sigma$ is a bijection for each $\sigma \in S$. (Proposition 2.2.2 implies that this definition is ‘correct’.) If it is clear which signature is involved then the name of the signature can be omitted, i.e., homomorphism (resp. isomorphism) will be written instead of Σ -homomorphism (resp. Σ -isomorphism) when Σ can be deduced from the context.

The Σ -algebras (X_S, p_N) and (Y_S, q_N) are said to be *isomorphic* if there exists an isomorphism from (X_S, p_N) to (Y_S, q_N) ; it is easily checked that the property of being isomorphic defines an equivalence relation on the class of all Σ -algebras.

In what follows consider the Σ -algebra (X_S, p_N) as being fixed. Let $\dot{X}_S \subset X_S$ (i.e., $\dot{X}_\sigma \subset X_\sigma$ for each $\sigma \in S$). Then the family of sets \dot{X}_S is said to be *invariant in* (X_S, p_N) , or just *invariant*, if $p_\nu(\dot{X}_\circ^L) \subset \dot{X}_\sigma$ whenever $\nu \in N$ is of type $L \rightarrow \sigma$. In particular, the family X_S is itself trivially invariant. The family \dot{X}_S being invariant means exactly that the following two conditions have to be satisfied:

- (i) $p_\nu(\varepsilon) \in \dot{X}_\sigma$ for each $\nu \in N$ of type $\emptyset \rightarrow \sigma$.
- (ii) If $\nu \in N$ is of type $L \rightarrow \sigma$ with $L \neq \emptyset$ and $c \in X_\circ^L$ is such that $c(\eta) \in \dot{X}_\eta$ for each $\eta \in L$ then $p_\nu(c) \in \dot{X}_\sigma$.

If Σ is an enumerated signature then these two conditions become the following:

- (i) $p_\nu(\varepsilon) \in \dot{X}_\sigma$ for each $\nu \in N$ of type $\varepsilon \rightarrow \sigma$.
- (ii) If $\nu \in N$ is of type $\sigma_1 \cdots \sigma_n \rightarrow \sigma$ with $n \geq 1$ and $x_j \in \dot{X}_{\sigma_j}$ for each j then $p_\nu(x_1, \dots, x_n) \in \dot{X}_\sigma$.

A related notion is that of a subalgebra: A Σ -algebra (Y_S, q_N) is said to be a *subalgebra* of (X_S, p_N) if $Y_S \subset X_S$ and q_ν is the restriction of p_ν to $\text{Dom}(q_\nu)$ for each $\nu \in N$. In this case the family Y_S is clearly invariant. Conversely, let \dot{X}_S be any invariant family and for each $\nu \in N$ of type $L \rightarrow \sigma$ let \dot{p}_ν denote the restriction of p_ν to \dot{X}_\circ^L , so \dot{p}_ν is a mapping from \dot{X}_\circ^L to \dot{X}_σ . Then (\dot{X}_S, \dot{p}_N) is a subalgebra of (X_S, p_N) .

This means that there is a one-to-one correspondence between invariant families and subalgebras of (X_S, p_N) . (The former tend to dominate, however, in this study.) If \check{X}_S is an invariant family then the corresponding subalgebra $(\check{X}_S, \check{p}_N)$ is called the *subalgebra associated with \check{X}_S* .

Lemma 2.2.1 *Let π_S be a homomorphism from (X_S, p_N) to a Σ -algebra (Y_S, q_N) .*

(1) *If the family \check{X}_S is invariant in (X_S, p_N) and $\check{Y}_\sigma = \pi_\sigma(\check{X}_\sigma)$ for each $\sigma \in S$ then \check{Y}_S is invariant in (Y_S, q_N) .*

(2) *If the family \check{Y}_S is invariant in (Y_S, q_N) and $\check{X}_\sigma = \pi_\sigma^{-1}(\check{Y}_\sigma)$ for each $\sigma \in S$ then \check{X}_S is invariant in (X_S, p_N) .*

Proof (1) It must be shown that if $\nu \in N$ is of type $L \rightarrow \sigma$ then $q_\nu(c) \in \check{Y}_\sigma$ for each $c \in \check{Y}_\sigma^L$. But if $c \in \check{Y}_\sigma^L$ then there exists $c' \in \check{X}_\sigma^L$ with $\pi_\sigma^L(c') = c$ (since Lemma 2.1.1 (3) implies that $\pi_\sigma^L(\check{X}_\sigma^L) = \check{Y}_\sigma^L$) and hence

$$q_\nu(c) = q_\nu(\pi_\sigma^L(c')) = \pi_\sigma(p_\nu(c')) \in \pi_\sigma(\check{X}_\sigma) = \check{Y}_\sigma.$$

(2) This time it must be shown that if $\nu \in N$ is of type $L \rightarrow \sigma$ and $c \in \check{X}_\sigma^L$ then $p_\nu(c) \in \check{X}_\sigma$, i.e., that $\pi_\sigma(p_\nu(c)) \in \check{Y}_\sigma$. But if $c \in \check{X}_\sigma^L$ then it is easy to check that $\pi_\sigma^L(c) \in \check{Y}_\sigma^L$ and therefore $\pi_\sigma(p_\nu(c)) = q_\nu(\pi_\sigma^L(c)) \in \check{Y}_\sigma$. \square

Lemma 2.2.1 says that both the image and the pre-image of a subalgebra under a homomorphism are again subalgebras.

Lemma 2.2.2 *Let $A_S \subset X_S$. Then the family \check{X}_S defined by*

$$\check{X}_\sigma = A_\sigma \cup \bigcup_{\nu \in N_\sigma} \text{Im}(p_\nu)$$

for each $\sigma \in S$ is invariant in (X_S, p_N) .

Proof If $\nu \in N$ is of type $L \rightarrow \sigma$ then $p_\nu(c) \in p_\nu(X_\sigma^L) = \text{Im}(p_\nu) \subset \check{X}_\sigma$ for all $c \in X_\sigma^L$, and so in particular for all $c \in \check{X}_\sigma^L$. \square

Lemma 2.2.3 *Let $A_S \subset X_S$. Then there exists a minimal invariant family containing A_S (i.e., an invariant family \hat{X}_S with $A_S \subset \hat{X}_S$ such that if Y_S is any invariant family containing A_S then $\hat{X}_S \subset Y_S$).*

Proof As already noted, the family X_S is itself invariant, and it contains of course A_S . Moreover, it is easy to see that an arbitrary intersection of invariant families is again invariant. (More precisely, if X_S^t is invariant for each $t \in T$ and $\check{X}_\sigma = \bigcap_{t \in T} X_\sigma^t$ for each $\sigma \in S$ then \check{X}_S is also invariant.) The intersection of all the invariant families containing A_S is thus the required minimal family. In fact, this minimal family \hat{X}_S

can be given somewhat more explicitly: For each $n \in \mathbb{N}$ define a family $\hat{X}_S^n \subset X_S$ by putting $\hat{X}_S^0 = A_S$ and for each $n \in \mathbb{N}$, $\sigma \in S$ letting

$$\hat{X}_\sigma^{n+1} = \hat{X}_\sigma^n \cup \bigcup_{\nu \in N_\sigma} \text{Im}(p_\nu^n),$$

where if ν is of type $L \rightarrow \sigma$ then p_ν^n is the restriction of p_ν to $(\hat{X}_\sigma^n)^L$. Then it is straightforward to check that $\hat{X}_\sigma = \bigcup_{n \in \mathbb{N}} \hat{X}_\sigma^n$ for each $\sigma \in S$. This shows that each element of \hat{X}_σ can be ‘constructed’ in a finite number of steps out of elements from the family A_S and elements which have already been ‘constructed’. \square

The main interest here is in the minimal invariant family (i.e., the family given by Lemma 2.2.3 with $A_\sigma = \emptyset$ for each $\sigma \in S$). The subalgebra associated with this family will be referred to as the *minimal subalgebra of (Y_S, q_N)* .

It is often the case that a Σ -algebra (X_S, p_N) is given and then Z_S is defined to be the minimal invariant family. This will then be accompanied by the statement that Z_S is defined by the following three rules:

- (i) If $\nu \in N$ is of type $\emptyset \rightarrow \sigma$ then $p_\nu(\varepsilon)$ is an element of Z_σ .
- (ii) If $\nu \in N$ is of type $L \rightarrow \sigma$ with $L \neq \emptyset$ and $c \in X_\sigma^L$ is such that $c(\eta) \in Z_\eta$ for each $\eta \in L$ then $p_\nu(c)$ is an element of Z_σ .
- (iii) The only elements in Z_σ are those which can be obtained using (i) and (ii).

Rules (i) and (ii) say that Z_S is invariant. Rule (iii) should be regarded as a somewhat imprecise reformulation of the final statement in the proof of Lemma 2.2.3. Of course, such a statement is really redundant, but it usually helps to clarify how the mappings in the family p_N operate in the particular case under consideration. (It is often convenient to divide up rules (i) and (ii) into various sub-cases, ending up with not three but four or more rules.)

If Σ is enumerated then the above rules take on the following form:

- (i) If $\nu \in N$ is of type $\varepsilon \rightarrow \sigma$ then $p_\nu(\varepsilon)$ is an element of Z_σ .
- (ii) If $\nu \in N$ is of type $\sigma_1 \cdots \sigma_n \rightarrow \sigma$ for some $n \geq 1$ and $x_j \in Z_{\sigma_j}$ for $j = 1, \dots, n$ then $p_\nu(x_1, \dots, x_n)$ is an element of Z_σ .
- (iii) The only elements in Z_σ are those which can be obtained using (i) and (ii).

The Σ -algebra (X_S, p_N) is now said to be *minimal* if X_S is the only invariant family. Such algebras play an important role, and so some of their elementary properties will now be presented. Note that the minimal subalgebra of (X_S, p_N) is always a minimal Σ -algebra.

Proposition 2.2.3 *Let (X_S, p_N) and (Y_S, q_N) be Σ -algebras.*

- (1) *If (X_S, p_N) is minimal then there exists at most one homomorphism from (X_S, p_N) to (Y_S, q_N) .*
- (2) *If (Y_S, q_N) is minimal then any homomorphism $\pi_S : (X_S, p_N) \rightarrow (Y_S, q_N)$ is surjective (i.e., the mapping $\pi_\sigma : X_\sigma \rightarrow Y_\sigma$ is surjective for each $\sigma \in S$).*

Proof (1) Let π_S and ϱ_S be homomorphisms from (X_S, p_N) to (Y_S, q_N) and for each $\sigma \in S$ let $\hat{X}_\sigma = \{x \in X_\sigma : \pi_\sigma(x) = \varrho_\sigma(x)\}$. Consider $\nu \in N$ of type $L \rightarrow \sigma$ and let $c \in \hat{X}_\sigma^L$; then $c(\eta) \in \hat{X}_\eta$ for each $\eta \in L$ and hence

$$\pi_\sigma^L(c)(\eta) = \pi_\eta(c(\eta)) = \varrho_\eta(c(\eta)) = \varrho_\sigma^L(c)(\eta)$$

which implies that $\pi_\sigma^L(c) = \varrho_\sigma^L(c)$. Therefore

$$\pi_\sigma(p_\nu(c)) = q_\nu(\pi_\sigma^L(c)) = q_\nu(\varrho_\sigma^L(c)) = \varrho_\sigma(p_\nu(c)),$$

i.e., $p_\nu(c) \in \hat{X}_\sigma$. This shows that the family \hat{X}_S is invariant and thus $\hat{X}_S = X_S$, since (X_S, p_N) is minimal. In other words, $\pi_S = \varrho_S$.

(2) This follows immediately from Lemma 2.2.1 (1). \square

Proposition 2.2.4 *If (X_S, p_N) is minimal then $\bigcup_{\nu \in N_\sigma} \text{Im}(p_\nu) = X_\sigma$ for each $\sigma \in S$.*

Proof This follows immediately from Lemma 2.2.2. \square

The converse of Proposition 2.2.4 does not hold in general. However, the condition occurring there can be combined with a second condition to give a useful sufficient criterion for minimality:

Proposition 2.2.5 *Suppose there exists a family of mappings $\#_S$ with $\#_\sigma : X_\sigma \rightarrow \mathbb{N}$ for each σ such that if $\nu \in N$ is of type $L \rightarrow \sigma$ then*

$$\#_\eta(c(\eta)) < \#_\sigma(p_\nu(c))$$

holds for all $c \in X_\sigma^L$, $\eta \in L$. For each $\sigma \in S$ let A_σ be a subset of X_σ such that $A_\sigma \cup \bigcup_{\nu \in N_\sigma} \text{Im}(p_\nu) = X_\sigma$. Then X_S is the only invariant family containing A_S . In particular, if $\bigcup_{\nu \in N_\sigma} \text{Im}(p_\nu) = X_\sigma$ for each $\sigma \in S$ then (X_S, p_N) is minimal.

Proof Let \hat{X}_S be the minimal invariant family containing A_S and suppose $\hat{X}_S \neq X_S$. There thus exists $\sigma \in S$ and $x \in X_\sigma \setminus \hat{X}_\sigma$ such that $\#_\sigma(x) \leq \#_\tau(x')$ whenever $x' \in X_\tau \setminus \hat{X}_\tau$ for some $\tau \in S$. Then $x \in \text{Im}(p_\nu)$ for some $\nu \in N_\sigma$, since $A_\sigma \subset \hat{X}_\sigma$. Let ν be of type $L \rightarrow \sigma$, so there exists $c \in X_\sigma^L$ with $x = p_\nu(c)$. But it then follows that $\#_\eta(c(\eta)) < \#_\sigma(x)$ and hence that $c(\eta) \in \hat{X}_\eta$ for each $\eta \in L$ (by the minimality of $\#_\sigma(x)$). However, this implies $x \in \hat{X}_\sigma$, since the family \hat{X}_S is invariant, which is a contradiction. \square

If Σ is enumerated then the condition involving the family $\#_S$ in Proposition 2.2.5 is that whenever $\nu \in N$ is of type $\sigma_1 \cdots \sigma_n \rightarrow \sigma$ then

$$\#_{\sigma_j}(x_j) < \#_\sigma(p_\nu(x_1, \dots, x_n))$$

must hold for each $j = 1, \dots, n$ for each $(x_1, \dots, x_n) \in X_{\sigma_1} \times \cdots \times X_{\sigma_n}$. It is easy to check that the Λ -algebra (X_B, p_K) defined in Example 2.2.1 is minimal.

A subset $S' \subset S$ is said to be *closed in Σ* or simply *closed* if $\sigma \in S'$ whenever there exists $\nu \in N$ of type $L \rightarrow \sigma$ for some L and $\langle \eta \rangle \in S'$ for each $\eta \in L$. (This means in particular that $\sigma \in S'$ if there exists an element $\nu \in N$ of type $\emptyset \rightarrow \sigma$.) The signature Σ will be called *pervasive* if the only closed subset of S is S itself.

It is easily checked that the signature Λ in Example 2.2.1 is pervasive. (Note that if Σ is enumerated then S' being closed means exactly that $\sigma \in S'$ must hold whenever there exists $\nu \in N$ of type $\sigma_1 \cdots \sigma_n \rightarrow \sigma$ and $\sigma_j \in S'$ for each $j = 1, \dots, n$.)

Proposition 2.2.6 *If Σ is pervasive then $X_\sigma \neq \emptyset$ for all $\sigma \in S$ for any Σ -algebra (X_S, p_N) . Conversely, if (X_S, p_N) minimal and $X_\sigma \neq \emptyset$ for all $\sigma \in S$ then Σ is pervasive.*

Proof The set $S_o = \{\sigma \in S : X_\sigma \neq \emptyset\}$ is clearly closed. Thus if Σ is pervasive then $S_o = S$, i.e., $X_\sigma \neq \emptyset$ for each $\sigma \in S$. Conversely, if S' is closed and the family X'_S is defined by $X'_\sigma = X_\sigma$ for $\sigma \in S'$ and $X'_\sigma = \emptyset$ for $\sigma \in S \setminus S'$ then X'_S is invariant. Therefore if (X_S, p_N) is minimal and $X_\sigma \neq \emptyset$ for each $\sigma \in S$ then $S' = S$. \square

Note that the set $S' = \{\sigma \in S : N_\sigma \neq \emptyset\}$ is always closed, and hence a necessary condition for pervasiveness is that $N_\sigma \neq \emptyset$ for each $\sigma \in S$.

To end the section the special case of a *single-sorted* signature will be looked at, i.e., a signature of the form $(\mathbb{I}, N, \Delta, \delta)$. In this case there is no choice for δ (since there is only one mapping possible from N to \mathbb{I}) and so a single-sorted signature can be regarded as being a pair (N, Δ) consisting of a set N and a mapping $\Delta : N \rightarrow \mathcal{F}_\mathbb{I}$ (recalling that $\mathcal{F}_\mathbb{I}$ is just considered to be the set of all finite subsets of the basic set of names \mathbb{M}).

Let $\Sigma = (N, \Delta)$ be a single-sorted signature; then (identifying a family $Z_\mathbb{I}$ with the single object Z_ε it contains) a Σ -algebra is here a pair (X, p_N) consisting of a set X and a family of mappings p_N with $p_\nu : X^{\Delta(\nu)} \rightarrow X$ for each $\nu \in N$.

Consider the very special case of an enumerated single-sorted signature $\Sigma = (N, \Delta)$. Then, since \mathbb{I}^* can clearly be identified with the set of natural numbers \mathbb{N} , Δ can here be regarded as a mapping from N to \mathbb{N} . If (X, p_N) is a Σ -algebra then p_ν is a mapping from the cartesian product $X^{\Delta(\nu)}$ to X , so $\Delta(\nu)$ is just the number of arguments taken by the operator p_ν .

If $\Sigma = (S, N, \Delta, \delta)$ is an arbitrary signature then a single-sorted signature $\Sigma^\circ = (N, \Delta^\circ)$ can be defined by letting $\Delta^\circ(\nu)$ be the underlying set involved in the S -typed set $\Delta(\nu)$. This means that Σ° is obtained from Σ by no longer distinguishing between the various types. Now let (X, p_N) be a Σ° -algebra and for each $\sigma \in S$ put $X_\sigma = X$. Then, since $|X_S| = X$ and any mapping from $\Delta(\nu)$ to $|X_S|$ is automatically typed, it follows that $X_\sigma^{\Delta(\nu)} = X^{\Delta^\circ(\nu)}$ for each $\nu \in N$, which means that (X_S, p_N) is a Σ -algebra. This almost trivial method of obtaining Σ -algebras turns out to be surprisingly useful.

2.3 Initial and free algebras

For the whole of the section let $\Sigma = (S, N, \Delta, \delta)$ be a signature. A Σ -algebra (X_S, p_N) is said to be *initial* if for each Σ -algebra (Y_S, q_N) there exists a unique homomorphism from (X_S, p_N) to (Y_S, q_N) . (The terminology ‘initial’ is the standard one used in category theory to describe such a situation.)

In Proposition 2.3.2 initial algebras are characterised as those that are minimal and possess a further property, here called unambiguity. In Proposition 2.3.3 it is shown there is a unique isomorphism class of initial Σ -algebras. This essentially amounts to showing that an initial Σ -algebra exists, which follows from Proposition 2.3.2 and the existence of a minimal unambiguous Σ -algebra.

A Σ -algebra (X_S, p_N) is said to be *unambiguous* if the following hold:

- (i) The mapping p_ν is injective for each $\nu \in N$.
- (ii) For each $\sigma \in S$ the sets $\text{Im}(p_\nu)$, $\nu \in N_\sigma$, are disjoint subsets of X_σ .

In particular, the Λ -algebra (X_B, p_K) in Example 2.2.1 is clearly unambiguous. Related to unambiguity is the following property: A Σ -algebra (X_S, p_N) is said to be *regular* if for each $x \in X_\sigma$ there exists a unique $\nu \in N_\sigma$ and a unique element $c \in \text{Dom}(p_\nu)$ such that $p_\nu(c) = x$. Thus (X_S, p_N) is regular if and only if the mapping p_ν is injective for each $\nu \in N$ and for each $\sigma \in S$ the sets $\text{Im}(p_\nu)$, $\nu \in N_\sigma$, form a partition of X_σ .

Lemma 2.3.1 *A minimal Σ -algebra is regular if and only if it is unambiguous.*

Proof This follows immediately from Proposition 2.2.4. \square

Before going any further consider again the natural number triples introduced at the beginning of the chapter. These are really the algebras corresponding to the enumerated signature consisting of a single type `nat` and two operator names `Zero` and `Succ` with `Zero` of type $\varepsilon \rightarrow \text{nat}$ and `Succ` of type $\text{nat} \rightarrow \text{nat}$; however, it was more convenient to represent an algebra $(\{X_{\text{nat}}\}, \{p_{\text{Zero}}, p_{\text{Succ}}\})$ using the natural number triple $(X_{\text{nat}}, p_{\text{Zero}}(\varepsilon), p_{\text{Succ}})$. The equivalence of initial and Peano triples is easily seen to be just a special case of Proposition 2.3.2.

In order to get started an unambiguous Σ -algebra is needed, and for this the following trivial observation is useful: Let $\Sigma^\circ = (N, \Delta^\circ)$ be the single-sorted signature defined at the end of the previous section (so $\Delta^\circ(\nu)$ is just the underlying set involved in the S -typed set $\Delta(\nu)$). Let (X, p_N) be a Σ° -algebra and (X_S, p_N) be the Σ -algebra with $X_\sigma = X$ for each $\sigma \in S$.

Lemma 2.3.2 *If the Σ° -algebra (X, p_N) is unambiguous then so is the Σ -algebra (X_S, p_N) .*

Proof This is clear. \square

Lemma 2.3.3 *There exists an unambiguous Σ -algebra.*

Proof By Lemma 2.3.2 it is enough to show that if $\Sigma' = (N, \Delta)$ is a single-sorted signature then there exists a unambiguous Σ' -algebra. The construction given below is just one of many possibilities of defining such an algebra.

Let $M = M_o \cup N$, where M_o is the set of all pairs of the form (ν, η) with $\nu \in N$ and $\eta \in \Delta(\nu)$, and let X be the set of all non-empty finite subsets of M^* . Now if $\nu \in N$ with $\Delta(\nu) = \emptyset$ then define $p_\nu : \mathbb{I} \rightarrow X$ by letting $p_\nu(\varepsilon) = \{\nu\}$ (where here ν is the list consisting of the single component ν), and if $\nu \in N$ with $\Delta(\nu) = L \neq \emptyset$ then define a mapping $p_\nu : X^L \rightarrow X$ by letting

$$p_\nu(c) = \{\nu\} \cup \bigcup_{\eta \in L} \{(\nu, \eta) \triangleleft s : s \in c(\eta)\}$$

for each $c \in X^L$, where $\triangleleft : M \times M^* \rightarrow M^*$ is the (infix) operation of adding an element to the beginning of a list. This gives a Σ' -algebra (X, p_N) . But it is easily checked that $\text{Im}(p_{\nu_1})$ and $\text{Im}(p_{\nu_2})$ are disjoint subsets of X whenever $\nu_1 \neq \nu_2$, and also that p_ν is injective for each $\nu \in N$. Hence (X, p_N) is unambiguous. \square

Proposition 2.3.1 *There exists a minimal unambiguous Σ -algebra.*

Proof By Lemma 2.3.3 there exists an unambiguous Σ -algebra (X_S, p_N) . But then any subalgebra of (X_S, p_N) is also unambiguous. In particular, the minimal subalgebra is minimal and unambiguous. \square

Proposition 2.3.2 *The following statements are equivalent for a Σ -algebra (X_S, p_N) :*

- (i) (X_S, p_N) is initial.
- (ii) (X_S, p_N) is minimal and unambiguous.
- (iii) (X_S, p_N) is minimal and regular.

Proof The equivalence of (ii) and (iii) is just Lemma 2.3.1. To show their equivalence to (i) a couple of simple facts will be needed:

Lemma 2.3.4 *Let (X_S, p_N) be a minimal regular Σ -algebra. Then there exists a unique family $\#_S$ with $\#_\sigma : X_\sigma \rightarrow \mathbb{N}$ for each $\sigma \in S$ such that $\#_\sigma(p_\nu(\varepsilon)) = 0$ if $\nu \in N$ is of type $\emptyset \rightarrow \sigma$ and such that*

$$\#_\sigma(p_\nu(c)) = 1 + \max\{\#_\eta(c(\eta)) : \eta \in L\}$$

for all $c \in X_\sigma^L$ whenever $\nu \in N$ is of type $L \rightarrow \sigma$ with $L \neq \emptyset$.

Proof For each $m \in \mathbb{N}$ a family of mappings $\#_S^m$ will be defined with $\#_\sigma^m : X_\sigma \rightarrow \mathbb{N}$ for each $\sigma \in S$, and then $\#_S$ will be obtained as the limit of the sequence $\{\#_S^m\}_{m \geq 0}$. The definition is by induction on m : First define $\#_\sigma^0 = 0$ for each $\sigma \in S$. Next suppose that the family $\#_S^m$ has already been defined for some $m \in \mathbb{N}$. Then since (X_S, p_N) is regular there is a unique family of mappings $\#_S^{m+1}$ such that $\#_\sigma^{m+1}(p_\nu(\varepsilon)) = 0$ if $\nu \in N$ is of type $\emptyset \rightarrow \sigma$ and such that

$$\#_\sigma^{m+1}(p_\nu(c)) = 1 + \max\{\#_\eta^m(c(\eta)) : \eta \in L\}$$

for all $c \in X_\sigma^L$ whenever $\nu \in N$ is of type $L \rightarrow \sigma$ with $L \neq \emptyset$.

Then $\#_S^m \leq \#_S^{m+1}$ for each $m \in \mathbb{N}$ (i.e., $\#_\sigma^m(x) \leq \#_\sigma^{m+1}(x)$ for all $x \in X_\sigma$, $\sigma \in S$): This follows by induction on m , since $\#_S^0 \leq \#_S^1$ holds by definition and if $\#_S^m \leq \#_S^{m+1}$ for some $m \in \mathbb{N}$ and $\nu \in N$ is of type $L \rightarrow \sigma$ with $L \neq \emptyset$ then

$$\begin{aligned} \#_\sigma^{m+1}(p_\nu(c)) &= 1 + \max\{\#_\eta^m(c(\eta)) : \eta \in L\} \\ &\leq 1 + \max\{\#_\eta^{m+1}(c(\eta)) : \eta \in L\} = \#_\sigma^{m+2}(p_\nu(c)) \end{aligned}$$

for all $c \in X_\sigma^L$, which implies that $\#_S^{m+1} \leq \#_S^{m+2}$.

The sequence $\{\#_\sigma^m(x)\}_{m \geq 0}$ is bounded for each $x \in X_\sigma$, $\sigma \in S$: Let \hat{X}_σ denote the set of those elements $x \in X_\sigma$ for which this is the case. Then it is easily checked that the family \hat{X}_S is invariant, and hence $\hat{X}_S = X_S$, since (X_S, p_N) is minimal.

Let $x \in X_\sigma$; then by the above $\{\#_\sigma^m(x)\}_{m \geq 0}$ is a bounded increasing sequence from \mathbb{N} , and so there exists an element $\#_\sigma(x) \in \mathbb{N}$ such that $\#_\sigma^m(x) = \#_\sigma(x)$ for all but finitely many m . This defines a mapping $\#_\sigma : X_\sigma \rightarrow \mathbb{N}$ for each $\sigma \in S$, and it immediately follows that the family $\#_S$ has the required property.

It remains to show the uniqueness, so suppose $\#'_S$ is another family of mappings with this property. For each $\sigma \in S$ let $X'_\sigma = \{x \in X_\sigma : \#'_\sigma(x) = \#_\sigma(x)\}$; then the family X'_S is clearly invariant and hence $X'_S = X_S$, since (X_S, p_N) is minimal. \square

Lemma 2.3.5 *Let (X_S, p_N) be a Σ -algebra isomorphic to a minimal unambiguous Σ -algebra. Then (X_S, p_N) is also minimal and unambiguous.*

Proof Let $\pi_S : (X_S, p_N) \rightarrow (Y_S, q_N)$ be an isomorphism, with (Y_S, q_N) a minimal unambiguous Σ -algebra. If X'_S is an invariant family in (X_S, p_N) then Lemma 2.2.1 (1) implies that $\pi_\sigma(X'_\sigma) = Y_\sigma$ for each $\sigma \in S$, since Y_S is the only invariant family in (Y_S, q_N) . Thus $X'_\sigma = \pi_\sigma^{-1}(\pi_\sigma(X'_\sigma)) = \pi_\sigma^{-1}(Y_\sigma) = X_\sigma$ for each $\sigma \in S$, i.e., $X'_S = X_S$, and this shows that (X_S, p_N) is minimal. Now if $\nu \in N_\sigma$ and $x \in \text{Im}(p_\nu)$ then by the definition of a homomorphism $\pi_\sigma(x) \in \text{Im}(q_\nu)$. It immediately follows that if $\nu_1, \nu_2 \in N_\sigma$ with $\nu_1 \neq \nu_2$ then $\text{Im}(p_{\nu_1}) \cap \text{Im}(p_{\nu_2}) = \emptyset$. Finally, if $\nu \in N$ is of type $L \rightarrow \sigma$ then $q_\nu \pi_\sigma^L = \pi_\sigma p_\nu$; moreover, q_ν is injective, π_σ is a bijection and by Lemma 2.1.1 (3) π_σ^L is also a bijection. Thus p_ν is injective. \square

Lemma 2.3.6 *Any two initial Σ -algebras are isomorphic.*

Proof Let (X_S, p_N) and (Y_S, q_N) be initial Σ -algebras. There then exists a unique homomorphism π_S from (X_S, p_N) to (Y_S, q_N) and a unique homomorphism π'_S from (Y_S, q_N) to (X_S, p_N) . However, by Proposition 2.2.1 $\{\pi'_\sigma \pi_\sigma : \sigma \in S\}$ is then a homomorphism from (X_S, p_N) to itself, and by assumption there is only one such homomorphism. Hence $\pi'_\sigma(\pi_\sigma(x)) = x$ for each $x \in X_\sigma$ (since the identity mappings also define a homomorphism from (X_S, p_N) to itself). In the same way $\pi_\sigma(\pi'_\sigma(y)) = y$ for each $y \in Y_\sigma$, and therefore π_S is an isomorphism. (This proof is quite general: It is just the proof that in any category possessing initial objects these objects are, up to isomorphism, unique.) \square

The proof of Proposition 2.3.2 can now be commenced. Suppose first that (X_S, p_N) is minimal and regular and let (Y_S, q_N) be any Σ -algebra. It is enough to construct a homomorphism π_S from (X_S, p_N) to (Y_S, q_N) , since Proposition 2.2.3 (1) then implies that this homomorphism is unique.

Let $\#_S$ be the family of mappings given by Lemma 2.3.4 (with $\#_\sigma : X_\sigma \rightarrow \mathbb{N}$ for each $\sigma \in S$) and for each $\sigma \in S$, $m \in \mathbb{N}$ let $X_\sigma^m = \{x \in X_\sigma : \#_\sigma(x) = m\}$. Define π_S by defining π_σ on X_σ^m for each $\sigma \in S$ using induction on m . Let $x \in X_\sigma^0$; then, since (X_S, p_N) is regular, there exists a unique $\nu \in N_\sigma$ and a unique element $c \in \text{Dom}(p_\nu)$ such that $p_\nu(c) = x$, and here ν must be of type $\emptyset \rightarrow \sigma$ and $x = p_\nu(\varepsilon)$. Thus put $\pi_\sigma(x) = q_\nu(\varepsilon)$, which defines π_σ on X_σ^0 for each $\sigma \in S$. Now let $m > 0$ and suppose π_τ has already been defined on X_τ^k for all $k < m$ and all $\tau \in S$. Let $x \in X_\sigma^m$; again using the regularity of (X_S, p_N) there exists a unique $\nu \in N_\sigma$ and a unique $c \in \text{Dom}(p_\nu)$ such that $x = p_\nu(c)$. Let ν be of type $L \rightarrow \sigma$, so here $L \neq \emptyset$ and $c \in X_\diamond^L$. Then by the defining property of $\#_S$ it follows that $\#_\eta(c) < m$ for each $\eta \in L$, which means $\pi_\eta(c)$ is already defined for each $\eta \in L$ and hence that $\pi_\diamond^L(c)$ is already defined (i.e., $\pi_\diamond^L(c)$ is the element c' of Y_\diamond^L given by $c'(\eta) = \pi_\eta(c)$ for each $\eta \in L$). Thus here put $\pi_\sigma(x) = q_\nu(\pi_\diamond^L(c))$. In this way π_σ is defined on X_σ^m for each $\sigma \in S$ and each $m \in \mathbb{N}$, and the family π_S is a homomorphism more-or-less by construction.

This shows that any minimal regular (and thus any minimal unambiguous) Σ -algebra is initial. The converse is a special case of a more general situation: Suppose a property P of Σ -algebras is given satisfying the following three conditions:

- (i) There exists a Σ -algebra having property P .
- (ii) Every Σ -algebra having property P is initial.
- (iii) A Σ -algebra isomorphic to a Σ -algebra having property P also has property P .

Then every initial Σ -algebra has property P , and so having property P is equivalent to being initial. (Let (X_S, p_N) be initial; by (i) there exists a Σ -algebra (Y_S, q_N) having property P , and by (ii) and Lemma 2.3.6 (X_S, p_N) and (Y_S, q_N) are isomorphic; thus by (iii) (X_S, p_N) has property P .) Now by Proposition 2.3.1 there exists a minimal unambiguous Σ -algebra, and the first part of the proof shows that each minimal unambiguous Σ -algebra is initial. Finally, by Lemma 2.3.5 a Σ -algebra isomorphic to a minimal unambiguous Σ -algebra is minimal and unambiguous, and therefore each initial Σ -algebra is minimal and unambiguous. \square

Proposition 2.3.2 implies that the Λ -algebra (X_B, p_K) in Example 2.2.1 is initial.

Proposition 2.3.3 *There is exactly one isomorphism class of initial Σ -algebras. (This means, more precisely, that an initial Σ -algebra (X_S, p_N) exists and that a Σ -algebra (Y_S, q_N) is initial if and only if it is isomorphic to (X_S, p_N) .)*

Proof Propositions 2.3.1 and 2.3.2 imply that an initial Σ -algebra exists, and by Lemma 2.3.6 any two initial Σ -algebras are isomorphic. Finally, it is also clear that a Σ -algebra isomorphic to an initial Σ -algebra is itself initial. \square

A type $\sigma \in S$ will be said to be *primitive* if ν is of type $\emptyset \rightarrow \sigma$ for each $\nu \in N_\sigma$. (In the signature Λ in Example 2.2.1 the primitive types are therefore `bool` and `int`.) Note that if (X_S, p_N) is an initial Σ -algebra and $\sigma \in S$ is a primitive type then the mapping $\nu \mapsto p_\nu(\varepsilon)$ maps N_σ bijectively onto X_σ . (For the Λ -algebra (X_B, p_K) in Example 2.2.1 this gives the obvious bijections between $K_{\text{bool}} = \{\text{True}, \text{False}\}$ and $X_{\text{bool}} = \mathbb{B} = \{T, F\}$ and between $K_{\text{int}} = \underline{\text{Int}}$ and $X_{\text{int}} = \mathbb{Z}$.)

Proposition 2.3.4 *The minimal subalgebra of an unambiguous Σ -algebra is initial.*

Proof This follows from Proposition 2.3.2, since, as already noted, any subalgebra of an unambiguous algebra is unambiguous. \square

The concept of being initial can be generalised and this leads to what is called a free algebra. For the rest of the section let I be an S -typed set and define a family I_S by putting $I_\sigma = \{\eta \in I : \langle \eta \rangle = \sigma\}$ for each $\sigma \in S$,

A Σ -algebra (X_S, p_N) is said to *contain* I if $I_S \subset X_S$ (which is of course equivalent to having $\eta \in X_\eta$ for each $\eta \in I$). A Σ -algebra (X_S, p_N) is then called *I -free* if it contains I and for each Σ -algebra (Y_S, q_N) and each $c \in Y^I$ there exists a unique Σ -homomorphism $\pi_S^c : (X_S, p_N) \rightarrow (Y_S, q_N)$ such that $\pi_\eta^c(\eta) = c(\eta)$ for all $\eta \in I$. In Proposition 2.3.6 it will be shown that an I -free Σ -algebra exists and that it is, in an appropriate sense, unique.

Note that a Σ -algebra (X_S, p_N) is initial if and only if it is \emptyset -free. Moreover, it will turn out that a Σ -algebra (X_S, p_N) containing I is I -free if and only if a related algebra (X_S, p_{N_I}) is initial with respect to an extended signature Σ_I .

For each $\eta \in I$ let \diamond_η be some element not in N and such that $\diamond_{\eta_1} \neq \diamond_{\eta_2}$ whenever $\eta_1 \neq \eta_2$. Put $N_I = N \cup \{\diamond_\eta : \eta \in I\}$ and define a signature $\Sigma_I = (S, N_I, \Delta_I, \delta_I)$ with mappings $\Delta_I : N_I \rightarrow \mathcal{F}_S$ and $\delta_I : N_I \rightarrow S$ given by

$$\Delta_I(\nu) = \begin{cases} \Delta(\nu) & \text{if } \nu \in N, \\ \emptyset & \text{if } \nu = \diamond_\eta \text{ for some } \eta \in I, \end{cases}$$

$$\delta_I(\nu) = \begin{cases} \delta(\nu) & \text{if } \nu \in N, \\ \langle \eta \rangle & \text{if } \nu = \diamond_\eta \text{ for some } \eta \in I. \end{cases}$$

Thus in Σ_I each $\nu \in N$ has the same type as it had in Σ and \diamond_η is of type $\emptyset \rightarrow \langle \eta \rangle$ for each $\eta \in I$.

Let (X_S, p_N) be a Σ -algebra containing I and for each $\eta \in I$ define a mapping $p_{\diamond_\eta} : \mathbb{I} \rightarrow X_\eta$ by letting $p_{\diamond_\eta}(\varepsilon) = \eta$. This results in a Σ_I -algebra (X_S, p_{N_I}) which will be called the Σ_I -algebra associated with (X_S, p_N) .

Lemma 2.3.7 *Let (Y_S, q_{N_I}) be any Σ_I -algebra and $\pi_S : X_S \rightarrow Y_S$ be a family of mappings. Then $\pi_S : (X_S, p_{N_I}) \rightarrow (Y_S, q_{N_I})$ is a Σ_I -homomorphism if and only if π_S is a Σ -homomorphism from (X_S, p_N) to (Y_S, q_N) with $\pi_\eta(\eta) = q_{\diamond_\eta}(\varepsilon)$ for each $\eta \in I$.*

Proof This follows directly from the definition of (X_S, p_{N_I}) . \square

Proposition 2.3.5 (X_S, p_N) is an I -free Σ -algebra if and only if (X_S, p_{N_I}) is an initial Σ_I -algebra.

Proof Suppose first that (X_S, p_N) is I -free and let (Y_S, q_{N_I}) be any Σ_I -algebra. Then, putting $c(\eta) = q_{\circ_\eta}(\varepsilon)$ for each $\eta \in I$ defines an assignment $c \in Y_\circ^I$, and thus there exists a unique Σ -homomorphism $\pi_S : (X_S, p_N) \rightarrow (Y_S, q_N)$ with $\pi_\eta(\eta) = q_{\circ_\eta}(\varepsilon)$ for all $\eta \in I$. Therefore by Lemma 2.3.7 π_S is a Σ_I -homomorphism from (X_S, p_{N_I}) to (Y_S, q_{N_I}) . Consider any Σ_I -homomorphism $\hat{\pi}_S : (X_S, p_{N_I}) \rightarrow (Y_S, q_{N_I})$. Then, again using Lemma 2.3.7, $\hat{\pi}_S$ is a Σ -homomorphism from (X_S, p_N) to (Y_S, q_N) with $\hat{\pi}_\eta(\eta) = q_{\circ_\eta}(\varepsilon)$ for all $\eta \in I$, and hence $\hat{\pi}_S = \pi_S$. This shows that (X_S, p_{N_I}) is an initial Σ_I -algebra.

Suppose conversely (X_S, p_{N_I}) is an initial Σ_I -algebra; let (Y_S, q_N) be a Σ -algebra and $c \in Y_\circ^I$. For each $\eta \in I$ define $q_{\circ_\eta} : \mathbb{1} \rightarrow Y_\eta$ by $q_{\circ_\eta}(\varepsilon) = c(\eta)$; then (Y_S, q_{N_I}) is a Σ_I -algebra and hence there is a unique Σ_I -homomorphism $\pi_S^c : (X_S, p_{N_I}) \rightarrow (Y_S, q_{N_I})$. But by Lemma 2.3.7 π_S^c is then a Σ -homomorphism from (X_S, p_N) to (Y_S, q_N) with $\pi_\eta^c(\eta) = q_{\circ_\eta}(\varepsilon) = c(\eta)$ for all $\eta \in I$. Finally, consider any Σ -homomorphism $\hat{\pi}_S^c$ from (X_S, p_N) to (Y_S, q_N) with $\hat{\pi}_\eta^c(\eta) = c(\eta)$ for all $\eta \in I$. Then Lemma 2.3.7 implies that $\hat{\pi}_S^c$ is also a Σ_I -homomorphism from (X_S, p_{N_I}) to (Y_S, q_{N_I}) and thus $\hat{\pi}_S^c = \pi_S^c$. This shows that (X_S, p_N) is I -free. \square

Proposition 2.3.6 There exists an I -free Σ -algebra (X_S, p_N) . Moreover, if (X'_S, p'_N) is any I -free Σ -algebra then there exists a unique Σ -isomorphism π_S from (X_S, p_N) to (X'_S, p'_N) such that $\pi_\eta(\eta) = \eta$ for each $\eta \in I$.

Proof By Proposition 2.3.3 there exists an initial Σ_I -algebra (X_S, p_{N_I}) , and clearly there then exists such an algebra in which the sets in the family X_S are disjoint. This implies in particular that $p_{\circ_\xi}(\varepsilon) \neq p_{\circ_\eta}(\varepsilon)$ whenever ξ and η are different elements of I . The set $\{p_{\circ_\eta}(\varepsilon) : \eta \in I\}$ can thus be identified with the set I and so $I_S \subset X_S$. In this way (X_S, p_N) can be regarded as a Σ -algebra containing I . Moreover, (X_S, p_{N_I}) is then the Σ_I -algebra associated with (X_S, p_N) , and hence Proposition 2.3.5 implies that (X_S, p_N) is I -free. There therefore exists an I -free Σ -algebra, and the final statement follows directly from the definition of being I -free. \square

Let (X_S, p_N) be a Σ -algebra containing I . Then (X_S, p_N) is said to be I -minimal if X_S is the only invariant family in (X_S, p_N) containing I_S . Moreover, (X_S, p_N) is said to be I -unambiguous if the mapping p_ν is injective for each $\nu \in N$ and for each $\sigma \in S$ the sets $\text{Im}(p_\nu)$, $\nu \in N_\sigma$, are disjoint subsets of $X_\sigma \setminus I_\sigma$. Finally, (X_S, p_N) is called I -regular if the mapping p_ν is injective for each $\nu \in N$ and for each $\sigma \in S$ the sets $\text{Im}(p_\nu)$, $\nu \in N_\sigma$, form a partition of $X_\sigma \setminus I_\sigma$.

Proposition 2.3.7 The following statements are equivalent:

- (i) (X_S, p_N) is I -free.
- (ii) (X_S, p_N) is I -minimal and I -unambiguous.
- (iii) (X_S, p_N) is I -minimal and I -regular.

Proof This follows from Propositions 2.3.2 and 2.3.5. \square

It is useful to express the equivalence between statements (i) and (iii) in Proposition 2.3.7 somewhat more explicitly:

Proposition 2.3.8 *A Σ -algebra (X_S, p_N) containing I is I -free if and only if the following three conditions hold:*

- (i) p_ν is injective for each $\nu \in N$.
- (ii) The sets $\text{Im}(p_\nu)$, $\nu \in N_\sigma$, form a partition of $X_\sigma \setminus I_\sigma$ for each $\sigma \in S$.
- (iii) X_S is the only invariant family containing I_S .

Proof Conditions (i) and (ii) are those occurring in the definition of being I -regular, and condition (iii) is that occurring in the definition of being I -minimal. Thus by Proposition 2.3.7 (X_S, p_N) is I -free if and only if (i), (ii) and (iii) hold. \square

2.4 Term algebras

This section introduces what are called term algebras. These provide the simplest explicit examples of initial algebras, and they can be used as the basic components of a ‘real’ programming language. Until further notice assume that $\Sigma = (S, N, \Delta, \delta)$ is an enumerated signature. (The general case is dealt with at the end of the section.)

Let Z be a set; then the concatenation of two lists $\ell, \ell' \in Z^*$ will be denoted by $\ell \ell'$. Thus $\ell \varepsilon = \varepsilon \ell = \ell$ and if $\ell = z_1 \cdots z_m, \ell' = z'_1 \cdots z'_n$ with $m, n \geq 1$ then

$$\ell \ell' = z_1 \cdots z_m z'_1 \cdots z'_n.$$

Concatenation is clearly associative, and hence if $p \geq 1$ and $\ell_1, \dots, \ell_p \in Z^*$ then *the* concatenation of ℓ_1, \dots, ℓ_p can be denoted simply by $\ell_1 \cdots \ell_p$. This notation is clearly compatible with the notation being employed for the elements of Z^* (in the sense that $\ell = z_1 \cdots z_n \in Z^*$ can be considered as the concatenation of the n one element lists z_1, \dots, z_n). If $z \in Z$ and $\ell \in Z^*$ then in particular there is the list $z \ell$ obtained by adding z to the beginning of the list ℓ (a list which is also being denoted by $z \triangleleft \ell$).

Let A be a set and $\Gamma : N \rightarrow A$ be a mapping. A Σ -algebra (Y_S, q_N) can then be obtained as follows: For each $\sigma \in S$ put $Y_\sigma = A^*$; if $\nu \in N$ is of type $\sigma_1 \cdots \sigma_n \rightarrow \sigma$ (with $n \geq 0$) then let $q_\nu : Y_{\sigma_1} \times \cdots \times Y_{\sigma_n} \rightarrow Y_\sigma$ be the mapping defined by

$$q_\nu(\alpha_1, \dots, \alpha_n) = \Gamma(\nu) \alpha_1 \cdots \alpha_n$$

for each $(\alpha_1, \dots, \alpha_n) \in Y_{\sigma_1} \times \cdots \times Y_{\sigma_n}$, i.e., $q_\nu(\alpha_1, \dots, \alpha_n)$ is the list obtained by concatenating the element $\Gamma(\nu)$ and the lists $\alpha_1, \dots, \alpha_n$. In particular, if $\nu \in N$ is of type $\varepsilon \rightarrow \sigma$ then $q_\nu(\varepsilon)$ is just the list consisting of the single component $\Gamma(\nu)$. (This is really just an instance of the construction described at the end of Section 2.2.)

Now let (E_S, \square_N) be the minimal subalgebra of (Y_S, q_N) , i.e., E_S is the minimal invariant family and \square_ν is the corresponding restriction of q_ν for each $\nu \in N$. This minimal Σ -algebra (E_S, \square_N) is called the *term Σ -algebra specified by Γ* , and Γ is then referred to as a *term algebra specifier*. Thus $E_\sigma \subset A^*$ for each $\sigma \in S$ and the family E_S can be thought of as being defined by the following rules:

- (i) If $\nu \in N$ is of type $\varepsilon \rightarrow \sigma$ then the list consisting of the single component $\Gamma(\nu)$ is an element of E_σ .
- (ii) If $\nu \in N$ is of type $\sigma_1 \cdots \sigma_n \rightarrow \sigma$ with $n \geq 1$ and $e_j \in E_{\sigma_j}$ for each $j = 1, \dots, n$ then $\Gamma(\nu) e_1 \cdots e_n$ is an element of E_σ .
- (iii) The only elements in E_σ are those which can be obtained using (i) and (ii).

By Proposition 2.2.4 it is clear that each element of E_σ contains at least one component, i.e., $\varepsilon \notin E_\sigma$ for each $\sigma \in S$.

The simplest example of this construction is with $A = N$ and with $\Gamma : N \rightarrow N$ the identity mapping: The term Σ -algebra specified by this mapping is called the *standard term Σ -algebra*.

In the special case when (E_S, \square_N) is the standard term Σ -algebra then $E_\sigma \subset N^*$ for each $\sigma \in S$ and the family E_S is defined by the following rules:

- (i) If $\nu \in N$ is of type $\varepsilon \rightarrow \sigma$ then the list consisting of the single component ν is an element of E_σ .
- (ii) If $\nu \in N$ is of type $\sigma_1 \cdots \sigma_n \rightarrow \sigma$ with $n \geq 1$ and $e_j \in E_{\sigma_j}$ for each $j = 1, \dots, n$ then $\nu e_1 \cdots e_n$ is an element of E_σ .
- (iii) The only elements in E_σ are those which can be obtained using (i) and (ii).

Example 2.4.1 Consider the standard term Λ -algebra (E_B, \square_K) arising from the enumerated signature Λ in Example 2.2.1. Then:

E_{bool} consists of the two elements `True` and `False` of K^* .

E_{nat} consists of exactly the following elements of K^* :

`Zero`, `Succ Zero`, `Succ Succ Zero`, `Succ Succ Succ Zero`, \dots .

E_{int} consists of all elements of K^* having the form `n` with $n \in \mathbb{Z}$.

E_{ipr} consists of all elements of K^* of the form `Pair m n` with $m, n \in \mathbb{Z}$. (Each element of E_{ipr} thus has exactly three components.)

E_{ilst} consists of the element `Nil` plus all elements of K^* having the form `Cons n e` with $n \in \mathbb{Z}$ and $e \in E_{\text{ilst}}$. For instance,

`Cons 42 Cons -128 Cons 0 Cons -21 Nil`

is an element of E_{ilst} .

It turns out that the standard term Σ -algebra is initial. This is a special case of Proposition 2.4.1 below.

Let $\Gamma : N \rightarrow A$ be a term algebra specifier and (E_S, \square_N) be the term Σ -algebra specified by Γ . For each $\nu \in N$ let $\chi_\nu : \text{Dom}(\square_\nu) \times A^* \rightarrow A^*$ be defined by

$$\chi_\nu(v, \alpha) = \square_\nu(v) \alpha.$$

Lemma 2.4.1 *Suppose $\text{Im}(\chi_{\nu_1})$ and $\text{Im}(\chi_{\nu_2})$ are disjoint subsets of A^* whenever $\sigma \in S$ and $\nu_1, \nu_2 \in N_\sigma$ with $\nu_1 \neq \nu_2$. Then (E_S, \square_N) is initial.*

Proof By Proposition 2.3.2 it is enough to show that (E_S, \square_N) is unambiguous. But

$$\text{Im}(\square_\nu) = \chi_\nu(\text{Dom}(\square_\nu) \times \{\varepsilon\}) \subset \text{Im}(\chi_\nu)$$

for each $\nu \in N$; thus if $\nu_1, \nu_2 \in N_\sigma$ with $\nu_1 \neq \nu_2$ then $\text{Dom}(\square_{\nu_1})$ and $\text{Dom}(\square_{\nu_2})$ are clearly disjoint. It therefore remains to show that \square_ν is injective for each $\nu \in N$, and for this the following lemma is required:

Lemma 2.4.2 *Suppose the assumption in the statement of Lemma 2.4.1 is satisfied and let $\chi_\sigma : E_\sigma \times A^* \rightarrow A^*$ be defined by $\chi_\sigma(e, \alpha) = e \alpha$. Then for each $\sigma \in S$ the mapping χ_σ is injective.*

Proof For each $\sigma \in S$ define a subset G_σ of A^* by

$$G_\sigma = \{e \in E_\sigma : e \alpha \in E_\sigma \text{ for some } \alpha \in A^* \text{ with } \alpha \neq \varepsilon\}$$

and put $G = \bigcup_{\sigma \in S} G_\sigma$. Then, since χ_σ is injective if and only if $G_\sigma = \emptyset$, it follows that $G \neq \emptyset$ if and only if χ_σ is not injective for some $\sigma \in S$.

Suppose that $G \neq \emptyset$ and let $m = \min\{|\alpha| : \alpha \in G\}$, where $|\alpha|$ denotes the number of components in the list $\alpha \in A^*$. There thus exists $\sigma \in S$, $e \in E_\sigma$ and $\alpha \in A^* \setminus \{\varepsilon\}$ such that $|e| = m$ and $e' = e \alpha \in E_\sigma$. Now by Proposition 2.2.4 there exist $\nu, \nu' \in N_\sigma$ and $v \in \text{Dom}(\square_\nu)$, $v' \in \text{Dom}(\square_{\nu'})$ such that $e = \square_\nu(v)$ and $e' = \square_{\nu'}(v')$, and this implies that $\chi_\nu(v, \alpha) = e \alpha = e' = e' \varepsilon = \chi_{\nu'}(v', \varepsilon)$, which by assumption is only possible if $\nu' = \nu$. Let ν be of type $\sigma_1 \cdots \sigma_n \rightarrow \sigma$ and put $\Gamma(\nu) = \alpha'$. Thus if $v = (e_1, \dots, e_n)$ and $v' = (e'_1, \dots, e'_n)$ then $e = \alpha' e_1 \cdots e_n$ and $e \alpha = \alpha' e'_1 \cdots e'_n$. There must therefore exist $1 \leq j \leq n$ with $e_j \neq e'_j$ (and so in particular $n > 0$). Let i be the least index with $e_i \neq e'_i$ and let \hat{e} be the shorter of the lists e_i and e'_i . But then $\hat{e} \in G_{\sigma_i}$ and $|\hat{e}| < |e|$, which by the minimality of $|e|$ is not possible.

This shows that $G = \emptyset$ and hence that χ_σ is injective for each $\sigma \in S$. \square

The proof of Lemma 2.4.1 can now be completed. Let $\nu \in N$ be of type $\sigma_1 \cdots \sigma_n \rightarrow \sigma$ and let $v, v' \in \text{Dom}(\square_\nu)$ with $\square_\nu(v) = \square_\nu(v')$. This means that if $v = (e_1, \dots, e_n)$ and $v' = (e'_1, \dots, e'_n)$ then $\Gamma(\nu) e_1 \cdots e_n = \Gamma(\nu) e'_1 \cdots e'_n$. Consider $1 \leq j \leq n$ and assume $e_i = e'_i$ for each $i < j$. Then

$$\begin{aligned} \chi_{\sigma_j}(e_j, e_{j+1} \cdots e_n) &= e_j e_{j+1} \cdots e_n \\ &= e'_j e'_{j+1} \cdots e'_n = \chi_{\sigma_j}(e'_j, e'_{j+1} \cdots e'_n) \end{aligned}$$

and so by Lemma 2.4.2 $e_j = e'_j$. Therefore by n applications of Lemma 2.4.2 it follows that $e_j = e'_j$ for each $j = 1, \dots, n$; i.e., $v = v'$. This shows that \square_ν is injective. \square

Proposition 2.4.1 *Suppose that the restriction of Γ to N_σ is injective for each $\sigma \in S$. Then the term Σ -algebra (E_S, \square_N) specified by Γ is initial.*

Proof This follows immediately from the fact that $\Gamma(\nu)$ is the first component of each element of $\text{Im}(\chi_\nu)$, and hence the hypothesis of Lemma 2.4.1 is satisfied. \square

Proposition 2.4.2 *The standard term Σ -algebra is initial.*

Proof This follows from Proposition 2.4.1, since the identity mapping used to specify the standard term Σ -algebra is trivially injective. \square

It will now no longer be assumed that Σ is enumerated. In order to apply the above results to Σ the problem of replacing the general signature $\Sigma = (S, N, \Delta, \delta)$ with an 'equivalent' enumerated signature $\Sigma' = (S, N, \Delta', \delta)$ must be considered. In order to carry this out fix for each $\nu \in N$ an enumeration of the elements in the set $\Delta(\nu)$: More precisely, choose a bijective mapping i_ν from $[n_\nu]$ to the underlying set $\Delta(\nu)$, where n_ν is the cardinality of $\Delta(\nu)$. There is then a mapping $\Delta' : N \rightarrow S^*$ given by

$$\Delta'(\nu) = \langle i_\nu(1) \rangle \cdots \langle i_\nu(n_\nu) \rangle$$

for each $\nu \in N$. This defines an enumerated signature $\Sigma' = (S, N, \Delta', \delta)$, which will be called the *signature obtained from Σ and the family of enumerations i_N* .

Lemma 2.4.3 *Let X_S be a family of sets and let $\nu \in N$ with $\Delta'(\nu) = \sigma_1 \cdots \sigma_{n_\nu}$. Then the mapping $i_\nu^* : X_{\diamond}^{\Delta(\nu)} \rightarrow X_{\sigma_1} \times \cdots \times X_{\sigma_{n_\nu}}$ given by*

$$i_\nu^*(c) = (c(i_\nu(1)), \dots, c(i_\nu(n_\nu)))$$

for each $c \in X_{\diamond}^{\Delta(\nu)}$ is a bijection.

Proof This is clear. \square

Proposition 2.4.3 *Let (X_S, p'_N) be a Σ' -algebra and for each $\nu \in N$ put $p_\nu = p'_\nu i_\nu^*$. Then (X_S, p_N) is a Σ -algebra. Conversely, if (X_S, p_N) is any Σ -algebra then there exists a unique Σ' -algebra (X_S, p'_N) such that $p_\nu = p'_\nu i_\nu^*$ for each $\nu \in N$.*

Proof This follows immediately from Lemma 2.4.3. \square

If (X_S, p'_N) is a Σ' -algebra and $p_\nu = p'_\nu i_\nu^*$ for each $\nu \in N$ then (X_S, p_N) is called the Σ -algebra associated with (X_S, p'_N) and i_N .

Proposition 2.4.3 says that there is a one-to-one correspondence between Σ -algebras and Σ' -algebras. Moreover, Proposition 2.4.4 implies that a Σ' -algebra has a property (such as being minimal or initial) if and only if the corresponding Σ -algebra also has this property. In this sense the signatures Σ' and Σ can be regarded as being equivalent.

Proposition 2.4.4 *Let (X_S, p'_N) be a Σ' -algebra and let (X_S, p_N) be the Σ -algebra associated with (X_S, p'_N) and i_N . Then:*

- (1) (X_S, p_N) is minimal if and only if (X_S, p'_N) is a minimal Σ' -algebra.
- (2) (X_S, p_N) is initial if and only if (X_S, p'_N) is an initial Σ' -algebra.

Proof (1) This follows from the easily verified fact that a family \hat{X}_S is invariant in (X_S, p'_N) if and only if it is invariant in (X_S, p_N) .

(2) This follows from (1), Lemma 2.3.3 and Proposition 2.3.2. \square

Now let $\Gamma : N \rightarrow A$ be a mapping, which will still be referred to as a *term algebra specifier*. Then Γ is also a term algebra specifier for the signature Σ' , so let (E_S, \square'_N) be the term Σ' -algebra specified by Γ . For each $\nu \in N$ put $\square_\nu = \square'_\nu i_\nu^*$ (with i_ν^* given by Lemma 2.4.3). This means that $E_\sigma \subset A^*$ for each $\sigma \in S$ and if $\nu \in N$ is of type $L \rightarrow \sigma$ then $\square_\nu : E_\sigma^L \rightarrow E_\sigma$ is the mapping given by

$$\square_\nu(c) = \Gamma(\nu) \ c(i_\nu(1)) \ \cdots \ c(i_\nu(m))$$

for each $c \in E_\sigma^L$, where $m = |L|$ (the cardinality of L). Moreover, the family E_S can be regarded as being defined by the following rules:

- (i) If $\nu \in N$ is of type $\emptyset \rightarrow \sigma$ then the list consisting of the single component $\Gamma(\nu)$ is an element of E_σ .
- (ii) If $\nu \in N$ is of type $L \rightarrow \sigma$ with $L \neq \emptyset$ and $e_j \in E_{\sigma_j}$ for $j = 1, \dots, m$, where $\sigma_j = \langle i_\nu(j) \rangle$ and $m = |L|$, then $\Gamma(\nu) \ e_1 \ \cdots \ e_m$ is an element of E_σ .
- (iii) The only elements in E_σ are those which can be obtained using (i) and (ii).

The algebra (E_S, \square_N) is called the *term Σ -algebra specified by Γ and the family of enumerations i_N* , or simply the *standard term Σ -algebra defined by the family i_N* in the special case when $\Gamma : N \rightarrow N$ is the identity mapping.

Lemma 2.4.4 *The Σ -algebra (E_S, \square_N) is initial if and only if (E_S, \square'_N) is an initial Σ' -algebra.*

Proof This follows from Proposition 2.4.4 (2), since by definition (E_S, \square_N) is the Σ -algebra associated with (E_S, \square'_N) and i_N . \square

Proposition 2.4.5 *If the restriction of Γ to N_σ is injective for each $\sigma \in S$ then (E_S, \square_N) is an initial Σ -algebra. In particular, the standard term Σ -algebra defined by any family of enumerations is initial.*

Proof This follows immediately from Proposition 2.4.1 and Lemma 2.4.4. \square

Suppose that the term Σ -algebra (E_S, \square_N) specified by $\Gamma : N \rightarrow A$ and the family of enumeration i_N is initial. Then for each Σ -algebra (X_S, p_N) there exists a unique homomorphism from (E_S, \square_N) to (X_S, p_N) . This homomorphism $\llbracket \cdot \rrbracket_S$ is the unique family of mappings satisfying the following conditions:

- (i) If $\nu \in N$ is of type $\emptyset \rightarrow \sigma$ then $\llbracket \Gamma(\nu) \rrbracket_\sigma = p_\nu(\varepsilon)$.
- (ii) If $\nu \in N$ is of type $L \rightarrow \sigma$ with $L \neq \emptyset$ and $e_j \in E_{\sigma_j}$ for $j = 1, \dots, m$, where $\sigma_j = \langle i_\nu(j) \rangle$ and $m = |L|$, then

$$\llbracket \Gamma(\nu) \ e_1 \ \cdots \ e_m \rrbracket_\sigma = p_\nu(\llbracket c \rrbracket_\sigma^L),$$

where $c \in E_\sigma^L$ is the assignment with $c(i_\nu(j)) = e_j$ for each $j = 1, \dots, m$.

Of course, if the signature Σ is enumerated and the above constructions are applied with the family of identity enumerations i_N then the end-result is that nothing happens, since in this case $\Sigma' = \Sigma$.

Example 2.4.2 There exists a unique homomorphism $\llbracket \cdot \rrbracket_B$ from the standard term Λ -algebra (E_B, \square_K) to the Λ -algebra (X_B, p_K) which was introduced in Example 2.2.1. Thus $\llbracket \cdot \rrbracket_B$ is the unique family of mappings such that:

$$\begin{aligned} \llbracket \text{True} \rrbracket_{\text{bool}} &= T, & \llbracket \text{False} \rrbracket_{\text{bool}} &= F, \\ \llbracket \text{Zero} \rrbracket_{\text{nat}} &= 0, & \llbracket \text{Succ } e \rrbracket_{\text{nat}} &= \llbracket e \rrbracket_{\text{nat}} + 1 \text{ for each } e \in E_{\text{nat}}, \\ \llbracket n \rrbracket_{\text{int}} &= n \text{ for each } n \in \mathbb{Z}, \\ \llbracket \text{Pair } m \ n \rrbracket_{\text{ipr}} &= (m, n) \text{ for each } (m, n) \in \mathbb{Z} \times \mathbb{Z}, \\ \llbracket \text{Nil} \rrbracket_{\text{ilst}} &= \varepsilon, \\ \llbracket \text{Cons } n \ e \rrbracket_{\text{ilst}} &= n \triangleleft \llbracket e \rrbracket_{\text{ilst}} \text{ for all } n \in \mathbb{Z}, e \in E_{\text{ilst}}. \end{aligned}$$

Note that $\llbracket \cdot \rrbracket_B$ is in fact an isomorphism, since (X_B, p_K) is also initial.

Term algebras also provide, via Proposition 2.3.6, simple examples of free Σ -algebras: Let I be an S -typed set which is assumed to be disjoint from the alphabet A and let $\Sigma_I = (S, N_I, \Delta_I, \delta_I)$ be the signature introduced in Section 2.3. Extend Γ to a mapping $\Gamma_I : N_I \rightarrow A \cup I$ by letting $\Gamma_I(\diamond_\eta) = \eta$ for each $\eta \in I$, and let i_{N_I} be the only extension possible of i_N to a family of enumerations (i.e., with i_{\diamond_η} the unique mapping from \emptyset to \emptyset for each $\eta \in I$). Let $(E_S^I, \square_{N_I}^I)$ be the term Σ_I -algebra specified by Γ_I and i_{N_I} ; the algebra (E_S^I, \square_N^I) obtained by omitting the mappings $\{\square_{\diamond_\eta}^I : \eta \in I\}$ will then be called the *term Σ -algebra specified by Γ , i_N and I* . In particular, (E_S^I, \square_N^I) contains I , since $I \subset (A \cup I)^*$.

Proposition 2.4.6 *If the restriction of Γ to N_σ is injective for each $\sigma \in S$ then (E_S^I, \square_N^I) is an I -free Σ -algebra.*

Proof This follows immediately from Propositions 2.3.6 and 2.4.5, since the restriction of Γ_I to $(N_I)_\sigma$ is injective for each $\sigma \in S$. \square

2.5 Extensions of signatures and algebras

Let $\Sigma = (S, N, \Delta, \delta)$ be a signature. A signature $\Sigma' = (S', N', \Delta', \delta')$ is said to be an *extension* of Σ if $S \subset S'$, $N \subset N'$ and Δ (resp. δ) is the restriction of Δ' (resp. δ') to N . Thus if $\nu \in N$ is of type $L \rightarrow \sigma$ in the signature Σ then it is still of type $L \rightarrow \sigma$ in the extension Σ' . In particular, Σ is trivially an extension of itself.

In what follows let $\Sigma = (S, N, \Delta, \delta)$ and $\Sigma' = (S', N', \Delta', \delta')$ be signatures with Σ' an extension of Σ .

A Σ' -algebra $(Y_{S'}, q_{N'})$ is now called an *extension* of a Σ -algebra (X_S, p_N) if $X_\sigma \subset Y_\sigma$ for each $\sigma \in S$ and p_ν is the restriction of q_ν to $\text{Dom}(p_\nu)$ for each $\nu \in N$. (The case $\Sigma' = \Sigma$ is certainly not excluded here, and in fact it will occur frequently in what follows.)

Note that if $(Y_{S'}, q_{N'})$ is a Σ' -algebra then (Y_S, q_N) , obtained by omitting the sets $\{Y_\sigma : \sigma \in S' \setminus S\}$ from the family $Y_{S'}$ and the mappings $\{q_\nu : \nu \in N' \setminus N\}$ from the family $q_{N'}$, is a Σ -algebra. Moreover, if $\pi_{S'}$ is a Σ' -homomorphism from $(Y_{S'}, q_{N'})$ to $(Z_{S'}, r_{N'})$ then the family π_S , obtained by omitting the mappings $\{\pi_\sigma : \sigma \in S' \setminus S\}$ from the family $\pi_{S'}$, is a Σ -homomorphism from (Y_S, q_N) to (Z_S, r_N) .

The following is a variation on Proposition 2.3.3:

Proposition 2.5.1 *Let (X_S, p_N) be an initial Σ -algebra. Then there exists an initial Σ' -algebra which is an extension of (X_S, p_N) .*

Proof By Proposition 2.3.3 there exists an initial Σ' -algebra $(Y_{S'}, q_{N'})$ and (Y_S, q_N) (obtained by omitting the sets $\{Y_\sigma : \sigma \in S' \setminus S\}$ and the mappings $\{q_\sigma : \sigma \in N' \setminus N\}$) is then a Σ -algebra; let (\hat{Y}_S, \hat{q}_N) be the minimal subalgebra of (Y_S, q_N) . Now (Y_S, q_N) is unambiguous, since by Proposition 2.3.2 $(Y_{S'}, q_{N'})$ is unambiguous. Hence (\hat{Y}_S, \hat{q}_N) is an initial Σ -algebra, and by construction $(Y_{S'}, q_{N'})$ is an extension of (\hat{Y}_S, \hat{q}_N) . The unique isomorphism from (X_S, p_N) to (\hat{Y}_S, \hat{q}_N) can therefore be used to construct an initial Σ' -algebra which is an extension of (X_S, p_N) . \square

Proposition 2.5.2 *Let $(X_{S'}, p'_{N'})$ be an initial Σ' -algebra which is an extension of an initial Σ -algebra (X_S, p_N) . Moreover, let $(Y'_{S'}, q'_{N'})$ be any Σ' -algebra which is an extension of some Σ -algebra (Y_S, q_N) . Then the unique Σ' -homomorphism $\pi'_{S'}$ from $(X_{S'}, p'_{N'})$ to $(Y'_{S'}, q'_{N'})$ is an extension of the unique Σ -homomorphism π_S from (X_S, p_N) to (Y_S, q_N) , i.e., $\pi'_\sigma(x) = \pi_\sigma(x)$ for all $x \in X_\sigma$, $\sigma \in S$.*

Proof For each $\sigma \in S$ let $\pi'_\sigma : X_\sigma \rightarrow Y'_\sigma$ be the restriction of $\pi'_{S'}$ to X_σ , and consider π_σ as a mapping from X_σ to Y'_σ . Then π'_S and π_S are both Σ -homomorphisms from the initial Σ -algebra (X_S, p_N) to the Σ -algebra (Y'_S, q'_N) , and hence $\pi'_S = \pi_S$, i.e., $\pi'_\sigma(x) = \pi_\sigma(x)$ for all $x \in X_\sigma$, $\sigma \in S$. \square

Now let I be an S -typed set and again let $I_\sigma = \{\eta \in I : \langle \eta \rangle = \sigma\}$ for each $\sigma \in S$. A Σ -algebra (Z_S, r_N) is said to be *disjoint* from I if $\eta \notin Z_\eta$ for each $\eta \in I$, which is of course equivalent to having $I_\sigma \cap Z_\sigma = \emptyset$ for each $\sigma \in S$.

Lemma 2.5.1 *Let (X_S, p_N) be an I -free Σ -algebra which is an extension of an initial Σ -algebra (Z_S, r_N) . Then (Z_S, r_N) is disjoint from I .*

Proof By Proposition 2.3.7 (X_S, p_N) is I -ambiguous and hence $\text{Im}(p_\nu) \subset X_\sigma \setminus I_\sigma$ for all $\nu \in N_\sigma$, $\sigma \in S$. But $\text{Im}(r_\nu) \subset \text{Im}(p_\nu)$ for each $\nu \in N$, and so by Proposition 2.3.2 $Z_\sigma \subset X_\sigma \setminus I_\sigma$ for all $\sigma \in S$, i.e., $I_\sigma \cap Z_\sigma = \emptyset$ for each $\sigma \in S$. \square

In the following two results let I be an S -typed set and (Z_S, r_N) be an initial Σ -algebra disjoint from I . The following is a variation on Proposition 2.5.2:

Proposition 2.5.3 *There exists an I -free Σ -algebra (X_S, p_N) which is an extension of (Z_S, r_N) . Moreover, if (X'_S, p'_N) is any such I -free Σ -algebra and π_S is the unique Σ -isomorphism from (X_S, p_N) to (X'_S, p'_N) such that $\pi_\eta(\eta) = \eta$ for each $\eta \in I$ then $\pi_\sigma(z) = z$ for all $z \in Z_\sigma$, $\sigma \in S$.*

Proof By Proposition 2.3.6 there exists an I -free Σ -algebra (\bar{X}_S, \bar{p}_N) . Let (\bar{Z}_S, \bar{r}_N) be the minimal subalgebra of (\bar{X}_S, \bar{p}_N) ; then by (i) and (ii) in Proposition 2.3.8 (\bar{Z}_S, \bar{r}_N) is also unambiguous and therefore Proposition 2.3.2 implies that (\bar{Z}_S, \bar{r}_N) is an initial Σ -algebra. Moreover, by Lemma 2.5.1 (\bar{Z}_S, \bar{r}_N) is disjoint from I and by construction (\bar{X}_S, \bar{p}_N) is an extension of (\bar{Z}_S, \bar{r}_N) . Now unique isomorphism from (Z_S, r_N) to (\bar{Z}_S, \bar{r}_N) can be used to construct an I -free Σ -algebra (X_S, p_N) which is an extension of (Z_S, r_N) (and note that here the assumption is needed that (Z_S, r_N) is disjoint from I). Finally, consider any Σ -algebra (X'_S, p'_N) which is an extension of (Z_S, r_N) and let π_S be any Σ -homomorphism from (X_S, p_N) to (X'_S, p'_N) . Then $\pi_\sigma(z) = z$ for all $z \in Z_\sigma$, $\sigma \in S$, since the family Z'_σ defined by $Z'_\sigma = \{z \in Z_\sigma : \pi_\sigma(z) = z\}$ for each $\sigma \in S$ is clearly invariant in (Z_S, r_N) . \square

Now consider a Σ -algebra (Y_S, q_N) and an assignment $c \in Y_\sigma^I$. Then there is both the unique Σ -homomorphism π_S^c from (X_S, p_N) to (Y_S, q_N) with $\pi_\eta^c(\eta) = c(\eta)$ for each $\eta \in I$ and also the unique Σ -homomorphism π_S from (Z_S, r_N) to (Y_S, q_N) .

Proposition 2.5.4 $\pi_\sigma^c(z) = \pi_\sigma(z)$ for all $z \in Z_\sigma$, $\sigma \in S$.

Proof For each $\sigma \in S$ let π'_σ denote the restriction of π_σ^c to Z_σ . Then π'_S is a Σ -homomorphism from (Z_S, r_N) to (Y_S, q_N) and thus $\pi'_S = \pi_S$. \square

The final topic of this section is the construction of term algebras. Suppose in what follows that $\Gamma' : N' \rightarrow A'$ is a term algebra specifier which is an *extension* of a term algebra specifier $\Gamma : N \rightarrow A$ in the sense that $A \subset A'$ and $\Gamma(\nu) = \Gamma'(\nu)$ for each $\nu \in N$.

Lemma 2.5.2 *Assume the signatures Σ and Σ' are both enumerated. Then the term Σ' -algebra specified by Γ' is an extension of the term Σ -algebra (E_S, \square_N) specified by Γ .*

Proof This follows from Lemma 2.5.3 below, because the Σ' -algebra corresponding to the Σ -algebra (Y_S, q_N) in the definition of (E_S, \square_N) is an extension of (Y_S, q_N) . \square

Lemma 2.5.3 *Let $(Y_{S'}, q_{N'})$ be a Σ' -algebra which is an extension of a Σ -algebra (X_S, p_N) , and let \hat{X}_S be the minimal invariant family in (X_S, p_N) and $\hat{Y}_{S'}$ the minimal invariant family in $(Y_{S'}, q_{N'})$. Then $\hat{X}_S \subset \hat{Y}_{S'}$.*

Proof Let $\dot{Y}_{S'}$ be any invariant family in $(Y_{S'}, q_{N'})$ and put $Z_\sigma = \dot{Y}_\sigma \cap X_\sigma$ for each $\sigma \in S$. Then the family Z_S is invariant in (X_S, p_N) , since

$$p_\nu(Z_\sigma^L) \subset q_\nu(\dot{Y}_\sigma^L) \cap X_\sigma \subset \dot{Y}_\sigma \cap X_\sigma = Z_\sigma$$

for each $\nu \in N$ of type $L \rightarrow \sigma$. Thus $\hat{X}_S \subset Z_S$ and hence $\hat{X}_S \subset \dot{Y}_{S'}$. In particular, $\hat{X}_S \subset \hat{Y}_{S'}$. \square

Lemma 2.5.2 can now be extended to the case where it not assumed that Σ and Σ' are enumerated. Let $i'_{N'}$ be an enumeration for Σ' which is an *extension* of an enumeration i_N for Σ in that $i'_\nu = i_\nu$ for each $\nu \in N$.

Proposition 2.5.5 *The term Σ' -algebra specified by Γ' and $i'_{N'}$ is an extension of the term Σ -algebra specified by Γ and i_N .*

Proof This follows from Lemma 2.5.2 since the enumerated signature obtained from Σ' and the family of enumerations $i'_{N'}$ is an extension of that obtained from Σ and the family i_N . \square

In particular Proposition 2.5.5 implies that the standard term Σ' -algebra defined by the family of enumerations $i'_{N'}$ is an extension of the standard term Σ -algebra defined by the family i_N .

2.6 Tree algebras

It has been seen that an initial Σ -algebra can be obtained explicitly as an appropriate term algebra. In the present section a further explicit method of constructing initial Σ -algebras is given, this time involving labelled trees. The material presented here will not be needed in the following chapters and so this section can be omitted.

For the whole of the section let $\Sigma = (S, N, \Delta, \delta)$ be a signature. Let $J = \bigcup_{\nu \in N} \Delta(\nu)$ (meaning the union of the underlying sets, ignoring the associated typings), and note that if the signature Σ is enumerated then J is either $[n]$ for some $n \in \mathbb{N}$ or the set of positive integers \mathbb{N}_+ . In the construction given below an important role is played by certain subsets of the set of lists J^* and the section thus starts by looking at some simple facts about the subsets of X^* for an arbitrary set X .

Let $\triangleleft : X \times X^* \rightarrow X^*$ and $\triangleright : X^* \times X \rightarrow X^*$ be the mappings defined by

$$\begin{aligned} x \triangleleft x_1 \cdots x_m &= x x_1 \cdots x_m \\ x_1 \cdots x_m \triangleright x &= x_1 \cdots x_m x \end{aligned}$$

for all $x, x_1, \dots, x_m \in X$ (and note that \triangleleft and \triangleright are written as infix operations). The facts which are needed about these operations are exactly those listed in the following lemma:

- Lemma 2.6.1** (1) $x \triangleleft \varepsilon = \varepsilon \triangleright x$ for all $x \in X$.
(2) $x \triangleleft (s \triangleright x') = (x \triangleleft s) \triangleright x'$ for all $x, x' \in X, s \in X^*$.
(3) The mappings \triangleleft and \triangleright are both injective.
(4) $\text{Im}(\triangleleft) = \text{Im}(\triangleright) = X^* \setminus \{\varepsilon\}$.
(5) Let Z be a subset of X^* with $\varepsilon \in Z$ and such that either $x \triangleleft s \in Z$ for all $x \in X, s \in Z$, or $s \triangleright x \in Z$ for all $s \in Z, x \in X$. Then $Z = X^*$.
(6) There exists a unique mapping $|\cdot| : X^* \rightarrow \mathbb{N}$ with $|\varepsilon| = 0$ such that $|x \triangleleft s| = 1 + |s|$ for all $x \in X, s \in X^*$.

Proof Straightforward. Of course in (6) $|s|$ is just the number of components in the list s . \square

Lemma 2.6.2 Each finite subset A of X^* containing ε has a unique representation

$$A = \{\varepsilon\} \cup \bigcup_{x \in B} x \triangleleft A_x$$

with B a finite subset of X and with A_x a non-empty finite subset of X^* for each $x \in B$. Moreover, the sets $x \triangleleft A_x, x \in B$, and $\{\varepsilon\}$ are all disjoint.

Proof If $s \in X^* \setminus \{\varepsilon\}$ then by Lemma 2.6.1 (3) and (4) there exists a unique $x \in X$ and a unique element $s' \in X^*$ such that $s = x \triangleleft s'$. The unique representation is therefore with $B = \{x \in X : (x \triangleleft X^*) \cap A \neq \emptyset\}$ and with $A_x = \{s \in X^* : x \triangleleft s \in A\}$ for each $x \in B$. \square

A subset A of X^* is said to be \triangleright -preinvariant if $s \in A$ whenever $s \triangleright x \in A$ for some $x \in X$.

Lemma 2.6.3 *Each non-empty \triangleright -preinvariant subset of X^* contains ε .*

Proof Let A be a \triangleright -preinvariant subset of X^* and put $B = X^* \setminus A$; thus $s \triangleright x \in B$ for all $s \in B$, $x \in X$. But if $\varepsilon \in B$ then Lemma 2.6.1 (5) implies that $B = X^*$, and hence $\varepsilon \in A$ whenever $A \neq \emptyset$. \square

The set of all non-empty finite \triangleright -preinvariant subsets of X^* will be denoted by \mathcal{S} . By Lemma 2.6.3 each subset in \mathcal{S} contains the element ε .

Proposition 2.6.1 (1) $\{\varepsilon\} \in \mathcal{S}$ and \mathcal{S} is closed under finite unions.

(2) Let B be a finite subset of X and for each $x \in B$ let $A_x \in \mathcal{S}$. Then

$$\{\varepsilon\} \cup \bigcup_{x \in B} x \triangleleft A_x$$

is an element of \mathcal{S} .

(3) Let $A \in \mathcal{S}$ and let $\{\varepsilon\} \cup \bigcup_{x \in B} x \triangleleft A_x$ be the unique representation of A given in Lemma 2.6.2. Then $A_x \in \mathcal{S}$ for each $x \in B$.

Proof (1) This is clear.

(2) By (1) it is enough to consider the case when $B = \{x\}$, so let $A \in \mathcal{S}$, $x \in X$ and put $A' = \{\varepsilon\} \cup (x \triangleleft A)$. Suppose that $s \triangleright x' \in A'$ for some $s \in X^*$, $x' \in X$. Then, since by Lemma 2.6.1 (4) $s \triangleright x' \neq \varepsilon$, it follows that $s \triangleright x' = x \triangleleft s'$ for some $s' \in A$. The first possibility here is that $s' = \varepsilon$; but then by Lemma 2.6.1 (1)

$$s \triangleright x' = x \triangleleft \varepsilon = \varepsilon \triangleright x,$$

which by Lemma 2.6.1 (3) implies that $x' = x$ and $s = \varepsilon$, and in particular $s \in A'$. Assume next that $s' \neq \varepsilon$; then by Lemma 2.6.1 (4) $s' = \check{s} \triangleright \check{x}$ for some $\check{s} \in X^*$, $\check{x} \in X$, and here by Lemma 2.6.1 (2)

$$s \triangleright x' = x \triangleleft (\check{s} \triangleright \check{x}) = (x \triangleleft \check{s}) \triangleright \check{x},$$

which by Lemma 2.6.1 (3) implies that $x' = \check{x}$ and $s = x \triangleleft \check{s}$. But $\check{s} \in A$ (because A is \triangleright -preinvariant and $s' = \check{s} \triangleright \check{x} \in A$), and hence $s \in A'$. This shows that $A' \in \mathcal{S}$.

(3) Let $x \in B$ and suppose $s \triangleright x' \in A_x$ for some $s \in X^*$, $x' \in X$. Then $x \triangleleft s \in A$, since by Lemma 2.6.1 (2) $(x \triangleleft s) \triangleright x' = x \triangleleft (s \triangleright x') \in A$ and A is \triangleright -preinvariant, which by Lemma 2.6.1 (3) and (4) is only possible if $s \in A_x$. Hence $A_x \in \mathcal{S}$. \square

With these preparations made, the construction of an initial Σ -algebra can now begin. As at the beginning of the section put $J = \bigcup_{\nu \in N} \Delta(\nu)$. In what follows the set X in the above discussion will be taken to be J . In particular, this means that \mathcal{S} is now the set of all non-empty finite \triangleright -preinvariant subsets of J^* .

Denote by \mathcal{L} the set of all those partial mappings from J^* to N whose domains are elements of \mathcal{S} . For each $\sigma \in \mathcal{S}$ put $\mathcal{L}_\sigma = \mathcal{L}$ and if $\nu \in N$ is of type $L \rightarrow \sigma$ then define a mapping $\Pi_\nu : \mathcal{L}_\sigma^L \rightarrow \mathcal{L}_\sigma$ by letting $\Pi_\nu(c) = \lambda$, where

$$\text{Dom}(\lambda) = \{\varepsilon\} \cup \bigcup_{\eta \in L} \eta \triangleleft \text{Dom}(c(\eta))$$

with $\lambda(\varepsilon) = \nu$ and $\lambda(\eta \triangleleft s) = c(\eta)(s)$ for all $s \in \text{Dom}(c(\eta))$, $\eta \in L$ (and note that by Lemma 2.6.1 (3) and (4) and Proposition 2.6.1 (2) this definition makes sense). In particular, if ν is of type $\emptyset \rightarrow \sigma$ then $\Pi_\nu : \mathbb{I} \rightarrow \mathcal{L}_\sigma$ is the mapping with $\Pi_\nu(\varepsilon) = \lambda$, where $\text{Dom}(\lambda) = \{\varepsilon\}$ and $\lambda(\varepsilon) = \nu$. Thus (\mathcal{L}_S, Π_N) is a Σ -algebra.

Proposition 2.6.2 *The Σ -algebra (\mathcal{L}_S, Π_N) is unambiguous.*

Proof If $\nu \in N$ and $\lambda \in \text{Im}(\Pi_\nu)$ then $\varepsilon \in \text{Dom}(\lambda)$ and $\lambda(\varepsilon) = \nu$, and from this it immediately follows that $\text{Im}(\Pi_{\nu_1}) \cap \text{Im}(\Pi_{\nu_2}) = \emptyset$ whenever $\nu_1 \neq \nu_2$. It remains to show that Π_ν is injective for each $\nu \in N$, so assume ν of type $L \rightarrow \sigma$ and consider $c, c' \in \mathcal{L}_\sigma^L$ with $\Pi_\nu(c) = \Pi_\nu(c')$. Then

$$\{\varepsilon\} \cup \bigcup_{\eta \in L} \eta \triangleleft \text{Dom}(c(\eta)) = \text{Dom}(\lambda) = \{\varepsilon\} \cup \bigcup_{\eta \in L} \eta \triangleleft \text{Dom}(c'(\eta)),$$

where $\lambda = \Pi_\nu(c)$, and hence by Lemma 2.6.2 $\text{Dom}(c(\eta)) = \text{Dom}(c'(\eta))$ for each $\eta \in L$. Let $\eta \in L$ and $s \in \text{Dom}(c(\eta))$; then $c(\eta)(s) = \lambda(\eta \triangleleft s) = c'(\eta)(s)$, and therefore $c(\eta) = c'(\eta)$. Thus $c = c'$, and this shows that Π_ν is injective. \square

Now denote the minimal subalgebra of (\mathcal{L}_S, Π_N) by $(\mathcal{T}_S, \nabla_N)$; thus $\mathcal{T}_\sigma \subset \mathcal{L}_\sigma = \mathcal{L}$ for each $\sigma \in \mathcal{S}$, \mathcal{T}_S is the minimal invariant family and for each $\nu \in N$ of type $L \rightarrow \sigma$ the mapping ∇_ν is the restriction of Π_ν to \mathcal{T}_σ^L . The family \mathcal{T}_S can be thought of as being defined by the following rules:

- (i) If $\nu \in N$ is of type $\emptyset \rightarrow \sigma$ then the mapping λ with $\text{Dom}(\lambda) = \{\varepsilon\}$ and $\lambda(\varepsilon) = \nu$ is an element of \mathcal{T}_σ .
- (ii) If $\nu \in N$ is of type $L \rightarrow \sigma$ with $L \neq \emptyset$ and $c \in \mathcal{T}_\sigma^L$ is such that $c(\eta) \in \mathcal{T}_\eta$ for each $\eta \in L$ then the mapping λ with

$$\text{Dom}(\lambda) = \{\varepsilon\} \cup \bigcup_{\eta \in L} \eta \triangleleft \text{Dom}(c(\eta)),$$

$\lambda(\varepsilon) = \nu$ and $\lambda(\eta \triangleleft s) = c(\eta)(s)$ for all $s \in \text{Dom}(c(\eta))$, $\eta \in L$, is an element of \mathcal{T}_σ .

- (iii) The only elements in \mathcal{T}_σ are those which can be obtained using (i) and (ii).

Proposition 2.6.3 *$(\mathcal{T}_S, \nabla_N)$ is an initial Σ -algebra.*

Proof This follows immediately from Propositions 2.3.4 and 2.6.2. \square

Because of the explicit description of \mathcal{T}_S given in Proposition 2.6.3 below $(\mathcal{T}_S, \nabla_N)$ is called the *tree Σ -algebra*. An element $\lambda \in \mathcal{L}$ is said to be a *labelled tree* if whenever $s \in \text{Dom}(\lambda)$ with $\lambda(s) \in N$ of type $L \rightarrow \sigma$ then $s \triangleright \eta \in \text{Dom}(\lambda)$ if and only if $\eta \in L$ and, moreover, $\lambda(\eta \triangleright s) \in N_\eta$ for each $\eta \in L$. A labelled tree λ is then said to be of *type σ* if $\lambda(\varepsilon) \in N_\sigma$, and the set of all labelled trees of type σ will be denoted by $\check{\mathcal{T}}_\sigma$.

Proposition 2.6.3 $\dot{\mathcal{T}}_S = \mathcal{T}_S$.

Proof The crucial step in the proof is the following fact:

Lemma 2.6.4 For $\nu \in N$ of type $L \rightarrow \sigma$ let $\dot{\Pi}_\nu$ denote the restriction of Π_ν to $\dot{\mathcal{T}}_\sigma^L$ (as a mapping from $\dot{\mathcal{T}}_\sigma^L$ to \mathcal{T}_σ). Then $\bigcup_{\nu \in N_\sigma} \text{Im}(\dot{\Pi}_\nu) = \dot{\mathcal{T}}_\sigma$ for each $\sigma \in S$.

Proof It will be shown first that if $\nu \in N_\sigma$ is of type $L \rightarrow \sigma$ and $c \in \dot{\mathcal{T}}_\sigma^L$ then $\Pi_\nu(c) \in \dot{\mathcal{T}}_\sigma$, and for this it is enough to show that $\lambda = \Pi_\nu(c)$ is a labelled tree, since $\lambda(\varepsilon) = \nu \in N_\sigma$. Hence consider an element $s \in \text{Dom}(\lambda) = \{\varepsilon\} \cup \bigcup_{\eta \in L} \eta \triangleleft \text{Dom}(c(\eta))$.

The first possibility is that $s = \varepsilon$; in this case $\lambda(s) = \nu$ is of type $L \rightarrow \sigma$ and $s \triangleright \eta = \eta \triangleleft \varepsilon \in \text{Dom}(\lambda)$ if and only if $\eta \in L$, and if $\eta \in L$ then (since $c(\eta) \in \dot{\mathcal{T}}_\eta$)

$$\lambda(s \triangleright \eta) = \lambda(\eta \triangleleft \varepsilon) = c(\eta)(\varepsilon) \in N_\eta.$$

The other possibility is that $s = \eta \triangleleft s'$ with $\eta \in L$ and $s' \in \text{Dom}(c(\eta))$, and suppose here $\lambda(s) = c(\eta)(s')$ is of type $L' \rightarrow \sigma$. Then by Lemma 2.6.1 (2)

$$s \triangleright \xi = (\eta \triangleleft s') \triangleright \xi = \eta \triangleleft (s' \triangleright \xi)$$

and $s' \triangleright \xi \in \text{Dom}(c(\eta))$ if and only if $\xi \in L'$, since $c(\eta)$ is a labelled tree, which implies that $s \triangleright \xi \in \text{Dom}(\lambda)$ if and only if $\xi \in L'$. Moreover, if $\xi \in L'$ then

$$\lambda(s \triangleright \xi) = \lambda(\eta \triangleleft (s' \triangleright \xi)) = c(\eta)(s' \triangleright \xi)$$

and $c(\eta)(s' \triangleright \xi)$ is an element of N_ξ , again because $c(\eta)$ is a labelled tree. This shows that λ is a labelled tree.

Next consider $\sigma \in S$ and $\lambda \in \dot{\mathcal{T}}_\sigma$ with $\lambda(\varepsilon) = \nu \in N$ of type $L \rightarrow \sigma$; it will be shown that there exists $c \in \dot{\mathcal{T}}_\sigma^L$ such that $\lambda = \Pi_\nu(c)$. By Proposition 2.6.1 (3) and Lemma 2.6.2 $\text{Dom}(\lambda)$ has a unique representation of the form

$$\text{Dom}(\lambda) = \{\varepsilon\} \cup \bigcup_{\eta \in B} \eta \triangleleft A_\eta$$

with B a finite subset of J and $A_\eta \in \mathcal{S}$ for each $\eta \in B$. But (since λ is a labelled tree) $\varepsilon \triangleright \eta = \eta \triangleleft \varepsilon \in \text{Dom}(\lambda)$ if and only if $\eta \in L$, and by Lemma 2.6.3 $\varepsilon \in A_\eta$ for each $\eta \in B$; it therefore follows that $B = L$. For each $\eta \in L$ define $\lambda_\eta : A_\eta \rightarrow N$ by $\lambda_\eta(s) = \lambda(\eta \triangleleft s)$, so $\lambda_\eta \in \mathcal{L}$ and $\text{Dom}(\lambda_\eta) = A_\eta$. In fact λ_η is a labelled tree: If $s \in \text{Dom}(\lambda_\eta)$ then $\eta \triangleleft s \in \text{Dom}(\lambda)$ and $\lambda_\eta(s) = \lambda(\eta \triangleleft s)$; moreover, $s \triangleright \xi \in \text{Dom}(\lambda_\eta)$ if and only if $\eta \triangleleft (s \triangleright \xi) = (\eta \triangleleft s) \triangleright \xi \in \text{Dom}(\lambda)$, and $\lambda_\eta(s \triangleright \xi) = \lambda((\eta \triangleleft s) \triangleright \xi)$. Hence $\lambda_\eta \in \dot{\mathcal{T}}_\eta$, since $\lambda_\eta(\varepsilon) = \lambda(\eta \triangleleft \varepsilon) = \lambda(\varepsilon \triangleright \eta) \in N_\eta$. An assignment $c \in \dot{\mathcal{T}}_\sigma^L$ can thus be defined by letting $c(\eta) = \lambda_\eta$ for each $\eta \in L$, and by construction $\lambda = \Pi_\nu(c)$.

Putting these two parts of the proof together now shows that $\bigcup_{\nu \in N_\sigma} \text{Im}(\dot{\Pi}_\nu) = \dot{\mathcal{T}}_\sigma$ for each $\sigma \in S$. \square

Lemma 2.6.4 implies in particular that the family $\check{\mathcal{T}}_S$ is invariant in $(\mathcal{T}_S, \nabla_N)$, so consider the subalgebra $(\check{\mathcal{T}}_S, \check{\nabla}_N)$ associated with $\check{\mathcal{T}}_S$. Now clearly $\check{\mathcal{T}}_S = \mathcal{T}_S$ will hold if and only if $(\check{\mathcal{T}}_S, \check{\nabla}_N)$ is minimal, and to establish this Proposition 2.2.5 can be used. The first requirement in Proposition 2.2.5, namely that $\bigcup_{\nu \in N_\sigma} \text{Im}(\check{\nabla}_\nu) = \check{\mathcal{T}}_\sigma$ for each $\sigma \in S$, is just a restatement of Lemma 2.6.4, and so it remains to satisfy the second requirement, which means a suitable family of mappings $\#_S$ must be found. Let $\# : \mathcal{L} \rightarrow \mathbb{N}$ be given by

$$\#(\lambda) = \max\{|s| : s \in \text{Dom}(\lambda)\},$$

where $|\cdot| : J^* \rightarrow \mathbb{N}$ is the mapping given in Lemma 2.6.1 (6), and for each $\sigma \in S$ put $\#_\sigma = \#$. Then the requirement in Proposition 2.2.5 is satisfied, since if $\nu \in N$ is of type $L \rightarrow \sigma$ then for all $c \in \mathcal{L}_\diamond^L$

$$\#_\sigma(\Pi_\nu(c)) = 1 + \max\{\#_\eta(c(\eta)) : \eta \in L\} > \max\{\#_\eta(c(\eta)) : \eta \in L\}.$$

Therefore $(\check{\mathcal{T}}_S, \check{\nabla}_N)$ is minimal. \square

2.7 Notes

The material presented in this chapter is all part of standard universal algebra. The classical field of universal algebra deals with the case of signatures having a single type, and so the only information required about each operator name is the number of arguments it takes. In this form the main problems were stated in Whitehead (1898) and solved in Birkhoff (1935); standard texts are Cohn (1965) and Grätzer (1968). (The term ‘universal algebra’ was coined by that great mathematical name-giver Sylvester.) The generalisation of the theory to multi-sorted algebras (i.e., to the algebras as they occur here) was made by Higgins (1963) and Birkhoff and Lipson (1970). Birkhoff and Lipson (who speak of heterogeneous algebras) showed that essentially the whole of the classical theory carries over to the more general case.

Some of the first uses of multi-sorted algebras in computer science can be found in Maibaum (1972) and Morris (1973). Their systematic use has been propagated by the ADJ group consisting of J.A. Goguen, J.W. Thatcher, E.G. Wagner and J.B. Wright, for example in the papers Goguen, Thatcher, Wagner and Wright (1977) and Goguen, Thatcher and Wagner (1978).

Minimal algebras are sometimes called *reachable* (Wirsing (1990)) or *term generated* (Bauer, Berghammer *et al* (1985)). This terminology comes from the fact that an algebra is minimal if and only if the unique homomorphism to this algebra from an initial algebra (and so in particular from the standard term algebra) is surjective. Pervasive signatures are called *sensible* in Huet and Oppen (1980).

The characterisation of initial algebras in Proposition 2.3.2 is sometimes referred to as stating that initial algebras are exactly those for which there is *no junk* and *no confusion* (i.e., those which are minimal and unambiguous). Proposition 2.4.2 is just a version of the classical result of Łukasiewicz (Łukasiewicz (1925)) concerning prefix or (left Polish) notation.

Bourbaki (1970) thinks that the word *algebra* is over-used and suggests the alternative *magma* for an algebra in the above sense. This suggestion has been followed by some French computer scientists, but it has of course no chance of being accepted in the English-speaking world. (What have algebras to do with volcanic lava?)

Chapter 3 Data objects

With the preparations made in Chapter 2 a start can now be made to building a framework for specifying data objects. This involves what will be called a bottomed extension of an initial algebra. The initial algebra is used to describe the basic data objects, and the bottomed extension then specifies which ‘undefined’ and ‘partially defined’ data objects are to be allowed. Bottom extensions are defined in Section 3.2. In Sections 3.3 and 3.4 two conditions, called regularity and monotonicity, are introduced which the extensions have to satisfy. Finally, in Section 3.5 cored extensions are introduced. These provide simple explicit examples of monotone regular extensions.

3.1 The basic data objects and the ground term algebra

The starting point is to choose a signature $\Lambda = (B, K, \Theta, \vartheta)$ and an initial Λ -algebra (X_B, p_K) . The basic data objects are then the elements in the family of sets X_B . Here Λ is called the *ground signature*, B the set of *ground types* and K the set of *constructor names*. In real applications the set B of ground types will be finite.

In all modern functional programming languages the signature Λ is not fixed and can be chosen by the user to fit the application at hand. Usually there are a few built-in types such as the types `int` and `bool` in Example 2.2.1 together with a primitive type `char` having 256 or more constructor names. (Recall that a type β is said to be primitive if κ is of type $\varnothing \rightarrow \beta$ for each $\kappa \in K_\beta$.) Once the signature Λ has been chosen the initial Λ -algebra (X_B, p_K) is uniquely determined up to isomorphism. The explicit choice of (X_B, p_K) can be seen as taking place in the user’s mind.

Bearing in mind practical applications it will be assumed that B contains the two primitive types `int` and `bool` as in Example 2.2.1. Thus $K_{\text{int}} = \underline{Int}$ (with \underline{Int} the subset of $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, -\}^*$ containing for each integer n its standard representation \underline{n} as a string of characters), $K_{\text{bool}} = \{\text{True}, \text{False}\}$ and each of these constructor names is of type $\varnothing \rightarrow \beta$ with β either `int` or `bool`. Moreover, X_{int} will always be taken to be \mathbb{Z} with $p_{\underline{n}}(\varepsilon) = n$ for each $n \in \mathbb{Z}$ and X_{bool} to be \mathbb{B} with $p_{\text{True}}(\varepsilon) = T$ and $p_{\text{False}}(\varepsilon) = F$.

To avoid some trivial complications later it is also convenient to suppose that Λ is pervasive. (Recall this means B is the only closed subset of itself, where a subset B' of B is closed if $\beta \in B'$ whenever there exists $\kappa \in K$ of type $L \rightarrow \beta$ for some L and $\langle \eta \rangle \in B'$ for each $\eta \in L$.) By Proposition 2.2.4 this is equivalent to requiring that $X_\beta \neq \varnothing$ for each $\beta \in B$. It is not assumed that the sets K_β , $\beta \in B$, are finite (since in fact K_{int} is infinite), although it will almost always be the case that K_β is finite whenever β is not a primitive type.

In all that follows the signature Λ and the initial Λ -algebra (X_B, p_K) are considered to be fixed.

As well as the initial Λ -algebra (X_B, p_K) it will also be assumed that a second initial Λ -algebra is given. This Σ -algebra (F_B, \odot_K) is called the *ground term algebra* and

is needed in order to be able to refer to the basic data objects: Since (X_B, p_K) and (F_B, \odot_K) are both initial Λ -algebras there exists a unique isomorphism from (F_B, \odot_K) to (X_B, p_K) which will be denoted by $\llbracket \cdot \rrbracket_B$. This means that each basic data object can be denoted by a unique element in the ground term algebra, i.e., the data object $x \in X_\beta$ can be denoted by the unique element $s \in F_\beta$ with $\llbracket s \rrbracket_\beta = x$.

Of course, (F_B, \odot_K) could just be taken to be (X_B, p_K) but, as its name suggests, it should be thought of rather as some kind of term algebra (as defined in Section 2.4). In fact, such a term algebra will now be introduced which will later be used as the basis for a programming language. This is done by choosing a family of enumerations i_K for the signature Λ , i.e., by choosing for each $\kappa \in K$ of type $L \rightarrow \beta$ a bijective mapping i_κ from $[m]$ to the set L , where $m = |L|$ is the cardinality of L . In the special case when $L = \beta_1 \cdots \beta_m \in B^*$ (and so $[m]$ is the underlying set of the B -typed set L) then $i_\kappa : [m] \rightarrow [m]$ will always be chosen to be the identity mapping. Let (E_B, \square_K) be the standard term Λ -algebra defined by the family i_K . Then by Proposition 2.4.5 (E_B, \square_K) is an initial Λ -algebra. Recall that $E_\beta \subset K^*$ for each $\kappa \in B$ and if $\kappa \in K$ is of type $L \rightarrow \beta$ then $\square_\kappa : E_\circ^L \rightarrow E_\beta$ is the mapping given by

$$\square_\kappa(c) = \kappa \ c(i_\kappa(1)) \ \cdots \ c(i_\kappa(m))$$

for each $c \in E_\circ^L$, where $m = |L|$. Moreover, the family E_B can be regarded as being defined by the following rules:

- (i) If $\kappa \in K$ is of type $\emptyset \rightarrow \beta$ then the list consisting of the single component κ is an element of E_β .
- (ii) If $\kappa \in K$ is of type $L \rightarrow \beta$ with $L \neq \emptyset$ and $e_j \in E_{\beta_j}$ for $j = 1, \dots, m$, where $\beta_j = \langle i_\kappa(j) \rangle$ and $m = |L|$, then $\kappa \ e_1 \ \cdots \ e_m$ is an element of E_β .
- (iii) The only elements in E_β are those which can be obtained using (i) and (ii).

In this special case the isomorphism $\llbracket \cdot \rrbracket_B$ from (E_B, \square_K) to (X_B, p_K) is the unique family of mappings satisfying:

- (i) If $\kappa \in K$ is of type $\emptyset \rightarrow \beta$ then $\llbracket \kappa \rrbracket_\beta = p_\kappa(\varepsilon)$.
- (ii) If $\kappa \in K$ is of type $L \rightarrow \beta$ with $L \neq \emptyset$ and $e_j \in E_{\beta_j}$ for $j = 1, \dots, m$, where $\sigma_j = \langle i_\kappa(j) \rangle$ and $m = |L|$, then

$$\llbracket \kappa \ e_1 \ \cdots \ e_m \rrbracket_\sigma = p_\kappa(\llbracket c \rrbracket_\circ^L),$$

where $c \in E_\circ^L$ is the assignment with $c(i_\kappa(j)) = e_j$ for each $j = 1, \dots, m$.

3.2 Bottomed algebras and bottomed extensions

The basic data objects were described in the previous section with the help of the initial Λ -algebra (X_B, p_K) . The next step is to introduce for each $\beta \in B$ a new element $\perp_\beta \notin X_\beta$ to play the role of an ‘undefined’ or ‘unknown’ object of type β . (The symbol \perp is called *bottom*.) Thus for each $\beta \in B$ there is now an ‘extended’ set $X_\beta \cup \{\perp_\beta\}$ of data objects of type β made up of the ‘real’ objects (i.e., the elements of the set X_β) together with the ‘undefined’ object \perp_β of type β . There are, however, good reasons to go at least one stage further and to also allow ‘partially defined’ data objects. For instance, considering the Λ -algebra introduced in Example 2.2.1, if $n \in \mathbb{Z} = X_{\text{int}}$ then the pair (n, \perp_{int}) could be regarded as a data object of type ipr which is not completely ‘undefined’ and so which should perhaps be different from \perp_{ipr} . (After all, it makes sense to consider the first component of (n, \perp_{int}) .) Similarly, it is often useful to allow ‘partially defined’ data objects of type ilst . (For example, the data object described as a list containing at least one component, with the first component having some specific value.)

In order to facilitate the systematic treatment of ‘partially defined’ data objects the following definition will be made: A Λ -algebra (D_B, f_K) is called a *bottomed extension* of (X_B, p_K) if (D_B, f_K) is an extension of (X_B, p_K) (i.e., $X_B \subset D_B$ and p_κ is the restriction of f_κ to $\text{Dom}(p_\kappa)$ for each $\kappa \in K$) such that $\perp_\beta \in D_\beta$ for each $\beta \in B$.

The elements in the set $D_\beta \setminus (X_\beta \cup \{\perp_\beta\})$ should be thought of for the moment as being the ‘partially defined’ data objects of type β . Later the possibility of having not only ‘partially defined’ but also ‘infinite’ data objects will be considered. This more general situation can still be described with the help of a bottomed extension. The elements in the set $D_\beta \setminus (X_\beta \cup \{\perp_\beta\})$ should then, however, be thought of as containing both the ‘partially defined’ and ‘infinite’ data objects of type β .

The simplest example of a bottomed extension is obtained by putting $D_\beta = X_\beta \cup \{\perp_\beta\}$ for each $\beta \in B$ and for each $\kappa \in K$ of type $L \rightarrow \beta$ letting $f_\kappa : D_\diamond^L \rightarrow D_\beta$ be given by

$$f_\kappa(c) = \begin{cases} p_\kappa(c) & \text{if } c \in X_\diamond^L, \\ \perp_\beta & \text{otherwise.} \end{cases}$$

Then (D_B, f_K) is a bottomed extension called the *flat extension* of (X_B, p_K) . (The flat extension of the Λ -algebra defined in Example 2.2.1 is given in Example 3.2.1 on the next page.)

The above definition of a bottomed extension will be reformulated somewhat, and for this it is useful to introduce the following simple notion: A *bottomed set* is a pair (D, \perp) consisting of a set D and a distinguished *bottom element* $\perp \in D$. The bottom element \perp should be thought of as representing an ‘undefined’ or ‘unknown’ value which, if it were to be ‘defined’ or ‘become known’, would take on the value of one of the remaining elements of D .

If (D, \perp) and (D', \perp') are bottomed sets then a mapping $f : D \rightarrow D'$ is said to be *bottomed* (or *strict*) if $f(\perp) = \perp'$.

If (D, \perp) is a bottomed set then it is usual to write just D instead of (D, \perp) and to assume that the bottom element \perp can be inferred from the context.

Example 3.2.1 The flat extension of the Λ -algebra (X_B, p_K) first introduced in Example 2.2.1 is the Λ -algebra (X_B^\perp, p_K^\perp) defined as follows:

$$X_{\text{bool}}^\perp = \mathbb{B}^\perp = \{T, F, \perp_{\text{bool}}\}, \quad X_{\text{nat}}^\perp = \mathbb{N} \cup \{\perp_{\text{nat}}\}, \\ X_{\text{int}}^\perp = \mathbb{Z} \cup \{\perp_{\text{int}}\}, \quad X_{\text{ipr}}^\perp = \mathbb{Z}^2 \cup \{\perp_{\text{ipr}}\}, \quad X_{\text{ilst}}^\perp = \mathbb{Z}^* \cup \{\perp_{\text{ilst}}\}.$$

$$p_{\text{True}}^\perp : \mathbb{I} \rightarrow X_{\text{bool}}^\perp \text{ with } p_{\text{True}}^\perp(\varepsilon) = T,$$

$$p_{\text{False}}^\perp : \mathbb{I} \rightarrow X_{\text{bool}}^\perp \text{ with } p_{\text{False}}^\perp(\varepsilon) = F,$$

$$p_{\text{Zero}}^\perp : \mathbb{I} \rightarrow X_{\text{nat}}^\perp \text{ with } p_{\text{Zero}}^\perp(\varepsilon) = 0,$$

$$p_{\text{Succ}}^\perp : X_{\text{nat}}^\perp \rightarrow X_{\text{nat}}^\perp \text{ with}$$

$$p_{\text{Succ}}^\perp(n) = \begin{cases} n + 1 & \text{if } n \in \mathbb{N}, \\ \perp_{\text{nat}} & \text{if } n = \perp_{\text{nat}}, \end{cases}$$

$$p_{\underline{n}}^\perp : \mathbb{I} \rightarrow X_{\text{int}}^\perp \text{ with } p_{\underline{n}}^\perp(\varepsilon) = n \text{ for each } n \in \mathbb{Z},$$

$$p_{\text{Pair}}^\perp : X_{\text{int}}^\perp \times X_{\text{int}}^\perp \rightarrow X_{\text{ipr}}^\perp \text{ with}$$

$$p_{\text{Pair}}^\perp(m, n) = \begin{cases} (m, n) & \text{if } m, n \in \mathbb{Z}, \\ \perp_{\text{ipr}} & \text{otherwise,} \end{cases}$$

$$p_{\text{Nil}}^\perp : \mathbb{I} \rightarrow X_{\text{ilst}}^\perp \text{ with } p_{\text{Nil}}^\perp(\varepsilon) = \varepsilon,$$

$$p_{\text{Cons}}^\perp : X_{\text{int}}^\perp \times X_{\text{ilst}}^\perp \rightarrow X_{\text{ilst}}^\perp \text{ with}$$

$$p_{\text{Cons}}^\perp(n, s) = \begin{cases} n \triangleleft s & \text{if } n \in \mathbb{Z} \text{ and } s \in \mathbb{Z}^*, \\ \perp_{\text{ilst}} & \text{otherwise.} \end{cases}$$

Let D_B be a family of bottomed sets with \perp_β the bottom element of D_β for each $\beta \in B$, and let I be a B -typed set. Then D_\diamond^I will be considered as a bottomed set by stipulating that its bottom element \perp be the assignment given by $\perp(\eta) = \perp_\eta$ for each $\eta \in I$. A special case is that of a cartesian product: Let $n \geq 2$ and for each $j = 1, \dots, n$ let D_j be a bottomed set with bottom element \perp_j ; then the product $D_1 \times \dots \times D_n$ is considered as a bottomed set with bottom element $(\perp_1, \dots, \perp_n)$.

A Λ -algebra (D_B, f_K) is said to be *bottomed* if D_β is a bottomed set for each $\beta \in B$. If (D_B, f_K) is a bottomed Λ -algebra then the bottom element of D_β will always be denoted by \perp_β . This means that if (D'_B, f'_K) is a further bottomed Λ -algebra then the bottom element of D'_β will also be denoted by \perp_β , although it is not to be assumed that the bottom elements of D_β and D'_β are the same.

The definition of a bottomed extension of (X_B, p_K) can now be expressed as follows: A *bottomed extension* of (X_B, p_K) is a bottomed Λ -algebra (D_B, f_K) which is an

extension of (X_B, p_K) such that $\perp_\beta \notin X_\beta$ for each $\beta \in B$.

If (D_B, f_K) and (D'_B, f'_K) are bottomed Λ -algebras then a homomorphism π_B from (D_B, f_K) to (D'_B, f'_K) is said to be *bottomed* if π_β is a bottomed mapping (i.e., if $\pi_\beta(\perp_\beta) = \perp_\beta$) for each $\beta \in B$.

If (D_B, f_K) and (D'_B, f'_K) are bottomed extensions of (X_B, p_K) then a bottomed homomorphism $\pi_B : (D_B, f_K) \rightarrow (D'_B, f'_K)$ is said to be an *extension homomorphism* if $\pi_\beta(x) = x$ for all $x \in X_\beta$, $\beta \in B$. In fact, the following simple result implies that a bottomed homomorphism is automatically an extension homomorphism:

Lemma 3.2.1 *Let (Z_B, r_K) and (Z'_B, r'_K) be extensions of a minimal Λ -algebra (Y_B, q_K) and $\pi_B : (Z_B, r_K) \rightarrow (Z'_B, r'_K)$ be a homomorphism. Then $\pi_\beta(y) = y$ for all $y \in Y_\beta$, $\beta \in B$.*

Proof The restriction of π_B to Y_B and the family of identity mappings id_B both define homomorphisms from (Y_B, q_K) to (Z'_B, r'_K) , and so by Proposition 2.2.3 (1) they must be equal, i.e., $\pi_\beta(y) = y$ for all $y \in Y_\beta$, $\beta \in B$. \square

If (D_B, f_K) is a bottomed Λ -algebra then the family of identity mappings id_B defines a bottomed homomorphism from (D_B, f_K) to itself. Also by Proposition 2.2.1 the composition of two bottomed homomorphisms is again a bottomed homomorphism. Moreover, if $\pi_B : (D_B, f_K) \rightarrow (D'_B, f'_K)$ is a bottomed homomorphism such that π_β is a bijection for each $\beta \in B$ then by Proposition 2.2.2 $\pi_B^{-1} : (D'_B, f'_K) \rightarrow (D_B, f_K)$ is again a bottomed homomorphism.

A bottomed homomorphism $\pi_B : (D_B, f_K) \rightarrow (D'_B, f'_K)$ is a *bottomed isomorphism* if the mapping $\pi_\beta : D_\beta \rightarrow D'_\beta$ is a bijection for each $\beta \in B$; (D_B, f_K) and (D'_B, f'_K) are then *isomorphic* if there exists a bottomed isomorphism from (D_B, f_K) to (D'_B, f'_K) , and the property of being isomorphic defines an equivalence relation on the class of all bottomed Λ -algebras.

Bottomed extensions (D_B, f_K) and (D'_B, f'_K) of (X_B, p_K) are said to be *conjugate* if there exists an extension homomorphism from (D_B, f_K) to (D'_B, f'_K) which is a bottomed isomorphism. Of course, Lemma 3.2.1 implies that (D_B, f_K) and (D'_B, f'_K) are conjugate if and only if they are isomorphic as bottomed Λ -algebras.

Some elementary results about bottomed Λ -algebras will now be collected together and then applied to the case of a bottomed extension of (X_B, p_K) . In what follows let (D_B, f_K) be a bottomed Λ -algebra. A family \check{D}_B with $\check{D}_B \subset D_B$ is said to be *bottomed* if $\perp_\beta \in \check{D}_\beta$ for each $\beta \in B$. There is clearly a one-to-one correspondence between bottomed invariant families and bottomed subalgebras of (D_B, f_K) (i.e., those subalgebras which are also bottomed Λ -algebras).

Lemma 3.2.2 *The bottomed family \check{D}_B defined by $\check{D}_\beta = \{\perp_\beta\} \cup \bigcup_{\kappa \in K_\beta} \text{Im}(f_\kappa)$ for each $\beta \in B$ is invariant.*

Proof This is a special case of Lemma 2.3.2. \square

Lemma 3.2.3 *There exists a minimal bottomed invariant family.*

Proof This is a special case of Lemma 2.2.3. \square

The subalgebra associated with the minimal bottomed invariant family will be referred to as the *minimal bottomed subalgebra* of (D_B, f_K) .

If D_B is the only bottomed invariant family then the bottomed Λ -algebra (D_B, f_K) is said to be *minimal*. Note that if (D_B, f_K) is a minimal Λ -algebra (as defined in Section 2.2) then it is also a minimal bottomed Λ -algebra, but of course in general the converse does not hold. In fact the minimal bottomed Λ -algebras occurring here will never be minimal Λ -algebras. The statement that a bottomed Λ -algebra is minimal will thus mean that it is a minimal bottomed Λ -algebra; minimality as defined in Section 2.2 will be referred to by stating explicitly that the algebra is a minimal Λ -algebra.

It is easy to see that the flat extension of (X_B, p_K) is a minimal bottomed Λ -algebra (since (X_B, p_K) is a minimal Λ -algebra).

Proposition 3.2.1 *Let (D'_B, f'_K) be a bottomed Λ -algebra.*

(1) *There exists at most one bottomed homomorphism from (D_B, f_K) to (D'_B, f'_K) when (D_B, f_K) is minimal.*

(2) *Any bottomed homomorphism from (D_B, f_K) to (D'_B, f'_K) is surjective when (D'_B, f'_K) is minimal.*

Proof This is the same as the proof of Proposition 2.2.4. \square

Proposition 3.2.2 *If (D_B, f_K) is minimal then $D_\beta = \{\perp_\beta\} \cup \bigcup_{\kappa \in K_\beta} \text{Im}(f_\kappa)$ for each $\beta \in B$.*

Proof This follows immediately from Lemma 3.2.2. \square

Proposition 3.2.3 *Suppose that $D_\beta = \{\perp_\beta\} \cup \bigcup_{\kappa \in K_\beta} \text{Im}(f_\kappa)$ for each $\beta \in B$ and there exists a family of mappings $\#_B$ with $\#_\beta : D_\beta \rightarrow \mathbb{N}$ for each β such that if $\kappa \in K$ is of type $L \rightarrow \beta$ then for all $c \in D_\beta^L$, $\eta \in L$*

$$\#_\eta(c(\eta)) < \#_\beta(f_\kappa(c)).$$

Then (D_B, f_K) is a minimal bottomed Λ -algebra.

Proof This is just a special case of Proposition 2.2.5. \square

As expected, (D_B, f_K) is said to be an *initial bottomed Λ -algebra* if for each bottomed Λ -algebra (D'_B, f'_K) there exists a unique bottomed homomorphism from (D_B, f_K) to (D'_B, f'_K) . The adjective ‘initial’ will be used in the same way as ‘minimal’: The statement that a bottomed Λ -algebra is initial thus means that it is an initial bottomed Λ -algebra, and initiality as defined in Section 2.3 will be referred to by stating explicitly that the algebra is an initial Λ -algebra.

Lemma 3.2.4 *Suppose the bottom elements \perp_β , $\beta \in B$, are all distinct, and let J be the B -typed set consisting of these elements with \perp_β of type β for each $\beta \in B$. Then (D_B, f_K) is an initial bottomed Λ -algebra if and only if it is J -free.*

Proof This is more-or-less clear. (Note that if (Y_B, q_K) is any Λ -algebra and $c \in Y_\diamond^J$ then (Y_B, q_K) can be considered as a bottomed Λ -algebra by declaring $c(\perp_\beta)$ to be the bottom element of Y_β for each $\beta \in B$.) \square

Proposition 3.2.4 *There exists an initial bottomed Λ -algebra, and any two initial bottomed Λ -algebras are isomorphic.*

Proof The existence follows from Lemma 3.2.4 and Proposition 2.3.6. The rest then follows directly from the definition of being initial. \square

Consider the construction given in the proof of Proposition 2.3.6, tailored somewhat to the special situation occurring here. For each $\beta \in B$ let b_β be some element not in K and such that $b_{\beta_1} \neq b_{\beta_2}$ whenever $\beta_1 \neq \beta_2$. Put $K^b = K \cup \{b_\beta : \beta \in B\}$ and define mappings $\Theta^b : K^b \rightarrow \mathcal{F}_B$ and $\vartheta^b : K^b \rightarrow B$ by

$$\Theta^b(\kappa) = \begin{cases} \Theta(\kappa) & \text{if } \kappa \in K, \\ \emptyset & \text{if } \kappa = b_\beta \text{ for some } \beta \in B, \end{cases}$$

$$\vartheta^b(\kappa) = \begin{cases} \vartheta(\kappa) & \text{if } \kappa \in K, \\ \beta & \text{if } \kappa = b_\beta \text{ for some } \beta \in B. \end{cases}$$

Then $\Lambda^b = (B, K^b, \Theta^b, \vartheta^b)$ is a signature which is clearly an extension of Λ and in which b_β is of type $\emptyset \rightarrow \beta$ for each $\beta \in B$.

Now by Proposition 2.3.3 there exists an initial Λ^b -algebra $(\check{D}_B, \check{f}_{K^b})$. The Λ -algebra $(\check{D}_B, \check{f}_K)$ (obtained by omitting the mappings \check{f}_{b_β} , $\beta \in B$) can then be considered as a bottomed Λ -algebra by declaring $\check{f}_{b_\beta}(\varepsilon)$ to be the bottom element of \check{D}_β for each $\beta \in B$.

Proposition 3.2.5 *$(\check{D}_B, \check{f}_K)$ is an initial bottomed Λ -algebra.*

Proof This follows directly from Proposition 2.3.5 and Lemma 3.2.4. \square

The bottomed Λ -algebra (D_B, f_K) is said to be *unambiguous* if the following hold:

- (i) The mapping f_κ is injective for each $\kappa \in K$.
- (ii) For each $\beta \in B$ the sets $\text{Im}(f_\kappa)$, $\kappa \in K_\beta$, are disjoint subsets of $D_\beta \setminus \{\perp_\beta\}$.

The following analogue of Proposition 2.3.2 then holds:

Proposition 3.2.6 *A bottomed Λ -algebra is initial if and only if it is minimal and unambiguous.*

Proof The general case can clearly be reduced to that in which the bottom elements are all distinct. The statement thus follows from Lemma 3.2.4 and Proposition 2.3.7. \square

Propositions 2.3.2 and 3.2.6 imply that an initial bottomed Λ -algebra is never an initial Λ -algebra and that, conversely, a bottomed Λ -algebra which is an initial Λ -algebra is never an initial bottomed Λ -algebra. Moreover, Proposition 3.2.6 implies that the flat extension of (X_B, p_K) is unlikely to be an initial bottomed Λ -algebra. (In fact this will hold only in the trivial case in which each type in Λ is primitive.)

Having said something about bottomed Λ -algebras, the next task is to look at the special case of bottomed extensions of the initial Λ -algebra (X_B, p_K) .

A bottomed extension (D_B, f_K) of (X_B, p_K) is said to be *minimal* if (D_B, f_K) is a minimal bottomed Λ -algebra. In particular, as was already mentioned, the flat extension of (X_B, p_K) is minimal.

Note that for any bottomed extension of (D_B, f_K) of (X_B, p_K) the minimal bottomed subalgebra (\hat{D}_B, \hat{f}_K) of (D_B, f_K) is still an extension, and thus a minimal bottomed extension of (X_B, p_K) .

In the same way as above a bottomed extension (D_B, f_K) of (X_B, p_K) will be called *initial* if (D_B, f_K) is an initial bottomed Λ -algebra. By Proposition 3.2.6 an initial extension is also minimal.

Proposition 3.2.7 *There exists an initial extension of (X_B, p_K) , and any two initial extensions of (X_B, p_K) are conjugate.*

Proof The existence follows from Proposition 2.5.3 and Lemma 3.2.4, and the second statement then follows from Proposition 3.2.4 and Lemma 3.2.1. \square

Example 3.2.2 on the following page gives an initial extension of the Λ -algebra (X_B, p_K) introduced in Example 2.2.1.

A bottomed extension was defined to be initial if it is an initial bottomed Λ -algebra, and it could be objected that this definition is not really correct. However, the next result shows that it is, even if not correct, equivalent to the correct definition.

Proposition 3.2.8 *Let (D_B, f_K) be a bottomed extension of (X_B, p_K) . Then (D_B, f_K) is initial if and only if for each bottomed extension (D'_B, f'_K) of (X_B, p_K) there exists a unique extension homomorphism from (D_B, f_K) to (D'_B, f'_K) .*

Proof Suppose that for each bottomed extension (D'_B, f'_K) of (X_B, p_K) there exists a unique extension homomorphism from (D_B, f_K) to (D'_B, f'_K) . Then this holds in particular with (D'_B, f'_K) an initial extension of (X_B, p_K) (and Proposition 3.2.7 states that such an extension exists). Thus (D_B, f_K) and (D'_B, f'_K) are isomorphic (since there also exists a unique bottomed homomorphism from (D'_B, f'_K) to (D_B, f_K)) and hence (D_B, f_K) is initial. The converse is clear. \square

Example 3.2.2 The following notation will be employed here (and also later): If Z is a set then $\text{bot}(Z)$ denotes a disjoint copy of Z ; the element in $\text{bot}(Z)$ corresponding to $z \in Z$ will be denoted by z^\perp (so $\text{bot}(Z) = \{z^\perp : z \in Z\}$).

For each $\beta \in B$ choose an element \perp_β not in X_β and consider the Λ -algebra (X_B^\top, p_K^\top) defined by

$$X_{\text{bool}}^\top = \mathbb{B}^\perp = \{T, F, \perp_{\text{bool}}\},$$

$$X_{\text{nat}}^\top = \mathbb{N} \cup \text{bot}(\mathbb{N}) \text{ with } 0^\perp = \perp_{\text{nat}},$$

$$X_{\text{int}}^\top = \mathbb{Z} \cup \{\perp_{\text{int}}\}, \quad X_{\text{ipr}}^\top = (X_{\text{int}}^\top)^2 \cup \{\perp_{\text{ipr}}\},$$

$$X_{\text{ilst}}^\top = (X_{\text{int}}^\top)^* \cup \text{bot}((X_{\text{int}}^\top)^*) \text{ with } \varepsilon^\perp = \perp_{\text{ilst}}.$$

$$p_{\text{True}}^\top : \mathbb{I} \rightarrow X_{\text{bool}}^\top \text{ with } p_{\text{True}}^\top(\varepsilon) = T,$$

$$p_{\text{False}}^\top : \mathbb{I} \rightarrow X_{\text{bool}}^\top \text{ with } p_{\text{False}}^\top(\varepsilon) = F,$$

$$p_{\text{Zero}}^\top : \mathbb{I} \rightarrow X_{\text{nat}}^\top \text{ with } p_{\text{Zero}}^\top(\varepsilon) = 0,$$

$$p_{\text{Succ}}^\top : X_{\text{nat}}^\top \rightarrow X_{\text{nat}}^\top \text{ with}$$

$$p_{\text{Succ}}^\top(x) = \begin{cases} n+1 & \text{if } x = n \text{ for some } n \in \mathbb{N}, \\ (n+1)^\perp & \text{if } x = n^\perp \text{ for some } n \in \mathbb{N}, \end{cases}$$

$$p_n^\top : \mathbb{I} \rightarrow X_{\text{int}}^\top \text{ with } p_n^\top(\varepsilon) = n \text{ for each } n \in \mathbb{Z},$$

$$p_{\text{Pair}}^\top : X_{\text{int}}^\top \times X_{\text{int}}^\top \rightarrow X_{\text{ipr}}^\top \text{ with } p_{\text{Pair}}^\top(x_1, x_2) = (x_1, x_2),$$

$$p_{\text{Nil}}^\top : \mathbb{I} \rightarrow X_{\text{ilst}}^\top \text{ with } p_{\text{Nil}}^\top(\varepsilon) = \varepsilon,$$

$$p_{\text{Cons}}^\top : X_{\text{int}}^\top \times X_{\text{ilst}}^\top \rightarrow X_{\text{ilst}}^\top \text{ with}$$

$$p_{\text{Cons}}^\top(x, z) = \begin{cases} x \triangleleft s & \text{if } z = s \text{ for some } s \in (X_{\text{int}}^\top)^*, \\ (x \triangleleft s)^\perp & \text{if } z = s^\perp \text{ for some } s \in (X_{\text{int}}^\top)^*. \end{cases}$$

It is left to the reader to check that (X_B^\top, p_K^\top) is an initial extension of the Λ -algebra (X_B, p_K) introduced in Example 2.2.1.

Note that an element of X_{ilst}^\top has either the form $x_1 \cdots x_n$ or the form $(x_1 \cdots x_n)^\perp$, where $n \geq 0$ and $x_j \in \mathbb{Z} \cup \{\perp_{\text{int}}\}$ for $j = 1, \dots, n$. The element $x_1 \cdots x_n$ describes a ‘real’ list with n components (although some or all of these components may be ‘undefined’). The element $(x_1 \cdots x_n)^\perp$, on the other hand, should be thought of as a ‘partial’ list containing at least n components, of which the first n components are ‘known’ to be x_1, \dots, x_n .

3.3 Regular extensions and the trace

As already indicated, ‘undefined’ and ‘partially defined’ data objects will be described with the help of a bottomed extension of the basic initial Λ -algebra (X_B, p_K) . However, as it stands, this notion is much too general, and two conditions will be imposed on the bottomed extensions to be actually used. In the present section the first of these conditions is introduced, which goes under the name of *regularity*.

A bottomed Λ -algebra (D_B, f_K) is said to be *regular* if for each $\beta \in B$ and each $u \in D_\beta \setminus \{\perp_\beta\}$ there exists a unique $\kappa \in K_\beta$ and a unique element $c \in \text{Dom}(f_\kappa)$ such that $f_\kappa(c) = u$. (This is a possible generalisation of the definition of a regular Λ -algebra given in Section 2.3. However, it is not the generalisation which would make the analogue of Proposition 2.3.2 hold.)

A bottomed extension (D_B, f_K) of (X_B, p_K) is now said to be *regular* if (D_B, f_K) is a regular bottomed Λ -algebra. In particular, the flat extension of (X_B, p_K) is regular (since (X_B, p_K) is a regular Λ -algebra). Moreover, by Propositions 3.2.2 and 3.2.6 each initial extension of (X_B, p_K) is regular. Note that if (D_B, f_K) is a regular bottomed extension of (X_B, p_K) then $D_\beta = X_\beta \cup \{\perp_\beta\}$ must hold for each primitive type $\beta \in B$, and so in particular $D_{\text{int}} = \mathbb{Z} \cup \{\perp_{\text{int}}\}$ and $D_{\text{bool}} = \mathbb{B} \cup \{\perp_{\text{bool}}\}$.

If (D_B, f_K) is a bottomed Λ -algebra then the family C_K given by

$$C_\kappa = \{c \in \text{Dom}(f_\kappa) : f_\kappa(c) \neq \perp_\beta\}$$

for each $\kappa \in K_\beta$ will be called the *core* of (D_B, f_K) . In particular, if (D_B, f_K) is initial then $C_\kappa = \text{Dom}(f_\kappa)$ for each $\kappa \in K$, and the core of the flat extension of (X_B, p_K) is the family C_K with $C_\kappa = \text{Dom}(p_\kappa)$ for each $\kappa \in K$. Regularity can be expressed in terms of the core as follows:

Lemma 3.3.1 *Let C_K be the core of the bottomed Λ -algebra (D_B, f_K) . Then (D_B, f_K) is regular if and only if the following three conditions hold:*

- (i) *The restriction of f_κ to C_κ is injective for each $\kappa \in K$.*
- (ii) *For each $\beta \in B$ the sets $f_\kappa(C_\kappa)$, $\kappa \in K_\beta$, are disjoint subsets of D_β .*
- (iii) $\bigcup_{\kappa \in K_\beta} f_\kappa(C_\kappa) = D_\beta \setminus \{\perp_\beta\}$ for each $\beta \in B$.

Proof Assume first that (i), (ii) and (iii) hold. Let $\beta \in B$ and $u \in D_\beta \setminus \{\perp_\beta\}$; by (iii) there then exists $\kappa \in K_\beta$ and $c \in C_\kappa$ such that $f_\kappa(c) = u$. Suppose also that $u = f_{\kappa'}(c')$ with $\kappa' \in K_\beta$ and $c' \in \text{Dom}(f_{\kappa'})$; then $c' \in C_{\kappa'}$ (since $u \neq \perp_\beta$), hence $u \in f_\kappa(C_\kappa) \cap f_{\kappa'}(C_{\kappa'})$ and by (ii) this is only possible if $\kappa' = \kappa$. Moreover, by (i) it now follows that $c' = c$, and therefore (D_B, f_K) is regular.

Assume conversely that (D_B, f_K) is regular. Let $\beta \in B$, $\kappa_1, \kappa_2 \in K_\beta$ and $c_1 \in C_{\kappa_1}$, $c_2 \in C_{\kappa_2}$ with $f_{\kappa_1}(c_1) = f_{\kappa_2}(c_2)$. Then $f_{\kappa_1}(c_1) \neq \perp_\beta$ and therefore $\kappa_1 = \kappa_2$ and $c_1 = c_2$. This shows that (i) and (ii) hold. Moreover, $D_\beta \setminus \{\perp_\beta\} \subset \bigcup_{\kappa \in K_\beta} \text{Im}(f_\kappa)$ for each $\beta \in B$ and hence by the definition of C_K (iii) holds. \square

Note by Proposition 3.2.2 condition (iii) holds automatically for a minimal bottomed Λ -algebra, which leads to the following observation: Let (D'_B, f'_K) be an extension of a bottomed Λ -algebra (D_B, f_K) , considered as a bottomed Λ -algebra in the obvious way, and suppose (D'_B, f'_K) is regular. Then in general this does not imply that (D_B, f_K) is regular. However, (D_B, f_K) will be regular if it is minimal (since conditions (i) and (ii) in Lemma 3.3.1 are inherited by (D_B, f_K)).

A regular bottomed Λ -algebra (D_B, f_K) will be called *fully regular* if $\perp_\beta \notin \text{Im}(f_\kappa)$ for each $\kappa \in K_\beta$, $\beta \in B$. Thus a regular bottomed Λ -algebra (D_B, f_K) is fully regular if and only if its core C_K is maximal, i.e., if and only if $C_\kappa = \text{Dom}(f_\kappa)$ for each $\kappa \in K$.

Proposition 3.3.1 *An initial bottomed Λ -algebra is fully regular. Conversely, a fully regular bottomed Λ -algebra is initial if and only if it is minimal.*

Proof This follows from Propositions 3.2.2 and 3.2.6. \square

Recall that in Section 3.1 the ground term algebra was introduced. This was a further initial Λ -algebra (F_B, \odot_K) , and the unique isomorphism $\llbracket \cdot \rrbracket_B$ from (F_B, \odot_K) to (X_B, p_K) is used to denote the basic data objects (i.e., the data object $x \in X_\beta$ can be denoted by the unique element $s \in F_\beta$ with $\llbracket s \rrbracket_\beta = x$).

By Proposition 3.2.7 there exists an initial bottomed extension of (F_B, \odot_K) (and, moreover, any two such initial extensions are conjugate). In what follows it can thus be supposed that an initial bottomed extension (F_B^b, \odot_K^b) of (F_B, \odot_K) has been chosen, which will be called the *bottomed ground term algebra*. The bottom element of F_β^b will be denoted by \flat_β rather than \perp_β (and to be thought of as a name with which to refer to the ‘undefined’ element \perp_β).

Let (D_B, f_K) be a bottomed Λ -algebra, which is considered to be fixed for the rest of the section. Then, since (F_B^b, \odot_K^b) is an initial bottomed Λ -algebra, there exists a unique bottomed homomorphism $\llbracket \cdot \rrbracket_B^\perp : (F_B^b, \odot_K^b) \rightarrow (D_B, f_K)$, i.e., $\llbracket \cdot \rrbracket_B^\perp$ is the unique homomorphism from (F_B^b, \odot_K^b) to (D_B, f_K) with $\llbracket \flat_\beta \rrbracket_\beta^\perp = \perp_\beta$ for each $\beta \in B$. In particular, if (D_B, f_K) is a bottomed extension of (X_B, p_K) then by Proposition 2.5.2 $\llbracket \cdot \rrbracket_B^\perp$ is an extension of $\llbracket \cdot \rrbracket_B$, i.e., $\llbracket s \rrbracket_\beta^\perp = \llbracket s \rrbracket_\beta$ for all $s \in F_\beta$, $\beta \in B$.

Proposition 3.3.2 (1) *The family $\llbracket \cdot \rrbracket_B^\perp$ is surjective (i.e., $\llbracket \cdot \rrbracket_\beta^\perp : F_\beta^b \rightarrow D_\beta$ is surjective for each $\beta \in B$) if and only if (D_B, f_K) is minimal.*

(2) *If (D_B, f_K) is fully regular then the family $\llbracket \cdot \rrbracket_B^\perp$ is injective, i.e., the mapping $\llbracket \cdot \rrbracket_\beta^\perp : F_\beta^b \rightarrow D_\beta$ is injective for each $\beta \in B$.*

(3) *The family $\llbracket \cdot \rrbracket_B^\perp$ is an isomorphism if and only if (D_B, f_K) is initial.*

Proof (1) For each $\beta \in B$ let $D'_\beta = \llbracket F_\beta^b \rrbracket_\beta^\perp$; then $\perp_\beta \in D'_\beta$ for each $\beta \in B$ and, as in Lemma 2.2.1 (1), D'_B is invariant in (D_B, f_K) . Moreover, D'_B is the minimal such family: Let D''_β be any family invariant in (D_B, f_K) and with $\perp_\beta \in D''_\beta$ for each $\beta \in B$, and for each $\beta \in B$ let $F''_\beta = \{s \in F_\beta^b : \llbracket s \rrbracket_\beta^\perp \in D''_\beta\}$. Then $\flat_\beta \in F''_\beta$

and, as in Lemma 2.2.1 (2), F_B^o is invariant in (F_B^b, \odot_K^b) ; thus $F_B^o = F_B^b$ (since by Proposition 3.2.6 (F_B^b, \odot_K^b) is a minimal extension of (F_B, \odot_K)). Hence $D'_B \subset D_B^o$. From this it follows that $\llbracket \cdot \rrbracket_B^\perp$ is surjective if and only if (D_B, f_K) is minimal.

(3) This is clear.

(2) Let (\hat{D}_B, \hat{f}_K) be the minimal bottomed subalgebra of (D_B, f_K) . Then (\hat{D}_B, \hat{f}_K) is a minimal fully regular and thus an initial bottomed Λ -algebra. Therefore by (3) $\llbracket \cdot \rrbracket_B^\perp$ must also be an isomorphism from (F_B^b, \odot_K^b) to (\hat{D}_B, \hat{f}_K) . In particular, the mapping $\llbracket \cdot \rrbracket_\beta^\perp : F_\beta^b \rightarrow D_\beta$ is injective for each $\beta \in B$. \square

Proposition 3.3.3 *Suppose (D_B, f_K) is a regular extension of (X_B, p_K) . Then*

$$\{s \in F_\beta^b : \llbracket s \rrbracket_\beta^\perp \in X_\beta\} = F_\beta$$

for each $\beta \in B$.

Proof It has already been noted that $\llbracket \cdot \rrbracket_B^\perp$ is an extension of $\llbracket \cdot \rrbracket_B$, which implies that $\llbracket s \rrbracket_\beta^\perp \in X_\beta$ for each $s \in F_\beta$ (and this holds even if (D_B, f_K) is not regular). For the converse consider the family \hat{F}_B^b with $\hat{F}_B^b \subset F_B^b$ given by

$$\hat{F}_\beta^b = F_\beta \cup \{s \in F_\beta^b \setminus F_\beta : \llbracket s \rrbracket_\beta^\perp \in D_\beta \setminus X_\beta\}$$

for each $\beta \in B$. If $\kappa \in K$ is of type $\emptyset \rightarrow \beta$ then $\odot_\kappa^b(\varepsilon) = \odot_\kappa(\varepsilon) \in F_\beta \subset \hat{F}_\beta^b$. Now let $\kappa \in K$ be of type $L \rightarrow \beta$ with $L \neq \emptyset$ and let $a \in (\hat{F}_\kappa^b)^L$, i.e., $a(\eta) \in \hat{F}_\eta^b$ for each $\eta \in L$. Suppose $\llbracket \odot_\kappa^b(a) \rrbracket_\beta^\perp = f_\kappa(\llbracket a \rrbracket_\kappa^\perp)^L) \in X_\beta$. Then, since (D_B, f_K) is regular, it follows that $(\llbracket a \rrbracket_\kappa^\perp)^L \in X_\kappa^L$, i.e., $\llbracket a(\eta) \rrbracket_\eta \in X_\eta$ for each $\eta \in L$. But then by assumption $a(\eta) \in F_\eta$ for each $\eta \in L$ and thus $\odot_\kappa^b(a) = \odot_\kappa(a) \in F_\beta$. This shows that $\odot_\kappa^b(a) \in \hat{F}_\beta^b$, and hence that the family \hat{F}_B^b is invariant in (F_B^b, \odot_K^b) . But clearly $b_\beta \in \hat{F}_\beta^b$ for each $\beta \in B$ and so by Proposition 3.2.6 $\hat{F}_B^b = F_B^b$. From this it follows that $\llbracket s \rrbracket_\beta^\perp \in D_\beta \setminus X_\beta$ for all $s \in F_\beta^b \setminus F_\beta$. \square

Lemma 3.3.2 *If (D_B, f_K) is the flat extension of (X_B, p_K) then*

$$\llbracket s \rrbracket_\beta^\perp = \begin{cases} \llbracket s \rrbracket_\beta & \text{if } s \in F_\beta, \\ \perp_\beta & \text{if } s \in F_\beta^b \setminus F_\beta. \end{cases}$$

Proof This follows immediately from Proposition 3.3.3 and the fact that $\llbracket \cdot \rrbracket_B^\perp$ is an extension of $\llbracket \cdot \rrbracket_B$. \square

Now define a family $R_B \subset F_B^b$, called the *trace* of (D_B, f_K) , by putting

$$R_\beta = \{s \in F_\beta^b : \llbracket s \rrbracket_\beta^\perp \neq \perp_\beta\},$$

thus in fact $R_\beta \subset F_\beta^b \setminus \{b_\beta\}$ for each $\beta \in B$ and, moreover, $F_B \subset R_B$ whenever (D_B, f_K) is a bottomed extension of (X_B, p_K) .

If (D_B, f_K) is the flat extension of (X_B, p_K) then by Lemma 3.3.2 $R_B = F_B$, and if (D_B, f_K) is fully regular then Proposition 3.3.2 (2) implies that $R_\beta = F_\beta^b \setminus \{v_\beta\}$ for each $\beta \in B$. The trace is thus minimal for the flat extension and maximal for a fully regular extension.

The next result shows that the trace is essentially a complete invariant for minimal regular bottomed Λ -algebras.

Proposition 3.3.4 *Minimal regular bottomed Λ -algebras are isomorphic if and only if they have the same trace. In particular, minimal regular extensions of (X_B, p_K) are conjugate if and only if they have the same trace.*

Proof It is clear that isomorphic bottomed Λ -algebras have the same trace. To prepare for the converse a couple of lemmas are needed. In the following let (D_B, f_K) be a minimal regular bottomed Λ -algebra with core C_K .

Lemma 3.3.3 *There exists a unique family of mappings $\#_B$ with $\#_\beta : D_\beta \rightarrow \mathbb{N}$ for each $\beta \in B$ such that $\#_\beta(\perp_\beta) = 0$ for each $\beta \in B$, $\#_\beta(f_\kappa(\varepsilon)) = 0$ whenever $\kappa \in K$ is of type $\emptyset \rightarrow \beta$ and such that*

$$\#_\beta(f_\kappa(c)) = 1 + \max\{\#_\eta(c(\eta)) : \eta \in L\}$$

for all $c \in D_\beta^L$ whenever $\kappa \in K$ is of type $L \rightarrow \beta$ with $L \neq \emptyset$.

Proof This is, with the obvious modifications, essentially the same as the proof of Lemma 2.3.5. \square

Lemma 3.3.4 *There exists a unique family of mappings ϱ_B with $\varrho_\beta : D_\beta \rightarrow F_\beta^b$ for each $\beta \in B$ such that $\varrho_\beta(\perp_\beta) = v_\beta$ and if $\kappa \in K$ is of type $L \rightarrow \beta$ then*

$$\varrho_\beta(f_\kappa(c)) = \odot_\kappa^b(\varrho_\beta^L(c))$$

for all $c \in C_\kappa$. Moreover, $\llbracket \varrho_\beta(u) \rrbracket_\beta^\perp = u$ for all $u \in D_\beta$, $\beta \in B$.

Proof Let $\#_B$ be the family of mappings given by Lemma 3.3.3 and for each $\beta \in B$, $m \in \mathbb{N}$ let $D_\beta^m = \{u \in D_\beta : \#_\beta(u) = m\}$. Define ϱ_β on D_β^m for each $\beta \in B$ using induction on m . If $u \in D_\beta^0$ then either $u = \perp_\beta$, in which case put $\varrho_\beta(u) = v_\beta$, or there exists a unique $\kappa \in K$ of type $\emptyset \rightarrow \beta$ with $u = f_\kappa(\varepsilon)$, and in this case put $\varrho_\beta(u) = \odot_\kappa^b(\varepsilon)$. Now let $m > 0$ and suppose ϱ_θ is already defined on D_θ^k for each $\theta \in B$ and for all $k < m$. Let $u \in D_\beta^m$ (so in particular $u \neq \perp_\beta$); then there exists a unique $\kappa \in K$ of type $L \rightarrow \beta$ for some $L \neq \emptyset$ and a unique element $c \in C_\kappa$ such that $u = f_\kappa(c)$. Moreover, $c(\eta) \in D_\eta^{m_\eta}$ for some m_η with $m_\eta < m$, which means that $\varrho_\eta(c(\eta))$ is already defined for each $\eta \in L$. It thus makes sense to put

$$\varrho_\beta(u) = \odot_\beta^b(\varrho_\beta^L(c))$$

(where of course $\varrho_\beta^L(c)$ is the element $c' \in (F_\beta^b)^L$ with $c'(\eta) = \varrho_\eta(c(\eta))$ for each $\eta \in L$). In this way ϱ_β is defined on D_β^m for each $m \in \mathbb{N}$ and the family ϱ_B has the

required property by construction. The uniqueness of the family ϱ_B also follows using induction on m .

Finally, for each $\beta \in B$ let $D'_\beta = \{u \in D_\beta : \llbracket \varrho_\beta(u) \rrbracket_\beta^\perp = u\}$; then $\perp_\beta \in D'_\beta$ for each $\beta \in B$ and the family D'_B is invariant. (Let $\kappa \in K$ be of type $L \rightarrow \beta$ and let $c \in D_\diamond^L$ with $c(\eta) \in D'_\eta$ for each $\eta \in L$; put $u = f_\kappa(c)$. If $u = \perp_\beta$ then $u \in D'_\beta$ holds trivially. On the other hand, if $u \neq \perp_\beta$ then $c \in C_\kappa$; in this case $\varrho_\beta(u) = \odot_\kappa^b(\varrho_\diamond^L(c))$ and hence, since $\llbracket \varrho_\eta(c(\eta)) \rrbracket_\eta^\perp = c(\eta)$ for each $\eta \in L$ and so by Lemma 2.1.1 (2) $(\llbracket \varrho_\diamond^L(c) \rrbracket_\diamond^\perp)^L = c$, it follows that

$$\llbracket \varrho_\beta(u) \rrbracket_\beta^\perp = \llbracket \odot_\kappa^b(\varrho_\diamond^L(c)) \rrbracket_\beta^\perp = f_\kappa((\llbracket \varrho_\diamond^L(c) \rrbracket_\diamond^\perp)^L) = f_\kappa(c) = u.$$

Therefore in both cases $u \in D'_\beta$.) But (D_B, f_K) is minimal and thus $D'_B = D_B$, i.e., $\llbracket \varrho_\beta(u) \rrbracket_\beta^\perp = u$ for all $u \in D_\beta$, $\beta \in B$. \square

Now consider a minimal regular bottomed Λ -algebra (D'_B, f'_K) having the same trace as (D_B, f_K) and denote by π_B the unique bottomed homomorphism from (F_B^b, \odot_K^b) to (D'_B, f'_K) .

Lemma 3.3.5 *Let $\varrho_B : D_B \rightarrow F_B^b$ be the family defined in Lemma 3.3.4. Then $\omega_B = \pi_B \varrho_B$ is a bottomed homomorphism from (D_B, f_K) to (D'_B, f'_K) .*

Proof Clearly $\omega_\beta(\perp_\beta) = \perp_\beta$ for each $\beta \in B$. Let $\kappa \in K$ be of type $L \rightarrow \beta$ and $c \in D_\diamond^L$. Then by Lemma 2.1.1 (2)

$$f'_\kappa(\omega_\diamond^L(c)) = f'_\kappa(\pi_\diamond^L(\varrho_\diamond^L(c))) = \pi_\beta(\odot_\kappa^b(\varrho_\diamond^L(c))).$$

Now if $c \in C_\kappa$ then $\omega_\beta(f_\kappa(c)) = \pi_\beta(\varrho_\beta(f_\kappa(c))) = \pi_\beta(\odot_\kappa^b(\varrho_\diamond^L(c)))$, and hence in this case $\omega_\beta(f_\kappa(c)) = f'_\kappa(\omega_\diamond^L(c))$. On the other hand, if $c \notin C_\kappa$ then $\llbracket \varrho_\eta(c(\eta)) \rrbracket_\eta^\perp = c(\eta)$ for each $\eta \in L$ and so by Lemma 2.1.1 (2) $(\llbracket \varrho_\diamond^L(c) \rrbracket_\diamond^\perp)^L = c$, thus

$$\llbracket \odot_\kappa^b(\varrho_\diamond^L(c)) \rrbracket_\beta^\perp = f_\kappa((\llbracket \varrho_\diamond^L(c) \rrbracket_\diamond^\perp)^L) = f_\kappa(c) = \perp_\beta,$$

which means that $\odot_\kappa^b(\varrho_\diamond^L(c)) \in R_\beta$, where R_B is the common trace of (D_B, f_K) and (D'_B, f'_K) . Therefore it again follows that

$$f'_\kappa(\omega_\diamond^L(c)) = \pi_\beta(\odot_\kappa^b(\varrho_\diamond^L(c))) = \perp_\beta = \omega_\beta(\perp_\beta) = \omega_\beta(f_\kappa(c)).$$

This shows that ω_B is a bottomed homomorphism. \square

Let (D_B, f_K) and (D'_B, f'_K) be minimal regular bottomed Λ -algebras having the same trace. By Lemma 3.3.5 there then exists a bottomed homomorphism ω_B from (D_B, f_K) to (D'_B, f'_K) and also a bottomed homomorphism ω'_B from (D'_B, f'_K) to (D_B, f_K) . By Proposition 2.2.1 $\omega_B \omega'_B$ is then a bottomed homomorphism from (D_B, f_K) to itself and the only such homomorphism is id_B , i.e., $\omega_B \omega'_B = \text{id}_B$. In the same way $\omega'_B \omega_B = \text{id}'_B$, and hence ω_B is a bottomed isomorphism, i.e., (D_B, f_K) and (D'_B, f'_K) are isomorphic. This completes the proof of Proposition 3.3.4. \square

3.4 Monotone extensions

In this section the second of the two conditions is introduced which will be imposed on the bottomed extensions. This is called *monotonicity* and, although its definition is somewhat technical, it is a simple matter to show (as will be done in Proposition 3.4.3) that the flat extension and any fully regular extension of (X_B, p_K) is monotone.

Let (D_B, f_K) be a bottomed Λ -algebra. If J is a B -typed set, (Z_B, r_K) a J -free Λ -algebra and $b \in D_\diamond^J$ then π_B^b will denote the unique homomorphism from (Z_B, r_K) to (D_B, f_K) such that $\pi_\eta^b(\eta) = b(\eta)$ for all $\eta \in J$. (D_B, f_K) is said to be *monotone* if whenever J is a B -typed set and (Z_B, r_K) is a J -free Λ -algebra and $\pi_\beta^{\perp^J}(z) \neq \perp_\beta$ for some $z \in Z_\beta$ then $\pi_\beta^b(z) \neq \perp_\beta$ for all $b \in D_\diamond^J$. Here \perp^J is the bottom element of D_\diamond^J given by $\perp^J(\eta) = \perp_\eta$ for each $\eta \in J$.

A bottomed extension (D_B, f_K) of (X_B, p_K) is now said to be *monotone* if (D_B, f_K) is a monotone bottomed Λ -algebra.

Monotonicity as defined above is not particularly easy to work with, and so an equivalent condition (to be called *strong monotonicity*) is now presented which turns out to be more tractable.

Let (D_B, f_K) be a bottomed Λ -algebra, (Y_B, q_K) be any Λ -algebra and let π_B be a homomorphism from (Y_B, q_K) to (D_B, f_K) . The family $U_B \subset Y_B$ with

$$U_\beta = \{y \in Y_\beta : \pi_\beta(y) = \perp_\beta\}$$

for each $\beta \in B$ is called the *kernel* of π_B and π_B is said to be *fat bottomed* if Y_B is the only invariant family containing U_B .

Now (D_B, f_K) is said to be *strongly monotone* if whenever (Y_B, q_K) is a Λ -algebra and $\check{\pi}_B : (Y_B, q_K) \rightarrow (D_B, f_K)$ a fat bottomed homomorphism then the kernel \check{U}_B of $\check{\pi}_B$ is maximal in the sense that if $\pi_B : (Y_B, q_K) \rightarrow (D_B, f_K)$ is any homomorphism with kernel U_B then $U_B \subset \check{U}_B$.

Proposition 3.4.1 *A strongly monotone bottomed Λ -algebra (D_B, f_K) is monotone.*

Proof Let J be a B -typed set and (Z_B, r_K) a J -free Λ -algebra; denote the kernel of the homomorphism $\pi_B^{\perp^J} : (Z_B, r_K) \rightarrow (D_B, f_K)$ by \check{U}_B . Then $\pi_B^{\perp^J}$ is fat bottomed, since $J_\beta = \{\eta \in J : \langle \eta \rangle = \beta\} \subset \check{U}_\beta$ for each $\beta \in B$, and by Proposition 2.3.7 (Z_B, r_K) is J -minimal. Thus if $b \in D_\diamond^J$ and U_B is the kernel of π_B^b then $U_B \subset \check{U}_B$. But this just means that $\pi_\beta^{\perp^J}(z) = \perp_\beta$ whenever $\pi_\beta^b(z) = \perp_\beta$, which shows that (D_B, f_K) is monotone. \square

Later it will be seen that the converse of Proposition 3.4.1 holds, i.e., that a monotone bottomed Λ -algebra is also strongly monotone.

It is also useful to consider a further notion which is somewhat stronger than that of being monotone: A bottomed Λ -algebra (D_B, f_K) is said to be *structurally monotone* if

whenever (Y_B, q_K) is a Λ -algebra and π_B and π'_B are homomorphisms from (Y_B, q_K) to (D_B, f_K) then the family Y'_B given by

$$Y'_\beta = \{ y \in Y_\beta : \pi'_\beta(y) = \perp_\beta \text{ whenever } \pi_\beta(y) = \perp_\beta \}$$

for each $\beta \in B$ is invariant in (Y_B, q_K) .

Proposition 3.4.2 *If (D_B, f_K) is structurally monotone then it is strongly monotone (and thus also monotone).*

Proof Let (Y_B, q_K) be a Λ -algebra, let $\check{\pi}_B$ be a fat bottomed and π_B an arbitrary homomorphism from (Y_B, q_K) to (D_B, f_K) . Put

$$Y'_\beta = \{ y \in Y_\beta : \check{\pi}_\beta(y) = \perp_\beta \text{ whenever } \pi_\beta(y) = \perp_\beta \}$$

for each $\beta \in B$. Then Y'_B is invariant, since (D_B, f_K) is structurally monotone. But if \check{U}_B is the kernel of $\check{\pi}_B$ then clearly $\check{U}_B \subset Y'_B$, and hence $Y'_B = Y_B$, since $\check{\pi}_B$ is fat bottomed. Therefore if U_B is the kernel of π_B then $U_B \subset \check{U}_B$. This shows that (D_B, f_K) is strongly monotone. \square

Proposition 3.4.3 *Any fully regular bottomed Λ -algebra is structurally monotone. Moreover, the flat extension of (X_B, p_K) is structurally monotone. In particular, these bottomed Λ -algebras are then monotone.*

Proof Let (Y_B, q_K) be a Λ -algebra and let π_B and π'_B be homomorphisms from (Y_B, q_K) to (D_B, f_K) . For each $\beta \in B$ put

$$Y'_\beta = \{ y \in Y_\beta : \pi'_\beta(y) = \perp_\beta \text{ whenever } \pi_\beta(y) = \perp_\beta \}.$$

If $\kappa \in K$ is of type $\emptyset \rightarrow \beta$ then $\pi_\beta(q_\kappa(\varepsilon)) = f_\kappa(\varepsilon) \neq \perp_\beta$, and so $q_\kappa(\varepsilon)$ is trivially an element of Y'_β . Thus consider $\kappa \in K$ of type $L \rightarrow \beta$ with $L \neq \emptyset$ and let $c \in Y_\diamond^L$ with $c(\eta) \in Y'_\eta$ for each $\eta \in L$. Now $\pi_\beta(q_\kappa(c)) = f_\kappa(\pi_\diamond^L(c))$, and so in particular $q_\kappa(c) \in Y'_\beta$ if (D_B, f_K) is fully regular, since in this case $f_\kappa(c') \neq \perp_\beta$ for all $c' \in D_\diamond^L$. On the other hand, if (D_B, f_K) is the flat extension of (X_B, p_K) and $\pi_\beta(q_\kappa(c)) = \perp_\beta$ then $\pi_\eta(c(\eta)) = \perp_\eta$ for some $\eta \in L$, thus $\pi'_\eta(c(\eta)) = \perp_\eta$, and hence

$$\pi'_\beta(q_\kappa(c)) = f_\kappa((\pi'_\diamond)^L(c)) = \perp_\beta,$$

which again implies that $q_\kappa(c) \in Y'_\beta$. The family Y'_B is thus invariant if (D_B, f_K) is either the flat extension of (X_B, p_K) or a fully regular bottomed Λ -algebra. \square

The proof of the converse of Proposition 3.4.1 requires the general construction involving free algebras given below which will also be needed several times later.

In what follows let (D_B, f_K) be a bottomed Λ -algebra and (Y_B, q_K) be an arbitrary Λ -algebra. For each $\beta \in B$ let \prec_β be a relation on the set $D_\beta \times Y_\beta$, i.e., \prec_β is just a subset of $D_\beta \times Y_\beta$. As usual, however, infix notation will be employed here and $u \prec_\beta y$ written to mean that (u, y) belongs to the relation \prec_β . The family of relations \prec_B is said to be *compatible* if whenever $\kappa \in K$ is of type $L \rightarrow \beta$ and $a \in Y_\diamond^L$, $u \in D_\beta \setminus \{\perp_\beta\}$

are such that $u \prec_\beta q_\kappa(a)$ then there exists $c \in D_\diamond^L$ with $u = f_\kappa(c)$ such that $c \prec_\diamond^L a$, i.e., such that $c(\eta) \prec_\eta a(\eta)$ for each $\eta \in L$.

Let $\beta \in B$; then $y \in Y_\beta$ is said to be a *base element* if the set $\{u \in D_\beta : u \prec_\beta y\}$ contains at most the bottom element \perp_β , and the set of base elements of Y_β will be denoted by U_β . A compatible family \prec_B is now said to be *full* if Y_B is the only invariant family containing U_B .

If J is a B -typed set and (Z_B, r_K) a J -free Λ -algebra then for each $b \in Y_\diamond^J$ let ϱ_B^b denote the unique homomorphism from (Z_B, r_K) to (Y_B, q_K) such that $\varrho_\eta^b(\eta) = b(\eta)$ for each $\eta \in L$. Moreover, ω_B^J will denote the unique homomorphism from (Z_B, r_K) to (D_B, f_K) such that $\omega_\eta^J(\eta) = \perp_\eta$ for each $\eta \in L$. (Thus, with the notation employed above, $\omega_B^J = \pi_B^{\perp^J}$.)

Proposition 3.4.4 *Suppose \prec_B is a full compatible family. Then for each $y \in Y_\beta$ and each $u \in D_\beta$ with $u \prec_\beta y$ there exists a finite B -typed set J , a J -free Λ -algebra (Z_B, r_K) , and elements $z \in Z_\beta$ and $b \in Y_\diamond^J$ such that $\varrho_\beta^b(z) = y$ and $\omega_\beta^J(z) = u$.*

Proof For each $\beta \in B$ let \dot{Y}_β be the set of those elements $y \in Y_\beta$ for which the above statement holds (i.e., those $y \in Y_\beta$ such that if $u \in D_\beta$ with $u \prec_\beta y$ then there exists a finite B -typed set J , a J -free Λ -algebra (Z_B, r_K) , and elements $z \in Z_\beta$ and $b \in Y_\diamond^J$ such that $\varrho_\beta^b(z) = y$ and $\omega_\beta^J(z) = u$). The aim is thus to show that $\dot{Y}_B = Y_B$.

A basic remark: For each $y \in Y_\beta$ there always exists a finite B -typed set J , a J -free Λ -algebra (Z_B, r_K) and elements $z \in Z_\beta$ and $b \in Y_\diamond^J$ such that $\varrho_\beta^b(z) = y$ and $\omega_\beta^J(z) = \perp_\beta$. In fact, just let $J = \{\eta\}$ with η of type β , let (Z_B, r_K) be any J -free Λ -algebra, let $b \in Y_\diamond^J$ be the assignment given by $b(\eta) = y$ and take $z = \eta$.

From the basic remark it immediately follows that $U_B \subset Y'_B$ and thus, since the family \prec_B is full, it is enough to show that the family Y'_B is invariant in (Y_B, q_K) .

It will be shown first that if $\kappa \in K$ is of type $\emptyset \rightarrow \beta$ then $q_\kappa(\varepsilon) \in Y'_\beta$. Thus consider $u \in D_\beta$ with $u \prec_\beta q_\kappa(\varepsilon)$; then either $u = f_\kappa(\varepsilon)$ or $u = \perp_\beta$ (since the family \prec_B is compatible) and the latter is dealt with by the basic remark. But if $u = f_\kappa(\varepsilon)$ then take any finite B -typed set J , any J -free Λ -algebra (Z_B, r_K) , any assignment $b \in Y_\diamond^J$ and put $z = r_\kappa(\varepsilon)$; then $\varrho_\beta^b(z) = \varrho_\beta^b(r_\kappa(\varepsilon)) = q_\kappa(\varepsilon)$ and $\omega_\beta^J(z) = \omega_\beta^J(r_\kappa(\varepsilon)) = f_\kappa(\varepsilon) = u$.

Now let $\kappa \in K$ be of type $L \rightarrow \beta$ with $L \neq \emptyset$ and let $a \in (Y'_\diamond)^L$. The task on hand is to show that $q_\kappa(a) \in Y'_\beta$ and so, again making use of the basic remark, it is only necessary to consider $u \in D_\beta \setminus \{\perp_\beta\}$ with $u \prec_\beta q_\kappa(a)$. In this case (since the family \prec_B is compatible) there exists $c \in D_\diamond^L$ with $u = f_\kappa(c)$ such that $c \prec_\diamond^L a$. Then $a(\eta) \in Y'_\eta$ and $c(\eta) \prec_\eta a(\eta)$ for each $\eta \in L$, and hence there exists a finite B -typed set J_η , a J_η -free Λ -algebra (Z_B^η, r_K^η) and elements $z_\eta \in Z_\eta^\eta$ and $b_\eta \in Y_\diamond^{J_\eta}$ with $\varrho_\eta^{b_\eta}(z_\eta) = a(\eta)$ and $\omega_\eta^{J_\eta}(z_\eta) = c(\eta)$. Moreover, the sets J_η , $\eta \in L$, can be arranged to be disjoint (by introducing duplicate variables if necessary). Put $J = \bigcup_{\eta \in L} J_\eta$, considered as a B -typed set in the obvious way and choose any J -free Λ -algebra (Z_B, r_K) ; let $b \in Y_\diamond^J$

be the sum of the assignments b_η , $\eta \in L$, i.e., b is given by $b(\xi) = b_\eta(\xi)$ for each $\xi \in J_\eta$, $\eta \in L$. At this point the following somewhat technical fact is needed:

Lemma 3.4.1 *There exists an assignment $d \in Z_\diamond^L$ such that $\varrho_\eta^b(d(\eta)) = \varrho_\eta^{b_\eta}(z_\eta)$ and $\omega_\eta^J(d(\eta)) = \omega_\eta^{J_\eta}(z_\eta)$ for each $\eta \in L$.*

Proof For each $\eta \in L$ let $\lambda_B^\eta : (Z_B^\eta, r_K^\eta) \rightarrow (Z_B, r_K)$ be the unique homomorphism such that $\lambda_\xi^\eta(\xi) = \xi$ for each $\xi \in J_\eta$ (i.e., the homomorphism corresponding to the assignment $i_\eta \in Z_\diamond^{J_\eta}$ given by $i_\eta(\xi) = \xi$ for each $\xi \in J_\eta$). Define $d \in Z_\diamond^L$ by putting $d(\eta) = \lambda_B^\eta(z_\eta)$ for each $\eta \in L$. Then it is easily checked (using the uniqueness in the definition of $\varrho_B^{b_\eta}$) that $\varrho_B^{b_\eta} = \varrho_B^b \lambda_B^\eta$, and hence that $\varrho_\eta^b(d(\eta)) = \varrho_\eta^{b_\eta}(z_\eta)$ for each $\eta \in L$. In the same way $\omega_B^{J_\eta} = \omega_B^J \lambda_B^\eta$, thus $\omega_\eta^J(d(\eta)) = \omega_\eta^{J_\eta}(z_\eta)$ for each $\eta \in L$. \square

Let $d \in Z_\diamond^L$ be the assignment in Lemma 3.4.1. Then $\varrho_\eta^b(d(\eta)) = \varrho_\eta^{b_\eta}(z_\eta) = a(\eta)$ and $\omega_\eta^J(d(\eta)) = \omega_\eta^{J_\eta}(z_\eta) = c(\eta)$ for each $\eta \in L$; in other words, $(\varrho_\diamond^b)^L(d) = a$ and $(\omega_\diamond^J)^L(d) = c$. Now put $z = r_\kappa(d)$; then

$$\varrho_\beta^b(z) = \varrho_\beta^b(r_\kappa(d)) = q_\kappa((\varrho_\diamond^b)^L(d)) = q_\kappa(a)$$

and in the same way $\omega_\beta^J(z) = f_\kappa(c) = u$, i.e., $q_\kappa(a) \in Y'_\beta$. This shows that the family Y'_B is invariant in (Y_B, q_K) , which completes the proof of Proposition 3.4.4. \square

The following corollary of Proposition 3.4.4 will be needed in Section 4.1:

Proposition 3.4.5 *Suppose \prec_B is a full compatible family. Let $\kappa \in K$ be of type $L \rightarrow \beta$ and let $c \in D_\diamond^L$, $a \in Y_\diamond^L$ with $c \prec_\diamond^L a$. Then there exists a finite B -typed set J , a J -free Λ -algebra (Z_B, r_K) , and elements $z \in Z_\beta$, $b \in D_\diamond^J$ such that $f_\kappa(c) = \omega_\beta^J(z)$ and $q_\kappa(a) = \varrho_\beta^b(z)$.*

Proof The case $L = \emptyset$ was already dealt with in the proof of Proposition 3.4.4, and so it can be assumed that $L \neq \emptyset$. Then $c(\eta) \prec_\eta a(\eta)$ for each $\eta \in L$ and hence by Proposition 3.4.4 there exists a finite B -typed set J_η , a J_η -free Λ -algebra (Z_B^η, r_K^η) , and elements $z_\eta \in Z_\eta^J$ and $b_\eta \in D_\diamond^{J_\eta}$ with $\omega_\eta^J(z_\eta) = c(\eta)$ and $\varrho_\eta^{b_\eta}(z_\eta) = a(\eta)$. The proof is now exactly the same as in the proof of Proposition 3.4.4: Again it can be supposed that the sets J_η , $\eta \in L$, are disjoint, put $J = \bigcup_{\eta \in L} J_\eta$, let (Z_B, r_K) be any J -free Λ -algebra and $b \in D_\diamond^J$ be the sum of the assignments b_η , $\eta \in L$. Moreover, take the same assignment $d \in Z_\diamond^L$ given by Lemma 3.4.1 and put $z = r_\kappa(d)$, and as before it follows that $f_\kappa(c) = \omega_\beta^J(z)$ and $q_\kappa(a) = \varrho_\beta^b(z)$. \square

The next result is a special case of Proposition 3.4.4 and is the key to proving the converse of Proposition 3.4.1.

Lemma 3.4.2 *Let $\check{\pi}_B$ and π_B be homomorphisms from (Y_B, q_K) to (D_B, f_K) with $\check{\pi}_B$ fat bottomed. Then for each $y \in Y_\beta$ there exists a finite B -typed set J , a J -free Λ -algebra (Z_B, r_K) , and elements $z \in Z_\beta$ and $b \in D_\circ^J$ such that $\pi_\beta(y) = \pi_\beta^b(z)$ and $\check{\pi}_\beta(y) = \pi_\beta^{\perp^J}(z)$.*

Proof For each $\beta \in B$ define a relation \prec_β on $D_\beta \times Y_\beta$ by stipulating that $u \prec_\beta y$ if and only if $u = \check{\pi}_\beta(y)$. Consider $\kappa \in K$ of type $L \rightarrow \beta$ and let $a \in Y_\circ^L$, $u \in D_\beta$ be such that $u \prec_\beta q_\kappa(a)$. Then $u = \check{\pi}_\beta(q_\kappa(a)) = f_\kappa(c)$ with $c = \check{\pi}_\circ^L(a)$, and so $c(\eta) = \check{\pi}_\eta(a(\eta))$ for each $\eta \in L$, i.e., $c \prec_\circ^L a$, which means the family of relations \prec_B is compatible. Moreover, $y \in Y_\beta$ is a base element if and only if $\check{\pi}_\beta(y) = \perp_\beta$, and hence the set of base elements of Y_β is just \check{U}_β , where \check{U}_B is the kernel of $\check{\pi}_B$. This implies that \prec_B is full, since $\check{\pi}_B$ is fat bottomed. Let $y \in Y_\beta$; then by Proposition 3.4.4 there exists a finite B -typed set J , a J -free Λ -algebra (Z_B, r_K) , and elements $z \in Z_\beta$ and $b' \in Y_\circ^J$ such that $\varrho_\beta^{b'}(z) = y$ and $\pi_\beta^{\perp^J}(z) = \omega_\beta^J(z) = \check{\pi}_\beta(y)$. Now let $b \in D_\circ^J$ be the assignment given by $b(\eta) = \pi_\eta(b'(\eta))$ for each $\eta \in L$. Then clearly $\pi_B^b = \pi_B \varrho_B^{b'}$, and therefore $\pi_\beta^b(z) = \pi_\beta(\varrho_\beta^{b'}(z)) = \pi_\beta(y)$. \square

Proposition 3.4.6 *A monotone bottomed Λ -algebra (D_B, f_K) is strongly monotone.*

Proof This follows immediately from Lemma 3.4.2. \square

The proof of Proposition 3.4.6 shows that in the definition of being monotone it would not change anything if the B -typed sets were restricted to being finite.

Proposition 3.4.7 *A bottomed Λ -algebra possessing a monotone extension is itself monotone.*

Proof This follows from the equivalence of monotonicity and strong monotonicity, and the fact that if (D'_B, f'_K) is an extension of (D_B, f_K) and $\pi_B : (Y_B, q_K) \rightarrow (D_B, f_K)$ is a homomorphism, and if π_B , considered as a homomorphism from (Y_B, q_K) to (D'_B, f'_K) , is denoted by π'_B then π'_B and π_B have the same kernel (and so in particular π'_B is fat bottomed if and only if π_B is). \square

On the following page two simple examples are presented which show that regularity and monotonicity are independent concepts, i.e., they give a regular extension which is not monotone and a monotone extension which is not regular.

Example 3.4.1 Let (X_B^\perp, p_K^\perp) be the flat extension of the Λ -algebra (X_B, p_K) from Example 2.2.1 and define Λ -algebras (Y_B, q_K) and (Y_B, q'_K) by letting

$$Y_{\mathbf{nat}} = \mathbb{N} \cup \{\perp_{\mathbf{nat}}, \perp_{\mathbf{nat}}^o\} \text{ with } \perp_{\mathbf{nat}}^o \notin \mathbb{N} \cup \{\perp_{\mathbf{nat}}\},$$

$$Y_\beta = X_\beta^\perp \text{ for all } \beta \in B \setminus \{\mathbf{nat}\},$$

$$q_{\mathbf{Zero}}, q'_{\mathbf{Zero}} : \mathbb{I} \rightarrow Y_{\mathbf{nat}} \text{ with } q_{\mathbf{Zero}}(\varepsilon) = q'_{\mathbf{Zero}}(\varepsilon) = 0,$$

$$q_{\mathbf{Succ}} : Y_{\mathbf{nat}} \rightarrow Y_{\mathbf{nat}} \text{ with}$$

$$q_{\mathbf{Succ}}(n) = \begin{cases} n+1 & \text{if } n \in \mathbb{N}, \\ \perp_{\mathbf{nat}} & \text{if } n = \perp_{\mathbf{nat}}^o, \\ \perp_{\mathbf{nat}}^o & \text{if } n = \perp_{\mathbf{nat}}, \end{cases}$$

$$q'_{\mathbf{Succ}} : Y_{\mathbf{nat}} \rightarrow Y_{\mathbf{nat}} \text{ with}$$

$$q'_{\mathbf{Succ}}(n) = \begin{cases} n+1 & \text{if } n \in \mathbb{N}, \\ \perp_{\mathbf{nat}}^o & \text{if } n \in \{\perp_{\mathbf{nat}}, \perp_{\mathbf{nat}}^o\}, \end{cases}$$

$$\text{and with } q_\kappa = q'_\kappa = p_\kappa^\perp \text{ for all } \kappa \in K \setminus \{\mathbf{Zero}, \mathbf{Succ}\}.$$

Then (Y_B, q_K) is clearly a minimal regular extension of (X_B, p_K) . However, (Y_B, q'_K) is not monotone. To see this consider a B -typed set $J = \{\mathbf{n}\}$ with \mathbf{n} of type \mathbf{nat} and let (Z_B, r_K) be any J -free Λ -algebra. Moreover, let z be the element $r_{\mathbf{Succ}}(\mathbf{n})$ of $Z_{\mathbf{nat}}$ and $b \in Y_\diamond^J$ be such that $b(\mathbf{n}) = \perp_{\mathbf{nat}}^o$. Then $\pi_{\mathbf{nat}}^b(z) = q_{\mathbf{Succ}}(b(\mathbf{n})) = q_{\mathbf{Succ}}(\perp_{\mathbf{nat}}^o) = \perp_{\mathbf{nat}}$ but

$$\pi_{\mathbf{nat}}^{\perp^J}(z) = q_{\mathbf{Succ}}(\perp^J(\mathbf{n})) = q_{\mathbf{Succ}}(\perp_{\mathbf{nat}}) = \perp_{\mathbf{nat}}^o \neq \perp_{\mathbf{nat}}.$$

On the other hand, (Y_B, q'_K) is a minimal extension of (X_B, p_K) which is not regular, since $q'_{\mathbf{Succ}}(\perp_{\mathbf{nat}}) = q'_{\mathbf{Succ}}(\perp_{\mathbf{nat}}^o) = \perp_{\mathbf{nat}}^o \neq \perp_{\mathbf{nat}}$. However, the reader is left to show that (Y_B, q'_K) is monotone. (This will later follow more easily from Proposition 4.1.6.)

3.5 Cored extensions

In this section some simple classes of bottomed extensions are introduced which can be seen as generalising the flat and fully regular extensions. They provide fairly explicit examples of monotone regular extensions.

For each $\beta \in B$ let \mathfrak{h}_β be an element different from \mathfrak{b}_β and put $H_\beta = \{\mathfrak{b}_\beta, \mathfrak{h}_\beta\}$; H_β is considered as a bottomed set with bottom element \mathfrak{b}_β . If L is a B -typed set then \mathfrak{h}^L will denote the element of H_\diamond^L defined by $\mathfrak{h}^L(\eta) = \mathfrak{h}_\eta$ for each $\eta \in L$. Let (H_B, \diamond_K) be a Λ -algebra; then (H_B, \diamond_K) is called a *core type* if $\diamond_\kappa(\mathfrak{h}^L) = \mathfrak{h}_\beta$ whenever $\kappa \in K$ is of type $L \rightarrow \beta$. In particular, $\diamond_\kappa(\varepsilon) = \mathfrak{h}_\beta$ whenever κ is of type $\emptyset \rightarrow \beta$ for some $\beta \in B$. A core type is thus essentially a finite object, provided the set of constructor names in K which do not have a type of the form $\emptyset \rightarrow \beta$ is finite. Now let (D_K, f_K) be a bottomed Λ -algebra and for each $\beta \in B$ define a mapping $\varepsilon_\beta : D_\beta \rightarrow H_\beta$ by

$$\varepsilon_\beta(u) = \begin{cases} \mathfrak{h}_\beta & \text{if } u \neq \perp_\beta, \\ \mathfrak{b}_\beta & \text{if } u = \perp_\beta. \end{cases}$$

Let (H_B, \diamond_K) be a core type; then (D_B, f_K) is said to be (H_B, \diamond_K) -*cored* if the family ε_B is actually a homomorphism (and thus a bottomed homomorphism) from (D_B, f_K) to (H_B, \diamond_K) . In other words, if for each $\kappa \in K$ of type $L \rightarrow \beta$

$$\varepsilon_\beta(f_\kappa(c)) = \diamond_\kappa(\varepsilon_\diamond^L(c))$$

for all $c \in D_\diamond^L$. This means that if $\kappa \in K$ is of type $L \rightarrow \beta$ then whether or not $f_\kappa(c) \neq \perp_\beta$ holds is completely determined by \diamond_κ and the set $\{\eta \in L : c(\eta) \neq \perp_\eta\}$.

A bottomed extension (D_B, f_K) of (X_B, p_K) is now said to be (H_B, \diamond_K) -*cored* if (D_B, f_K) is an (H_B, \diamond_K) -cored bottomed Λ -algebra. Note that such an extension can be (H_B, \diamond_K) -cored for at most one core type (H_B, \diamond_K) : This follows since the signature Λ is assumed to be pervasive and hence $X_\beta \neq \emptyset$ for each $\beta \in B$.

The class of those bottomed Λ -algebras which are (H_B, \diamond_K) -cored for some core type (H_B, \diamond_K) includes each fully regular bottomed Λ -algebra and the flat extension of (X_B, p_K) : For each $\kappa \in K$ of type $L \rightarrow \beta$ let $\diamond_\kappa^\top : H_\diamond^L \rightarrow H_\beta$ be the mapping with $\diamond_\kappa^\top(c) = \mathfrak{h}_\beta$ for all $c \in H_\diamond^L$, and let $\diamond_\kappa^\perp : H_\diamond^L \rightarrow H_\beta$ be given by

$$\diamond_\kappa^\perp(c) = \begin{cases} \mathfrak{h}_\beta & \text{if } c = \mathfrak{h}^L, \\ \mathfrak{b}_\beta & \text{otherwise.} \end{cases}$$

Then (H_B, \diamond_K^\top) and (H_B, \diamond_K^\perp) are both core types. Moreover, the flat extension is (H_B, \diamond_K^\perp) -cored and each fully regular bottomed Λ -algebra is (H_B, \diamond_K^\top) -cored.

Proposition 3.5.1 *Let (H_B, \diamond_K) be a core type; then there is a unique isomorphism class of minimal regular (H_B, \diamond_K) -cored bottomed Λ -algebras. Moreover, there is a unique conjugacy class of minimal regular (H_B, \diamond_K) -cored extensions of (X_B, p_K) .*

Proof This is given at the end of the section. \square

It will be shown next that if (H_B, \diamond_K) is a core type and the mappings in the family \diamond_K are monotone (in a sense to be defined below) then any (H_B, \diamond_K) -cored bottomed Λ -algebra is monotone.

For each $\beta \in B$ let \leq_β be the partial order defined on the set $H_\beta = \{b_\beta, \natural_\beta\}$ by requiring that $b_\beta \leq_\beta \natural_\beta$. Let $\kappa \in K$ be of type $L \rightarrow \beta$; then $\diamond_\kappa : H_\diamond^L \rightarrow H_\beta$ is *monotone* if $\diamond_\kappa(b) \leq_\beta \diamond_\kappa(b')$ whenever $b, b' \in H_\diamond^L$ are such that $b \leq^L b'$, where $b \leq^L b'$ if and only if $b(\eta) \leq_\eta b'(\eta)$ for each $\eta \in L$. The core type (H_B, \diamond_K) is then said to be *monotone* if \diamond_κ is monotone for each $\kappa \in K$.

In particular the core types (H_B, \diamond_K^\perp) and (H_B, \diamond_K^\top) are monotone. The following is thus a generalisation of Proposition 3.4.3:

Proposition 3.5.2 *Suppose that (D_B, f_K) is (H_B, \diamond_K) -cored for some monotone core type (H_B, \diamond_K) . Then (D_B, f_K) is structurally monotone (and thus monotone).*

Proof Let (Y_B, q_K) be a Λ -algebra, let π_B and π'_B be homomorphisms from (Y_B, q_K) to (D_B, f_K) , and define a family Y'_B with $Y'_B \subset Y_B$ by letting

$$Y'_\beta = \{y \in Y_\beta : \pi'_\beta(y) = \perp_\beta \text{ whenever } \pi_\beta(y) = \perp_\beta\}$$

for each $\beta \in B$. Note that if $u \in D_\beta$ then $u = \perp_\beta$ if and only if $\varepsilon_\beta(u) = b_\beta$, thus

$$Y'_\beta = \{y \in Y_\beta : \varepsilon_\beta(\pi'_\beta(y)) \leq_\beta \varepsilon_\beta(\pi_\beta(y))\}.$$

If $\kappa \in K$ is of type $\emptyset \rightarrow \beta$ then $\pi_\beta(q_\kappa(\varepsilon)) = f_\kappa(\varepsilon) \neq \perp_\beta$, and so $q_\kappa(\varepsilon)$ is trivially an element of Y'_β . Thus consider $\kappa \in K$ of type $L \rightarrow \beta$ with $L \neq \emptyset$ and let $c \in Y_\diamond^L$ with $c(\eta) \in Y'_\eta$ for each $\eta \in L$. Thus $\varepsilon_\eta(\pi'_\eta(c(\eta))) \leq_\eta \varepsilon_\eta(\pi_\eta(c(\eta)))$ for each $\eta \in L$, which implies that

$$\begin{aligned} \varepsilon_\beta(\pi'_\beta(q_\kappa(c))) &= \varepsilon_\beta(f_\kappa((\pi'_\diamond)^L(c))) = \diamond_\kappa(\varepsilon_\diamond^L((\pi'_\diamond)^L(c))) \\ &\leq_\beta \diamond_\kappa(\varepsilon_\diamond^L(\pi_\diamond^L(c))) = \varepsilon_\beta(f_\kappa(\pi_\diamond^L(c))) = \varepsilon_\beta(\pi_\beta(q_\kappa(c))), \end{aligned}$$

and therefore $q_\kappa(c) \in Y'_\beta$. This shows the family Y'_B is invariant in (Y_B, q_K) and thus that (D_B, f_K) is structurally monotone. \square

The next topic considered is the trace of an (H_B, \diamond_K) -cored bottomed Λ -algebra. This turns out to only depend on (H_B, \diamond_K) and it can be computed more-or-less explicitly. In what follows let the core type (H_B, \diamond_K) be fixed. Since the bottomed ground term algebra is initial there exists a unique bottomed homomorphism δ_B^b from (F_B^b, \odot_K^b) to (H_B, \diamond_K) , i.e., $\delta_B^b : (F_B^b, \odot_K^b) \rightarrow (H_B, \diamond_K)$ is the unique homomorphism such that $\delta_B^b(b_\beta) = b_\beta$ for each $\beta \in B$.

Proposition 3.5.3 *Suppose that (D_B, f_K) is (H_B, \diamond_K) -cored. Then*

$$\delta_B^b = \varepsilon_B \llbracket \cdot \rrbracket_B^\perp,$$

i.e., $\delta_B^b(s) = \varepsilon_\beta(\llbracket s \rrbracket_\beta^\perp)$ for all $s \in F_\beta^b$, $\beta \in B$. In particular, if R_B is the trace of (D_B, f_K) then $R_\beta = \{s \in F_\beta^b : \delta_B^b(s) = \natural_\beta\}$ for each $\beta \in B$.

Proof By Proposition 2.2.1 $\varepsilon_B \llbracket \cdot \rrbracket_B^\perp$ is a homomorphism from $(F_B^\flat, \odot_K^\flat)$ to (H_B, \diamond_K) and by definition $\varepsilon_B(\llbracket \flat_\beta \rrbracket_B^\perp) = \flat_\beta$ for each $\beta \in B$. But δ_B^\flat is the unique such homomorphism and therefore $\delta_B^\flat = \varepsilon_B \llbracket \cdot \rrbracket_B^\perp$. \square

Proposition 3.5.3 implies that if (D_B, f_K) is (H_B, \diamond_K) -cored then the trace of (D_B, f_K) is determined by (H_B, \diamond_K) . In fact, more explicitly, the following holds:

Proposition 3.5.4 *There is a unique family $R_B \subset F_B^\flat$ satisfying the following:*

- (i) $\flat_\beta \notin R_B$ for each $\beta \in B$.
- (ii) If $\kappa \in K$ is of type $\emptyset \rightarrow \beta$ for some $\beta \in B$ then $\odot_\kappa^\flat(\varepsilon) \in R_B$.
- (iii) If $\kappa \in K$ is of type $L \rightarrow \beta$ with $L \neq \emptyset$ and $c \in (F_\diamond^\flat)^L$ then $\odot_\kappa^\flat(c) \in R_B$ if and only if $\diamond_\kappa(b) = \flat_\beta$, where $b \in H_\diamond^L$ is given by

$$b(\eta) = \begin{cases} \flat_{\beta_j} & \text{if } c(\eta) \in R_\eta, \\ \flat_\eta & \text{otherwise.} \end{cases}$$

Moreover, R_B is the trace of each (H_B, \diamond_K) -cored bottomed Λ -algebra.

Proof For each $\beta \in B$ let $R_\beta = \{s \in F_\beta^\flat : \delta_\beta^\flat(s) = \flat_\beta\}$; then the definition of the family δ_B^\flat implies that R_B satisfies (i), (ii) and (iii). Moreover, by Proposition 3.5.3 R_B is the trace of each (H_B, \diamond_K) -cored bottomed Λ -algebra. It thus remains to show that R_B is the unique family satisfying these conditions and this easily follows from the uniqueness of the homomorphism δ_B^\flat . \square

Besides (H_B, \diamond_K^\top) and (H_B, \diamond_K^\perp) there is a further core type which should be mentioned. If $\kappa \in K$ is of type $L \rightarrow \beta$ with $L \neq \emptyset$ then let $\diamond_\kappa^c : H_\diamond^L \rightarrow H_\beta$ be given by

$$\diamond_\kappa^c(b) = \begin{cases} \flat_\beta & \text{if } b(\eta) = \flat_\eta \text{ for at least one } \eta \in L, \\ \flat_\beta & \text{otherwise,} \end{cases}$$

and note that \diamond_κ^c is monotone. A type $\beta \in B$ is called a *product type* if K_β consists of exactly one constructor name κ and κ is of type $L \rightarrow \beta$ with L containing at least two elements and $\langle \eta \rangle \neq \beta$ for each $\eta \in L$. (In the signature Λ in Example 2.2.1, for instance, `ipr` is a product type.) Now define a core type (H_B, \diamond'_K) by letting

$$\diamond'_\kappa = \begin{cases} \diamond_\kappa^c & \text{if } \kappa \text{ is the single constructor for some product type,} \\ \diamond_\kappa^\top & \text{otherwise.} \end{cases}$$

Then (H_B, \diamond'_K) is a monotone core type which is sometimes used instead of (H_B, \diamond_K^\top) , and the reason for perhaps preferring (H_B, \diamond'_K) to (H_B, \diamond_K^\top) will now be explained. Let $\beta \in B$ be a product type with $K_\beta = \{\kappa\}$ and with κ of type $L \rightarrow \beta$. Then by Proposition 2.3.2 the mapping $p_\kappa : X_\diamond^L \rightarrow X_\beta$ is a bijection, and in most cases X_β will just taken to be X_\diamond^L (and hence the name *product type*). For simplicity suppose that this has been done. Consider any regular extension (D_B, f_K) of (X_B, p_K) with core

C_K , and note that $X_\beta = X_\diamond^L \subset C_\kappa$. Then by Lemma 3.3.1 f_κ maps C_κ bijectively onto $D_\beta \setminus \{\perp_\beta\}$ and thus, assuming $\perp_\beta \notin C_\kappa$, D_β can be identified with $C_\kappa \cup \{\perp_\beta\}$. Now if (D_B, f_K) is (H_B, \diamond_K^\top) -cored then $C_\kappa = \text{Dom}(f_\kappa) = D_\diamond^L$, which means that $D_\beta = D_\diamond^L \cup \{\perp_\beta\}$. On the other hand, if (D_B, f_K) is (H_B, \diamond'_K) -cored then

$$C_\kappa = D_\diamond^L \setminus \{\perp^L\}$$

and hence $D_\beta = D_\diamond^L$, provided \perp^L is identified with \perp_β . A reason for preferring (H_B, \diamond'_K) to (H_B, \diamond_K^\top) is thus to obtain a more ‘natural’ bottomed product in this case.

Proof of Proposition 3.5.1 It is easily checked that a bottomed Λ -algebra which is isomorphic to a (H_B, \diamond_K) -cored bottomed Λ -algebra is itself (H_B, \diamond_K) -cored. Thus by Propositions 3.3.4 and 3.5.3 it is enough to show that there exists a minimal regular (H_B, \diamond_K) -cored extension of (X_B, p_K) .

To establish this start with any initial bottomed extension $(\check{D}_B, \check{f}_K)$ of (X_B, p_K) (whose existence is guaranteed by Proposition 3.2.7) and let $\delta_B : (\check{D}_B, \check{f}_K) \rightarrow (H_B, \diamond_K)$ be the unique homomorphism with $\delta_\beta(\perp_\beta) = \flat_\beta$ for each $\beta \in B$. For each $\beta \in B$ put

$$D'_\beta = \{u \in \check{D}_\beta : \delta_\beta(u) = \varepsilon_\beta(u)\};$$

thus $D'_\beta = \{u \in \check{D}_\beta : \delta_\beta(u) = \flat_\beta\} \cup \{\perp_\beta\}$, and in particular $X_\beta \subset D'_\beta$. For each $\kappa \in K$ of type $L \rightarrow \beta$ define a mapping $f'_\kappa : \check{D}_\diamond^L \rightarrow \check{D}_\beta$ by

$$f'_\kappa(c) = \begin{cases} \check{f}_\kappa(c) & \text{if } \check{f}_\kappa(c) \in D'_\beta, \\ \perp_\beta & \text{otherwise.} \end{cases}$$

If $c \in X_\diamond^L$ then $\check{f}_\kappa(c) = p_\kappa(c) \in X_\beta \subset D'_\beta$ and therefore $f'_\kappa(c) = \check{f}_\kappa(c) = p_\kappa(c)$, and thus (\check{D}_B, f'_K) is a Λ -algebra which is a bottomed extension of (X_B, p_K) . Moreover, $\varepsilon_\beta(f'_\kappa(c)) = \delta_\beta(\check{f}_\kappa(c))$ for all $c \in \check{D}_\diamond^L$ whenever $\kappa \in K$ is of type $L \rightarrow \beta$: If $\check{f}_\kappa(c) \in D'_\beta$ then $f'_\kappa(c) = \check{f}_\kappa(c)$ and so $\varepsilon_\beta(f'_\kappa(c)) = \varepsilon_\beta(\check{f}_\kappa(c)) = \delta_\beta(\check{f}_\kappa(c))$; on the other hand, if $\check{f}_\kappa(c) \notin D'_\beta$ then $\delta_\beta(\check{f}_\kappa(c)) = \flat_\beta$ and hence $\delta_\beta(\check{f}_\kappa(c)) = \varepsilon_\beta(\perp_\beta) = \varepsilon_\beta(f'_\kappa(c))$.

Let C_K be the core of (\check{D}_B, f'_K) ; then the restriction of f'_κ to C_κ is injective for each $\kappa \in K$ and $f'_{\kappa_1}(C_{\kappa_1})$ and $f'_{\kappa_2}(C_{\kappa_2})$ are disjoint subsets of \check{D}_β whenever $\kappa_1, \kappa_2 \in K_\beta$ with $\kappa_1 \neq \kappa_2$ (since by Lemma 3.3.1 the corresponding statements hold for the family \check{f}_K if C_κ is replaced by $\text{Dom}(\check{f}_\kappa)$ for each $\kappa \in K$). In other words, conditions (i) and (ii) in Lemma 3.3.1 hold for (\check{D}_B, f'_K) . Now let (\hat{D}_B, \hat{f}_K) be the minimal bottomed subalgebra of (\check{D}_B, f'_K) , so (\hat{D}_B, \hat{f}_K) is a minimal bottomed extension of (X_B, p_K) . But if \hat{C}_K is the core of (\hat{D}_B, \hat{f}_K) then $\hat{C}_\kappa \subset C_\kappa$ for each $\kappa \in K$ and hence conditions (i) and (ii) in Lemma 3.3.1 also hold for (\hat{D}_B, \hat{f}_K) . Therefore by Lemma 3.3.1 and Proposition 3.2.2 (\hat{D}_B, \hat{f}_K) is regular.

Note that $\hat{D}_B \subset D'_B$, since D'_B is invariant (in (\check{D}_B, f'_K)) and $\perp_\beta \in D'_\beta$ for each $\beta \in B$, and hence $\delta_\beta(u) = \varepsilon_\beta(u)$ for all $u \in \hat{D}_\beta$, $\beta \in B$. Let $\kappa \in K$ be of type

$L \rightarrow \beta$ and $c \in \hat{D}_\diamond^L$. Then

$$\diamond_\kappa(\varepsilon_\diamond^L(c)) = \diamond_\kappa(\delta_\diamond^L(c)) = \delta_\beta(\hat{f}_\kappa(c)) = \varepsilon_\beta(\hat{f}'_\kappa(c)) = \varepsilon_\beta(\hat{f}_\kappa(c))$$

and this implies that (\hat{D}_B, \hat{f}_K) is (H_B, \diamond_K) -cored, i.e., (\hat{D}_B, \hat{f}_K) is a minimal regular (H_B, \diamond_K) -cored extension of (X_B, p_K) . \square

3.6 Notes

The form of the basic data objects (i.e., given by the family of sets in an initial algebra) corresponds to that occurring in the functional programming languages this study is trying to explain. It could be objected that what is also required are data types obtained through equivalence relations defined on initial algebras. For example, this is necessary in order to implement the rational numbers. However, the implementation of such data types has been left to the user, who must make do with data types without equivalence relations and then check that all the functions he or she defines actually respect the equivalence relation specifying the intended data type.

The concept of a bottomed extension has been introduced to give a framework which covers both ‘lazy’ and ‘eager’ programming languages (and also everything between these two extremes). The condition of regularity is exactly what is needed for the ‘case’ operators to make sense.

David Turner’s language *Miranda* (see Turner (1985)) uses the monotone core type (H_B, \diamond'_K) introduced in Section 3.5, whereas *Haskell* and most other ‘lazy’ languages use (H_B, \diamond_K^\top) . The expression *fat bottomed* occurring in the definition of monotonicity is taken of course from Queen (1978).

Chapter 4 Completion of bottomed extensions

In Chapter 3 a framework was introduced for specifying data objects. This involved starting with a ground signature Λ (whose choice depends on the application at hand) and then fixing an initial Λ -algebra (X_B, p_K) to describe the basic data objects. A bottomed extension (D_B, f_K) of (X_B, f_K) was then chosen in order to specify which ‘undefined’ and ‘partially defined’ data objects are to be allowed. For now it will always be assumed that (D_B, f_K) is monotone and regular (for reasons which should become more apparent in the present and the following chapter).

Now it is often convenient, and even necessary, to also consider ‘infinite’ data objects. Such an object can be thought of as representing the limit of some infinite process (for example, a process which produces a new data element at each step), and so they will be typically involved in the description of calculations which continue to produce new data without ever terminating. Moreover, there are also purely mathematical reasons for including a ‘correct’ set of limiting objects, since this results in a set-up which is easier to deal with (in the same way that real analysis, i.e., analysis done over the real numbers \mathbb{R} , is usually much simpler than rational analysis, i.e., analysis done over the rational numbers \mathbb{Q}).

The ‘infinite’ data objects which will occur in this study are described with the help of complete posets (i.e., complete partially ordered sets); these are studied in Section 4.2. The complete posets which will be used arise as initial (or ideal) completions, and so in Section 4.3 it is shown how to construct the initial completion of an arbitrary poset.

Recall that a *partial order* \sqsubseteq on a set Y is a binary relation \sqsubseteq on Y satisfying:

- (i) $y \sqsubseteq y$ for all $y \in Y$.
- (ii) If $y_1 \sqsubseteq y_2$ and $y_2 \sqsubseteq y_1$ then $y_1 = y_2$.
- (iii) If $y_1 \sqsubseteq y_2$ and $y_2 \sqsubseteq y_3$ then $y_1 \sqsubseteq y_3$.

How posets come into the picture at all is explained in Section 4.1. There it is shown that if (D_B, f_K) is a regular bottomed Λ -algebra then for each $\beta \in B$ there exists a partial order \sqsubseteq_β on the set D_β such that $u \sqsubseteq_\beta u'$ can be sensibly interpreted as meaning that u is ‘less-defined’ than u' . More precisely, a family of partial orders \sqsubseteq_B is said to be an ordering associated with (D_B, f_K) if the following two conditions hold:

- (i) D_β is a bottomed poset with bottom element \perp_β , i.e., $\perp_\beta \sqsubseteq_\beta u$ for each $u \in D_\beta$.
- (ii) If $\kappa, \kappa' \in K_\beta$ and $c \in C_\kappa$, $c' \in C_{\kappa'}$ then $f_\kappa(c) \sqsubseteq_\beta f_{\kappa'}(c')$ if and only if $\kappa = \kappa'$ and $c \sqsubseteq_\kappa c'$.

Here C_K is the core of (D_B, f_K) and \sqsubseteq_κ is the appropriate ‘product’ partial order on $\text{Dom}(f_\kappa)$. (More precisely, if κ is of type $L \rightarrow \beta$ then \sqsubseteq_κ is just the partial order \sqsubseteq_\diamond^L on D_\diamond^L .) In Proposition 4.1.2 it is shown that there exists an ordering associated with (D_B, f_K) . Moreover, if (D_B, f_K) is also minimal then it turns out that this associated ordering is unique.

An associated ordering \sqsubseteq_B is said to be monotone if the mappings f_κ , $\kappa \in K$, are monotone with respect to \sqsubseteq_B , i.e., if $\kappa \in K$ is of type $L \rightarrow \beta$ and $c, c' \in D_\diamond^L$ with $c \sqsubseteq_\kappa c'$ then $f_\kappa(c) \sqsubseteq_\beta f_\kappa(c')$. Now, although it might be expected that an associated ordering is automatically monotone, this is not the case. However, Proposition 4.1.5 states that if (D_B, f_K) is a minimal monotone regular bottomed Λ -algebra then the (unique) associated ordering \sqsubseteq_B is monotone.

The results of Sections 4.2 and 4.3 are applied in Section 4.4, where the initial completion of a Λ -algebra is defined. More precisely, if (Y_B, q_K) is a monotone regular extension of (X_B, p_K) and \sqsubseteq_B is an associated ordering which is monotone then Y_β is regarded as a poset with the partial order \sqsubseteq_β for each $\beta \in B$. The initial completion of (Y_B, q_K) is then a Λ -algebra (D_B, f_K) in which D_β is an initial completion of the poset Y_β for each $\beta \in B$ and such that f_κ is the unique continuous extension of q_κ for each $\kappa \in K$. Proposition 4.3.2 states that an initial completion (D_B, f_K) exists, and Proposition 4.4.1 implies it is, in the appropriate sense, unique. Moreover, by Proposition 4.4.2 (D_B, f_K) is then a monotone regular extension of (X_B, p_K) , and the partial orders occurring in the completion constitute an ordering associated with (D_B, f_K) which is monotone.

The results of this chapter can be applied to give the following ‘canonical’ procedure for obtaining a complete bottomed extension (D_B, f_K) of (X_B, p_K) :

1. Start by choosing a minimal monotone regular extension (Y_B, q_K) of (X_B, p_K) : Proposition 3.3.4 implies that such an extension is essentially uniquely determined by its trace.
2. By Proposition 4.1.2 there is then a unique ordering \sqsubseteq_B associated with (Y_B, q_K) , which by Proposition 4.1.5 is monotone.
3. Next let (D_B, f_K) be an initial completion of (Y_B, q_K) : The existence of such a completion is guaranteed by Proposition 4.3.2, and Proposition 4.4.1 implies that it is essentially unique.
4. By Proposition 4.4.2 (D_B, f_K) is then a monotone regular extension of (X_B, p_K) . Moreover, it follows that (D_B, f_K) has the same trace as (Y_B, q_K) , which implies that (D_B, f_K) is essentially uniquely determined by this trace.

4.1 Associated orderings for regular extensions

In this section it is shown that if (D_B, f_K) is a regular bottomed Λ -algebra then for each $\beta \in B$ there exists a partial order \sqsubseteq_β on the set D_β such that $u \sqsubseteq_\beta u'$ can be sensibly interpreted as meaning that u is 'less-defined' than u' . Moreover, if (D_B, f_K) is also minimal then it turns out that this family \sqsubseteq_B of partial orders is unique.

First some notation and conventions concerning partial orders must be fixed. If \sqsubseteq is a partial order on a non-empty set Y then the pair (Y, \sqsubseteq) is called a *partially ordered set* or, for short, a *poset*. However, instead of (Y, \sqsubseteq) it is usual just to write Y , and to assume that \sqsubseteq can be determined from the context. Something like ' Y is a poset with partial order \sqsubseteq ' or ' \sqsubseteq is the partial order on Y ' can be employed when it is necessary to refer to the partial order explicitly. It is useful to consider the set \mathbb{I} as a poset (naturally with respect to the unique partial order on \mathbb{I}).

Let $\{(D_\beta, \sqsubseteq_\beta) : \beta \in B\}$ be a family of posets and let I be a B -typed set. Then D_\diamond^I is considered as a poset with the partial order \sqsubseteq_\diamond^I defined by stipulating that $c \sqsubseteq_\diamond^I c'$ if and only if $c(\eta) \sqsubseteq_\eta c'(\eta)$ for each $\eta \in I$. If $\kappa \in K$ is of type $L \rightarrow \beta$ then it is convenient to allow \sqsubseteq_κ as an alternative notation for the partial order \sqsubseteq_\diamond^L on D_\diamond^L .

Let Y be a poset with partial order \sqsubseteq ; then there can be at most one element $\perp_Y \in Y$ with $\perp_Y \sqsubseteq y$ for all $y \in Y$. If such an element \perp_Y exists then Y is said to be a *bottomed poset* with *bottom element* \perp_Y .

Let (D_B, f_K) be a bottomed Λ -algebra with core C_K (so

$$C_\kappa = \{c \in \text{Dom}(f_\kappa) : f_\kappa(c) \neq \perp_\beta\}$$

for each $\kappa \in K_\beta$). For each $\beta \in B$ let \sqsubseteq_β be a partial order on the set D_β . Then the family of partial orders \sqsubseteq_B will be called an *ordering associated* with (D_B, f_K) (or just an *associated ordering*) if the following two conditions hold:

- (i) D_β is a bottomed poset with bottom element \perp_β , i.e., $\perp_\beta \sqsubseteq_\beta u$ for each $u \in D_\beta$.
- (ii) If $\kappa, \kappa' \in K_\beta$ and $c \in C_\kappa$, $c' \in C_{\kappa'}$ then $f_\kappa(c) \sqsubseteq_\beta f_{\kappa'}(c')$ if and only if $\kappa = \kappa'$ and $c \sqsubseteq_\kappa c'$.

The definition of an ordering associated with (D_B, f_K) is chosen to be compatible with the interpretation mentioned above: If $u, u' \in D_\beta$ then $u \sqsubseteq_\beta u'$ should be thought of as meaning that u is 'less-defined' than u' .

Proposition 4.1.2 states that there exists a unique ordering associated with each minimal regular bottomed Λ -algebra. In the simple case where (D_B, f_K) is the flat extension of (X_B, p_K) it can be seen directly that the unique associated ordering is obtained by letting $u \sqsubseteq_\beta u'$ if and only if $u \in \{\perp_\beta, u'\}$. For the initial extension (X_B^\top, p_K^\top) introduced in in Example 3.2.2 the unique associated ordering is given in Example 4.1.1 below.

Before coming to Proposition 4.1.2 it will be shown that the converse is essentially true.

Example 4.1.1 It is straightforward to check that the following defines an (and thus the) ordering \sqsubseteq_B associated with the initial extension (X_B^\top, p_K^\top) of the Λ -algebra (X_B, p_K) introduced in Example 3.2.2.

If $b, b' \in X_{\text{bool}}^\top$ with $b \neq b'$ then $b \sqsubseteq_{\text{bool}} b'$ if and only if $b = \perp_{\text{bool}}$.

Let $n \in \mathbb{N}$ and $x \in X_{\text{nat}}^\top$. Then $x \sqsubseteq_{\text{nat}} n$ if and only if either $x = n$ or $x = m^\perp$ for some $m \leq n$. Moreover, $x \sqsubseteq_{\text{nat}} n^\perp$ if and only if $x = m^\perp$ for some $m \leq n$.

If $n, n' \in X_{\text{int}}^\top$ with $n \neq n'$ then $n \sqsubseteq_{\text{int}} n'$ if and only if $n = \perp_{\text{int}}$.

If $p, p' \in X_{\text{ipr}}^\top$ with $p' \neq \perp_{\text{ipr}}$ then $p' \sqsubseteq_{\text{ipr}} p$ if and only if $p' = (x', y')$ and $p = (x, y)$ with $x' \sqsubseteq_{\text{int}} x$ and $y' \sqsubseteq_{\text{int}} y$.

Let $\ell = x_1 \cdots x_n \in (X_{\text{int}}^\top)^*$ and $z' \in X_{\text{ilst}}^\top$. Then $z' \sqsubseteq_{\text{ilst}} \ell$ if and only if either $z' = x'_1 \cdots x'_n$ with $x'_j \sqsubseteq_{\text{int}} x_j$ for $j = 1, \dots, n$ or $z' = (x'_1 \cdots x'_m)^\perp$ with $m \leq n$ and $x'_j \sqsubseteq_{\text{int}} x_j$ for $j = 1, \dots, m$. Moreover, $z' \sqsubseteq_{\text{ilst}} \ell^\perp$ if and only if $z' = (x'_1 \cdots x'_m)^\perp$ with $m \leq n$ and $x'_j \sqsubseteq_{\text{int}} x_j$ for $j = 1, \dots, m$.

(Recall here that

$$\begin{aligned} X_{\text{bool}}^\top &= \mathbb{B}^\perp = \{T, F, \perp_{\text{bool}}\}, \\ X_{\text{nat}}^\top &= \mathbb{N} \cup \text{bot}(\mathbb{N}) \text{ with } \mathbf{0}^\perp = \perp_{\text{nat}}, \\ X_{\text{int}}^\top &= \mathbb{Z} \cup \{\perp_{\text{int}}\}, \quad X_{\text{ipr}}^\top = (X_{\text{int}}^\top)^2 \cup \{\perp_{\text{ipr}}\}, \\ X_{\text{ilst}}^\top &= (X_{\text{int}}^\top)^* \cup \text{bot}((X_{\text{int}}^\top)^*) \text{ with } \varepsilon^\perp = \perp_{\text{ilst}}. \end{aligned}$$

Proposition 4.1.1 *Let (D_B, f_K) be a bottomed Λ -algebra with core C_K for which there exists an associated ordering \sqsubseteq_B . Then:*

- (i) *The restriction of f_κ to C_κ is injective for each $\kappa \in K$.*
- (ii) *If $\beta \in B$ and $\kappa_1, \kappa_2 \in K_\beta$ with $\kappa_1 \neq \kappa_2$ then $f_{\kappa_1}(C_{\kappa_1})$ and $f_{\kappa_2}(C_{\kappa_2})$ are disjoint subsets of D_β .*

In particular, any minimal bottomed Λ -algebra possessing an associated ordering is regular.

Proof Let $\kappa \in K_\beta$ and $c, c' \in C_\kappa$ with $f_\kappa(c) = f_\kappa(c')$. Then $f_\kappa(c) \sqsubseteq_\beta f_\kappa(c')$ and hence $c \sqsubseteq_\kappa c'$; in the same way $c' \sqsubseteq_\kappa c$. Thus $c = c'$, which shows the restriction of f_κ to C_κ is injective. Next consider $\kappa_1, \kappa_2 \in K_\beta$ and let $u \in f_{\kappa_1}(C_{\kappa_1}) \cap f_{\kappa_2}(C_{\kappa_2})$; choose $c_1 \in C_{\kappa_1}$, $c_2 \in C_{\kappa_2}$ with $f_{\kappa_1}(c_1) = u = f_{\kappa_2}(c_2)$. Then $f_{\kappa_1}(c_1) \sqsubseteq_\beta f_{\kappa_2}(c_2)$, which is only possible if $\kappa_1 = \kappa_2$. Therefore if $\kappa_1 \neq \kappa_2$ then $f_{\kappa_1}(C_{\kappa_1})$ and $f_{\kappa_2}(C_{\kappa_2})$ are

disjoint subsets of D_β . The final statement now follows directly from Proposition 3.2.2 and Lemma 3.3.1. \square

Proposition 4.1.2 *Let (D_B, f_K) be a regular bottomed Λ -algebra. Then there exists an ordering associated with (D_B, f_K) . Moreover, if in addition (D_B, f_K) is minimal then this associated ordering is unique.*

Proof In order to show the existence of an associated ordering it is useful to first introduce a somewhat weaker concept. If \sqsubseteq_β is a partial order on the set D_β for each $\beta \in B$ then the family \sqsubseteq_B will be called a *weak ordering* if the following hold:

- (i) $\perp_\beta \sqsubseteq_\beta u$ for each $u \in D_\beta$.
- (ii) If $\kappa, \kappa' \in K_\beta$ and $c \in C_\kappa, c' \in C_{\kappa'}$ with $f_\kappa(c) \sqsubseteq_\beta f_{\kappa'}(c')$ then $\kappa = \kappa'$ and $c \sqsubseteq_\kappa c'$.

For each $\beta \in B$ let \sqsubseteq'_β be a partial order on D_β . Then (making use of Lemma 3.3.1) a binary relation \sqsubseteq'_B can be defined on D_B for each $\beta \in B$ as follows:

- (i) $\perp_\beta \sqsubseteq'_\beta u$ for each $u \in D_\beta$.
- (ii) $u \sqsubseteq'_\beta \perp_\beta$ holds (if and) only if $u = \perp_\beta$.
- (iii) If $\kappa, \kappa' \in K_\beta$ and $c \in C_\kappa, c' \in C_{\kappa'}$ then $f_\kappa(c) \sqsubseteq'_\beta f_{\kappa'}(c')$ if and only if $\kappa = \kappa'$ and $c \sqsubseteq_\kappa c'$.

Lemma 4.1.1 *The relation \sqsubseteq'_β is a partial order for each $\beta \in B$.*

Proof It is clear that \sqsubseteq'_β is reflexive. To show that \sqsubseteq'_β is anti-symmetric consider $u_1, u_2 \in D_\beta$ with $u_1 \sqsubseteq'_\beta u_2$ and $u_2 \sqsubseteq'_\beta u_1$. If $\perp_\beta \in \{u_1, u_2\}$ then $u_1 = u_2 (= \perp_\beta)$ by (ii). Thus suppose that $u_1, u_2 \in D_\beta \setminus \{\perp_\beta\}$. Then by (iii) and Lemma 3.3.1 there exists $\kappa \in K_\beta$ and $c_1, c_2 \in C_\kappa$ such that $u_1 = f_\kappa(c_1), u_2 = f_\kappa(c_2)$, $c_1 \sqsubseteq_\kappa c_2$ and $c_2 \sqsubseteq_\kappa c_1$. Hence $c_1 = c_2$ and so $u_1 = f_\kappa(c_1) = f_\kappa(c_2) = u_2$. A similar argument shows that \sqsubseteq'_β is transitive. \square

Lemma 4.1.2 *Suppose \sqsubseteq_B is a weak ordering. Then \sqsubseteq'_B is also a weak ordering and $u_1 \sqsubseteq'_\beta u_2$ whenever $u_1 \sqsubseteq_\beta u_2$.*

Proof It will be shown first that if $u_1, u_2 \in D_\beta$ with $u_1 \sqsubseteq_\beta u_2$ then $u_1 \sqsubseteq'_\beta u_2$. Now since $\perp_\beta \sqsubseteq_\beta u_2$ holds trivially it can be assumed here that $u_1 \neq \perp_\beta$. Then $u_1, u_2 \in D_\beta \setminus \{\perp_\beta\}$, and hence by Lemma 3.3.1 and the definition of a weak ordering there exists $\kappa \in K_\beta$ and $c_1, c_2 \in C_\kappa$ such that $u_1 = f_\kappa(c_1), u_2 = f_\kappa(c_2)$ and $c_1 \sqsubseteq_\kappa c_2$. But then $u_1 = f_\kappa(c_1) \sqsubseteq'_\beta f_\kappa(c_2) = u_2$ holds by the definition of \sqsubseteq'_β .

It remains to show that \sqsubseteq'_B is a weak ordering. Thus consider $\kappa, \kappa' \in K_\beta$ and $c \in C_\kappa, c' \in C_{\kappa'}$ with $f_\kappa(c) \sqsubseteq'_\beta f_{\kappa'}(c')$. Then (by the definition of \sqsubseteq'_β) $\kappa = \kappa'$ and $c \sqsubseteq_\kappa c'$, and by the first part of the proof $c \sqsubseteq'_\kappa c'$. This implies that \sqsubseteq'_B is a weak ordering (since by definition $\perp_\beta \sqsubseteq'_\beta u$ holds for each $u \in D_\beta$). \square

If \sqsubseteq_B is a weak ordering then \sqsubseteq'_B will be called the *first refinement* of \sqsubseteq_B . Let \sqsubseteq_β^0 be the ‘trivial’ partial order on D_β defined so that $u_1 \sqsubseteq_\beta^0 u_2$ if and only if u_1 is either \perp_β or u_2 . Then \sqsubseteq_B^0 is clearly a weak ordering, and thus by Lemma 4.1.2 a sequence of weak orderings \sqsubseteq_B^m , $m \geq 0$, can be defined by letting \sqsubseteq_B^{m+1} be the first refinement of \sqsubseteq_B^m for each $m \geq 0$. Now define a partial order \sqsubseteq_β on D_β for each $\beta \in B$ by stipulating that $u_1 \sqsubseteq_\beta u_2$ if and only if $u_1 \sqsubseteq_\beta^m u_2$ for some (and thus for all sufficiently large) $m \geq 0$. It is easy to see that \sqsubseteq_B is a weak ordering.

In fact \sqsubseteq_B is an associated ordering: Let $\kappa \in K$ be of type $L \rightarrow \beta$ and let $c, c' \in C_\kappa$ with $c \sqsubseteq_\kappa c'$. Then $c(\eta) \sqsubseteq_\eta c'(\eta)$ for each $\eta \in L$, and so for each η there exists $m_\eta \geq 0$ such that $c(\eta) \sqsubseteq_\eta^{m_\eta} c'(\eta)$. Put $m = \max\{m_\eta : \eta \in L\}$; then $c(\eta) \sqsubseteq_\eta^m c'(\eta)$ for each η , and this means that $c \sqsubseteq_\kappa^m c'$. Hence $f_\kappa(c) \sqsubseteq_\beta^{m+1} f_\kappa(c')$ by the definition of \sqsubseteq_β^{m+1} , and therefore $f_\kappa(c) \sqsubseteq_\beta f_\kappa(c')$.

The uniqueness in the case when (D_B, f_K) is minimal still has to be considered. Thus let \sqsubseteq_B and \sqsubseteq'_B be two orderings associated with (D_B, f_K) and for each $u \in D_\beta$ define $L_\beta(u) = \{u' \in D_\beta : u' \sqsubseteq_\beta u\}$ and $L'_\beta(u) = \{u' \in D_\beta : u' \sqsubseteq'_\beta u\}$; put $D'_\beta = \{u \in D_\beta : L'_\beta(u) = L_\beta(u)\}$. Then $\perp_\beta \in D'_\beta$ for each $\beta \in B$ and it is straightforward to check that the family D'_β is invariant. Hence if (D_B, f_K) is minimal then $D'_\beta = D_\beta$, i.e., $\sqsubseteq'_B = \sqsubseteq_B$. \square

Proposition 4.1.3 *Let (D_B, f_K) be a regular bottomed extension of (X_B, p_K) and \sqsubseteq_B be an ordering associated with (D_B, f_K) ; let $\beta \in B$.*

- (1) *Each element of X_β is a maximal element of the poset D_β , i.e., if $x \in X_\beta$ and $u \in D_\beta$ with $x \sqsubseteq_\beta u$ then $u = x$.*
- (2) *For each $x \in X_\beta$ the set $\{u \in D_\beta : u \sqsubseteq_\beta x\}$ is finite.*

Proof (1) For each $\beta \in B$ let X'_β denote the set of elements in X_β which are maximal. Then the family X'_β is clearly invariant in (X_B, p_K) , and thus $X'_\beta = X_\beta$, i.e., each element of X_β is a maximal element of D_β .

(2) For each $\beta \in B$ let X'_β denote the set of those elements $x \in X_\beta$ for which the set $\{u \in D_\beta : u \sqsubseteq_\beta x\}$ is finite. Then it is easily checked that the family X'_β is invariant in (X_B, p_K) , and thus $X'_\beta = X_\beta$, i.e., the set $\{u \in D_\beta : u \sqsubseteq_\beta x\}$ is finite for each $x \in X_\beta$. \square

If (D_B, f_K) is a minimal regular extension of (X_B, p_K) then by Proposition 4.1.2 there exists a unique ordering \sqsubseteq_B associated with (D_B, f_K) ; thus in this case it makes sense to speak of *the* associated ordering. Recall the assumption made in Section 3.1 that the signature Λ is pervasive, from which it follows that $X_\beta \neq \emptyset$ for each $\beta \in B$.

Proposition 4.1.4 *Let (D_B, f_K) be a minimal regular extension of (X_B, p_K) and \sqsubseteq_B be the associated ordering; let $\beta \in B$.*

- (1) *For each $u \in D_\beta$ there exists $x \in X_\beta$ with $u \sqsubseteq_\beta x$ (and note here the fact that $X_\beta \neq \emptyset$ is needed).*

(2) For each $u \in D_\beta$ the set $\{u' \in D_\beta : u' \sqsubseteq_\beta u\}$ is finite.

In particular, (1) and Proposition 4.1.3 (1) imply that X_β is exactly the set of maximal elements of the poset D_β .

Proof (1) For each $\beta \in B$ let $D'_\beta = \{u \in D_\beta : u \sqsubseteq_\beta x \text{ for some } x \in X_\beta\}$. Then it easily checked that the family D'_B is invariant in (D_B, f_K) and $\perp_\beta \in D'_\beta$ for each $\beta \in B$. (Note that $\perp_\beta \in D'_\beta$ is equivalent to having $X_\beta \neq \emptyset$.) Hence $D'_B = D_B$.

(2) This also follows by considering an appropriate invariant family. \square

If Y_1 and Y_2 are posets then a mapping $h : Y_1 \rightarrow Y_2$ is said to be *monotone* if $h(y) \sqsubseteq_2 h(y')$ for all $y, y' \in Y_1$ with $y \sqsubseteq_1 y'$ (where \sqsubseteq_j is the partial order on Y_j for $j = 1, 2$). Let (D_B, f_K) be a regular bottomed Λ -algebra and \sqsubseteq_B be an associated ordering. Then \sqsubseteq_B is said to be *monotone* if the mappings f_κ , $\kappa \in K$, are monotone with respect to \sqsubseteq_B , i.e., if $\kappa \in K_\beta$ and $c, c' \in \text{Dom}(f_\kappa)$ with $c \sqsubseteq_\kappa c'$ then $f_\kappa(c) \sqsubseteq_\beta f_\kappa(c')$.

Now it might be expected that an associated ordering \sqsubseteq_B is automatically monotone. But, as Example 4.1.2 below shows, this is not true in general, even if (D_B, f_K) is a minimal regular bottomed Λ -algebra and \sqsubseteq_B is the (unique) associated ordering.

Example 4.1.2 Let Λ and (X_B, p_K) be as in Example 2.2.1 and let (Y_B, q_K) be the Λ -algebra defined (as in Example 3.4.1) by

$$Y_{\text{nat}} = \mathbb{N} \cup \{\perp_{\text{nat}}, \perp_{\text{nat}}^o\} \text{ with } \perp_{\text{nat}}^o \notin \mathbb{N} \cup \{\perp_{\text{nat}}\},$$

$$Y_\beta = X_\beta^\perp \text{ for all } \beta \in B \setminus \{\text{nat}\},$$

$$q_{\text{Zero}} : \mathbb{I} \rightarrow Y_{\text{nat}} \text{ with } q_{\text{Zero}}(\varepsilon) = 0,$$

$$q_{\text{Succ}} : Y_{\text{nat}} \rightarrow Y_{\text{nat}} \text{ with}$$

$$q_{\text{Succ}}(n) = \begin{cases} n + 1 & \text{if } n \in \mathbb{N}, \\ \perp_{\text{nat}} & \text{if } n = \perp_{\text{nat}}^o, \\ \perp_{\text{nat}}^o & \text{if } n = \perp_{\text{nat}}, \end{cases}$$

and with $q_\kappa = p_\kappa^\perp$ for all $\kappa \in K \setminus \{\text{Zero}, \text{Succ}\}$,

where (X_B^\perp, p_K^\perp) is the flat extension of the Λ -algebra (X_B, p_K) . Then (Y_B, q_K) is clearly a minimal regular extension of (X_B, p_K) , and so let \sqsubseteq_B be the ordering associated with (Y_B, q_K) . However, q_{Succ} is not monotone, since $\perp_{\text{nat}} \sqsubseteq_{\text{nat}} \perp_{\text{nat}}^o$ but $q_{\text{Succ}}(\perp_{\text{nat}}) = \perp_{\text{nat}}^o$ and $q_{\text{Succ}}(\perp_{\text{nat}}^o) = \perp_{\text{nat}}$.

However, it will now be shown that if (D_B, f_K) is a minimal monotone regular bottomed Λ -algebra then the associated ordering \sqsubseteq_B is monotone.

Proposition 4.1.5 *Let (D_B, f_K) be a minimal monotone regular bottomed Λ -algebra with (unique) associated ordering \sqsubseteq_B . Then \sqsubseteq_B is monotone.*

Proof The family of relations \sqsubseteq_B is compatible (as defined in Section 3.4): Let $\kappa \in K$ be of type $L \rightarrow \beta$ and consider $a \in D_\circ^L$, $u \in D_\beta \setminus \{\perp_\beta\}$ with $u \sqsubseteq_\beta f_\kappa(a)$. Then u has a unique representation of the form $f_{\kappa'}(c)$ (since $u \neq \perp_\beta$ and (D_B, f_K) is regular) and hence $\kappa' = \kappa$ and $c \sqsubseteq_\circ^L a$ (since \sqsubseteq_B is an associated ordering). Moreover, \sqsubseteq_B is full: This follows from the fact that \perp_β is a base element of D_β for each $\beta \in B$ and because (D_B, f_K) is minimal.

Let $\kappa \in K$ be of type $L \rightarrow \beta$ and let $c, a \in D_\circ^L$ with $c \sqsubseteq_\circ^L a$, and since the task at hand is to show that $f_\kappa(c) \sqsubseteq_\beta f_\kappa(a)$ it can clearly be assumed that $f_\kappa(c) \neq \perp_\beta$. Now by Proposition 3.4.5 there exists a finite B -typed set J , a J -free Λ -algebra (Z_B, r_K) , and elements $z \in Z_\beta$, $b \in D_\circ^J$ such that $f_\kappa(c) = \pi_\beta^{\perp_J}(z)$ and $f_\kappa(a) = \pi_\beta^b(z)$. But (D_B, f_K) is monotone and $\pi_\beta^{\perp_J}(z) = f_\kappa(c) \neq \perp_\beta$; hence $f_\kappa(a) = \pi_\beta^b(z) \neq \perp_\beta$, and so by the definition of an associated ordering it follows that $f_\kappa(c) \sqsubseteq_\beta f_\kappa(a)$. \square

The following result can be considered as a converse to Proposition 4.1.5.

Proposition 4.1.6 *Let (D_B, f_K) be a bottomed Λ -algebra and for each $\beta \in B$ let \sqsubseteq_β be a partial order on the set D_β such that $\perp_\beta \sqsubseteq_\beta u$ for all $u \in D_\beta$. Suppose that the mappings $f_\kappa, \kappa \in K$, are monotone with respect to the family \sqsubseteq_B , i.e., if $\kappa \in K_\beta$ and $c, c' \in \text{Dom}(f_\kappa)$ with $c \sqsubseteq_\kappa c'$ then $f_\kappa(c) \sqsubseteq_\beta f_\kappa(c')$. Then (D_B, f_K) is monotone.*

Proof By Proposition 3.4.1 it is enough to show that (D_B, f_K) is strongly monotone. Let (Y_B, q_K) be a Λ -algebra, let $\check{\pi}_B$ be a fat bottomed and π_B be an arbitrary homomorphism from (Y_B, q_K) to (D_B, f_K) . For each $\beta \in B$ put

$$Y'_\beta = \{y \in Y_\beta : \check{\pi}_\beta(y) \sqsubseteq_\beta \pi_\beta(y)\}.$$

If $\kappa \in K$ is of type $\emptyset \rightarrow \beta$ then $q_\kappa(\varepsilon) \in Y'_\beta$, because $\check{\pi}_\beta(q_\kappa(\varepsilon)) = f_\kappa(\varepsilon) = \pi_\beta(q_\kappa(\varepsilon))$. Next consider $\kappa \in K$ of type $L \rightarrow \beta$ with $L \neq \emptyset$ and let $c \in Y_\circ^L$ be such that $c(\eta) \in Y'_\eta$ for each $\eta \in L$. Then $\check{\pi}_\circ^L(c) \sqsubseteq_\kappa \pi_\circ^L(c)$ and thus

$$\check{\pi}_\beta(q_\kappa(c)) = f_\kappa(\check{\pi}_\circ^L(c)) \sqsubseteq_\beta f_\kappa(\pi_\circ^L(c)) = \pi_\beta(q_\kappa(c)),$$

i.e., $q_\kappa(c) \in Y'_\beta$. This shows the family Y'_B is invariant in (Y_B, q_K) .

Let \check{U}_B (resp. U_B) be the kernel of $\check{\pi}_B$ (resp. the kernel of π_B). If $y \in \check{U}_\beta$ then $\check{\pi}_\beta(y) = \perp_\beta \sqsubseteq_\beta \pi_\beta(y)$, and hence $\check{U}_B \subset Y'_B$. Thus $Y'_B = Y_B$, since $\check{\pi}_B$ is fat bottomed. In particular, if $y \in U_\beta$ then it follows that $\check{\pi}_\beta(y) \sqsubseteq_\beta \pi_\beta(y) = \perp_\beta$, i.e., $y \in \check{U}_\beta$. Therefore $U_B \subset \check{U}_B$, i.e., (D_B, f_K) is strongly monotone. \square

4.2 Complete partially ordered sets

Let Y_1 and Y_2 be posets and $h : Y_1 \rightarrow Y_2$ a mapping; as before h is *monotone* if $h(y) \sqsubseteq_2 h(y')$ for all $y, y' \in Y_1$ with $y \sqsubseteq_1 y'$ (where \sqsubseteq_j is the partial order on Y_j for $j = 1, 2$). The mapping h is said to be an *embedding* if $h(y) \sqsubseteq_2 h(y')$ holds if and only if $y \sqsubseteq_1 y'$. An embedding is, in particular, monotone and it is also automatically injective.

A bijective mapping $h : Y_1 \rightarrow Y_2$ is called an *order isomorphism* if h and the inverse mapping $h^{-1} : Y_2 \rightarrow Y_1$ are both monotone. Thus a mapping is an order isomorphism if and only if it is a surjective embedding. (Note that if $h : Y_1 \rightarrow Y_2$ is a monotone mapping which is a bijection then the inverse mapping $h^{-1} : Y_2 \rightarrow Y_1$ is not necessarily monotone.) The posets Y_1 and Y_2 are said to be *isomorphic* if there exists an order isomorphism $h : Y_1 \rightarrow Y_2$.

Let Y be a poset with partial order \sqsubseteq and let A be a non-empty subset of Y . An element $y \in Y$ is called an *upper bound* of A if $y' \sqsubseteq y$ for all $y' \in A$; y is called the *least upper bound* of A if y is an upper bound of A and $y \sqsubseteq y'$ for each upper bound y' of A . (It is clear that there can be at most one element $y \in Y$ having these properties.) If the least upper bound of A exists then it will be denoted by $\bigsqcup A$.

A subset A of a poset Y is said to be *directed* if it is non-empty and for each $y_1, y_2 \in A$ there exists $y \in A$ such that $y_1 \sqsubseteq y$ and $y_2 \sqsubseteq y$. The set of directed subsets of Y will be denoted by $\mathbf{d}(Y)$. If $h : Y_1 \rightarrow Y_2$ is monotone then clearly $h(A) \in \mathbf{d}(Y_2)$ for each $A \in \mathbf{d}(Y_1)$.

A poset D is said to be *complete* if the least upper bound $\bigsqcup A$ of A exists for each $A \in \mathbf{d}(D)$. If D_1 and D_2 are complete posets then a mapping $h : D_1 \rightarrow D_2$ is said to be *continuous* if h is monotone and $h(\bigsqcup A) = \bigsqcup h(A)$ for each $A \in \mathbf{d}(D_1)$. Note that if $h : D_1 \rightarrow D_2$ is monotone then $\bigsqcup h(A) \sqsubseteq_2 h(\bigsqcup A)$ always holds (with \sqsubseteq_2 the partial order on D_2). Thus a monotone mapping h is continuous if and only if $h(\bigsqcup A) \sqsubseteq_2 \bigsqcup h(A)$ for each $A \in \mathbf{d}(D_1)$. If D_1, D_2 and D_3 are complete posets and $h : D_1 \rightarrow D_2$ and $h' : D_2 \rightarrow D_3$ are continuous then the mapping $h' \circ h : D_1 \rightarrow D_3$ is clearly also continuous. In this section some of the basic properties of complete posets and continuous mappings will be looked at.

Proposition 4.2.1 *Let D be a complete poset and Y a poset isomorphic to D ; then Y is also complete. Moreover, if $h : Y \rightarrow D$ is any order isomorphism then h and the inverse mapping $h^{-1} : D \rightarrow Y$ are both automatically continuous.*

Proof Let $A \in \mathbf{d}(Y)$; then $B = h(A) \in \mathbf{d}(D)$, thus let $u = \bigsqcup B$ and put $y = h^{-1}(u)$. Now u' is an upper bound of B if and only if $h^{-1}(u')$ is an upper bound of A , and hence $y = \bigsqcup A$. This shows that Y is complete and that h is continuous (since $h(\bigsqcup A) = h(y) = u = \bigsqcup B = \bigsqcup h(A)$). The continuity of h^{-1} follows by reversing the roles of D and Y . \square

Proposition 4.2.2 *Let D_S be a family of complete posets and let J be an S -typed set. Then the poset D_S^J is also complete.*

Proof This follows immediately from Lemma 4.2.1. \square

Lemma 4.2.1 *Let D_S and J be as in Proposition 4.2.2. Let $A \in \mathbf{d}(D_\diamond^J)$ and for each $\eta \in J$ put $A_\eta = \{c(\eta) : c \in A\}$. Then $A_\eta \in \mathbf{d}(D_\eta)$, and the least upper bound of A is the assignment $c \in D_\diamond^J$ defined by $c(\eta) = \bigsqcup A_\eta$ for each $\eta \in J$.*

Proof Straightforward. \square

Proposition 4.2.3 *Let D_S and D'_S be two families of complete posets and let $\varphi_S : D_S \rightarrow D'_S$ be a family of continuous mappings. Then for each S -typed set J the mapping $\varphi_\diamond^J : D_\diamond^J \rightarrow (D'_\diamond)^J$ is also continuous.*

Proof If $c, c' \in D_\diamond^J$ with $c \sqsubseteq_\diamond^J c'$ then for each $\eta \in J$

$$\varphi_\diamond^J(c)(\eta) = \varphi_\eta(c(\eta)) \sqsubseteq_\eta \varphi_\eta(c'(\eta)) = \varphi_\diamond^J(c')(\eta),$$

since φ_η is monotone, and hence φ_\diamond^J is monotone. Now let $A \in \mathbf{d}(D_\diamond^J)$; then, with two applications of Lemma 4.2.1 and since φ_η is continuous, it follows that

$$\begin{aligned} \varphi_\diamond^J(\bigsqcup A)(\eta) &= \varphi_\eta(\bigsqcup A(\eta)) = \varphi_\eta(\bigsqcup \{c(\eta) : c \in A\}) \\ &= \bigsqcup \{\varphi_\eta(c(\eta)) : c \in A\} = \bigsqcup \{\varphi_\diamond^J(c)(\eta) : c \in A\} = (\bigsqcup \varphi_\diamond^J(A))(\eta) \end{aligned}$$

for each $\eta \in J$, and therefore $\varphi_\diamond^J(\bigsqcup A) = \bigsqcup \varphi_\diamond^J(A)$. Thus φ_\diamond^J is continuous. \square

Note the following special case of Proposition 4.2.3: Let $n \geq 2$ and for $j = 1, \dots, n$ let D_j and D'_j be complete posets and $h_j : D_j \rightarrow D'_j$ be a continuous mapping. Then the mapping $h : D_1 \times \dots \times D_n \rightarrow D'_1 \times \dots \times D'_n$ defined by

$$h(u_1, \dots, u_n) = (h_1(u_1), \dots, h_n(u_n))$$

for each $(u_1, \dots, u_n) \in D_1 \times \dots \times D_n$ is continuous.

Let Y_S be a family of posets and L be an S -typed set. Then for each $\eta \in L$ there is the projection mapping $\mathbf{p}_\eta^L : Y_\diamond^L \rightarrow Y_\eta$ defined by $\mathbf{p}_\eta^L(c) = c(\eta)$ for each $c \in Y_\diamond^L$, and \mathbf{p}_η^L is clearly monotone (since if $c, c' \in Y_\diamond^L$ with $c \sqsubseteq_\diamond^L c'$ then by definition $c(\eta) \sqsubseteq_\eta c'(\eta)$ for each $\eta \in L$).

Lemma 4.2.2 *A mapping $h : Y \rightarrow Y_\diamond^L$ from a poset Y to the poset Y_\diamond^L is monotone if and only if for each $\eta \in L$ the mapping $\mathbf{p}_\eta^L h$ from Y to Y_η is monotone.*

Proof Straightforward. \square

Now let D_S be a family of complete posets and L be an S -typed set. Then by Lemma 4.2.1 the projection mapping $\mathbf{p}_\eta^L : D_\diamond^L \rightarrow D_\eta$ is continuous for each $\eta \in L$.

Proposition 4.2.4 *A mapping $h : D \rightarrow D_\diamond^L$ from a complete poset D to the complete poset D_\diamond^L is continuous if and only if for each $\eta \in L$ the mapping $\mathbf{p}_\eta^L h$ from D to D_η is continuous.*

Proof If h is continuous then $\mathfrak{p}_\eta^L h$ is also continuous for each $\eta \in L$ (since it is the composition of two continuous mappings). Conversely, suppose $\mathfrak{p}_\eta^L h$ is continuous for each $\eta \in L$; in particular, h is then monotone (as in Lemma 4.2.2). Let $A \in \mathfrak{d}(D)$; then by Lemma 4.2.1 it follows that for each $\eta \in L$

$$\begin{aligned} (\bigsqcup h(A))(\eta) &= \bigsqcup \{c(\eta) : c \in h(A)\} \\ &= \bigsqcup \mathfrak{p}_\eta^L(h(A)) = (\mathfrak{p}_\eta^L h)(\bigsqcup A) = h(\bigsqcup A)(\eta) \end{aligned}$$

and so $\bigsqcup h(A) = h(\bigsqcup A)$. This shows h is continuous. \square

If Y_1 and Y_2 are posets then the set of all monotone mappings from Y_1 to Y_2 will be denoted by $(Y_1 \rightarrow Y_2)$. Note that the set $(Y_1 \rightarrow Y_2)$ is non-empty, since the constant mappings are clearly monotone.

A partial order \sqsubseteq on the set $(Y_1 \rightarrow Y_2)$ can be defined by stipulating that $h \sqsubseteq h'$ if and only if $h(y) \sqsubseteq_2 h'(y)$ for all $y \in Y_1$ (where \sqsubseteq_2 is the partial order on Y_2), and if $(Y_1 \rightarrow Y_2)$ is considered as a poset then it will always be with respect to this partial order. Note that if Y_2 is a bottomed poset with bottom element \perp_2 then $(Y_1 \rightarrow Y_2)$ is also a bottomed poset: The bottom element \perp of $(Y_1 \rightarrow Y_2)$ is the constant mapping with $\perp(y) = \perp_2$ for all $y \in Y_1$.

Let Y be a poset; a poset Y' is called a *subposet* of Y if $Y' \subset Y$ and the partial order on Y' is obtained by restricting the partial order on Y . If Y is a bottomed poset then a subposet Y' of Y is said to be *proper bottomed* if Y' contains the bottom element \perp of Y . Then \perp is also the bottom element of Y' and so Y' is a bottomed poset. (The reason for making this definition is that it is possible for a subposet of a bottomed poset Y to be bottomed but to have a different bottom element from that in Y .)

If D_1 and D_2 are complete posets then the set of all continuous mappings from D_1 to D_2 will be denoted by $[D_1 \rightarrow D_2]$. Then $[D_1 \rightarrow D_2]$ is a subset of $(D_1 \rightarrow D_2)$, and so $[D_1 \rightarrow D_2]$ can be considered as a poset with the partial order induced by the partial order on $(D_1 \rightarrow D_2)$. (This means that $[D_1 \rightarrow D_2]$ is a subposet of $(D_1 \rightarrow D_2)$.)

Note that if in addition D_2 is bottomed with bottom element \perp_2 then the bottom element of $(D_1 \rightarrow D_2)$ (i.e., the constant mapping \perp with $\perp(u) = \perp_2$ for all $u \in D_1$) is an element of $[D_1 \rightarrow D_2]$. In other words, in this case $[D_1 \rightarrow D_2]$ is a proper bottomed subposet of $(D_1 \rightarrow D_2)$.

Proposition 4.2.5 *If D_1, D_2 are complete posets then the poset $D = [D_1 \rightarrow D_2]$ is also complete. Moreover, if $A \in \mathfrak{d}(D)$ with $h = \bigsqcup A$ then $h(u) = \bigsqcup \{f(u) : f \in A\}$ for each $u \in D_1$.*

Proof Let $A \in \mathfrak{d}(D)$, and for each $u \in D_1$ put $A_u = \{f(u) : f \in A\}$. Then $A_u \in \mathfrak{d}(D_2)$ and so define a mapping $h : D_1 \rightarrow D_2$ by putting $h(u) = \bigsqcup A_u$ for each $u \in D_1$. It will be shown that h is continuous, and, since h is clearly monotone, for this it is enough to show that if $B \in \mathfrak{d}(D_1)$ with $u' = \bigsqcup B$ then $h(u') \sqsubseteq_2 \bigsqcup h(B)$. Let $f \in A$; then

$$\begin{aligned} f(u') &= \bigsqcup f(B) = \bigsqcup \left(\bigcup_{u \in B} f(u) \right) \\ &\sqsubseteq_2 \bigsqcup \left(\bigcup_{u \in B} \bigsqcup A_u \right) = \bigsqcup \left(\bigcup_{u \in B} h(u) \right) = \bigsqcup h(B). \end{aligned}$$

Hence $h(u') = \bigsqcup A_{u'} \sqsubseteq_2 \bigsqcup h(B)$. Now (as in Lemma 4.2.1) it is straightforward to check that h is the least upper bound of A in $[D_1 \rightarrow D_2]$. \square

In what follows let D_1, D_2 and D be complete posets.

Lemma 4.2.3 *A monotone mapping $h : D_1 \times D_2 \rightarrow D$ is continuous if and only if*

$$h(\bigsqcup A_1, \bigsqcup A_2) = \bigsqcup h(A_1 \times A_2)$$

for all $A_1 \in \mathbf{d}(D_1)$, $A_2 \in \mathbf{d}(D_2)$.

Proof If $A \in \mathbf{d}(D_1 \times D_2)$ then it is easy to see that there exist $A_1 \in \mathbf{d}(D_1)$ and $A_2 \in \mathbf{d}(D_2)$ such that $A' = A_1 \times A_2$ is an element of $\mathbf{d}(D_1 \times D_2)$ which is equivalent to A in the sense that for each $v \in A$ there exists $v' \in A'$ with $v \sqsubseteq v'$ and for each $v' \in A'$ there exists $v \in A$ with $v' \sqsubseteq v$. Then $\bigsqcup A = \bigsqcup A' = (\bigsqcup A_1, \bigsqcup A_2)$ and $\bigsqcup h(A') = \bigsqcup h(A)$, and this shows that if $h(\bigsqcup A_1, \bigsqcup A_2) = \bigsqcup h(A_1 \times A_2)$ for all $A_1 \in \mathbf{d}(D_1)$, $A_2 \in \mathbf{d}(D_2)$ then h is continuous. The converse is clear, since $A_1 \times A_2 \in \mathbf{d}(D_1 \times D_2)$ whenever $A_1 \in \mathbf{d}(D_1)$ and $A_2 \in \mathbf{d}(D_2)$. \square

Lemma 4.2.4 *A mapping $h : D_1 \times D_2 \rightarrow D$ is continuous if and only if it is separately continuous in each argument, i.e., if and only if $u_2 \mapsto h(u_1, u_2)$ is a continuous mapping from D_2 to D for each $u_1 \in D_1$ and $u_1 \mapsto h(u_1, u_2)$ is a continuous mapping from D_1 to D for each $u_2 \in D_2$.*

Proof Clearly h is monotone if and only if it is separately monotone in each argument. Moreover, if h is continuous then it is separately continuous, since if $A \in \mathbf{d}(D_2)$ then $\{u_1\} \times A \in \mathbf{d}(D_1 \times D_2)$ and $\bigsqcup (\{u_1\} \times A) = (u_1, \bigsqcup A)$, with of course an analogous statement holding when the roles of the two components are reversed. Suppose then that h is separately continuous in each argument. Lemma 4.2.3 will be used to show that h is continuous; thus let $A_1 \in \mathbf{d}(D_1)$ and $A_2 \in \mathbf{d}(D_2)$ and put $u_1 = \bigsqcup A_1$ and $u_2 = \bigsqcup A_2$. Then $A_1 \times \{u_2\} \in \mathbf{d}(D_1 \times D_2)$ and

$$h(u_1, u_2) = \bigsqcup \{h(v, u_2) : v \in A_1\} = \bigsqcup h(A_1 \times \{u_2\}),$$

since h is continuous in its first argument. Moreover, since h is continuous in its second argument, it follows that

$$\begin{aligned} \bigsqcup h(A_1 \times \{u_2\}) &= \bigsqcup \{h(v_1, \bigsqcup A_2) : v_1 \in A_1\} \\ &= \bigsqcup \{ \bigsqcup \{h(v_1, v_2) : v_2 \in A_2\} : v_1 \in A_1 \} \\ &= \bigsqcup \{h(v_1, v_2) : v_1 \in A_1, v_2 \in A_2\} = \bigsqcup h(A_1 \times A_2). \end{aligned}$$

Therefore $h(u_1, u_2) = \bigsqcup h(A_1 \times \{u_2\}) = \bigsqcup h(A_1 \times A_2)$ and thus by Lemma 4.2.3 h is continuous. \square

Proposition 4.2.6 *The application operator $\mathbf{a} : [D_1 \rightarrow D_2] \times D_1 \rightarrow D_2$ given by*

$$\mathbf{a}(h, u) = h(u)$$

for all $h \in [D_1 \rightarrow D_2]$, $u \in D_1$, is continuous.

Proof This follows immediately from Lemma 4.2.4, since \mathbf{a} is clearly continuous in its second argument, and by Proposition 4.2.5 it is also continuous in its first argument. \square

Now consider a continuous mapping $h : D_1 \times D_2 \rightarrow D$; then by Lemma 4.2.4 the mapping $u_2 \mapsto h(u_1, u_2)$ from D_2 to D is continuous for each $u_1 \in D_1$, which means a mapping $\mathbf{c}(h) : D_1 \rightarrow [D_2 \rightarrow D]$ can be defined by letting

$$\mathbf{c}(h)(u_1)(u_2) = h(u_1, u_2)$$

for each $u_1 \in D_1$, $u_2 \in D_2$.

Lemma 4.2.5 $\mathbf{c}(h) : D_1 \rightarrow [D_2 \rightarrow D]$ is continuous for each continuous mapping $h : D_1 \times D_2 \rightarrow D$.

Proof It is clear that $\mathbf{c}(h)$ is monotone. Let $A \in \mathbf{d}(D_1)$ and put $h = \bigsqcup \mathbf{c}(h)(A)$ (so $h \in [D_2 \rightarrow D]$). Then by Proposition 4.2.5 it follows for each $u_2 \in D_2$ that

$$h(u_2) = \bigsqcup \{ \mathbf{c}(h)(u_1)(u_2) : u_1 \in A \} = \bigsqcup \{ h(u_1, u_2) : u_1 \in A \},$$

and by Lemma 4.2.4 that

$$\bigsqcup \{ h(u_1, u_2) : u_1 \in A \} = h(\bigsqcup A, u_2) = \mathbf{c}(h)(\bigsqcup A)(u_2).$$

Hence $\bigsqcup \mathbf{c}(h)(A) = \mathbf{c}(h)(\bigsqcup A)$, and thus $\mathbf{c}(h)$ is continuous. \square

Lemma 4.2.5 allows the so-called *currying operator* to be defined; this is the mapping $\mathbf{c} : [D_1 \times D_2 \rightarrow D] \rightarrow [D_1 \rightarrow [D_2 \rightarrow D]]$ given by

$$\mathbf{c}(h, u_1)(u_2) = h(u_1, u_2)$$

for all $h \in [D_1 \times D_2 \rightarrow D]$, $u_1 \in D_1$, $u_2 \in D_2$.

Proposition 4.2.7 *The currying operator \mathbf{c} is an order isomorphism whose inverse is the uncurrying operator $\mathbf{u} : [D_1 \rightarrow [D_2 \rightarrow D]] \rightarrow [D_1 \times D_2 \rightarrow D]$ which is defined by*

$$\mathbf{u}(g)(u_1, u_2) = g(u_1)(u_2)$$

for all $g \in [D_1 \rightarrow [D_2 \rightarrow D]]$, $(u_1, u_2) \in D_1 \times D_2$. (Proposition 4.2.1 then implies that the mappings \mathbf{c} and \mathbf{u} are both continuous.)

Proof The inverse mapping to \mathbf{c} will be constructed. For each $g \in [D_1 \rightarrow [D_2 \rightarrow D]]$ define $\mathbf{u}(g) : D_1 \times D_2 \rightarrow D$ by $\mathbf{u}(g)(u_1, u_2) = g(u_1)(u_2)$. Thus $\mathbf{u}(g)$ is monotone, and if $A_1 \in \mathbf{d}(D_1)$, $A_2 \in \mathbf{d}(D_2)$ then

$$\begin{aligned}
u(g)(\bigsqcup A_1, \bigsqcup A_2) &= g(\bigsqcup A_1)(\bigsqcup A_2) = \bigsqcup \{g(u_1)(\bigsqcup A_2) : u_1 \in A_1\} \\
&= \bigsqcup \{\bigsqcup \{g(u_1)(u_2) : u_2 \in A_2\} : u_1 \in A_1\} \\
&= \bigsqcup \{g(u_1)(u_2) : u_1 \in A_1, u_2 \in A_2\} \\
&= \bigsqcup \{u(g)(u_1, u_2) : u_1 \in A_1, u_2 \in A_2\} = \bigsqcup u(g)(A_1 \times A_2);
\end{aligned}$$

hence by Lemma 4.2.3 $u(g)$ is continuous. This gives a mapping

$$u : [D_1 \rightarrow [D_2 \rightarrow D]] \rightarrow [D_1 \times D_2 \rightarrow D]$$

and it is clear that u is monotone. Moreover, it is also clear that $c(u(g)) = g$ for each $g \in [D_1 \rightarrow [D_2 \rightarrow D]]$ and $u(c(h)) = h$ for each $h \in [D_1 \times D_2 \rightarrow D]$. Thus c is an order isomorphism and u is the corresponding inverse mapping. \square

Proposition 4.2.8 *The mapping $\text{pa} : [D_1 \times D_2 \rightarrow D] \times D_1 \rightarrow [D_2 \rightarrow D]$ given by*

$$\text{pa}(h, u_1)(u_2) = h(u_1, u_2)$$

for all $h \in [D_1 \times D_2 \rightarrow D]$, $u_1 \in D_1$, $u_2 \in D_2$, is continuous. This mapping is called a *partial application operator*.

Proof This follows immediately from Propositions 4.2.6 and 4.2.7, since

$$\text{pa}(h, u_1) = \mathbf{a}(c(h), u_1)$$

for all $h \in [D_1 \times D_2 \rightarrow D]$, $u_1 \in D_1$ (with the appropriate currying and application operators), making use of the special case of Proposition 4.2.3 and the fact that the identity mapping $\text{id} : D_1 \rightarrow D_1$ is continuous. \square

Proposition 4.2.9 *Let D be a bottomed complete poset and $h : D \rightarrow D$ be a continuous mapping. Then h possesses a least fixed point. More precisely, the set*

$$\text{Fix}(h) = \{u \in D : h(u) = u\}$$

is non-empty and contains a (unique) least element, i.e., $\text{Fix}(h)$ contains an element \hat{u} such that $\hat{u} \sqsubseteq u$ for all $u \in \text{Fix}(h)$.

Proof Define a sequence $\{u_n\}_{n \geq 0}$ of elements of D by $u_0 = \perp$ (with \perp the bottom element of D) and $u_{n+1} = h(u_n)$ for each $n \geq 0$. Then, since h is monotone, it follows by induction that $u_n \sqsubseteq u_{n+1}$ for all $n \geq 0$ (since $u_0 = \perp \sqsubseteq u_1$ and if $u_n \sqsubseteq u_{n+1}$ then $u_{n+1} = h(u_n) \sqsubseteq h(u_{n+1}) = u_{n+2}$). Thus the set

$$A = \{u \in D : u = u_n \text{ for some } n \geq 0\}$$

is directed and, because $h(A) = \{u \in D : u = u_n \text{ for some } n \geq 1\}$, it follows that $\bigsqcup A = \bigsqcup h(A)$. But $\bigsqcup h(A) = h(\bigsqcup A)$, since h is continuous, and this implies that $\hat{u} = \bigsqcup A \in \text{Fix}(h)$. Now let u be any element of $\text{Fix}(h)$; again by induction it follows that $u_n \sqsubseteq \hat{u}$ for all $n \geq 0$ (since $u_0 = \perp \sqsubseteq \hat{u}$ and $u_{n+1} = h(u_n) \sqsubseteq h(\hat{u}) = \hat{u}$ whenever $u_n \sqsubseteq \hat{u}$). Hence u is an upper bound of A and therefore $\hat{u} = \bigsqcup A \sqsubseteq u$. \square

The proof of Proposition 4.2.9 is just as important as its statement: This shows how the minimal fixed point \hat{u} can be constructed ‘explicitly’.

4.3 Initial completions

A poset Y_2 is said to be an *extension* of a poset Y_1 if Y_1 is a subposet of Y_2 . (This means that if \sqsubseteq_j is the partial order on Y_j for $j = 1, 2$ and $y, y' \in Y_1$ then $y \sqsubseteq_1 y'$ if and only if $y \sqsubseteq_2 y'$.) A poset D is said to be a *completion* of a poset Y if D is a complete extension of Y and each element of D is the least upper bound (in D) of some element of $d(Y)$.

Warning: Let D be a complete extension of a poset Y and let $A \in d(Y)$. Then it is possible that A has a least upper bound in Y which is not equal to the least upper bound of A in D . (Note, however, that if the least upper bound of A in D is an element of Y then in this case it must also be the least upper bound of A in Y .) When speaking just of the least upper bound of A or writing $\sqcup A$ then the least upper bound in D is always meant.

If Y is a poset with partial order \sqsubseteq and D is an extension of Y then, unless something explicit to the contrary is stated, the partial order on D will also be denoted by \sqsubseteq .

Lemma 4.3.1 *Let Y be a bottomed poset with bottom element \perp and let D be a completion of Y . Then D is bottomed and \perp is also the bottom element of D .*

Proof If $u \in D$ then $u = \sqcup A$ for some $A \in d(Y)$. But $\perp \sqsubseteq y$ for each $y \in A$, and hence also $\perp \sqsubseteq u$. \square

Proposition 4.3.1 *Let Y_σ be a family of posets, for each $\sigma \in S$ let D_σ be a completion of Y_σ , and let J be an S -typed set. Then D_\diamond^J is a completion of Y_\diamond^J .*

Proof Let $c \in D_\diamond^J$. Then for each $\eta \in J$ there exists $A_\eta \in d(Y_\eta)$ with $c(\eta) = \sqcup A_\eta$ (since D_η is a completion of Y_η). Put $A = \{b \in Y_\diamond^J : b(\eta) \in A_\eta \text{ for each } \eta \in J\}$; then $A \in d(Y_\diamond^J)$: If $c_1, c_2 \in A$ then for each $\eta \in J$ there exists $y_\eta \in A_\eta$ with $c_1(\eta) \sqsubseteq_\eta y_\eta$ and $c_2(\eta) \sqsubseteq_\eta y_\eta$; thus if $c \in Y_\diamond^J$ is defined by letting $c(\eta) = y_\eta$ for each $\eta \in J$ then $c \in A$ and $c_1 \sqsubseteq_\diamond^J c$, $c_2 \sqsubseteq_\diamond^J c$. Moreover, $c = \sqcup A$: If $b \in A$ then $b(\eta) \in A_\eta$ and so $b(\eta) \sqsubseteq_\eta \sqcup A_\eta = c(\eta)$ for each $\eta \in J$, i.e., $b \sqsubseteq_\diamond^J c$. On the other hand, if $c' \in D_\diamond^J$ is an upper bound of A then $b(\eta) \sqsubseteq_\eta c'(\eta)$ for each $b \in A$ and this implies that $y \sqsubseteq_\eta c'(\eta)$ for all $y \in A_\eta$ (since for each $y \in A_\eta$ there exists $b \in A$ with $b(\eta) = y$) and thus that $c(\eta) = \sqcup A_\eta \sqsubseteq_\eta c'(\eta)$ for each $\eta \in J$, i.e., $c \sqsubseteq_\diamond^J c'$. This, together with Proposition 4.2.2, shows that D_\diamond^J is a completion of Y_\diamond^J . \square

Extensions Y_1 and Y_2 of a poset Y are said to be *conjugate* if there exists an order isomorphism $h : Y_1 \rightarrow Y_2$ with $h(y) = y$ for all $y \in Y$. Conjugacy clearly defines an equivalence relation on the class of all extensions of Y . If D_1 and D_2 are conjugate extensions of Y then D_1 is a completion of Y if and only if D_2 is.

A completion D of a poset Y is said to be *initial* if for each completion D' of Y there exists a unique continuous mapping $h : D \rightarrow D'$ with $h(y) = y$ for all $y \in Y$. (Note that the mapping h is then automatically surjective.) It is easily checked that if D_1 and D_2 are conjugate completions of Y then D_2 is initial if and only if D_1 is.

Proposition 4.3.2 below states that each poset Y possesses an essentially unique initial completion, which is then often called *the* initial completion of Y . In fact, for reasons to be explained at the end of the section, this is also referred to as *the ideal completion of Y* .

A seemingly stronger requirement than being initial on the completion of a poset is the following: A completion D of Y is said to have the *continuous extension property* if whenever $h : Y \rightarrow D'$ is a monotone mapping from Y to a complete poset D' then h can be uniquely extended to a continuous mapping $h' : D \rightarrow D'$ (i.e., there exists a unique continuous mapping $h' : D \rightarrow D'$ with $h'(y) = h(y)$ for all $y \in Y$). It is clear that a completion D of Y with the continuous extension property is initial, since if D' is any completion of Y then the mapping $i : Y \rightarrow D'$ with $i(y) = y$ for all $y \in Y$ is monotone. The next result shows in particular that the converse is true.

Proposition 4.3.2 *Let Y be a poset. Then there is exactly one conjugacy class of initial completions of Y . Moreover, each initial completion of Y has the continuous extension property.*

Proof Let \sqsubseteq be the partial order on Y . It will be shown first that there is at most one conjugacy class of initial completions.

Lemma 4.3.2 *Initial completions D and D' of Y are conjugate.*

Proof There exists a continuous mapping $h : D \rightarrow D'$ with $h(y) = y$ for all $y \in Y$ and, reversing the roles of D and D' , also a continuous mapping $h' : D' \rightarrow D$ with $h'(y) = y$ for all $y \in Y$. But if $\text{id}_D : D \rightarrow D$ is the identity mapping then $h'h = \text{id}_D$, since $h'h$ and id_D are both continuous mappings from D to D with $h'h(y) = \text{id}_D(y) = y$ for all $y \in Y$. In the same way $hh' = \text{id}_{D'}$. Hence h is a bijection and both h and $h^{-1} = h'$ are continuous. This shows that D and D' are conjugate. \square

If D_1 and D_2 are conjugate completions of Y then it is easy to see that D_2 has the continuous extension property if and only if D_1 does. The proof of Proposition 4.3.2 thus reduces to showing there exists a completion of Y having this property. Moreover, the following lemma implies that in the definition of the continuous extension property the statement ‘can be uniquely extended to a continuous mapping’ can be replaced by just ‘can be extended to a continuous mapping’.

Lemma 4.3.3 *Let D be a completion of a poset Y and $h : Y \rightarrow D'$ be a monotone mapping from Y to a complete poset D' . Then there exists at most one extension of h to a continuous mapping $h' : D \rightarrow D'$ (i.e., there exists at most one continuous mapping $h' : D \rightarrow D'$ with $h'(y) = h(y)$ for all $y \in Y$).*

Proof Let $u \in D$; then there exists $A \in \mathbf{d}(Y)$ with $u = \bigsqcup A$ and therefore follows that $h'(u) = \bigsqcup h(Y)$. \square

Let $A_1, A_2 \in \mathbf{d}(Y)$; then A_1 is said to be *cofinal* in A_2 if for each $y_1 \in A_1$ there exists $y_2 \in A_2$ with $y_1 \sqsubseteq y_2$. If D is a complete extension of Y and $A_1, A_2 \in \mathbf{d}(Y)$ with A_1 cofinal in A_2 then clearly $\bigsqcup A_1 \sqsubseteq \bigsqcup A_2$.

Lemma 4.3.4 *Let D be a complete extension of Y (but not necessarily a completion of Y) and suppose that whenever A_1, A_2 are elements of $\mathbf{d}(Y)$ with $\sqcup A_1 \sqsubseteq \sqcup A_2$ then A_1 is cofinal in A_2 . Let D' denote the set of all elements of D having the form $\sqcup A$ for some $A \in \mathbf{d}(Y)$. Then for each $B \in \mathbf{d}(D')$ there exists $A \in \mathbf{d}(Y)$ with A cofinal in B and $\sqcup A = \sqcup B$.*

Proof Let $B \in \mathbf{d}(D')$ and for each $u \in B$ choose $A_u \in \mathbf{d}(Y)$ with $u = \sqcup A_u$. Then $A = \bigcup_{u \in B} A_u \in \mathbf{d}(Y)$. To see this let $y_1, y_2 \in A$; then there exist $u_1, u_2 \in B$ with $y_1 \in A_{u_1}$ and $y_2 \in A_{u_2}$ and, since $B \in \mathbf{d}(D')$, there exists $u \in B$ such that $u_1 \sqsubseteq u$ and $u_2 \sqsubseteq u$. But then by assumption A_{u_1} is cofinal in A_u and A_{u_2} is cofinal in A_u . Hence $y'_1, y'_2 \in A_u$ can be found with $y_1 \sqsubseteq y'_1$ and $y_2 \sqsubseteq y'_2$ and, since $A_u \in \mathbf{d}(Y)$, it follows that $y_1 \sqsubseteq y'_1 \sqsubseteq y$ and $y_2 \sqsubseteq y'_2 \sqsubseteq y$ for some $y \in A_u \subset A$.

Now if $y \in A$ then $y \in A_u$ for some $u \in B$, and so $y \sqsubseteq u$; this implies A is cofinal in B and thus also that $\sqcup A \sqsubseteq \sqcup B$. On the other hand, if $u \in B$ then $u \sqsubseteq \sqcup A$, since $u = \sqcup A_u$ and $A_u \subset A$, and therefore $\sqcup B \sqsubseteq \sqcup A$. Hence $\sqcup A = \sqcup B$. \square

Lemma 4.3.5 *Let D be a completion of Y such that whenever $A_1, A_2 \in \mathbf{d}(Y)$ with $\sqcup A_1 \sqsubseteq \sqcup A_2$ then A_1 is cofinal in A_2 . Then D has the continuous extension property.*

Proof Let $h : Y \rightarrow D'$ be a monotone mapping of Y to a complete poset D' . If $A_1, A_2 \in \mathbf{d}(Y)$ with $\sqcup A_1 = \sqcup A_2$ then each of A_1 and A_2 is cofinal in the other and therefore $\sqcup h(A_1) = \sqcup h(A_2)$. Thus, since each element of D has the form $\sqcup A$ for some $A \in \mathbf{d}(Y)$, a mapping $h' : D \rightarrow D'$ can be defined by putting $h'(u) = \sqcup h(A)$, where A is any element of $\mathbf{d}(Y)$ with $u = \sqcup A$. Then h' is monotone and an extension of h , because $\{y\} \in \mathbf{d}(Y)$ and $\sqcup \{y\} = y$ for each $y \in Y$.

It remains to show that the mapping h' is continuous, and for this it is enough to show that $h'(\sqcup B) \sqsubseteq \sqcup h'(B)$ for each $B \in \mathbf{d}(D)$ (with \sqsubseteq also denoting the partial order on D'). Let $B \in \mathbf{d}(D)$; then by Lemma 4.3.4 there exists $A \in \mathbf{d}(Y)$ with A cofinal in B and $\sqcup A = \sqcup B$. Hence $h'(\sqcup B) = h'(\sqcup A) = \sqcup h(A) = \sqcup h'(A) \sqsubseteq \sqcup h'(B)$. \square

A completion of Y will now be constructed and Lemma 4.3.5 then used to show that it has the continuous extension property. Elements $A_1, A_2 \in \mathbf{d}(Y)$ are said to be *mutually cofinal* if each is cofinal in the other. Mutual cofinality clearly defines an equivalence relation on the set $\mathbf{d}(Y)$. Let D denote the set of equivalence classes, and if $A \in \mathbf{d}(Y)$ then denote by $[A]$ the equivalence class containing A . If A_1 is equivalent to A'_1 and A_2 equivalent to A'_2 then clearly A_1 is cofinal in A_2 if and only if A'_1 is cofinal in A'_2 . Thus define a relation \preceq on D by letting $[A_1] \preceq [A_2]$ if A_1 is cofinal in A_2 ; it is immediate that \preceq is a partial order.

Lemma 4.3.6 *The poset D (with the partial order \preceq) is complete.*

Proof Let $B \in \mathbf{d}(D)$. For each $u \in B$ choose $A_u \in \mathbf{d}(Y)$ with $[A_u] = u$ and put $A = \bigcup_{u \in B} A_u$. Then $A \in \mathbf{d}(Y)$. (Let $y_1, y_2 \in A$; then there exist $u_1, u_2 \in B$ such that $y_1 \in A_{u_1}$ and $y_2 \in A_{u_2}$ and, since $B \in \mathbf{d}(D)$, there exists $u \in B$ with $u_1 \preceq u$ and $u_2 \preceq u$. But this means that $y'_1, y'_2 \in A_u$ can be found with $y_1 \sqsubseteq y'_1$ and $y_2 \sqsubseteq y'_2$

and hence there exists $y \in A_u \subset A$ such that $y_1 \sqsubseteq y'_1 \sqsubseteq y$ and $y_2 \sqsubseteq y'_2 \sqsubseteq y$, because $A_u \in \mathbf{d}(Y)$. It is now enough to show that $\hat{u} = [A]$ is the least upper bound of B in D . If $u \in B$ and $y \in A_u$ then $y \sqsubseteq \hat{u}$ and $y \in A$; thus A_u is cofinal in A . This implies that $u \preceq \hat{u}$, i.e., \hat{u} is an upper bound of B . Now let $u' = [A']$ be any upper bound of B . If $y \in A$ then $y \in A_u$ for some $u \in B$ and A_u is cofinal in A' , since $u \preceq u'$. There therefore exists $y' \in A'$ with $y \sqsubseteq y'$ and hence A is cofinal in A' , i.e., $\hat{u} \preceq u'$. \square

Now $\{y\} \in \mathbf{d}(Y)$ for each $y \in Y$, which means a mapping $i : Y \rightarrow D$ can be defined by putting $i(y) = [\{y\}]$ for each $y \in Y$. Then i is an embedding, since $y_1 \sqsubseteq y_2$ if and only if $\{y_1\}$ is cofinal in $\{y_2\}$, and Y can thus be considered as a subposet of D by identifying Y with $i(Y)$. But this just means that D is an extension of Y . Moreover, if $u = [A] \in D$ and $B = \bigcup_{y \in A} \{y\}$ then B is a directed subset of $Y \subset D$ and, as in the proof of Lemma 4.3.6, $u = \bigsqcup B$. This shows D is a completion of Y .

Let $A_1, A_2 \in \mathbf{d}(Y)$ with $\bigsqcup A_1 \preceq \bigsqcup A_2$. Then $[A_1] \preceq [A_2]$, since $\bigsqcup A = [A]$ for each $A \in \mathbf{d}(Y)$, which by definition means that A_1 is cofinal in A_2 . Therefore by Lemma 4.3.5 D has the continuous extension property. This completes the proof of Proposition 4.3.2. \square

Warning: If D is an initial completion of a poset Y then in general D is not an initial completion of itself. In fact, the class of complete posets which are initial completions of themselves is very special and will be characterised in Proposition 4.3.7.

The next result gives various properties which are equivalent to a completion being initial.

Proposition 4.3.3 *Let D be a completion of a poset Y . Then the following five statements are equivalent:*

- (i) D is initial.
- (ii) D has the continuous extension property.
- (iii) A_1 is cofinal in A_2 whenever A_1, A_2 are elements of $\mathbf{d}(Y)$ with $\bigsqcup A_1 \sqsubseteq \bigsqcup A_2$.
- (iv) $y \sqsubseteq y'$ for some $y' \in A$ whenever $y \in Y$ and $A \in \mathbf{d}(Y)$ with $y \sqsubseteq \bigsqcup A$.
- (v) $y \sqsubseteq u$ for some $u \in A$ whenever $y \in Y$ and $A \in \mathbf{d}(D)$ with $y \sqsubseteq \bigsqcup A$.

Proof To show that the first three statements are equivalent it is enough to prove the converse of Lemma 4.3.5. Thus assume that D has the continuous extension property and let A_1, A_2 be elements of $\mathbf{d}(Y)$ with A_1 not cofinal in A_2 ; then there exists $y_1 \in A_1$ such that $y_1 \not\sqsubseteq y$ for all $y \in A_2$. Choose some complete poset D' containing elements u, u' with $u \neq u'$ and $u' \sqsubseteq u$; then

$$h(y) = \begin{cases} u & \text{if } y_1 \sqsubseteq y, \\ u' & \text{otherwise,} \end{cases}$$

defines a monotone mapping $h : Y \rightarrow D'$ which by assumption extends to a continuous mapping $h' : D \rightarrow D'$. But $h'(\bigsqcup A_2) = \bigsqcup h'(A_2) = u'$ and $h'(\bigsqcup A_1) = \bigsqcup h'(A_1) = u$

and hence $\bigsqcup A_1 \sqsubseteq \bigsqcup A_2$ cannot hold. Thus if $\bigsqcup A_1 \sqsubseteq \bigsqcup A_2$ then A_1 must be cofinal in A_2 .

This shows that (i), (ii) and (iii) are equivalent. But it is clear that (iii) and (iv) are equivalent and that (v) implies (iv); it thus remains to show that the last statement is implied by the others and this follows immediately from Lemma 4.3.4. \square

Proposition 4.3.4 *Let D be an initial completion of a poset Y and for each $u \in D$ let $A_u = \{y \in Y : y \sqsubseteq u\}$. Then $A_u \in \mathbf{d}(Y)$ and $\bigsqcup A_u = u$.*

Proof Let $u \in D$; then, since D is a completion of Y , there exists $A \in \mathbf{d}(Y)$ with $\bigsqcup A = u$. Thus $A \subset A_u$ and so in particular $A_u \neq \emptyset$. Let $y \in A_u$; then $y \sqsubseteq u = \bigsqcup A$ and hence by Proposition 4.3.3 there exists $y' \in A$ with $y \sqsubseteq y'$. Therefore if $y_1, y_2 \in A_u$ then there exist $y'_1, y'_2 \in A$ with $y_1 \sqsubseteq y'_1$ and $y_2 \sqsubseteq y'_2$. But $A \in \mathbf{d}(Y)$ and so there exists $y' \in A$ with $y'_1 \sqsubseteq y'$ and $y'_2 \sqsubseteq y'$, i.e., there exists $y' \in A_u$ with $y_1 \sqsubseteq y'$ and $y_2 \sqsubseteq y'$. This shows that $A_u \in \mathbf{d}(Y)$. It is now clear that $\bigsqcup A_u = u$, since $u = \bigsqcup A \sqsubseteq \bigsqcup A_u$ and by definition u is an upper bound of A_u . \square

Proposition 4.3.5 *Let D be an initial completion of a poset Y and Y' be a subposet of Y ; let D' be the set of all elements of D having the form $\bigsqcup A'$ for some $A' \in \mathbf{d}(Y')$. Then D' is an initial completion of Y' . Moreover, if $B \in \mathbf{d}(D')$ then $\bigsqcup B$ (the least upper bound of B in D) is an element of D' , and hence $\bigsqcup B$ is also the least upper bound of B in D' .*

Proof If A'_1, A'_2 are elements of $\mathbf{d}(Y')$ (and thus also of $\mathbf{d}(Y)$) with $\bigsqcup A'_1 \sqsubseteq \bigsqcup A'_2$ then by Proposition 4.3.3 A'_1 is cofinal in A'_2 . Let $B \in \mathbf{d}(D')$; then Lemma 4.3.4 (applied with D considered as a complete extension of Y') implies that there exists $A' \in \mathbf{d}(Y')$ with A' cofinal in B and $\bigsqcup A' = \bigsqcup B$. Hence $\bigsqcup B$ is an element of D' and so $\bigsqcup B$ is also the least upper bound of B in D' . In particular, D' is complete, and therefore a completion of Y' . But now Proposition 4.3.3 implies that D' is an initial completion of Y' . (If $A_1, A_2 \in \mathbf{d}(Y')$ with $\bigsqcup A_1 \sqsubseteq \bigsqcup A_2$ then A_1 is cofinal in A_2 , since $A_1, A_2 \in \mathbf{d}(Y)$ and D is an initial completion of Y .) \square

Proposition 4.3.6 *Let Y_S be a family of posets, for each $\sigma \in S$ let D_σ be an initial completion of Y_σ and let J be a finite S -typed set. Then D_\diamond^J is an initial completion of Y_\diamond^J .*

Proof By Proposition 4.3.1 D_\diamond^J is a completion of Y_\diamond^J . Let A and A' be elements of $\mathbf{d}(Y_\diamond^J)$ with $\bigsqcup A \sqsubseteq_\diamond^J \bigsqcup A'$. For each $\eta \in J$ let $A_\eta = \{b(\eta) : b \in A\}$ and $A'_\eta = \{b(\eta) : b \in A'\}$. Then $A_\eta, A'_\eta \in \mathbf{d}(Y_\eta)$ and $\bigsqcup A_\eta \sqsubseteq_\eta \bigsqcup A'_\eta$ and thus, since D_η is initial, A_η is cofinal in A'_η for each $\eta \in J$. From this it follows that A is cofinal in A' : Let $c \in A$; then for each $\eta \in J$ there exists $c_\eta \in A'$ with $c(\eta) \sqsubseteq_\eta c_\eta(\eta)$, and therefore, since J is finite and A' is directed, there exists $c' \in A'$ with $c_\eta \sqsubseteq_\diamond^J c'$ for all $\eta \in J$. But then $c \sqsubseteq_\diamond^J c'$. Hence by Proposition 4.3.3 D_\diamond^J is an initial completion of Y_\diamond^J . \square

Proposition 4.3.7 *Let D be a complete poset. Then D is an initial completion of itself if and only if each directed subset of D contains a maximum element (i.e., if and only if for each $A \in \mathfrak{d}(D)$ then there exists $u \in A$ with $u' \sqsubseteq u$ for all $u' \in A$).*

Proof Suppose first that D is an initial completion of itself and let $A \in \mathfrak{d}(D)$ with $\bigsqcup A = u$. Then $u \sqsubseteq \bigsqcup A$ and so by Proposition 4.3.3 $u \sqsubseteq u'$ for some $u' \in A$. But this is only possible if $u = u'$ and hence $u \in A$, i.e., A contains a maximum element (namely u). The converse follows directly from Proposition 4.3.3. \square

If Y is a poset then a sequence of elements $\{y_n\}_{n \geq 0}$ from Y is called a *chain* in Y if $y_n \sqsubseteq y_{n+1}$ for each $n \geq 0$. A chain $\{y_n\}_{n \geq 0}$ is said to be *finite* if $y_m = y_n$ for all $m \geq n$ for some $n \geq 0$. Now let D be a complete poset; if D is an initial completion of itself then by Proposition 4.3.7 each chain in D must be finite (since the elements in a chain form a directed set). The converse is in fact also true: If each chain in D is finite then D is an initial completion of itself. This follows from Proposition 4.3.7 together with a standard application of Zorn's lemma.

Let D be a complete poset and D' be an initial completion of D . Then the identity mapping $i_D : D \rightarrow D$ is monotone and so it extends uniquely to a continuous mapping $i_D^* : D' \rightarrow D$, which is obviously surjective. However, if $D' \neq D$ then i_D^* cannot be injective. The following simple example indicates the kind of behaviour which occurs here: Let X be a set and consider the set $D = \mathcal{P}(X)$ of all subsets of X , regarded as a poset with the inclusion ordering. Then D is complete, since if $A \in \mathfrak{d}(D)$ then clearly $\bigsqcup A = \bigcup_{B \in A} B$. However, by Proposition 4.3.7 D is an initial completion of itself if and only if X is finite. Let D' be an initial completion of D and let $i_D^* : D' \rightarrow D$ be the unique continuous extension of the identity mapping i_D . The proof of Proposition 4.3.2 shows that if B is an infinite subset of X then the set $\{u \in D' : i_D^*(u) = B\}$ is infinite (since there are then infinitely many non-equivalent directed subsets of D having B as least upper bound).

In what follows some remarks will be made about the initial completion of the poset $(Y_1 \rightarrow Y_2)$. The purpose of these remarks is mainly to warn the reader that such an initial completion is more complicated than perhaps would be expected. Let Y_1 and Y_2 be posets and for $j = 1, 2$ let D_j be an initial completion of Y_j . Then each monotone mapping from Y_1 to Y_2 extends uniquely to a continuous mapping from D_1 to D_2 , a mapping $\psi : (Y_1 \rightarrow Y_2) \rightarrow [D_1 \rightarrow D_2]$ can thus be defined by letting $\psi(f)$ be the unique continuous extension of $f \in (Y_1 \rightarrow Y_2)$.

Lemma 4.3.7 *The mapping $\psi : (Y_1 \rightarrow Y_2) \rightarrow [D_1 \rightarrow D_2]$ is an embedding, i.e., $f_1 \sqsubseteq f_2$ if and only if $\psi(f_1) \sqsubseteq \psi(f_2)$.*

Proof This is straightforward. \square

By Lemma 4.3.7 $(Y_1 \rightarrow Y_2)$ can be considered as a subposet of $[D_1 \rightarrow D_2]$ (by identifying $f \in (Y_1 \rightarrow Y_2)$ with the continuous mapping $\psi(f)$). However, in general $[D_1 \rightarrow D_2]$ will not be an initial completion of $(Y_1 \rightarrow Y_2)$. For example, if the partial order on Y_1 is trivial (i.e., $y \sqsubseteq y'$ if and only if $y = y'$) and $Y_2 = \{0, 1\}$ with $0 \sqsubseteq 1$ then $(Y_1 \rightarrow Y_2)$ can clearly be identified with the poset $\mathcal{P}(Y_1)$ (with the

inclusion ordering). But Proposition 4.3.7 implies that $D_1 = Y_1$ and $D_2 = Y_2$, and here $[D_1 \rightarrow D_2] = (Y_1 \rightarrow Y_2)$. On the other hand, if Y_1 is infinite then, as noted above, $\mathcal{P}(Y_1)$ is not an initial completion of itself.

Now let D be an initial completion of $(Y_1 \rightarrow Y_2)$. Then, the monotone mapping ψ extends uniquely to a continuous mapping $\psi^* : D \rightarrow [D_1 \rightarrow D_2]$ (since $[D_1 \rightarrow D_2]$ is complete). As noted above, ψ^* is in general not injective, and so more than one (and often infinitely many) elements of D will be associated with the same continuous mapping from D_1 to D_2 .

Consider the application operator $\mathbf{a} : (Y_1 \rightarrow Y_2) \times Y_1 \rightarrow Y_2$. This is clearly monotone and by Proposition 4.3.6 $D \times D_1$ is an initial extension of $(Y_1 \rightarrow Y_2) \times Y_1$. Thus \mathbf{a} extends uniquely to a continuous mapping $\mathbf{a}^* : D \times D_1 \rightarrow D_2$. Moreover,

$$\mathbf{a}^*(u, u_1) = \psi^*(u)(u_1)$$

for all $u \in D$, $u_1 \in D_1$, since $\mathbf{a}(f, y_1) = f(y_1) = \psi(f)(y_1)$ for all $f \in (Y_1 \rightarrow Y_2)$, $y_1 \in Y_1$, and hence $(u, u_1) \mapsto \mathbf{a}^*(u, u_1)$ and $(u, u_1) \mapsto \psi^*(u)(u_1)$ are both continuous extensions of the same monotone mapping. However, \mathbf{a}^* should not really be regarded as an application operator, since in general it fails to have the *extensionality* property enjoyed by \mathbf{a} , namely that if $f, f' \in (Y_1 \rightarrow Y_2)$ and $\mathbf{a}(f, y) = \mathbf{a}(f', y)$ for all $y \in Y_1$ then $f = f'$.

The final topic of this section explains why the initial completion also goes under the name of the ideal completion. In what follows let Y be a fixed poset. A non-empty subset I of Y is called an *ideal of Y* if $y \in I$ whenever $y \sqsubseteq y'$ for some $y' \in I$. An ideal I is said to be *directed* if $I \in \mathbf{d}(Y)$. In particular, the set

$$I(y) = \{y' \in Y : y' \sqsubseteq y\}$$

is a directed ideal for each $y \in Y$ (*the principal ideal generated by y*).

Lemma 4.3.8 *Let $y_1, y_2 \in Y$. Then $y_1 \sqsubseteq y_2$ if and only if $I(y_1) \subset I(y_2)$.*

Proof This is clear. \square

Now denote by $\mathcal{I}_d(Y)$ the set of directed ideals of Y , regarded as a poset with the inclusion ordering. By Lemma 4.3.8 $\mathcal{I}_d(Y)$ can be considered as an extension of Y (by identifying the element y with the principal ideal $I(y)$ for each $y \in Y$).

Proposition 4.3.8 *$\mathcal{I}_d(Y)$ is an initial completion (called the ideal completion) of Y .*

Proof It is clearly enough to show that $\mathcal{I}_d(Y)$ is conjugate to the initial completion constructed in the proof of Proposition 4.3.2 (since an extension conjugate to an initial completion is also an initial completion). For each non-empty subset A of Y let

$$I(A) = \{y \in Y : y \sqsubseteq y' \text{ for some } y' \in A\}.$$

Lemma 4.3.9 (1) $I(A)$ is an ideal with $A \subset I(A)$. Moreover, if I is any ideal with $A \subset I$ then $I(A) \subset I$, i.e., $I(A)$ is the smallest ideal containing A .

(2) If $A \in \mathfrak{d}(Y)$ then $I(A) \in \mathfrak{d}(Y)$; moreover, A and $I(A)$ are mutually cofinal.

(3) If $A_1, A_2 \in \mathfrak{d}(Y)$ then $I(A_1) \subset I(A_2)$ if and only if A_1 is cofinal in A_2 .

Proof This is straightforward. \square

Now let D be as in the proof of Proposition 4.3.2. Recall that mutual cofinality defines an equivalence relation on the set $\mathfrak{d}(Y)$ and that D is the set of equivalence classes, considered as a poset with the partial order \preceq defined by stipulating that $[A_1] \preceq [A_2]$ if A_1 is cofinal in A_2 (and where $[A]$ denotes the equivalence class containing A for each $A \in \mathfrak{d}(Y)$).

If $A_1, A_2 \in \mathfrak{d}(Y)$ then by Lemma 4.3.9 (3) $I(A_1) = I(A_2)$ if and only if A_1 and A_2 are equivalent, thus by Lemma 4.3.9 (2) a mapping $h : D \rightarrow \mathcal{I}_{\mathfrak{d}}(Y)$ can be defined by letting $h([A]) = I(A)$ for each $A \in \mathfrak{d}(Y)$. Moreover, Lemma 4.3.9 (3) then implies that h is an embedding. In fact h is also surjective, since by Lemma 4.3.9 (1) and (2)

$$A = I(A) = h([A])$$

for each $A \in \mathcal{I}_{\mathfrak{d}}(Y)$. Hence h is an order isomorphism. But $h(y) = y$ for each $y \in Y$ (because y is identified with $[\{y\}]$ in D , with $I(y)$ in $\mathcal{I}_{\mathfrak{d}}(Y)$, and $I(\{y\}) = I(y)$), and therefore D and $\mathcal{I}_{\mathfrak{d}}(Y)$ are conjugate extensions of Y . \square

4.4 Initial completion of monotone regular extensions

For the whole of this section let (Y_B, q_K) be a monotone regular bottomed Λ -algebra and \sqsubseteq_B be an associated ordering which is monotone; Y_β will always be considered as a poset with the partial order \sqsubseteq_β for each $\beta \in B$.

Lemma 4.4.1 *For each $\beta \in B$ let D_β be an initial completion of the poset Y_β and let $\kappa \in K$ be of type $L \rightarrow \beta$. Then the mapping $q_\kappa : Y_\diamond^L \rightarrow Y_\beta$ extends uniquely to a continuous mapping $f_\kappa : D_\diamond^L \rightarrow D_\beta$.*

Proof By assumption the mapping $q_\kappa : Y_\diamond^L \rightarrow Y_\beta$ is monotone and so it is still monotone considered as a mapping from Y_\diamond^L to D_β ; moreover, by Proposition 4.3.6 D_\diamond^L is an initial completion of Y_\diamond^L . The result thus follows from Proposition 4.3.3. \square

Lemma 4.4.1 allows the following definition to be made: A Λ -algebra (D_B, f_K) is said to be an *initial completion* of (Y_B, q_K) if D_β is an initial completion of the poset Y_β for each $\beta \in B$ and f_κ is the unique continuous extension of q_κ for each $\kappa \in K$. (Of course, whether or not (D_B, f_K) is an initial completion of (Y_B, q_K) depends on the associated ordering \sqsubseteq_B . However, it will be assumed that this can be determined from the context.) It follows immediately from Proposition 4.3.2 that there exists an initial completion of (Y_B, q_K) .

An initial completion (D_B, f_K) of (Y_B, q_K) is, in particular, an extension of (Y_B, q_K) . Moreover, by Lemma 4.3.1 the bottom element \perp_β of Y_β is also the the bottom element of the poset D_β . The partial order on D_β will always be denoted by \sqsubseteq_β .

If (Y_B, q_K) is a bottomed extension of (X_B, p_K) then, of course, any initial completion of (Y_B, q_K) is also a bottomed extension of (X_B, p_K) .

Proposition 4.4.1 *Let (D_B, f_K) and (D'_B, f'_K) be initial completions of (Y_B, q_K) . Then for each $\beta \in B$ there exists a unique order isomorphism $\pi_\beta : D_\beta \rightarrow D'_\beta$ with $\pi_\beta(y) = y$ for each $y \in Y_\beta$ (and by Proposition 4.2.1 π_β and the inverse mapping π_β^{-1} are then both continuous). Moreover, the family π_B is also an isomorphism from (D_B, f_K) to (D'_B, f'_K) .*

Proof There exists a unique continuous mapping $\pi_\beta : D_\beta \rightarrow D'_\beta$ with $\pi_\beta(y) = y$ for all $y \in Y_\beta$ because D_β is an initial completion of Y_β . Moreover, exactly as in the proof of Lemma 4.3.2, π_β is an order isomorphism. The uniqueness follows from Proposition 4.2.1. and Lemma 4.3.3.

It remains to show that $\pi_B : (D_B, f_K) \rightarrow (D'_B, f'_K)$ is a homomorphism (and thus an isomorphism). Let $\kappa \in K$ be of type $L \rightarrow \beta$; then Proposition 4.2.3 implies that $\pi_\diamond^L : D_\diamond^L \rightarrow (D'_\diamond)^L$ is continuous, and so $\pi_\beta f_\kappa$ and $f'_\kappa \pi_\diamond^L$ are both continuous mappings from D_\diamond^L to D'_β with

$$\pi_\beta f_\kappa(c) = \pi_\beta q_\kappa(c) = q_\kappa(c) = q_\kappa \pi_\diamond^L(c) = f'_\kappa \pi_\diamond^L(c)$$

for all $c \in Y_\diamond^L$. Therefore by Lemma 4.3.3 $\pi_\beta f_\kappa = f'_\kappa \pi_\diamond^L$, since D_\diamond^L is a completion of Y_\diamond^L . \square

Note that in the special case when (Y_B, q_K) is the flat extension of (X_B, p_K) (and so there is a unique associated ordering) then (Y_B, q_K) itself is the (only possible) initial completion of (Y_B, q_K) : The poset Y_β is clearly complete and therefore by Proposition 4.3.7 Y_β is an initial completion of itself, since a directed subset of Y_β can contain at most two elements and a finite directed set always contains a maximum element. Example 4.4.1 on the next page gives an initial completion of the Λ -algebra (X_B^\top, p_K^\top) introduced in Example 3.2.2.

In what follows let (D_B, f_K) be an initial completion of (Y_B, q_K) ; Proposition 4.4.1 implies that this initial completion is, in the appropriate sense, unique.

Proposition 4.4.2 *(D_B, f_K) is a monotone regular bottomed Λ -algebra and \sqsubseteq_B is an ordering associated with (D_B, f_K) which is monotone. Moreover, (Y_B, q_K) and (D_B, f_K) have the same trace, and if (Y_B, q_K) is (H_B, \diamond_K) -cored for some core type (H_B, \diamond_K) then so is (D_B, f_K) .*

Proof Denote the core of (Y_B, q_K) by C'_K and that of (D_B, f_K) by C_K . The following fact relating the cores C'_K and C_K is needed:

Lemma 4.4.2 *Let $\kappa \in K$ be of type $L \rightarrow \beta$. Then*

$$C_\kappa = \{c \in D_\diamond^L : c' \sqsubseteq_\kappa c \text{ for some } c' \in C'_\kappa\}.$$

Moreover, C_κ is also the set of all elements of D_\diamond^L having the form $\bigsqcup A$ for some $A \in \mathbf{d}(Y_\diamond^L)$ with $A \subset C'_\kappa$.

Proof Let $c \in D_\diamond^L$ and suppose $c' \sqsubseteq_\kappa c$ for some $c' \in C'_\kappa$; then $\perp_\beta \neq f_\kappa(c') \sqsubseteq_\beta f_\kappa(c)$ and hence $f_\kappa(c) \neq \perp_\beta$, i.e., $c \in C_\kappa$. Conversely, suppose $c \in D_\diamond^L$ with $f_\kappa(c) \neq \perp_\beta$. Then, since D_\diamond^L is a completion of Y_\diamond^L , there exists $A \in \mathbf{d}(Y_\diamond^L)$ with $c = \bigsqcup A$ and therefore $\bigsqcup f_\kappa(A) = f_\kappa(\bigsqcup A) \neq \perp_\beta$ (because f_κ is continuous). But this implies that $f_\kappa(c') \neq \perp_\beta$ for some $c' \in A$, and then $c' \in C'_\kappa$ with $c' \sqsubseteq_\kappa c$.

If $c \in D_\diamond^L$ has the form $\bigsqcup A$ for some $A \in \mathbf{d}(Y_\diamond^L)$ with $A \subset C'_\kappa$ then in particular $c' \sqsubseteq_\kappa c$ for all $c' \in A$; hence $c \in C_\kappa$. Suppose conversely that $c \in C_\kappa$, and let $B \in \mathbf{d}(Y_\diamond^L)$ with $\bigsqcup B = c$. Then, putting $A = B \cap C'_\kappa$, it is easily checked that also $A \in \mathbf{d}(Y_\diamond^L)$ and $\bigsqcup A = c$. \square

The proof of Proposition 4.4.2 can now be commenced, and it will be shown first that \sqsubseteq_B is an ordering associated with (D_B, f_K) . Clearly $\perp_\beta \sqsubseteq_\beta c$ for each $c \in D_\beta$. Moreover, if $\kappa \in K_\beta$ and $c, c' \in C_\kappa$ with $c \sqsubseteq_\kappa c'$ then $f_\kappa(c) \sqsubseteq_\beta f_\kappa(c')$, because f_κ is monotone. Thus consider $\kappa, \kappa' \in K$ having types $L \rightarrow \beta$ and $L' \rightarrow \beta$ respectively, and let $c \in C_\kappa, c' \in C_{\kappa'}$ with $f_\kappa(c) \sqsubseteq_\beta f_{\kappa'}(c')$. By Lemma 4.4.2 there exist $A \in \mathbf{d}(Y_\diamond^L)$ and $A' \in \mathbf{d}(Y_\diamond^{L'})$ with $A \subset C'_\kappa, A' \subset C'_{\kappa'}, \bigsqcup A = c$ and $\bigsqcup A' = c'$, and then

$$\bigsqcup q_\kappa(A) = \bigsqcup f_\kappa(A') = f_\kappa(\bigsqcup A) \sqsubseteq_\beta f_{\kappa'}(\bigsqcup A') = \bigsqcup f_{\kappa'}(A') = \bigsqcup q_{\kappa'}(A')$$

(since f_κ is continuous). Therefore by Proposition 4.3.3 $q_\kappa(A)$ is cofinal in $q_{\kappa'}(A')$, and from the definition of an ordering associated with (Y_B, q_K) this is only possible if $\kappa = \kappa'$. Hence in fact $q_\kappa(A)$ is cofinal in $q_\kappa(A')$ which, again using the definition of an

Example 4.4.1 Define a Λ -algebra (D_B^\top, f_K^\top) , which is an extension of the Λ -algebra (X_B^\top, p_K^\top) introduced in Example 3.2.2, as follows (where if Z is a set then Z^∞ denotes the set of all infinite lists of elements from Z):

$$D_{\text{bool}}^\top = X_{\text{bool}}^\top, \quad D_{\text{nat}}^\top = X_{\text{nat}}^\top \cup \{\infty\}, \text{ where } \infty \notin X_{\text{nat}}^\top,$$

$$D_{\text{int}}^\top = X_{\text{int}}^\top, \quad D_{\text{ipr}}^\top = X_{\text{ipr}}^\top,$$

$$D_{\text{ilst}}^\top = X_{\text{ilst}}^\top \cup (X_{\text{int}}^\top)^\infty,$$

$$f_{\text{True}}^\top = p_{\text{True}}^\top, \quad f_{\text{False}}^\top = p_{\text{False}}^\top,$$

$$f_{\text{Zero}}^\top : \mathbb{I} \rightarrow D_{\text{nat}}^\top \text{ with } f_{\text{Zero}}^\top(\varepsilon) = 0,$$

$$f_{\text{Succ}}^\top : D_{\text{nat}}^\top \rightarrow D_{\text{nat}}^\top \text{ with}$$

$$f_{\text{Succ}}^\top(n) = \begin{cases} p_{\text{Succ}}^\top(n) & \text{if } n \in X_{\text{nat}}^\top, \\ \infty & \text{if } n = \infty, \end{cases}$$

$$f_{\underline{n}}^\top = p_{\underline{n}}^\top \text{ for each } n \in \mathbb{Z}, \quad f_{\text{Pair}}^\top = p_{\text{Pair}}^\top,$$

$$f_{\text{Nil}}^\top : \mathbb{I} \rightarrow D_{\text{ilst}}^\top \text{ with } f_{\text{Nil}}^\top(\varepsilon) = \varepsilon,$$

$$f_{\text{Cons}}^\top : D_{\text{int}}^\top \times D_{\text{ilst}}^\top \rightarrow D_{\text{ilst}}^\top \text{ with}$$

$$f_{\text{Cons}}^\top(n, s) = \begin{cases} p_{\text{Cons}}^\top(n, s) & \text{if } s \in X_{\text{ilst}}^\top, \\ n \triangleleft s & \text{if } s \in (X_{\text{int}}^\top)^\infty. \end{cases}$$

Moreover, for each $\beta \in B$ the partial order \sqsubseteq_β on X_β^\top can be extended to a partial order \sqsubseteq_β on D_β^\top (which means that $\sqsubseteq_{\text{bool}}$, \sqsubseteq_{int} and \sqsubseteq_{ipr} are just the corresponding partial orders on X_{bool}^\top , X_{int}^\top and X_{ipr}^\top):

\sqsubseteq_{nat} is defined to be the extension of the partial order \sqsubseteq_{nat} on X_{nat}^\top such that $\infty \sqsubseteq_{\text{nat}} n$ if and only if $n = \infty$ and $n \sqsubseteq_{\text{nat}} \infty$ if and only if $n \in \text{bot}(\mathbb{N}) \cup \{\infty\}$.

$\sqsubseteq_{\text{ilst}}$ is defined to be the extension of the partial order $\sqsubseteq_{\text{ilst}}$ on X_{ilst}^\top such that if $s = x_1 x_2 \cdots \in (X_{\text{int}}^\top)^\infty$ and $z \in X_{\text{ilst}}^\top$ then $s \sqsubseteq_{\text{ilst}} z'$ if and only if $z' = x'_1 x'_2 \cdots \in (X_{\text{int}}^\top)^\infty$ with $x_n \sqsubseteq_{\text{int}} x'_n$ for all $n \in \mathbb{N}$, and $z \sqsubseteq_{\text{ilst}} s$ if and only if $z = (x'_1 \cdots x'_m)^\perp$ for some $m \in \mathbb{N}$ and $x'_n \sqsubseteq_{\text{int}} x_n$ for each $n = 0, \dots, m$.

It is left to the reader to check that (D_B^\top, f_K^\top) (where D_β^\top is considered as a poset with the partial order \sqsubseteq_β) is an initial completion of (X_B^\top, p_K^\top) .

ordering associated with (Y_B, q_K) , implies that A is cofinal in A' , and so in particular $c = \bigsqcup A \sqsubseteq_{\kappa} \bigsqcup A' = c'$. This shows that \sqsubseteq_B is an ordering associated with (D_B, f_K) , and by definition \sqsubseteq_B is then monotone (since the mapping f_{κ} , being continuous, is in particular monotone).

It will be shown next that $\bigcup_{\kappa \in K_{\beta}} f_{\kappa}(C_{\kappa}) = D_{\beta} \setminus \{\perp_{\beta}\}$ for each $\beta \in B$. The regularity of (D_B, f_K) then follows from Proposition 4.1.1 and Lemma 3.3.1. Let $u \in D_{\beta} \setminus \{\perp_{\beta}\}$ and let $A \in \mathbf{d}(Y_{\beta})$ with $\bigsqcup A = u$. It can clearly be assumed that $\perp_{\beta} \notin A$ and hence for each $y \in A$ there exists a unique $\kappa \in K_{\beta}$ and a unique element $u \in C'_{\kappa}$ such that $q_{\kappa}(u) = y$. But from the definition of an ordering associated with (Y_B, q_K) this is only possible if κ does not depend on y . Thus there exists $\kappa \in K$ having type $L \rightarrow \beta$ for some L and for each $y \in A$ a unique element $c_y \in C'_{\kappa}$ such that $q_{\kappa}(c_y) = y$. Let $B = \{c_y : y \in A\}$; then, again using the definition of an associated ordering, it follows that $B \in \mathbf{d}(Y_{\circ}^L)$. Let $c = \bigsqcup B$; then by Lemma 4.4.2 $c \in C_{\kappa}$ and

$$f_{\kappa}(c) = f_{\kappa}(\bigsqcup B) = \bigsqcup f_{\kappa}(B) = \bigsqcup q_{\kappa}(B) = \bigsqcup A = u,$$

since f_{κ} is continuous. Therefore $u \in q_{\kappa}(C_{\kappa})$.

The monotonicity of (D_B, f_K) follows directly from Proposition 4.1.6, and (Y_B, q_K) and (D_B, f_K) have the same trace because there is essentially no difference between the homomorphism $\llbracket \cdot \rrbracket_B^{\perp} : (F_B^{\flat}, \odot_K^{\flat}) \rightarrow (Y_B, q_K)$ and the corresponding homomorphism from $(F_B^{\flat}, \odot_K^{\flat})$ to (D_B, f_K) .

It remains to show that if (Y_B, q_K) is (H_K, \diamond_K) -cored for some core type (H_B, \diamond_K) then so is (D_B, f_K) . Let $\kappa \in K$ be of type $L \rightarrow \beta$ and let $c \in D_{\circ}^L$. Then it is easy to see that there exists $c' \in Y_{\circ}^L$ such that $\varepsilon_{\eta}(c'(\eta)) = \varepsilon_{\eta}(c(\eta))$ for each $\eta \in L$ and such that $\varepsilon_{\beta}(f_{\kappa}(c')) = \varepsilon_{\beta}(f_{\kappa}(c))$. Thus

$$\varepsilon_{\beta}(f_{\kappa}(c)) = \varepsilon_{\beta}(f_{\kappa}(c')) = \varepsilon_{\beta}(q_{\kappa}(c')) = \diamond_{\kappa}(\varepsilon_{\circ}^L(c')) = \diamond_{\kappa}(\varepsilon_{\circ}^L(c))$$

and hence (D_B, f_K) is also (H_K, \diamond_K) -cored. \square

Suppose now that (Y_B, p_K) (and thus (D_B, f_K)) is a bottomed extension of (X_B, p_K) .

Proposition 4.4.3 *Each element of X_{β} is a maximal element of D_{β} .*

Proof This follows immediately from Proposition 4.1.3, since by Proposition 4.4.2 (D_B, f_K) is a regular extension of (X_B, p_K) and \sqsubseteq_B is an ordering associated with (D_B, f_K) . \square

Proposition 4.4.4 *Suppose in addition that (Y_B, q_K) is a minimal extension of (X_B, p_K) ; let $\beta \in B$. Then:*

- (1) $\{u \in D_{\beta} : u \sqsubseteq_{\beta} y\} \subset Y_{\beta}$ for each $y \in Y_{\beta}$.
- (2) For each $u \in D_{\beta}$ the set $\{u' \in D_{\beta} : u' \sqsubseteq_{\beta} u\}$ is finite if and only if $u \in Y_{\beta}$.

Proof (1) Let $y \in Y_\beta, u \in D_\beta$ with $u \sqsubseteq_\beta y$. Then there exists $A \in \mathbf{d}(Y_\beta)$ with $u = \bigsqcup A$ and here Proposition 4.3.4 implies that $A \sqsubseteq_\beta \{y\}$, i.e., $y' \sqsubseteq_\beta y$ for all $y' \in A$. But then by Proposition 4.1.4 (2) A is finite and hence $u \in A$, i.e., $u \in Y$.

(2) If $y \in Y_\beta$ then by (1) and Proposition 4.1.4 (2) $\{u' \in D_\beta : u' \sqsubseteq_\beta y\}$ is finite. The converse follows from the fact that if $u \in D_\beta$ with $u = \bigsqcup A$ for some $A \in \mathbf{d}(Y_\beta)$ then $A \subset \{u' \in D_\beta : u' \sqsubseteq_\beta u\}$. But, as in (1), if A is finite then $u \in Y_\beta$. \square

4.5 Notes

The special case of an associated ordering for a fully regular extension can be found in Goguen, Thatcher, Wagner and Wright (1977) and in Courcelle and Nivat (1976).

Section 4.2 contains the elementary results which will be required from what is called *domain theory*. This was originated by Dana Scott in the late sixties, and is the main tool for dealing with denotational semantics. The reader interested in finding out more about domain theory should consult Scott and Gunter (1990); an account of its origins can be found in Scott (1976).

The results in Sections 4.3 concerning the initial completion of a poset are well-known and exist in a bewildering variety of forms; see, for example, Markowsky and Rosen (1976), Markowsky (1977), Wright, Wagner and Thatcher (1978) and Banaschewski and Nelson (1982). The idea of an ideal completion goes back to Birkhoff (1976) (which was first published in 1940).

Chapter 5 Functions

Chapters 3 and 4 dealt with the data objects. These were obtained by starting with a signature $\Lambda = (B, K, \Theta, \vartheta)$ and fixing an initial Λ -algebra (X_B, p_K) to describe the basic data objects. In a further step, a monotone regular extension (D_B, f_K) of (X_B, p_K) was then chosen to specify which ‘undefined’, ‘partially defined’ and, when appropriate, which ‘infinite’ data objects are to be allowed. In order to make things more specific the following three cases will be concentrated on throughout the study:

Case 1 (D_B, f_K) is any monotone regular extension of (X_B, p_K) . Here D_B will be considered just as a family of bottomed sets, and the mappings in the family f_K have in general no further structure.

Case 2 (D_B, f_K) is a minimal monotone regular extension of (X_B, p_K) . Then by Proposition 4.1.2 there exists a unique ordering \sqsubseteq_B associated with (D_B, f_K) which by Proposition 4.1.5 is monotone. Here D_B will always be considered to be a family of bottomed posets with respect to this family of partial orders, and hence f_K is a family of monotone mappings.

Case 3 (D_B, f_K) is the initial completion of a minimal monotone regular extension (Y_B, q_K) of (X_B, p_K) . There is then the ordering \sqsubseteq_B associated with (D_B, f_K) which arises from this completion process (i.e., from the initial completion of (Y_B, q_K) using its unique associated ordering). Here D_B will always be considered to be a family of bottomed complete posets with respect to this family of partial orders, and hence f_K is a family of continuous mappings. (This means that (D_B, f_K) is obtained using the ‘canonical’ procedure described at the beginning of Chapter 4.)

The next step is to define functions over the data objects. These will be obtained by first introducing a set S called the *set of functional types over B* . Somewhat informally, the set S can be thought of as being defined by the following rules:

- (i) If $L \in \mathcal{F}_S$ and $\beta \in B$ then $L \rightarrow \beta$ is an element of S .
- (ii) Each element of S can be uniquely constructed using rule (i).

The set B is considered as a subset of S by identifying each ground type $\beta \in B$ with the element $\emptyset \rightarrow \beta$ of S . The formal definition of S is given in Section 5.1.

The family D_B is then extended to a family D_S in such a way that for each $\sigma \in S \setminus B$ it is natural to regard D_σ as a set of functions of type σ . More precisely, D_S is chosen so that if $\sigma = L \rightarrow \beta$ is a functional type with $L \neq \emptyset$ then D_σ is a subset of $\langle D_\diamond^L \rightarrow D_\beta \rangle$ containing the bottom element \perp_σ defined by $\perp_\sigma(c) = \perp_\beta$ for all $c \in D_\diamond^L$, and where $\langle D_\diamond^L \rightarrow D_\beta \rangle$ is an alternative notation for the set of all mappings from D_\diamond^L to D_β . Such a family D_S is called a *functional extension* of D_B .

The most obvious example of a functional extension is the *full functional extension* of D_B . This is the unique family D_S such that $D_{L \rightarrow \beta} = \langle D_\diamond^L \rightarrow D_\beta \rangle$ whenever $L \neq \emptyset$, and it will always be employed in Case 1 described above. In Cases 2 and 3 this definition is modified by only considering monotone and continuous functions respectively. The

resulting families are then called the *full monotone functional extension* and the *full continuous functional extension* of D_B .

If D_S is a functional extension of D_B then for each type $\sigma = L \rightarrow \beta$ with $L \neq \emptyset$ there is the (total) *functional application operator* $\mathfrak{a}_\sigma^L : D_\sigma \times D_\circ^L \rightarrow D_\beta$ defined by $\mathfrak{a}_\sigma^L(h, c) = h(c)$ for all $h \in D_\sigma$, $c \in D_\circ^L$. Now if J is a non-empty proper subset of L and $c \in D_\circ^J$ then it is possible to apply just the ‘arguments’ c to the function h to obtain a ‘value’ $h(c)$ which is a function from $D_\circ^{L \setminus J}$ to β . But here it would be desirable if $h(c)$ were not just a function from $D_\circ^{L \setminus J}$ to D_β but actually an element of $D_{L \setminus J \rightarrow \beta}$. If this is the case then D_S is said to *support partial application*, and such an extension will always be used. In particular, each of the three full functional extensions mentioned above supports partial application. If D_S supports partial application then there is a corresponding *partial functional application operator* $\mathfrak{a}_\sigma^J : D_\sigma \times D_\circ^J \rightarrow D_{L \setminus J \rightarrow \beta}$ for each $\sigma = L \rightarrow \beta$ and each non-empty proper subset J of L .

In Section 5.2 the signature Λ is extended to a signature $\Sigma = (S, N, \Delta, \delta)$ by adding names for the functional application operators; the Λ -algebra (D_B, f_K) can then be extended to a Σ -algebra (D_S, f_N) by adding these operators to the family f_K . Σ is referred to as the *functional signature associated with Λ* and (D_S, f_N) the *functional Σ -algebra associated with (D_B, f_K) and D_S* .

In the Σ -algebra (D_S, f_N) certain relations hold automatically, since, roughly speaking, the functional application operators can often be written as the composition of two such operators. This leads to the notion of a *functional Σ -algebra*, defined to be a Σ -algebra for which the analogous relations hold. In Section 5.3 *functionally free Σ -algebras* are considered; these correspond to free algebras for the class of functional Σ -algebras. More precisely, if I is an S -typed set then a functional Σ -algebra (Y_S, q_N) is said to be *functionally I -free* if (Y_S, q_N) contains I (i.e., $\eta \in Y_\eta$ for each $\eta \in I$) and for each functional Σ -algebra (Z_S, r_N) and each $c \in Z_\circ^I$ there exists a unique Σ -homomorphism π_S^c from (Y_S, q_N) to (Z_S, r_N) such that $\pi_\eta^c(\eta) = c(\eta)$ for each $\eta \in I$.

The class of functionally free Σ -algebras includes the term algebras which will provide the basic building blocks for the rudimentary programming language to be defined in Chapter 6. Most of the properties of these algebra which are needed in Chapters 6 and 7 are worked out in Section 5.3.

In Section 5.4 the *primitive* or *built-in* functions are specified. These are the functions occurring in the equations which are really part of what is given (as opposed to the functions which the equations are supposed to define). There are two kinds of primitive functions which are needed. For practical reasons it is useful to have a few *integer operators*. More importantly, a suitable family of *case operators* is also required. These will facilitate what corresponds to pattern matching or to the ‘case’ construction in *Haskell*. These operators (or something equivalent) are absolutely essential: They are the analogue of the `if then else` construction found in all imperative programming languages and without them it is impossible to define any ‘non-trivial’ functions.

5.1 Functional types and functional extensions

This chapter deals with the functions to be defined over the data objects. These will be obtained by first introducing a set S called the set of functional types over B and then extending the family D_B to a family D_S in such a way that it is natural to regard D_σ as a set of functions of type σ for each $\sigma \in S \setminus B$.

For any set S denote by \mathcal{F}_S^o the set of non-empty finite S -typed sets, in other words $\mathcal{F}_S^o = \mathcal{F}_S \setminus \{\emptyset\}$. Somewhat informally, the set S of functional types over B can be thought of as being defined by the following rules:

- (i) Each ground type $\beta \in B$ is an element of S .
- (ii) If $L \in \mathcal{F}_S^o$ and $\beta \in B$ then $L \rightarrow \beta$ is an element of S .
- (iii) Each element of S can be uniquely constructed using rules (i) and (ii).

The functions of type $L \rightarrow \beta$ will be defined to be some kind of functions whose arguments are named by the elements of the set L , in which the values of the argument corresponding to $\eta \in L$ range over the set of elements of type $\langle \eta \rangle$, and which take their values in the set D_β . In particular, if $\sigma_1 \cdots \sigma_n \in S^*$ with $n \geq 1$ then the functions of type $\sigma_1 \cdots \sigma_n \rightarrow \beta$ will be functions of n arguments with the values of the j -th argument ranging over the set of elements of type σ_j .

Now it is convenient to identify the ground type $\beta \in B$ with a functional type having no arguments, i.e., with the type $\emptyset \rightarrow \beta$, and this reduces the informal definition to just the following two rules:

- (i) If $L \in \mathcal{F}_S$ and $\beta \in B$ then $L \rightarrow \beta$ is an element of S .
- (ii) Each element of S can be uniquely constructed using rule (i).

This must be made precise. Thus suppose S is a set which is going to be a candidate for the set of functional types over B . Then a mapping $\# : \mathcal{F}_S \times B \rightarrow S$ must also be given which ‘constructs’ the new types, i.e., such that $\#(L, \beta)$ is the new type $L \rightarrow \beta$ occurring in (i) for each $L \in \mathcal{F}_S$, $\beta \in B$.

Now let S be any set and let $\# : \mathcal{F}_S \times B \rightarrow S$ be a mapping; then, formalising conditions (i) and (ii) above, the pair $(S, \#)$ is said to be a *set of functional types over B* if the following hold:

- (i) The mapping $\# : \mathcal{F}_S \times B \rightarrow S$ is bijective.
- (ii) If $T \subset S$ with $\#(\mathcal{F}_T \times B) \subset T$ then $T = S$.

Note that by (ii) ‘bijective’ could be replaced in (i) by ‘injective’, since if $T = \text{Im}(\#)$ then clearly $\#(\mathcal{F}_T \times B) \subset \#(\mathcal{F}_S \times B) = T$.

Proposition 5.1.1 *There exists a set of functional types over B . Moreover, this set is unique in the sense that if $(S, \#)$ and $(S', \#')$ are sets of functional types over B then there exists a unique bijective mapping $\pi : S \rightarrow S'$ with*

$$\pi(\#(L, \beta)) = \#'(\pi L, \beta)$$

for all $L \in \mathcal{F}_S$, $\beta \in B$, where πL is the S' -typed set having the same underlying set as L together with the typing $\langle \cdot \rangle' : L \rightarrow S'$ defined by $\langle \eta \rangle' = \pi(\langle \eta \rangle)$ for each $\eta \in L$ (with $\langle \cdot \rangle$ the typing on L).

Proof Consider the single-sorted signature $\Xi = (\mathcal{F}_{\mathbb{I}} \times B, \Phi)$ in which the mapping $\Phi : \mathcal{F}_{\mathbb{I}} \times B \rightarrow \mathcal{F}_{\mathbb{I}}$ is just the projection onto the first factor (i.e., $\Phi(A, \beta) = A$ for all $A \in \mathcal{F}_{\mathbb{I}}$, $\beta \in B$). Thus Ξ contains a single type ε and for each $A \in \mathcal{F}_{\mathbb{I}}$ and each $\beta \in B$ an operator name (A, β) of type $A \rightarrow \varepsilon$. Proposition 2.3.3 implies there exists a unique isomorphism class of initial Ξ -algebras, so fix an initial Ξ -algebra $(S, h_{\mathcal{F}_{\mathbb{I}} \times B})$. Note that $h_{(A, \beta)}$ is then a mapping from S^A to S for each $A \in \mathcal{F}_{\mathbb{I}}$, $\beta \in B$, and that the elements of the set S^A can be considered as typings on the set A . A mapping $\# : \mathcal{F}_S \times B \rightarrow S$ can thus be defined by letting

$$\#(L, \beta) = h_{(A, \beta)}(\langle \cdot \rangle)$$

for each $L = (A, \langle \cdot \rangle) \in \mathcal{F}_S$, $\beta \in B$. Then by Proposition 2.3.2 $(S, \#)$ is a set of functional types over B : Condition (i) holds because the Ξ -algebra $(S, h_{\mathcal{F}_{\mathbb{I}} \times B})$ is unambiguous and (ii) holds because it is minimal.

Now suppose $(S, \#)$ and $(S', \#')$ are sets of functional types over B . The construction of a set of functional types over B out of an initial Ξ -algebra can be reversed: For each $A \in \mathcal{F}_{\mathbb{I}}$ and each $\beta \in B$ define $h_{(A, \beta)} : S^A \rightarrow S$ by $h_{(A, \beta)}(v) = \#(L, \beta)$, where L is the S -typed set with underlying set A and typing v . Then $(S, h_{\mathcal{F}_{\mathbb{I}} \times B})$ is a Ξ -algebra which is unambiguous, since (i) holds, and minimal, since (ii) holds. Thus by Proposition 2.3.2 $(S, h_{\mathcal{F}_{\mathbb{I}} \times B})$ is initial. The same procedure, but starting with the set of functional types $(S', \#')$, produces an initial Ξ -algebra $(S', h'_{\mathcal{F}_{\mathbb{I}} \times B})$, and there then exists a unique Ξ -isomorphism π from $(S, h_{\mathcal{F}_{\mathbb{I}} \times B})$ to $(S', h'_{\mathcal{F}_{\mathbb{I}} \times B})$. But such an isomorphism is just a bijective mapping $\pi : S \rightarrow S'$ with

$$\pi(\#(L, \beta)) = \#'(\pi L, \beta)$$

for all $L \in \mathcal{F}_S$, $\beta \in B$. Finally, suppose that $\pi' : S \rightarrow S'$ is another mapping with this property, and let $T = \{ \sigma \in S : \pi(\sigma) = \pi'(\sigma) \}$. Then

$$\pi(\#(L, \beta)) = \#'(\pi L, \beta) = \#'(\pi' L, \beta) = \pi'(\#(L, \beta))$$

for all $L \in \mathcal{F}_S$, $\beta \in B$, i.e., $\#(\mathcal{F}_T \times B) \subset T$. Hence by (ii) $T = S$, which implies that $\pi' = \pi$, and this shows π is unique. \square

Because of the uniqueness as formulated in Proposition 5.1.1 it makes sense to speak of *the* set of functional types over B , and this will be denoted by $(S, \#)$ or, more usually, just by S . Moreover, the notation $L \rightarrow \beta$ will almost always be used instead of $\#(L, \beta)$. The symbol \rightarrow thus has two different usages; it will be seen, however, that these two usages are, in a certain sense, compatible.

By (i) each element of S has a unique representation of the form $L \rightarrow \beta$ with $L \in \mathcal{F}_S$ and $\beta \in B$, which implies, in particular, that $\emptyset \rightarrow \beta$ and $\emptyset \rightarrow \beta'$ are different elements of S whenever $\beta \neq \beta'$. This means that $\emptyset \rightarrow \beta$ can (and will) be identified with β for each $\beta \in B$; in this way B is considered to be a subset of S .

Finally, although S is referred to as the set of functional types, the statement that $\sigma \in S$ is a *functional type* will mean that $\sigma \in S \setminus B$, i.e., that σ has the form $L \rightarrow \beta$ with $L \in \mathcal{F}_S^o$. The set S is therefore divided up into functional types (the elements of $S \setminus B$) and ground types (the elements of B).

Example 5.1.1 In the signature Λ in Example 2.2.1 the set of ground types is the set $B = \{\text{bool}, \text{nat}, \text{int}, \text{ipr}, \text{ilst}\}$. The set S thus contains the functional types $\text{int} \rightarrow \text{int}$, $\text{ipr} \rightarrow \text{ipr}$, $\text{ipr} \rightarrow \text{int}$, $\text{int int} \rightarrow \text{int}$, $\text{int int} \rightarrow \text{ipr}$ and $\text{int ipr} \rightarrow \text{ipr}$ which already occurred in the example in Chapter 1.

Lemma 5.1.1 *There is a unique mapping $|\cdot| : S \rightarrow \mathbb{N}$ with $|\beta| = 0$ for each ground type $\beta \in B$ and such that*

$$|L \rightarrow \beta| = 1 + \sum_{\eta \in L} |\langle \eta \rangle|$$

for each functional type $L \rightarrow \beta$.

Proof As in the proof of Proposition 5.1.1 let $(S, h_{\mathcal{F}_1 \times B})$ be the initial Ξ -algebra with $h_{(A, \beta)}(v) = \#(L, \beta)$ for each $v : A \rightarrow S$, where $L = (A, v)$. Define a further Ξ -algebra $(\mathbb{N}, g_{\mathcal{F}_1 \times B})$ by letting $g_{(\emptyset, \beta)}(\varepsilon) = 0$ for each $\beta \in B$ and

$$g_{(A, \beta)}(v) = \sum_{x \in A} v(x)$$

for each $v : A \rightarrow \mathbb{N}$ whenever $A \neq \emptyset$. There thus exists a unique Ξ -homomorphism from $(S, h_{\mathcal{F}_1 \times B})$ to $(\mathbb{N}, g_{\mathcal{F}_1 \times B})$, in this case consisting of a single mapping $|\cdot| : S \rightarrow \mathbb{N}$, and by the definition of a homomorphism it follows that $|\beta| = 0$ for each ground type $\beta \in B$ and $|L \rightarrow \beta| = \sum_{\eta \in L} |\langle \eta \rangle|$ for each functional type $L \rightarrow \beta$. The uniqueness follows from the uniqueness of this homomorphism, or directly from condition (ii) in the definition of a set of functional types. \square

The elements σ of S with $|\sigma| = 1$ are called *first order functional types*; these are exactly the types having the form $L \rightarrow \beta$ with $L \in \mathcal{F}_S^o$ and $\langle \eta \rangle \in B$ for each $\eta \in L$ (i.e., each of the ‘parameters’ of a first order functional type is of ground type). The elements σ of S with $|\sigma| > 1$ are then called *higher order functional types*: These are therefore the types having the form $L \rightarrow \beta$ with $L \in \mathcal{F}_S^o$ and $\langle \eta \rangle \in S \setminus B$ for at least one $\eta \in L$ (i.e., a higher order functional type has at least one ‘parameter’ which is of functional type).

Having now constructed the set S of functional types, the next step is to extend the family D_B to a family D_S in such a way that it is natural to regard D_σ as a set of functions of type σ for each $\sigma \in S \setminus B$.

In what follows let Y_B be a family of bottomed sets, with as usual the bottom element of Y_β being denoted by \perp_β for each $\beta \in B$. The constructions to be given in terms of Y_B will then be applied in the following section to the family D_B occurring in the extension (D_B, f_K) . If X and Y are sets then it is useful to allow $\langle X \rightarrow Y \rangle$ (as well as Y^X) to denote the set of all mappings from X to Y .

Let Y_S be a family of bottomed sets extending the family Y_B , i.e., for each functional type $\sigma \in S \setminus B$ a bottomed set Y_σ (with bottom element \perp_σ) has been added to the family Y_B . The family Y_S is said to be a *functional extension* of Y_B if for each functional type $\sigma = L \rightarrow \beta$ (and so $L \in \mathcal{F}_S^0$) the set Y_σ is a subset of $\langle Y_\circ^L \rightarrow Y_\beta \rangle$ with \perp_σ the bottom element of $\langle Y_\circ^L \rightarrow Y_\beta \rangle$, i.e., with $\perp_\sigma : Y_\circ^L \rightarrow Y_\beta$ the function defined by $\perp_\sigma(c) = \perp_\beta$ for all $c \in Y_\circ^L$.

If Y_S is a functional extension of Y_B and $\sigma = \sigma_1 \cdots \sigma_n \rightarrow \beta$ with $n \geq 1$ then each element of Y_σ is just a function from $Y_{\sigma_1} \times \cdots \times Y_{\sigma_n}$ to Y_β .

The most obvious example of a functional extension is provided by the following result:

Proposition 5.1.2 *There exists a unique functional extension Y_S of Y_B such that $Y_{L \rightarrow \beta} = \langle Y_\circ^L \rightarrow Y_\beta \rangle$ for each functional type $L \rightarrow \beta$.*

Proof Let $|\cdot| : S \rightarrow \mathbb{N}$ be the mapping given in Lemma 5.1.1; then Y_σ can be defined using induction on $n = |\sigma|$: If $|\sigma| = 0$ then $\sigma \in B$ and so Y_σ is the corresponding member of the family Y_B . Thus suppose $n \geq 1$ and that Y_τ has already been defined whenever $|\tau| < n$; let $\sigma = L \rightarrow \beta$ with $|\sigma| = n$. Put $S' = \{\tau \in S : |\tau| < n\}$; then the family $Y_{S'}$ is already defined and L is an S' -typed set (since $|\langle \eta \rangle| < n$ for each $\eta \in L$). Therefore $Y_{L \rightarrow \beta}$ can be defined to be $\langle Z \rightarrow Y_\beta \rangle$, where Z is the set of all typed mappings from L to $|Y_{S'}|$. Then, regardless of how Y_τ is defined for $\tau \in S \setminus S'$, the set Z can be identified with the set of assignments Y_\circ^L . The family Y_S thus has the required property by construction, and the uniqueness also follows directly using induction on $n = |\sigma|$. \square

The family Y_S given in Proposition 5.1.2 will be referred to as the *full functional extension* of Y_B .

There are two more special classes of functional extensions which must be considered. Recall that if Y_1 and Y_2 are posets then $(Y_1 \rightarrow Y_2)$ is used to denote the set of all monotone mappings from Y_1 to Y_2 and that $(Y_1 \rightarrow Y_2)$ is regarded as a poset with the partial order \sqsubseteq , where $g \sqsubseteq h$ if and only if $g(y) \sqsubseteq_2 h(y)$ for all $y \in Y_1$ (with \sqsubseteq_2 the partial order on Y_2). Recall also that a poset Y' is said to be a subposet of Y if $Y' \subset Y$ and the partial order on Y' is obtained by restricting the partial order on Y . Moreover, if Y is a bottomed poset then a subposet Y' of Y is said to be proper bottomed if Y' contains the bottom element \perp of Y .

Suppose now that Y_B is a family of bottomed posets, and let Y_S be a family of bottomed posets extending the family Y_B , i.e., a bottomed poset Y_σ has been added

Example 5.1.2 Let (D_B, f_K) be the flat extension of the Λ -algebra (X_B, p_K) introduced in Example 2.2.1 and D_S be the full functional extension of D_B . Then in particular

$$\begin{aligned} D_{\text{ipr} \rightarrow \text{ipr}} &= \langle \mathbb{P}^\perp \rightarrow \mathbb{P}^\perp \rangle, \quad D_{\text{int int} \rightarrow \text{ipr}} = \langle \mathbb{Z}^\perp \times \mathbb{Z}^\perp \rightarrow \mathbb{P}^\perp \rangle, \\ D_{\text{ipr} \rightarrow \text{int}} &= \langle \mathbb{P}^\perp \rightarrow \mathbb{Z}^\perp \rangle, \quad D_{\text{int int} \rightarrow \text{int}} = \langle \mathbb{Z}^\perp \times \mathbb{Z}^\perp \rightarrow \mathbb{Z}^\perp \rangle, \\ D_{\text{int ipr} \rightarrow \text{ipr}} &= \langle \mathbb{Z}^\perp \times \mathbb{P}^\perp \rightarrow \mathbb{P}^\perp \rangle \quad \text{and} \quad D_{\text{int} \rightarrow \text{int}} = \langle \mathbb{Z}^\perp \rightarrow \mathbb{Z}^\perp \rangle, \end{aligned}$$

with $\mathbb{P} = \mathbb{Z}^2$. This is what is needed to deal with the third interpretation of the equations considered in Chapter 1.

to the family Y_B for each functional type $\sigma \in S \setminus B$. The family Y_S will be called a *monotone functional extension* of Y_B if Y_σ is a proper bottomed poset of $(Y_\sigma^L \rightarrow Y_\beta)$ for each functional type $\sigma = L \rightarrow \beta$. This means that the bottom element of \perp_σ of Y_σ is the function defined by $\perp_\sigma(c) = \perp_\beta$ for all $c \in D_\sigma^L$.

A monotone functional extension Y_S of Y_B is in particular a functional extension of Y_B . The canonical example of a monotone functional extension is provided by the following result:

Proposition 5.1.3 *There exists a unique monotone functional extension Y_S of Y_B such that $Y_{L \rightarrow \beta} = (Y_\sigma^L \rightarrow Y_\beta)$ for each functional type $L \rightarrow \beta$.*

Proof This is the same as the proof of Proposition 5.1.2. \square

The family Y_S given in Proposition 5.1.3 will be called the *full monotone functional extension* of Y_B .

The second special class of functional extensions involves continuous functions. Recall that if Y_1 and Y_2 are complete posets then $[Y_1 \rightarrow Y_2]$ denotes the set of all continuous mappings from Y_1 to Y_2 . Thus $[Y_1 \rightarrow Y_2] \subset (Y_1 \rightarrow Y_2)$ and $[Y_1 \rightarrow Y_2]$ is considered as a poset with the partial order induced by the partial order on $(Y_1 \rightarrow Y_2)$. By Proposition 4.2.5 the poset $[Y_1 \rightarrow Y_2]$ is then complete.

Suppose now that Y_B is a family of bottomed complete posets, and let Y_S be a family of bottomed complete posets extending the family Y_B , i.e., a bottomed complete poset Y_σ has been added to the family Y_B for each functional type $\sigma \in S \setminus B$. The family Y_S will be called a *continuous functional extension* of Y_B if Y_σ is a proper bottomed poset of $[Y_\sigma^L \rightarrow Y_\beta]$ for each functional type $\sigma = L \rightarrow \beta$. This again means that the bottom element of \perp_σ of Y_σ is the function defined by $\perp_\sigma(c) = \perp_\beta$ for all $c \in D_\sigma^L$.

A continuous functional extension Y_S of Y_B is in particular a monotone functional extension and thus also a functional extension of Y_B . The canonical example of a continuous functional extension is provided by the following result:

Proposition 5.1.4 *There exists a unique continuous functional extension Y_S of Y_B such that $Y_{L \rightarrow \beta} = [Y_{\diamond}^L \rightarrow Y_{\beta}]$ for each functional type $L \rightarrow \beta$.*

Proof This is the same as the proof of Proposition 5.1.2. \square

The family Y_S given in Proposition 5.1.4 will be called the *full continuous functional extension* of Y_B .

Now let Y_B be any family of bottomed sets and let Y_S be a functional extension of Y_B . The fundamental operation associated with functions is that of functional application: If $\sigma = L \rightarrow \beta$ is a functional type then there is the (*total*) *functional application operator* $\mathbf{a}_{\sigma}^L : Y_{\sigma} \times Y_{\diamond}^L \rightarrow Y_{\beta}$ defined for all $h \in Y_{\sigma}$, $c \in Y_{\diamond}^L$ by

$$\mathbf{a}_{\sigma}^L(h, c) = h(c).$$

But if J is a non-empty proper subset of L and $c \in Y_{\diamond}^J$ then it is possible to apply just the ‘arguments’ c to the function h to obtain a ‘value’ $h(c)$ which is a function from $Y_{\diamond}^{L \setminus J}$ to Y_{β} . But here it would be desirable if $h(c)$ were not just a function from $Y_{\diamond}^{L \setminus J}$ to Y_{β} but actually an element of $Y_{L \setminus J \rightarrow \beta}$. However, for a general functional extension of Y_B this will not always be the case, and so an extra condition must now be imposed.

If L and L' are disjoint elements of \mathcal{F}_S then $L \cup L'$ is considered as an S -typed set with the typing induced by the typings on L and L' . If $c \in Y_{\diamond}^L$, $c' \in Y_{\diamond}^{L'}$ then $c \oplus c'$ will denote the element of $Y_{\diamond}^{L \cup L'}$ defined by

$$(c \oplus c')(\eta) = \begin{cases} c(\eta) & \text{if } \eta \in L, \\ c'(\eta) & \text{if } \eta \in L'. \end{cases}$$

Lemma 5.1.2 *Each element of $Y_{\diamond}^{L \cup L'}$ has a unique representation of the form $c \oplus c'$ with $c \in Y_{\diamond}^L$ and $c' \in Y_{\diamond}^{L'}$.*

Proof This is clear. \square

Let L and L' be disjoint elements of \mathcal{F}_S° and $\beta \in B$. Then for each $h \in Y_{L \cup L' \rightarrow \beta}$ and each $c \in Y_{\diamond}^L$ there is a mapping $h(c) : Y_{\diamond}^{L'} \rightarrow Y_{\beta}$ defined by letting

$$h(c)(c') = h(c \oplus c')$$

for each $c' \in Y_{\diamond}^{L'}$, and by definition $Y_{L' \rightarrow \beta}$ is a subset of $(Y_{\diamond}^{L'} \rightarrow Y_{\beta})$. The family Y_S is said to *support partial application* if it is always the case that $h(c)$ is actually an element of $Y_{L' \rightarrow \beta}$. Thus Y_S supporting partial application means that whenever L and L' are disjoint elements of \mathcal{F}_S° and $\beta \in B$ then the mapping

$$c' \mapsto h(c \oplus c')$$

must define an element of $Y_{L' \rightarrow \beta}$ for each $h \in Y_{L \cup L' \rightarrow \beta}$ and each $c \in Y_{\diamond}^L$.

It is clear that the full functional extension supports partial application. Moreover, the other two ‘full’ classes of functional extensions also have this property.

Proposition 5.1.5 *Let Y_B be a family of bottomed posets. Then the full monotone functional extension Y_S of Y_B supports partial application.*

Proof Let L and L' be disjoint elements of \mathcal{F}_\diamond^0 and $\beta \in B$; then it must be shown that the mapping $b \mapsto h(c \oplus b)$ from $Y_\diamond^{L'}$ to Y_β is monotone for each $h \in Y_{L \cup L' \rightarrow \beta}$ and each $c \in Y_\diamond^L$. But this is clearly the case, since if $b, b' \in Y_\diamond^{L'}$ with $b \sqsubseteq_\diamond^{L'} b'$ then $b \oplus c \sqsubseteq_\diamond^{L \cup L'} b' \oplus c$ and hence $h(b \oplus c) \sqsubseteq_\beta h(b' \oplus c)$. \square

Proposition 5.1.6 *Let Y_B be a family of bottomed complete posets. Then the full continuous functional extension Y_S of Y_B supports partial application.*

Proof Let L and L' be disjoint elements of \mathcal{F}_\diamond^0 and $\beta \in B$; here it must be shown that the mapping $b \mapsto h(c \oplus b)$ from $Y_\diamond^{L'}$ to Y_β is continuous for each $h \in Y_{L \cup L' \rightarrow \beta}$ and each $c \in Y_\diamond^L$. Fix $h \in Y_{L \cup L' \rightarrow \beta}$ and $c \in Y_\diamond^L$ and denote this mapping by g . Now it is easily checked that the mapping $(b, b') \mapsto b \oplus b'$ defines an order isomorphism from $Y_\diamond^L \times Y_\diamond^{L'}$ to $Y_\diamond^{L \cup L'}$ and hence by Proposition 4.2.1 its composition with h defines a continuous mapping h' from $Y_\diamond^L \times Y_\diamond^{L'}$ to Y_β . But then $g = \mathbf{pa}(h', c)$, with \mathbf{pa} the appropriate partial application operator as defined in Section 4.2, and so in particular Proposition 4.2.8 implies that g is continuous. \square

Again let Y_B be any family of bottomed sets and suppose now that Y_S is a functional extension of Y_B which supports partial application. Let $\sigma = L \rightarrow \beta$ be a functional type and J be a non-empty proper subset of L . Then there is the *partial functional application operator* $\mathbf{a}_\sigma^J : Y_\sigma \times Y_\diamond^J \rightarrow Y_{L \setminus J \rightarrow \beta}$ defined by letting

$$\mathbf{a}_\sigma^J(h, c) = h(c)$$

for all $h \in Y_\sigma$, $c \in Y_\diamond^J$. In other words this means that $\mathbf{a}_\sigma^J(h, c)(c') = h(c \oplus c')$ for all $h \in Y_\sigma$, $c \in Y_\diamond^J$, $c' \in Y_\diamond^{L \setminus J}$.

The following relationships satisfied by the (total and partial) functional application operators will later play an important role:

Lemma 5.1.3 *Let $L \rightarrow \beta$ be a functional type and J and J' be non-empty disjoint subsets of L (and so in fact L must contain at least two elements). Then*

$$\mathbf{a}_{L \rightarrow \beta}^{J \cup J'}(h, c \oplus c') = \mathbf{a}_{L \setminus J \rightarrow \beta}^{J'}(\mathbf{a}_{L \rightarrow \beta}^J(h, c), c')$$

for all $h \in Y_{L \rightarrow \beta}$, $c \in Y_\diamond^J$ and all $c' \in Y_\diamond^{J'}$.

Proof Suppose first that $J \cup J'$ is a proper subset of L . Then

$$\begin{aligned} \mathbf{a}_{L \rightarrow \beta}^{J \cup J'}(h, c \oplus c')(b) &= h((c \oplus c') \oplus b) = h(c \oplus (c' \oplus b)) \\ &= \mathbf{a}_{L \rightarrow \beta}^J(h, c)(c' \oplus b) = \mathbf{a}_{L \setminus J \rightarrow \beta}^{J'}(\mathbf{a}_{L \rightarrow \beta}^J(h, c), c')(b) \end{aligned}$$

for each $b \in Y_\diamond^{L \setminus (J \cup J')}$. On the other hand, if $J \cup J' = L$ then

$$\begin{aligned} \mathbf{a}_{L \rightarrow \beta}^{J \cup J'}(h, c \oplus c') &= h(c \oplus c') \\ &= \mathbf{a}_{L \rightarrow \beta}^J(h, c)(c') = \mathbf{a}_{L \setminus J \rightarrow \beta}^{J'}(\mathbf{a}_{L \rightarrow \beta}^J(h, c), c'). \quad \square \end{aligned}$$

5.2 Functional algebras

The constructions made in the previous section will now be applied to the case when $Y_B = D_B$. Thus in addition to the monotone regular bottomed extension (D_B, f_K) of (X_B, p_K) it is also necessary to choose a functional extension D_S of D_B which supports partial application. For the three basic Cases (which were introduced at the beginning of the chapter) the functional extensions D_S will be chosen as follows:

Case 1 (with (D_B, f_K) any monotone regular extension of (X_B, p_K)) Here D_S will always be taken to be the full functional extension of D_B .

Case 2 (with (D_B, f_K) a minimal monotone regular extension of (X_B, p_K)) Here D_S will always be taken to be the full monotone functional extension of D_B (and so by Proposition 5.1.5 D_S supports partial application).

Case 3 (with (D_B, f_K) the initial completion of a minimal monotone regular extension of (X_B, p_K)) Here D_S will always be taken to be the full continuous functional extension of D_B (and so by Proposition 5.1.6 D_S supports partial application).

Having chosen a functional extension D_S of D_B which supports partial application, the signature Λ will be extended to a signature $\Sigma = (S, N, \Delta, \delta)$ by adding names for the functional application operators, and then the Λ -algebra (D_B, f_K) will be extended to a Σ -algebra (D_S, f_N) by adding these operators to the family f_K .

Before doing this, though, there is one small point which has to be dealt with. The application operators as defined above have two arguments, the first being a function and the remainder the function's arguments. However, to conform with the set-up being employed for the mappings occurring in algebras, they should only have a single argument, and to achieve this it is convenient to introduce the following simple device: For each functional type $\sigma = L \rightarrow \beta \in S$ fix some element \diamond_σ not in L , and for each non-empty subset J of L put $\sigma \cdot J = J \cup \{\diamond_\sigma\}$, considered as an S -typed set with the typing induced from L and with \diamond_σ of type σ . If Y_S is a family of sets and $y \in Y_\sigma$, $c \in Y_\sigma^J$ then $y \triangleleft c$ will denote the element of $Y_\sigma^{\sigma \cdot J}$ given by

$$(y \triangleleft c)(\eta) = \begin{cases} c(\eta) & \text{if } \eta \in J, \\ y & \text{if } \eta = \diamond_\sigma. \end{cases}$$

Lemma 5.2.1 *Each element of $Y_\sigma^{\sigma \cdot J}$ has a unique representation of the form $y \triangleleft c$ with $y \in Y_\sigma$ and $c \in Y_\sigma^J$.*

Proof This is clear. \square

Lemma 5.2.1 implies that the mapping $(y, c) \mapsto y \triangleleft c$ defines a bijection between $Y_\sigma \times Y_\sigma^J$ and $Y_\sigma^{\sigma \cdot J}$. (Moreover, if Y_S is a family of posets then it is easy to see that the mapping is actually an order isomorphism.) This allows the application operator originally defined as a mapping with domain $Y_\sigma \times Y_\sigma^J$ to be converted into a mapping with domain $Y_\sigma^{\sigma \cdot J}$.

For each functional type $\sigma = L \rightarrow \beta$ and each non-empty subset J of L let $\triangleleft\{\sigma, J\}$ be some element not in K and such that $\triangleleft\{\sigma, J\} \neq \triangleleft\{\tau, J'\}$ whenever $(\sigma, J) \neq (\tau, J')$. Put $N = K \cup A$, where A is the set consisting of these new elements and define mappings $\Delta : N \rightarrow \mathcal{F}_S$ and $\delta : N \rightarrow S$ by

$$\Delta(\nu) = \begin{cases} \Theta(\nu) & \text{if } \nu \in K, \\ \sigma \cdot J & \text{if } \nu = \triangleleft\{\sigma, J\}, \end{cases}$$

$$\delta(\nu) = \begin{cases} \vartheta(\nu) & \text{if } \nu \in K, \\ L \setminus J \rightarrow \beta & \text{if } \nu = \triangleleft\{\sigma, J\} \text{ with } \sigma = L \rightarrow \beta. \end{cases}$$

Then $\Sigma = (S, N, \Delta, \delta)$ is a signature which is an extension of Λ , and in which $\triangleleft\{\sigma, J\}$ is of type $\sigma \cdot J \rightarrow (L \setminus J \rightarrow \beta)$ whenever $\sigma = L \rightarrow \beta \in S \setminus B$ and J is a non-empty subset of L . Σ will be called the *functional signature associated with Λ* .

Now if $\sigma = L \rightarrow \beta$ is a functional type and J is a non-empty subset of L then there is a mapping $f_{\triangleleft\{\sigma, J\}} : D_{\diamond}^{\sigma \cdot J} \rightarrow D_{L \setminus J \rightarrow \beta}$ defined by letting

$$f_{\triangleleft\{\sigma, J\}}(h \triangleleft c) = \mathbf{a}_{\sigma}^J(h, c)$$

for all $h \in D_{\sigma}$, $c \in D_{\diamond}^J$. Thus if J a non-empty proper subset of L then

$$f_{\triangleleft\{\sigma, J\}}(h \triangleleft c)(c') = h(c \oplus c')$$

for all $h \in D_{\sigma}$, $c \in D_{\diamond}^J$, $c' \in D_{\diamond}^{L \setminus J}$, whereas $f_{\triangleleft\{\sigma, L\}} : D_{\diamond}^{\sigma \cdot L} \rightarrow D_{\beta}$ is given by

$$f_{\triangleleft\{\sigma, L\}}(h \triangleleft c) = h(c)$$

for each $h \in D_{\sigma}$, $c \in D_{\diamond}^L$. This results in a Σ -algebra (D_S, f_N) which will be called the *functional Σ -algebra associated with (D_B, f_K) and D_S* ; it is clearly an extension of the Λ -algebra (D_B, f_K) .

Proposition 5.2.1 *In Case 2 the mapping f_{ν} is monotone for each $\nu \in N$.*

Proof If $\kappa \in K$ then f_{κ} is monotone by definition. Thus consider a functional type $\sigma = L \rightarrow \beta$ and a non-empty subset J of L . Suppose first that $J \neq L$. Let $h, h' \in D_{\sigma}$ and $c, c' \in D_{\diamond}^J$ with $h \sqsubseteq_{\sigma} h'$ and $c \sqsubseteq_{\diamond}^J c'$. Then for each $b \in D_{\diamond}^{L \setminus J}$

$$f_{\triangleleft\{\sigma, J\}}(h \triangleleft c)(b) = h(c \oplus b) \sqsubseteq_{\beta} h(c' \oplus b) \sqsubseteq_{\beta} h'(c' \oplus b) = f_{\triangleleft\{\sigma, J\}}(h' \triangleleft c')(b)$$

since h is monotone and $c \oplus b \sqsubseteq_{\diamond}^L c' \oplus b$, and this implies that $f_{\triangleleft\{\sigma, J\}}$ is monotone. A similar, but simpler, argument shows that $f_{\triangleleft\{\sigma, L\}}$ is monotone. \square

Proposition 5.2.2 *In Case 3 the mapping f_{ν} is continuous for each $\nu \in N$.*

Proof Let $\sigma = L \rightarrow \beta$ and J be a non-empty subset of L , and again suppose first that $J \neq L$. Then the mapping $f_{\triangleleft\{\sigma, J\}} : D_{\diamond}^{\sigma \cdot J} \rightarrow D_{L \setminus J \rightarrow \beta}$ can be written as the composition of mappings ϱ , ϱ' and pa , where $\varrho : D_{\diamond}^{\sigma \cdot J} \rightarrow D_{\sigma} \times D_{\diamond}^J = [D_{\diamond}^L \rightarrow D_{\beta}] \times D_{\diamond}^J$ and $\varrho' : [D_{\diamond}^L \rightarrow D_{\beta}] \times D_{\diamond}^J \rightarrow [D_{\diamond}^J \times D_{\diamond}^{L \setminus J} \rightarrow D_{\beta}] \times D_{\diamond}^J$ are (easily determined) order isomorphisms and $\text{pa} : [D_{\diamond}^J \times D_{\diamond}^{L \setminus J} \rightarrow D_{\beta}] \times D_{\diamond}^J \rightarrow [D_{\diamond}^{L \setminus J} \rightarrow D_{\beta}] = D_{L \setminus J \rightarrow \beta}$ is a

partial application operator as defined in Section 4.2. Thus by Propositions 4.2.1 and 4.2.8 $f_{\triangleleft\{\sigma,J\}}$ is continuous. A similar argument, but using Proposition 4.2.6 instead of Proposition 4.2.8, shows that $f_{\triangleleft\{\sigma,L\}}$ is continuous. Finally, if $\kappa \in K$ then f_κ is continuous by definition. \square

The functional Σ -algebra (D_S, f_N) possesses a couple of properties which will play an important role in what follows. These are given in the next two results.

Proposition 5.2.3 *Let $\sigma = L \rightarrow \beta$ be a functional type and J be a non-empty subset of L . Then $f_{\triangleleft\{\sigma,J\}}(\perp_\sigma \triangleleft c) = \perp_{L \setminus J \rightarrow \beta}$ for all $c \in D_\diamond^J$.*

Proof Let $c \in D_\diamond^J$; then for all $c' \in D_\diamond^{L \setminus J}$

$$f_{\triangleleft\{\sigma,J\}}(\perp_\sigma \triangleleft c)(c') = \perp_\sigma(c \oplus c') = \perp_\beta,$$

and hence $f_{\triangleleft\{\sigma,J\}}(\perp_\sigma \triangleleft c) = \perp_{L \setminus J \rightarrow \beta}$ by the definition of $\perp_{L \setminus J \rightarrow \beta}$. \square

Proposition 5.2.4 *Let $L \rightarrow \beta$ be a functional type and J and J' be non-empty disjoint subsets of L (and so in fact L must contain at least two elements). Then*

$$f_{\triangleleft\{L \rightarrow \beta, J \cup J'\}}(h \triangleleft (c \oplus c')) = f_{\triangleleft\{L \setminus J \rightarrow \beta, J'\}}(f_{\triangleleft\{L \rightarrow \beta, J\}}(h \triangleleft c) \triangleleft c')$$

for all $h \in D_{L \rightarrow \beta}$, $c \in D_\diamond^J$ and all $c' \in D_\diamond^{J'}$.

Proof This follows immediately from Lemma 5.1.3. \square

The relations occurring in Proposition 5.2.4 will be used as the basis for a definition: A Σ -algebra (Y_S, q_N) will be called *functional* if whenever $L \rightarrow \beta$ is a functional type, J and J' are non-empty disjoint subsets of L then

$$q_{\triangleleft\{L \rightarrow \beta, J \cup J'\}}(y \triangleleft (c \oplus c')) = q_{\triangleleft\{L \setminus J \rightarrow \beta, J'\}}(q_{\triangleleft\{L \rightarrow \beta, J\}}(y \triangleleft c) \triangleleft c')$$

for all $y \in Y_{L \rightarrow \beta}$, $c \in Y_\diamond^J$ and all $c' \in Y_\diamond^{J'}$. In other words, (Y_S, q_N) is functional if the analogous relations to those in Proposition 5.2.4 hold. Thus (D_S, f_N) is functional (as well as being *the* functional Σ -algebra associated with (D_B, f_K) and D_S).

Now consider a bottomed functional Σ -algebra (Y_S, q_N) , i.e., a functional Σ -algebra which is also a bottomed Σ -algebra (and as usual the bottom element of Y_σ will be denoted by \perp_σ for each $\sigma \in S$). Then (Y_S, q_N) will be called a *monotone regular functional extension* of (X_B, p_K) if the following two conditions hold:

- (i) If $\sigma = L \rightarrow \beta$ is a functional type and J is a non-empty subset of L then

$$q_{\triangleleft\{\sigma,J\}}(\perp_\sigma \triangleleft c) = \perp_{L \setminus J \rightarrow \beta}$$

for all $c \in Y_\diamond^J$.

- (ii) The bottomed Λ -algebra (Y_B, q_K) (obtained from (Y_S, q_N) by omitting the sets $\{Y_\sigma : \sigma \in S \setminus B\}$ and the mappings $\{q_\nu : \nu \in N \setminus K\}$) is a monotone regular extension of (X_B, p_K) .

Such an extension (Y_S, q_N) of (X_B, p_K) will be said to be *extensional* if in addition the following condition holds:

(iii) If $\sigma = L \rightarrow \beta$ is a functional type and $y, y' \in Y_\sigma$ then $y' = y$ if and only if

$$q_{\triangleleft\{\sigma, J\}}(y \triangleleft c) = q_{\triangleleft\{\sigma, L\}}(y' \triangleleft c)$$

for all $c \in Y_\sigma^L$.

Thus by Propositions 5.2.3 and 5.2.4 the Σ -algebra (D_S, f_N) is a monotone regular functional extension of (X_B, p_K) which, moreover, is clearly extensional.

For most of the study, however, it will just be assumed that (D_S, f_N) is simply a monotone regular functional extension of (X_B, p_K) (and not that (D_S, f_N) is the functional Σ -algebra associated with (D_B, f_K) and D_S). In fact, in general it will not even be assumed that (D_S, f_N) is extensional.

There are two reasons for working with this more general framework. The first is that it emphasises the role of the functional application operators rather than that of the individual functions. The second is that there is at least one important example of a monotone regular functional extension of (X_B, p_K) which is not extensional.

This example will now be looked at. Let (Y_S, q_N) be as in Case 2; thus (Y_B, q_K) is a minimal monotone regular extension of (X_B, p_K) , Y_S is the full monotone functional extension of Y_B and (Y_S, q_N) is the Σ -algebra associated with (Y_B, q_K) and Y_S . Moreover, Y_β is here considered as a poset with the partial order \sqsubseteq_β , where \sqsubseteq_B is the unique ordering associated with (Y_B, q_K) .

Lemma 5.2.2 *For each $\sigma \in S$ let D_σ be an initial completion of the poset Y_σ and let $\nu \in N$ be of type $L \rightarrow \sigma$. Then the mapping $q_\nu : Y_\sigma^L \rightarrow Y_\sigma$ extends uniquely to a continuous mapping $f_\nu : D_\sigma^L \rightarrow D_\sigma$.*

Proof This is the same as the proof of Lemma 4.4.1. \square

Lemma 5.2.2 allows the following definition to be made: A Σ -algebra (D_S, f_N) is said to be an *initial completion* of (Y_S, q_N) if D_σ is an initial completion of the poset Y_σ for each $\sigma \in S$ and f_ν is the unique continuous extension of q_ν for each $\nu \in N$. It follows immediately from Proposition 4.3.2 that there exists an initial completion of (Y_S, q_N) .

An initial completion (D_S, f_N) of (Y_S, q_N) is, in particular, an extension of (Y_S, q_N) . Moreover, by Lemma 4.3.1 the bottom element \perp_σ of Y_σ is also the the bottom element of the poset D_σ . The partial order on D_σ will always be denoted by \sqsubseteq_σ .

Proposition 5.2.5 *Let (D_S, f_N) and (D'_S, f'_N) be initial completions of (Y_S, q_N) . Then for each $\sigma \in S$ there exists a unique order isomorphism $\pi_\sigma : D_\sigma \rightarrow D'_\sigma$ with $\pi_\sigma(y) = y$ for each $y \in Y_\sigma$ (and by Proposition 4.2.1 π_σ and the inverse mapping π_σ^{-1} are then both continuous). Moreover, the family π_S is also an isomorphism from (D_S, f_N) to (D'_S, f'_N) .*

Proof This is the same as the proof of Proposition 4.4.1. \square

In what follows let (D_S, f_N) be an initial completion of (Y_S, q_N) ; this situation will be referred to as Case 2'. Proposition 5.2.5 implies that (D_S, f_N) is, in the appropriate sense, unique. Note that the Λ -algebra (D_B, f_K) occurring here is the same as the corresponding Λ -algebra in Case 3.

Lemma 5.2.3 *The initial completion (D_S, f_N) is a functional Σ -algebra.*

Proof Let $L \rightarrow \beta$ be a functional type, J and J' non-empty disjoint subsets of L . Then it is easily checked that the mappings $(u, c, c') \mapsto f_{\triangleleft\{L \rightarrow \beta, J \cup J'\}}(u \triangleleft (c \oplus c'))$ and $(u, c, c') \mapsto f_{\triangleleft\{L \setminus J \rightarrow \beta, J'\}}(f_{\triangleleft\{L \rightarrow \beta, J\}}(u \triangleleft c) \triangleleft c')$ from $D_\sigma \times D_\sigma^J \times D_\sigma^{J'}$ to $D_{L \setminus (J \cup J') \rightarrow \beta}$ are both continuous. Moreover, (since (Y_S, q_N) is functional) each is an extension of the monotone mapping $(y, b, b') \mapsto q_{\triangleleft\{L \rightarrow \beta, J \cup J'\}}(y \triangleleft (b \oplus b'))$ from $Y_\sigma \times Y_\sigma^J \times Y_\sigma^{J'}$ to $Y_{L \setminus (J \cup J') \rightarrow \beta}$. Thus by Lemma 4.3.3 these two mappings are equal, and this shows that (D_S, f_N) is a functional Σ -algebra. \square

Proposition 5.2.6 *The initial completion (D_S, f_N) is a monotone regular functional extension of (X_B, p_K) .*

Proof By Lemma 5.2.3 (D_S, f_N) is a functional Σ -algebra. Now if $\sigma = L \rightarrow \beta$ is a functional type and J a non-empty subset of L then the mapping $c \mapsto f_{\triangleleft\{\sigma, J\}}(\perp_\sigma \triangleleft c)$ from D_σ^J to $D_{L \setminus J \rightarrow \beta}$ is continuous, and by Proposition 5.2.3 it is an extension of the constant mapping $b \mapsto \perp_{L \setminus J \rightarrow \beta}$ from Y_σ^J to $Y_{L \setminus J \rightarrow \beta}$. Thus by Lemma 4.3.3 it follows that $f_{\triangleleft\{\sigma, J\}}(\perp_\sigma \triangleleft c) = \perp_{L \setminus J \rightarrow \beta}$ for all $c \in D_\sigma^J$. Finally, the fact that the Λ -algebra (D_B, f_K) is a monotone regular extension of (X_B, p_K) was already shown in Proposition 4.4.2. \square

The remarks made towards the end of Section 4.3 indicate that in general (D_S, f_N) will not be extensional. Some care should thus be taken when thinking of the elements in the sets D_σ , $\sigma \in S \setminus B$, as functions and the mappings f_ν , $\nu \in N \setminus K$, as functional application operators.

5.3 Functionally free algebras

The set-up introduced in Section 5.2 will be employed in this section, which means (D_S, f_N) is a monotone regular functional extension of (X_B, p_K) . In particular, it is a functional Σ -algebra (which is all that is really needed here). The elements occurring in the sets in the family $D_{S \setminus B}$ will be referred to as *functions*, since that is what they are in the three basic Cases (although of course in general this is really a misnomer).

The present section introduces what will be called functionally free Σ -algebras. This class of algebras includes the term algebras which will provide the basic building blocks for the rudimentary programming language to be defined in Chapter 6.

Very roughly, the aim is to find some kind of generalisation of the ground term algebra (F_B, \odot_K) which will allow references to be made to the functions occurring in the sets in the family $D_{S \setminus B}$. This, however, cannot be done directly, one reason being because of the following simple fact:

Lemma 5.3.1 *If (Y_S, q_N) is a minimal Σ -algebra (and in particular if (Y_S, q_N) is an initial Σ -algebra) then $Y_\sigma = \emptyset$ for each functional type $\sigma \in S \setminus B$.*

Proof Let Z_S be the family with $Z_\beta = Y_\beta$ for each $\beta \in B$ and $Z_\sigma = \emptyset$ for each $\sigma \in S \setminus B$. Then it is easily checked that Z_S is invariant in (Y_S, q_N) . (Note that if $\sigma = L \rightarrow \beta$ is a functional type and J a non-empty subset of L then $Z_\sigma^J = \emptyset$, since $\sigma \cdot J$ contains the element \diamond_σ of type σ). Thus $Y_\sigma = \emptyset$ for each functional type $\sigma \in S \setminus B$. \square

The only obvious way of referring to functions is the trivial one of simply giving some of them names: This at least provides a meaning to expressions involving these names, and such expressions or terms are the basic component of any functional programming language.

The set of names of the functions of interest (together with their types) specifies an S -typed set I which is considered to be fixed in what follows. Define a family I_S by putting $I_\sigma = \{\eta \in I : \langle \eta \rangle = \sigma\}$ for each $\sigma \in S$. What is required is a Σ -algebra (Y_S, q_N) containing I (i.e., $I_S \subset Y_S$) with the property that for each $c \in D_\diamond^I$ there exists a unique Σ -homomorphism $\pi_S^c : (Y_S, q_N) \rightarrow (D_S, f_N)$ such that $\pi_S^c(\xi) = c(\xi)$ for each $\xi \in I$. Then π_S^c can be regarded as giving a meaning, relative to the assignment c , to the elements in the sets in the family Y_S : For each $y \in Y_\sigma$ the ‘value’ $\pi_S^c(y)$ is interpreted as the meaning of the ‘expression’ y , given that ξ has already been assigned the meaning $c(\xi)$ for each $\xi \in I$.

Of course, (Y_S, q_N) could simply chosen to be an I -free Σ -algebra. However, this turns out not to be the best choice to make here. One reason is that an I -free Σ -algebra (Y_S, q_N) cannot be functional (since Proposition 2.3.8 implies that the sets $\text{Im}(q_\nu)$, $\nu \in N_\sigma$, form a partition of $Y_\sigma \setminus I_\sigma$ for each $\sigma \in S$). This fact would later cause some difficulties and suggests only considering functional Σ -algebras. But the definition of being free should then be modified accordingly: A functional Σ -algebra (Y_S, q_N) will thus be called *functionally I -free* if it contains I and if for each functional Σ -algebra

(Z_S, r_N) and each $c \in Z_\circ^I$ there exists a unique Σ -homomorphism π_ξ^c from (Y_S, q_N) to (Z_S, r_N) such that $\pi_\xi^c(\xi) = c(\xi)$ for all $\xi \in I$.

This modified definition provides a reasonable framework to work with, at least in as much as functionally free Σ -algebras exist:

Proposition 5.3.1 *There exists a functionally I -free Σ -algebra (Y_S, q_N) which is, in the usual sense, unique: If (Y'_S, q'_N) is a further functionally I -free Σ -algebra then there exists a unique isomorphism $\pi_S : (Y_S, q_N) \rightarrow (Y'_S, q'_N)$ such that $\pi_\eta(\xi) = \xi$ for all $\xi \in I$.*

Proof Later in this section. \square

A functionally I -free Σ -algebra (Y_S, q_N) clearly meets the requirements stated above: If $c \in D_\circ^I$ then, since the Σ -algebra (D_S, f_N) is functional, there exists a unique Σ -homomorphism $\pi_\xi^c : (Y_S, q_N) \rightarrow (D_S, f_N)$ such that $\pi_\xi^c(\xi) = c(\xi)$ for each $\xi \in I$.

At this point the reader is justified in being somewhat sceptical about the need for functionally free rather than just free algebras. However, it will now be shown that functionally free algebras also arise when taking a different approach, and this gives a much more convincing reason for employing them.

This alternative approach begins by noting that in all modern functional programming languages the functional application operators do not have to be named explicitly. The reason is essentially that the names of the functions involved (i.e., the elements of I) are themselves used as names for operators. More precisely, $\xi \in I$ is also used as the name of the operation of applying the function which ξ denotes. This turns out to be enough, since any expression involving general functional application operators can be expressed equivalently using only these more elementary operations.

To implement this approach formally names have to be introduced for the operations of applying elements of I to some or all of their arguments. Thus for each $\xi \in I$ of type $L \rightarrow \beta$ and each subset J of L let $\xi\{J\}$ be some element not in K and such that $\xi\{J\} \neq \xi'\{J'\}$ whenever $(\xi, J) \neq (\xi', J')$. The ‘name’ $\xi\{J\}$ will be used for the operation of applying the function which ξ denotes just to the arguments in J . This means that if ξ is of type $L \rightarrow \beta$ then $\xi\{L\}$ refers to a total application operation, whereas $\xi\{J\}$ refers to a partial application operation if J is a non-empty proper subset of L , and $\xi\{\emptyset\}$ will be essentially a synonym for ξ .

In line with this interpretation there is then the following new signature Σ^I : Put $N^I = K \cup A^I$, where A^I is the set consisting of the new elements introduced above, and define mappings $\Delta^I : N^I \rightarrow \mathcal{F}_S$ and $\delta^I : N^I \rightarrow S$ by

$$\Delta^I(\nu) = \begin{cases} \Delta(\nu) & \text{if } \nu \in K, \\ J & \text{if } \nu = \xi\{J\}, \end{cases}$$

$$\delta^I(\nu) = \begin{cases} \vartheta(\nu) & \text{if } \nu \in K, \\ L \setminus J \rightarrow \beta & \text{if } \nu = \xi\{J\} \text{ with } \xi \text{ of type } L \rightarrow \beta. \end{cases}$$

Then $\Sigma^I = (S, N^I, \Delta^I, \delta^I)$ is a signature which is an extension of Λ . If $\xi \in I$ is of type $L \rightarrow \beta$ and $J \subset L$ then $\xi\{J\}$ is of type $J \rightarrow (L \setminus J \rightarrow \beta)$ in Σ^I . In particular,

this means that $\xi\{L\}$ is of type $L \rightarrow \beta$ and $\xi\{\emptyset\}$ is of type $\emptyset \rightarrow (L \rightarrow \beta)$ (i.e., $\xi\{\emptyset\}$ is of type $\emptyset \rightarrow \langle \xi \rangle$).

Proposition 5.3.2 *Let (Y_S, q_{N^I}) be an initial Σ^I -algebra and $c \in D_\circ^I$. Then there exists a unique family of mappings π_S^c with $\pi_\sigma^c : Y_\sigma \rightarrow D_\sigma$ for each $\sigma \in S$ such that*

- (i) π_B^c is a Λ -homomorphism from (Y_B, q_K) to (D_B, f_K) .
- (ii) $\pi_\xi^c(q_{\xi\{\emptyset\}}(\varepsilon)) = c(\xi)$ for each $\xi \in I$.
- (iii) If $\xi \in I$ is of type $L \rightarrow \beta$ and J is a non-empty subset of L then

$$\pi_{L \setminus J \rightarrow \beta}^c(q_{\xi\{J\}}(b)) = f_{\triangleleft\{\sigma, J\}}(c(\xi) \triangleleft (\pi^c)^J(b))$$

for all $b \in Y_\circ^J$.

(Note in (i) that (Y_B, q_K) , obtained by omitting the sets $\{Y_\sigma : \sigma \in S \setminus B\}$ from the family Y_S and the mappings $\{q_\nu : \nu \in N^I \setminus K\}$ from the family q_{N^I} , is a Λ -algebra.)

Proof Define a Σ^I -algebra $(D_S, f_{N^I}^c)$ with $f_\kappa^c = f_\kappa$ for each $\kappa \in K$ as follows: If $\xi \in I$ is of type $\sigma = L \rightarrow \beta$ and J is a non-empty subset of L then $f_{\xi\{J\}}^c : D_\circ^J \rightarrow D_{L \setminus J \rightarrow \beta}$ is the mapping given by $f_{\xi\{J\}}^c(b) = f_{\triangleleft\{\sigma, J\}}(c(\xi) \triangleleft b)$ for all $b \in D_\circ^J$, and for each $\xi \in I$ the mapping $f_{\xi\{\emptyset\}}^c : \mathbb{I} \rightarrow D_\xi$ is given by $f_{\xi\{\emptyset\}}^c(\varepsilon) = c(\xi)$. By construction $(D_S, f_{N^I}^c)$ is then an extension of (D_B, f_K) . Let π_S^c be the unique Σ^I -homomorphism from (Y_S, q_{N^I}) to $(D_S, f_{N^I}^c)$. Then it is easy to see that the family π_S^c satisfies (i), (ii) and (iii). The uniqueness follows since, conversely, any family satisfying (i), (ii) and (iii) is actually a Σ^I -homomorphism from (Y_S, q_{N^I}) to $(D_S, f_{N^I}^c)$. \square

If (Y_S, q_{N^I}) is an initial Σ^I -algebra and $c \in D_\circ^I$ then the unique family of mappings π_S^c given in Proposition 5.3.2 can be used in the same way as the corresponding families defined in terms of either an I -free or a functionally I -free Σ -algebra: π_S^c is considered as giving a meaning, relative to the assignment c , to the elements in the sets in the family Y_S , i.e., for each $y \in Y_\sigma$ the ‘value’ $\pi_\sigma^c(y)$ is interpreted as the meaning of the ‘expression’ y , given that ξ has already been assigned the meaning $c(\xi)$ for each $\xi \in I$.

There are now at least two seemingly different approaches to constructing ‘expressions’ involving the names in I . The first involves functionally I -free Σ -algebras (or even just I -free Σ -algebras) and the second initial Σ^I -algebras. However, as will be seen below, initial Σ^I -algebras and functionally I -free Σ -algebras are essentially the same objects, and so the choice of which to use is really just a matter of taste. This is the real reason for using functionally I -free rather than just I -free Σ -algebras.

A procedure will be developed for switching between functional Σ -algebras containing I and Σ^I -algebras, and it will be shown that under this procedure the initial Σ^I -algebras correspond exactly to the functionally I -free Σ -algebras. These transformations only change the families of mappings in the algebras and not the family of sets, i.e., there is a family of sets Y_S which occurs unchanged in all the algebras which will be constructed.

First consider any Σ -algebra (Y_S, q_N) and an assignment $c \in Y_\diamond^I$. A Σ^I -algebra can then be defined as follows: Let $\xi \in I$ be of type $\sigma = L \rightarrow \beta$. If J is a non-empty subset of L then let $q_{\xi\{J\}} : Y_\diamond^J \rightarrow Y_{L \setminus J \rightarrow \beta}$ be the mapping given by

$$q_{\xi\{J\}}(b) = q_{\triangleleft\{\sigma, J\}}(c(\xi) \triangleleft b)$$

for each $b \in Y_\diamond^J$. Moreover, let $q_{\xi\{\emptyset\}} : \mathbb{I} \rightarrow Y_{L \rightarrow \beta}$ be the mapping defined by

$$q_{\xi\{\emptyset\}}(\varepsilon) = c(\xi).$$

In this way a Σ^I -algebra (Y_S, q_{NI}) is obtained, which will be called the Σ^I -algebra *associated* with (Y_S, q_N) and the assignment c : The family q_{NI} is derived from the family q_N by omitting the mappings in $\{q_\nu : \nu \in A\}$ and replacing them with the mappings $\{q_\mu : \mu \in A^I\}$.

Consider the special case of the above construction in which (Y_S, q_N) is a Σ -algebra containing I and $i \in Y_\diamond^I$ is the assignment given by $i(\xi) = \xi$ for each $\xi \in I$. The resulting Σ^I -algebra (Y_S, q_{NI}) will be called simply the Σ^I -algebra *associated* with (Y_S, q_N) . Note here that if $\xi \in I$ is of type $\sigma = L \rightarrow \beta$ and J is a non-empty subset of L then $q_{\xi\{J\}} : Y_\diamond^J \rightarrow Y_{L \setminus J \rightarrow \beta}$ is given by

$$q_{\xi\{J\}}(b) = q_{\triangleleft\{\sigma, J\}}(\xi \triangleleft b)$$

for each $b \in Y_\diamond^J$, and $q_{\xi\{\emptyset\}} : \mathbb{I} \rightarrow Y_{L \rightarrow \beta}$ is the mapping defined by $q_{\xi\{\emptyset\}}(\varepsilon) = \xi$.

A Σ^I -algebra (Y_S, q_{NI}) will be said to be *adapted to I* if $q_{\xi\{\emptyset\}}(\varepsilon) = \xi$ for each $\xi \in I$ (which in particular implies that (Y_S, q_{NI}) must contain I). Note that if (Y_S, q_N) is a Σ -algebra containing I then by definition the Σ^I -algebra associated with (Y_S, q_N) is adapted to I .

Until further notice let (Y_S, q_{NI}) now be an initial Σ^I -algebra adapted to I . It will be shown that in this case the above construction can be reversed to obtain a functional Σ -algebra (Y_S, q_N) , which in fact turns out to be functionally I -free.

Lemma 5.3.2 *Let $\sigma = L \rightarrow \beta$ be a functional type and $y \in Y_\sigma$; then there exists a unique $\xi \in I$ having type $L' \cup L \rightarrow \beta$ for some $L' \in \mathcal{F}_S$ disjoint from L and a unique assignment $b \in Y_\diamond^{L'}$ such that $y = q_{\xi\{L'\}}(b)$.*

Proof This follows since by Proposition 2.3.2 (Y_S, q_{NI}) is a regular Σ^I -algebra (and because $\sigma \notin B$). \square

Note that if $\xi \in I_\sigma$ then the unique representation of ξ given in Lemma 5.3.2 is just $q_{\xi\{\emptyset\}}(\varepsilon)$. Conversely, if $y \in Y_\sigma \setminus I_\sigma$ then the unique representation of y has the form $y = q_{\xi\{L'\}}(b)$ with $L' \in \mathcal{F}_S^\circ$.

Let $\sigma = L \rightarrow \beta$ be a functional type and J be a non-empty subset of L ; consider $y \in Y_\sigma$ and $c \in Y_\diamond^J$. By Lemma 5.3.2 there then exists a unique $\xi \in I$ having type $L' \cup L \rightarrow \beta$ for some $L' \in \mathcal{F}_S$ disjoint from L and a unique assignment $b \in Y_\diamond^{L'}$ such that $y = q_{\xi\{L'\}}(b)$. There is thus an element $q_{\xi\{L' \cup J\}}(b \oplus c)$ of $Y_{L \setminus J \rightarrow \beta}$ which is it convenient to denote simply by $y(c)$. In particular, if $y = \xi \in I_\sigma$ then $y(c) = q_{\xi\{J\}}(c)$.

Lemma 5.3.3 *Let $\sigma = L \rightarrow \beta$ be a functional type and J and J' be non-empty disjoint subsets of L . Then for all $y \in Y_\sigma$, $c \in D_\diamond^J$ and all $c' \in D_\diamond^{J'}$*

$$y(c \oplus c') = y(c)(c').$$

Proof Let $y \in Y_\sigma$ and let $q_{\xi\{L'\}}(b)$ be the unique representation of y given in Lemma 5.3.2. Then it follows that $y(c) = q_{\xi\{L' \cup J\}}(b \oplus c)$, which means $q_{\xi\{L' \cup J\}}(b \oplus c)$ is the unique representation of the element $y(c)$, and hence that

$$\begin{aligned} y(c)(c') &= q_{\xi\{(L' \cup J) \cup J'\}}((b \oplus c) \oplus c') \\ &= q_{\xi\{L' \cup (J \cup J')\}}(b \oplus (c \oplus c')) = y(c \oplus c') \end{aligned}$$

for all $c \in Y_\diamond^J$, $c' \in Y_\diamond^{J'}$. \square

Let $\sigma = L \rightarrow \beta$ be a functional type and J be a non-empty subset of L . Then a mapping $q_{\triangleleft\{\sigma, J\}} : Y_\diamond^{\sigma \cdot J} \rightarrow Y_{L \setminus J \rightarrow \beta}$ can be defined by letting

$$q_{\triangleleft\{\sigma, J\}}(y \triangleleft c) = y(c)$$

for all $y \in Y_\sigma$, $c \in Y_\diamond^J$. (In particular, if $\xi \in I_\sigma$ then $q_{\triangleleft\{\sigma, J\}}(\xi \triangleleft c) = q_{\xi\{J\}}(c)$ for each $c \in Y_\diamond^J$.) This defines a Σ -algebra (Y_S, q_N) : The family q_N is obtained from the family q_{N^I} by omitting the mappings in $\{q_\mu : \mu \in A^I\}$ and replacing them with the mappings $\{q_\nu : \nu \in A\}$. Of course, (Y_S, q_N) contains I because (Y_S, q_{N^I}) is adapted to I .

Lemma 5.3.4 *(Y_S, q_N) is a functional Σ -algebra.*

Proof Let $L \rightarrow \beta$ be a functional type and J and J' be non-empty disjoint subsets of L . Then by Lemma 5.3.3 $y(c \oplus c') = y(c)(c')$, which just means that

$$q_{\triangleleft\{L \rightarrow \beta, J \cup J'\}}(y \triangleleft (c \oplus c')) = q_{\triangleleft\{L \setminus J \rightarrow \beta, J'\}}(q_{\triangleleft\{L \rightarrow \beta, J\}}(y \triangleleft c) \triangleleft c')$$

for all $y \in Y_{L \rightarrow \beta}$, $c \in Y_\diamond^J$ and $c' \in Y_\diamond^{J'}$. \square

Let us call (Y_S, q_N) the *functional Σ -algebra associated with (Y_S, q_{N^I})* , and note that the first construction introduced above is in the following sense the inverse of the second:

Lemma 5.3.5 *If (Y_S, q_N) is the functional Σ -algebra associated with (Y_S, q_{N^I}) then (Y_S, q_{N^I}) is the Σ^I -algebra associated with (Y_S, q_N) .*

Proof This follows because if $\xi \in I$ is of type $\sigma = L \rightarrow \beta$ and J is a non-empty subset of L then by definition $q_{\triangleleft\{\sigma, J\}}(\xi \triangleleft c) = q_{\xi\{J\}}(c)$ for each $c \in Y_\diamond^J$. \square

The following result gives the first indication of the relationship between functionally I -free Σ -algebras and initial Σ^I -algebras.

Proposition 5.3.3 *The functional Σ -algebra (Y_S, q_N) associated with the initial Σ^I -algebra (Y_S, q_{N^I}) is functionally I -free.*

Proof Let (Z_S, r_N) be a functional Σ -algebra and $c \in Z_\diamond^I$, and let (Z_S, r_{N^I}) be the Σ^I -algebra associated with (Z_S, r_N) and c . Then, since (Y_S, q_{N^I}) is initial, there exists a unique Σ^I -homomorphism $\pi_S : (Y_S, q_{N^I}) \rightarrow (Z_S, r_{N^I})$, and it will be shown that π_S is actually the unique Σ -homomorphism from (Y_S, q_N) to (Z_S, r_N) with $\pi_\eta(\eta) = c(\eta)$ for each $\eta \in I$.

Thus consider a functional type $\sigma = L \rightarrow \beta$, let J be a non-empty subset of L , and let $y \in Y_\sigma$ and $b \in Y_\diamond^J$. Let $q_{\xi\{L'\}}(b')$ be the unique representation of y given in Lemma 5.3.2 (so $L' \in \mathcal{F}_S$ is disjoint from L , $\xi \in I$ is of type $L' \cup L \rightarrow \beta$ and $b' \in Y_\diamond^{L'}$). Assume first that $L' \neq \emptyset$ and put $L^\circ = L' \cup L$. Then, since π_S is a Σ^I -homomorphism, it follows that

$$\begin{aligned} \pi_{L \setminus J \rightarrow \beta}(q_{\triangleleft\{\sigma, J\}}(y \triangleleft b)) &= \pi_{L \setminus J \rightarrow \beta}(q_{\xi\{L' \cup J\}}(b' \oplus b)) \\ &= r_{\xi\{L' \cup J\}}(\pi_\diamond^{L' \cup J}(b' \oplus b)) = r_{\triangleleft\{L^\circ \rightarrow \beta, L' \cup J\}}(c(\xi) \triangleleft \pi_\diamond^{L' \cup J}(b' \oplus b)). \end{aligned}$$

But $\pi_\diamond^{L' \cup J}(b' \oplus b) = \pi_\diamond^{L'}(b') \oplus \pi_\diamond^J(b)$ and (Z_S, q_N) is a functional Σ -algebra, thus

$$\begin{aligned} r_{\triangleleft\{L^\circ \rightarrow \beta, L' \cup J\}}(c(\xi) \triangleleft \pi_\diamond^{L' \cup J}(b' \oplus b)) \\ &= r_{\triangleleft\{L^\circ \rightarrow \beta, L' \cup J\}}(c(\xi) \triangleleft (\pi_\diamond^{L'}(b') \oplus \pi_\diamond^J(b))) \\ &= r_{\triangleleft\{L^\circ \rightarrow \beta, L' \cup J\}}((c(\xi) \triangleleft \pi_\diamond^{L'}(b')) \oplus \pi_\diamond^J(b)) \\ &= r_{\triangleleft\{L \rightarrow \beta, J\}}(r_{\triangleleft\{L^\circ \rightarrow \beta, L'\}}(c(\xi) \triangleleft \pi_\diamond^{L'}(b')) \triangleleft \pi_\diamond^J(b)). \end{aligned}$$

Moreover, again using that π_S is a Σ^I -homomorphism,

$$r_{\triangleleft\{L^\circ \rightarrow \beta, L'\}}(c(\xi) \triangleleft \pi_\diamond^{L'}(b')) = r_{\xi\{L'\}}(\pi_\diamond^{L'}(b')) = \pi_{L \rightarrow \beta}(q_{\xi\{L'\}}(b')) = \pi_\sigma(y)$$

and therefore

$$\begin{aligned} r_{\triangleleft\{L \rightarrow \beta, J\}}(r_{\triangleleft\{L^\circ \rightarrow \beta, L'\}}(c(\xi) \triangleleft \pi_\diamond^{L'}(b')) \triangleleft \pi_\diamond^J(b)) \\ = r_{\triangleleft\{\sigma, J\}}(\pi_\sigma(y) \triangleleft \pi_\diamond^J(b)) = r_{\triangleleft\{\sigma, J\}}(\pi_\diamond^{\sigma \cdot J}(y \triangleleft b)). \end{aligned}$$

Putting these steps together then gives the equality

$$\pi_{L \setminus J \rightarrow \beta}(q_{\triangleleft\{\sigma, J\}}(y \triangleleft b)) = r_{\triangleleft\{\sigma, J\}}(\pi_\diamond^{\sigma \cdot J}(y \triangleleft b)),$$

and a somewhat easier calculation shows that this also holds when $L' = \emptyset$. Therefore π_S is a Σ -homomorphism from (Y_S, q_N) to (Z_S, r_N) (since π_B is automatically a Λ -homomorphism from (Y_B, q_K) to (Z_B, r_K)). Moreover,

$$\pi_\eta(\eta) = \pi_\eta(q_{\eta\{\emptyset\}}(\varepsilon)) = r_{\eta\{\emptyset\}}(\varepsilon) = c(\eta)$$

for each $\eta \in I$, since (Y_S, q_{N^I}) is adapted to I .

To show the uniqueness consider any invariant family Y'_S in (Y_S, q_N) containing I_S . Then, since $q_{\xi\{\emptyset\}}(\varepsilon) = \xi$ for each $\xi \in I$ and since

$$q_{\xi\{J\}}(b) = q_{\triangleleft\{\sigma, J\}}(\xi \triangleleft b)$$

for each $b \in Y_\diamond^J$ whenever $\xi \in I$ is of type $\sigma = L \rightarrow \beta$ and J is a non-empty subset of L , it follows that the family Y'_S is also invariant in (Y_S, q_{N^I}) , and so $Y'_S = Y_S$,

since (Y_S, q_{N^I}) is a minimal Σ^I -algebra. But if π_S and π'_S are Σ -homomorphisms from (Y_S, q_N) to (Z_S, r_N) with $\pi_\eta(\eta) = c(\eta) = \pi'_\eta(\eta)$ for each $\eta \in I$ then the family Y'_S defined by $Y'_\sigma = \{y \in Y_\sigma : \pi'_\sigma(y) = \pi_\sigma(y)\}$ for each $\sigma \in S$ contains I_S and is invariant in (Y_S, q_N) . Thus from the above $Y'_S = Y_S$, and therefore $\pi'_S = \pi_S$. \square

Proposition 5.3.3 will now be applied to prove Proposition 5.3.1.

Proof of Proposition 5.3.1 By Proposition 2.3.2 there exists an initial Σ^I -algebra and there then also exists such a Σ^I -algebra (Y_S, q_{N^I}) which is adapted to I . (An initial Σ^I -algebra (Y_S, q_{N^I}) is unambiguous, and hence $q_{\xi\{\emptyset\}}(\varepsilon) \neq q_{\xi'\{\emptyset\}}(\varepsilon)$ whenever ξ and ξ' are different elements of I_σ for some $\sigma \in S$. It is thus easy to arrange that $q_{\xi\{\emptyset\}}(\varepsilon) \neq q_{\xi'\{\emptyset\}}(\varepsilon)$ whenever ξ and ξ' are different elements of I , and in this case $q_{\xi\{\emptyset\}}(\varepsilon)$ can be identified with ξ for each $\xi \in I$.) By Proposition 5.3.3 the functional Σ -algebra associated with (Y_S, q_{N^I}) is then functionally I -free. The uniqueness follows in the usual way. \square

At this point the assumption that (Y_S, q_{N^I}) is an initial Σ^I -algebra will be dropped. The next result is the key to understanding the relationship between initial Σ^I -algebras and functionally I -free Σ -algebras.

Proposition 5.3.4 *A functional Σ -algebra (Y_S, q_N) containing I is functionally I -free if and only if the Σ^I -algebra (Y_S, q_{N^I}) associated with (Y_S, q_N) is initial.*

Proof As in the proof of Proposition 5.3.1 there exists an initial Σ^I -algebra (Y'_S, q'_{N^I}) which is adapted to I . Let π_S be the unique Σ^I -homomorphism from (Y'_S, q'_{N^I}) to (Y_S, q_{N^I}) ; then (Y_S, q_{N^I}) is initial if and only if π_S is a Σ^I -isomorphism, which is the case if and only if the mapping π_σ is a bijection for each $\sigma \in S$.

Consider also the functional Σ -algebra (Y'_S, q'_N) associated with (Y'_S, q'_{N^I}) . Then by Proposition 5.3.3 (Y'_S, q'_N) is functionally I -free and therefore there exists a unique Σ -homomorphism π'_S from (Y'_S, q'_N) to (Y_S, q_N) such that $\pi'_\xi(\xi) = \xi$ for all $\xi \in I$. Moreover, it is easy to see that (Y_S, q_N) is functionally I -free if and only if π'_S is a Σ -isomorphism, which is again the case if and only if the mapping π'_σ is a bijection for each $\sigma \in S$.

This means it is enough to show that $\pi_S = \pi'_S$ or, equivalently, that $Y_S^o = Y'_S$, where $Y_S^o = \{y \in Y'_\sigma : \pi_\sigma(y) = \pi'_\sigma(y)\}$ for each $\sigma \in S$. Consider $\xi \in I$ having functional type $\sigma = L \rightarrow \beta$ and a non-empty subset J of L , and let $c \in (Y_\sigma^o)^J$. Then, putting $\tau = L \setminus J \rightarrow \beta$, it follows that

$$\begin{aligned} \pi_\tau(q'_{\xi\{J\}}(c)) &= q_{\xi\{J\}}(\pi_\sigma^J(c)) = q_{\xi\{J\}}((\pi'_\sigma)^J(c)) \\ &= q_{\triangleleft\{\sigma, J\}}(\xi \triangleleft (\pi'_\sigma)^J(c)) = q_{\triangleleft\{\sigma, J\}}((\pi'_\sigma)^{\sigma \cdot J}(\xi \triangleleft c)) \\ &= \pi'_\tau(q'_{\triangleleft\{\sigma, J\}}(\xi \triangleleft c)) = \pi'_\tau(q'_{\xi\{J\}}(c)) \end{aligned}$$

and hence $q'_{\xi\{J\}}(c) \in Y_\tau^o$. The family Y_S^o is therefore invariant in (Y'_S, q'_{N^I}) (noting that the other cases which should be checked hold trivially because π_B and π'_B are both Λ -homomorphisms from (Y'_B, q'_K) to (Y_B, q_K)). Thus $Y_S^o = Y'_S$, since (Y'_S, q'_{N^I}) is a minimal Σ^I -algebra. \square

The following result complements Lemma 5.3.5.

Lemma 5.3.6 *Let (Y_S, q_N) be a functionally I -free Σ -algebra and let (Y_S, q_{N^I}) be the Σ^I -algebra associated with (Y_S, q_N) . Then (Y_S, q_N) is the functional Σ -algebra associated with (Y_S, q_{N^I}) .*

Proof Let $\sigma = L \rightarrow \beta$ be a functional type and J be a non-empty subset of L ; let $y \in Y_\sigma$ and $c \in Y_\sigma^J$. Then y has a unique representation of the form $y = q_{\xi\{L'\}}(b)$ with $\xi \in I$ of type $L' \cup L \rightarrow \beta$ for some L' disjoint from L and with $b \in Y_\sigma^{L'}$. But then by definition $q_{\xi\{L'\}}(b) = q_{\triangleleft\{L' \cup L \rightarrow \beta, L'\}}(\xi \triangleleft b)$ and

$$q_{\xi\{L' \cup J\}}(b \oplus c) = q_{\triangleleft\{L' \cup L \rightarrow \beta, L' \cup J\}}(\xi \triangleleft (b \oplus c))$$

and therefore, since (Y_S, q_N) is a functional Σ -algebra, it follows that

$$\begin{aligned} q_{\triangleleft\{\sigma, J\}}(y \triangleleft c) &= q_{\triangleleft\{L \rightarrow \beta, J\}}(q_{\triangleleft\{L' \cup L \rightarrow \beta, L'\}}(\xi \triangleleft b) \triangleleft c) \\ &= q_{\triangleleft\{L' \cup L \rightarrow \beta, L' \cup J\}}(\xi \triangleleft (b \oplus c)) = q_{\xi\{L' \cup J\}}(b \oplus c), \end{aligned}$$

i.e., $q_{\triangleleft\{\sigma, J\}}(y \triangleleft c) = q_{\xi\{L' \cup J\}}(b \oplus c)$. \square

Propositions 5.3.3 and 5.3.4 together with Lemmas 5.3.5 and 5.3.6 imply that there is a natural one-to-one correspondence between functionally I -free Σ -algebras and initial Σ^I -algebras adapted to I . Moreover, the following result shows that for each $c \in D_\sigma^I$ the meaning, relative to the assignment c , of the elements in the sets in the family Y_S does not depend on which approach is taken.

Proposition 5.3.5 *Let (Y_S, q_N) be a functionally I -free Σ -algebra, let $c \in D_\sigma^I$ and $\pi_S^c : (Y_S, q_N) \rightarrow (D_S, f_N)$ be the unique Σ -homomorphism such that $\pi_\eta^c(\eta) = c(\eta)$ for each $\eta \in I$. Then π_S^c is also the unique family of mappings given by Proposition 5.3.2 in terms of the Σ^I -algebra (Y_S, q_{N^I}) associated with (Y_S, q_N) .*

Proof Condition (iii) in Proposition 5.3.2 holds because if $\xi \in I$ is of type $\sigma = L \rightarrow \beta$ and J is a non-empty subset of L then

$$\pi_{L \setminus J \rightarrow \beta}^c(q_{\xi\{J\}}(b)) = \pi_{L \setminus J \rightarrow \beta}^c(q_{\triangleleft\{\sigma, J\}}(\xi \triangleleft b)) = f_{\triangleleft\{\sigma, J\}}(c(\xi) \triangleleft (\pi^c)_\sigma^J(b))$$

for all $b \in Y_\sigma^J$. Conditions (i) and (ii) hold trivially. \square

The above results imply that there is very little difference between a functionally I -free Σ -algebra and an initial Σ^I -algebra adapted to I . The choice has been made to work formally with functionally I -free Σ -algebras, but continual use will be made of the fact that the associated Σ^I -algebras are initial. One advantage of making definitions and stating results in terms of functionally I -free Σ -algebras rather than initial Σ^I -algebras is that there is only one signature Σ involved (independent of I) and that the family π_S^c is in this case explicitly a homomorphism

The following variation on Proposition 5.3.1 will be needed:

Proposition 5.3.6 *Let (Z_B, r_K) be an initial Λ -algebra disjoint from I (i.e., such that $I_\beta \cap Z_\beta = \emptyset$ for each $\beta \in B$). Then there exists a functionally I -free Σ -algebra (Y_S, q_N) which is an extension of (Z_B, r_K) . Moreover, if (Y'_S, q'_N) is any such algebra and π_S is the unique isomorphism from (Y_S, q_N) to (Y'_S, q'_N) such that $\pi_\eta(\xi) = \xi$ for all $\xi \in I$ then $\pi_\beta(z) = z$ for all $z \in Z_\beta$, $\beta \in B$.*

Proof By Proposition 2.5.1 there exists an initial Σ^I -algebra which is an extension of (Z_B, r_K) , and exactly as in the proof of Proposition 5.3.1 there then also exists such a Σ^I -algebra (Y_S, q_{N^I}) which is adapted to I . Let (Y_S, q_N) be the functional Σ -algebra associated with (Y_S, q_{N^I}) . Proposition 5.3.3 then implies that (Y_S, q_N) is functionally I -free and by definition (Y_S, q_N) is an extension of (Z_B, r_K) . The final statement follows by considering the appropriate invariant family in (Z_B, r_K) . \square

Finally, the following fact will be needed several times:

Proposition 5.3.7 *Let (Y_S, q_N) be a functionally I -free Σ -algebra. Then Y_S is the only invariant family in (Y_S, q_N) containing I_S .*

Proof If Y'_S is invariant in (Y_S, q_N) and contains I_S then, as already noted in the proof of Proposition 5.3.3, Y'_S is also invariant in the Σ^I -algebra (Y_S, q_{N^I}) associated with (Y_S, q_N) . Thus $Y'_S = Y_S$, since by Proposition 5.3.4 (Y_S, q_{N^I}) is initial and so in particular minimal. (In fact, this result can be proved directly from the definition of being functionally I -free: If Y'_S is as above then it is easy to see that the subalgebra (Y'_S, q'_N) associated with Y'_S is also functionally I -free, and from this it follows that $Y'_S = Y_S$.) \square

Note that Proposition 5.3.7 implies in particular that a functionally I -free Σ -algebra (Y_S, q_N) satisfies the hypotheses of Proposition 5.2.3.

5.4 Primitive functions

For the whole of the section assume (D_S, f_N) is a monotone regular functional extension of (X_B, p_K) . If $\sigma = L \rightarrow \beta$ is a functional type then $u \in D_\sigma$ is said to be *associated* with a mapping $h : D_\sigma^L \rightarrow D_\beta$ if $f_{\triangleleft\{\sigma, L\}}(u \triangleleft c) = h(c)$ for all $c \in D_\sigma^L$.

Lemma 5.4.1 *Let $\sigma = L \rightarrow \beta$ be a functional type and let $h : D_\sigma^L \rightarrow D_\beta$ be a mapping. Then in Case 1 there always exists $u \in D_\sigma$ associated with h . In Case 2 u exists if and only if h is monotone and in Case 3 if and only if h is continuous. Moreover, in each of these cases u , if it exists, is just the mapping h considered as an element of D_σ .*

Proof This is clear. \square

Before introducing the equations in Chapter 6 there is a further point that has to be considered which concerns the so-called *primitive* or *built-in* functions. These are the functions occurring in the equations which are really part of what is given (as opposed to the functions which the equations are supposed to define). For instance, the symbols $+$ and $-$ occur in the original equations in Chapter 1 and in the discussion there it was implicitly assumed that $+$ and $-$ refer to the usual arithmetical operations of addition and subtraction. Somewhat more importantly, the right-hand sides of the equations for it , fst and pfb clearly also involve some kind of built-in ‘case’ operation.

As in these original equations, in general there are two kinds of primitive functions which are needed. On the one hand it is useful to have a few *integer operators*, and on the other hand nothing non-trivial can be achieved without a suitable family of *case operators*. This section explains exactly what these operators are.

Integer operators: Let $Add : D_{\mathbf{int}} \times D_{\mathbf{int}} \rightarrow D_{\mathbf{int}}$ be the strict extension of the arithmetical operator $+$: $\mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{Z}$, i.e.,

$$Add(m, n) = \begin{cases} m + n & \text{if } m, n \in \mathbb{Z}, \\ \perp_{\mathbf{int}} & \text{otherwise,} \end{cases}$$

(recalling that $D_{\mathbf{int}} = \mathbb{Z} \cup \{\perp_{\mathbf{int}}\}$, since (D_B, f_K) is a regular extension of (X_B, p_K) and \mathbf{int} is a primitive type). Similarly let Sub and Mul be the strict extensions of the corresponding arithmetical operators $-$ and \times .

Moreover, let $Eq : D_{\mathbf{int}} \times D_{\mathbf{int}} \rightarrow D_{\mathbf{bool}}$ be the strict extension of the relational operator $=$: $\mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{B}$, i.e.,

$$Eq(m, n) = \begin{cases} T & \text{if } m, n \in \mathbb{Z} \text{ and } m = n, \\ F & \text{if } m, n \in \mathbb{Z} \text{ and } m \neq n, \\ \perp_{\mathbf{bool}} & \text{otherwise.} \end{cases}$$

Similarly let Neq , Le and Ge be the strict extensions of the relational operators \neq , \leq and \geq .

From now on int_2 will be used to denote the list int int (this list being considered as a B -typed set). The Σ -algebra (D_S, f_N) will be said to *support integer operators* if there exist elements add , sub and mul of $D_{\text{int}_2 \rightarrow \text{int}}$ associated respectively with Add , Sub and Mul and elements eq , neq , le and ge of $D_{\text{int}_2 \rightarrow \text{bool}}$ associated respectively with Eq , Neq , Le and Ge . This means that

$$f_{\triangleleft\{\text{int}_2 \rightarrow \text{int}, \text{int}_2\}}(\text{add} \triangleleft (m, n)) = \text{Add}(m, n)$$

for all $m, n \in D_{\text{int}}$, and that the analogous statements hold for the other operators.

In the three basic Cases (D_S, f_N) does support integer operators. This is trivially true in Case 1, and the following two results, together with Lemma 5.4.1, show that it also true in Cases 2 and 3.

Proposition 5.4.1 *In Case 2 the mappings Add , Sub , Mul , Eq , Neq , Le and Ge are all monotone.*

Proof If $n \in \mathbb{Z}$ then $n' \sqsubseteq_{\text{int}} n$ if and only if n' is either n or \perp_{int} ; moreover, the analogous statement holds for D_{bool} . From this it follows that the strict extension of any mapping from $\mathbb{Z} \times \mathbb{Z}$ to either \mathbb{Z} or \mathbb{B} is automatically monotone. \square

Proposition 5.4.2 *In Case 3 the mappings Add , Sub , Mul , Eq , Neq , Le and Ge are all continuous.*

Proof This is the same as the proof of Proposition 5.4.1. \square

Case operators: Let B_f denote the set of ground types $\theta \in B$ with K_θ finite. For each $\theta \in B_f$, $\beta \in B$ let $K_{\theta, \beta}$ be the S -typed set with underlying set K_θ in which κ has type $L \rightarrow \beta$ in $K_{\theta, \beta}$ whenever κ is of type $L \rightarrow \theta$ in Λ .

If $u \neq \perp_\theta$ then, since (D_B, f_K) is a regular extension of (X_B, p_K) , there exists a unique $\kappa \in K$ having type $L \rightarrow \theta$ for some $L \in \mathcal{F}_B$ and a unique $b \in D_\diamond^L$ such that $u = f_\kappa(b)$. A mapping $\text{Case}_\beta^\theta : D_\theta \times D_\diamond^{K_{\theta, \beta}} \rightarrow D_\beta$ can thus be defined by letting

$$\text{Case}_\beta^\theta(u, c) = \begin{cases} f_{\triangleleft\{L \rightarrow \beta, L\}}(c(\kappa) \triangleleft b) & \text{if } u \neq \perp_\theta \text{ and } u = f_\kappa(b) \\ & \text{with } \kappa \text{ of type } L \rightarrow \theta \text{ and } b \in D_\diamond^L, \\ \perp_\beta & \text{if } u = \perp_\theta, \end{cases}$$

noting that in the three basic Cases $f_{\triangleleft\{L \rightarrow \beta, L\}}(c(\kappa) \triangleleft b)$ is just equal to $c(\kappa)(b)$.

The device introduced in Section 5.2 will also be employed here to regard Case_β^θ as a function of a single argument: Put $\theta \cdot K_{\theta, \beta} = K_{\theta, \beta} \cup \{\diamond_\theta\}$, where \diamond_θ is some element not in K_θ ; consider $\theta \cdot K_{\theta, \beta}$ as an S -typed set with the typing induced from $K_{\theta, \beta}$ and with \diamond_θ of type θ . If $u \in D_\theta$ and $c \in D_\diamond^{K_{\theta, \beta}}$ then let $u \triangleleft c \in D_\diamond^{\theta \cdot K_{\theta, \beta}}$ be given by

$$(u \triangleleft c)(\eta) = \begin{cases} c(\eta) & \text{if } \eta \in K_{\theta, \beta}, \\ u & \text{if } \eta = \diamond_\theta. \end{cases}$$

Thus, exactly as in Lemma 5.2.1, each element of $D_\diamond^{\theta \cdot K_{\theta, \beta}}$ has a unique representation of the form $u \triangleleft c$ with $u \in D_\theta$ and $c \in D_\diamond^{K_{\theta, \beta}}$.

The functional Σ -algebra (D_S, f_N) is said to *support case operators* if for each $\theta \in B_f$ and each $\beta \in B$ there exists an element $\text{case}_\beta^\theta \in D_{L \rightarrow \beta}$ with $L = \theta \cdot K_{\theta, \beta}$ such that

$$f_{\triangleleft\{L \rightarrow \beta, L\}}(\text{case}_\beta^\theta \triangleleft (u \triangleleft c)) = \text{Case}_\beta^\theta(u, c)$$

for all $u \in D_\theta$, $c \in D_\diamond^{K_{\theta, \beta}}$. This just means that case_β^θ should be an element associated with the mapping $u \triangleleft c \mapsto \text{Case}_\beta^\theta(u, c)$.

In the three basic Cases (D_S, f_N) supports case operators. This is trivially true in Case 1, and the following two results show, again together with Lemma 5.4.1, that it is also true in Cases 2 and 3 (noting that the mapping $u \triangleleft c \mapsto (u, c)$ from $D_\diamond^{\theta \cdot K_{\theta, \beta}}$ to $D_\theta \times D_\diamond^{K_{\theta, \beta}}$ is an order isomorphism, and so is monotone in Case 2 and continuous in Case 3).

Proposition 5.4.3 *In Case 2 the mapping $\text{Case}_\beta^\theta : D_\theta \times D_\diamond^{K_{\theta, \beta}} \rightarrow D_\beta$ is monotone.*

Proof Let $u, u' \in D_\theta$ with $u \sqsubseteq_\theta u'$ and $c, c' \in D_\diamond^{K_{\theta, \beta}}$ with $c \sqsubseteq^{K_{\theta, \beta}} c'$ (i.e., with $c(\kappa) \sqsubseteq_{L \rightarrow \beta} c'(\kappa)$ whenever κ is of type $L \rightarrow \theta$). If $u = \perp_\theta$ then

$$\text{Case}_\beta^\theta(u, c) = \perp_\beta \sqsubseteq_\beta \text{Case}_\beta^\theta(u', c'),$$

and so assume that $u \neq \perp_\theta$ (and hence that $u' \neq \perp_\theta$). But by the definition of an associated ordering there then exists a unique κ of type $L \rightarrow \theta$ for some $L \in \mathcal{F}_B$ and unique elements $b, b' \in D_\diamond^L$ with $b \sqsubseteq_\diamond^L b'$ such that $u = f_\kappa(b)$ and $u' = f_\kappa(b')$. Thus

$$\text{Case}_\beta^\theta(u, c) = c(\kappa)(b) \sqsubseteq_\beta c(\kappa)(b') \sqsubseteq_\beta c'(\kappa)(b') = \text{Case}_\beta^\theta(u', c'),$$

i.e., the mapping Case_β^θ is monotone. \square

Proposition 5.4.4 *In Case 3 the mapping $\text{Case}_\beta^\theta : D_\theta \times D_\diamond^{K_{\theta, \beta}} \rightarrow D_\beta$ is continuous.*

Proof By Lemma 4.2.4 it is enough to show that Case_β^θ is separately continuous in each of its arguments, and it is clear that the mapping $c \mapsto \text{Case}_\beta^\theta(u, c)$ is continuous for each $u \in D_\theta$. It therefore remains to show that the mapping $u \mapsto \text{Case}_\beta^\theta(u, c)$ is continuous for each $c \in D_\diamond^{K_{\theta, \beta}}$. As in the proof of Proposition 5.4.3 this mapping is monotone, hence it must be shown that if $A \in \mathbf{d}(D_\theta)$ with $v = \bigsqcup A$ then

$$\text{Case}_\beta^\theta(v, c) = \bigsqcup \{ \text{Case}_\beta^\theta(u, c) : u \in A \},$$

and clearly it can be assumed here that $v \neq \perp_\theta$. There thus exists a unique $\kappa \in K$ having type $L \rightarrow \theta$ for some $L \in \mathcal{F}_B$ and a unique element $b \in D_\diamond^L$ such that $v = f_\kappa(b)$, and by the definition of an associated ordering it then follows that each element in $A \setminus \{\perp_\theta\}$ has a unique representation of the form $f_\kappa(b')$ with $b' \in D_\diamond^L$. Let

$$A' = \{ b' \in D_\diamond^L : f_\kappa(b') \in A \setminus \{\perp_\theta\} \};$$

then it is easily checked that $A' \in \mathbf{d}(D_\diamond^L)$ and hence

$$f_\kappa(\bigsqcup A') = \bigsqcup \{ f_\kappa(b') : b' \in A' \} = \bigsqcup (A \setminus \{\perp_\theta\}) = \bigsqcup A = v,$$

from which it follows that $b = \bigsqcup A'$. Therefore

$$\begin{aligned}
\sqcup \{ \text{Case}_\beta^\theta(u, c) : u \in A \} &= \sqcup \{ \text{Case}_\beta^\theta(u, c) : u \in A \setminus \{\perp_\theta\} \} \\
&= \sqcup \{ \text{Case}_\beta^\theta(f_\kappa(b'), c) : b' \in A' \} = \sqcup \{ c(\kappa)(b') : b' \in A' \} \\
&= c(\kappa)(b) = \text{Case}_\beta^\theta(v, c) .
\end{aligned}$$

This shows the mapping $\text{Case}_\beta^\theta : D_\theta \times D_\circ^{K_\theta, \beta} \rightarrow D_\beta$ is continuous. \square

Starting in Chapter 6 it will always be assumed that the Σ -algebra (D_S, f_N) supports both integer and case operators. This requirement is thus met in the three basic Cases, and below it will be shown that (D_S, f_N) also supports both integer and case operators in Case 2'.

The requirement that certain integer operators be ‘built-in’ is important if the equations are to be used as the basis for a practical programming language. In fact, since `int` is a primitive type with infinitely many constructor names, the only way to make use of this type is via the integer operators. However, it is essentially a requirement about one specific data type.

On the other hand, the inclusion of the ‘case’ operators is of a much more fundamental character and will facilitate in the present set-up what corresponds to pattern matching or to the ‘case’ construction in *Haskell*. These ‘case’ operators (or something equivalent) are absolutely essential: They are the analogue of the `if then else` construction occurring in all imperative programming languages and without them it is impossible to define any ‘non-trivial’ functions.

The choice of three arithmetical and four relational operators is, of course, somewhat arbitrary. There is no reason to prevent further operators being added, provided that each of these is the strict extension of a corresponding (total) operator with domain $\mathbb{Z} \times \mathbb{Z}$ and codomain either \mathbb{Z} or \mathbb{B} .

Now choose a set P to consist of the distinct elements

$$\{ \text{add, sub, mul, eq, neq, le, ge} \} \cup \{ \text{case}_\beta^\theta : \theta \in B_f, \beta \in B \}$$

and which is disjoint from K . Regard P as an S -typed set by stipulating that `add`, `sub` and `mul` be of type $\text{int}_2 \rightarrow \text{int}$, that `eq`, `neq`, `le` and `ge` be of type $\text{int}_2 \rightarrow \text{bool}$ and that case_β^θ be of type $\theta \cdot K_{\theta, \beta} \rightarrow \beta$ for each $\theta \in B_f, \beta \in B$.

Finally, let $p \in D_\circ^P$ be the assignment which assigns the corresponding ‘operator’ to each name of an integer operator (i.e., $p(\text{add}) = \text{add}$, $p(\text{sub}) = \text{sub}$ and so on) and for which $p(\text{case}_\beta^\theta) = \text{case}_\beta^\theta$ for each $\theta \in B_f, \beta \in B$.

Example 5.4.1 on the next page indicates how some of the primitive functions are involved in the equations introduced in Chapter 1.

Proposition 5.4.5 (D_S, f_N) supports both integer and case operators in Case 2'.

Example 5.4.1 Let (D_B, f_K) be some monotone regular bottomed extension of the Λ -algebra (X_B, p_K) introduced in Example 2.2.1 and let (D_S, f_N) be as in Case 1 with D_S the full functional extension of D_B .

Let $\text{ita} \in D_{\text{int int} \rightarrow \text{ipr}}$; then for each $p \in D_{\text{ipr}}$

$$\text{case}_{\text{ipr}}^{\text{ipr}}(p, \{\text{Pair} \rightarrow \text{ita}\}) = \begin{cases} \text{ita}(k, \ell) & \text{if } p \neq \perp_{\text{ipr}} \\ & \text{and } p = f_{\text{Pair}}(k, \ell), \\ \perp_{\text{ipr}} & \text{if } p = \perp_{\text{ipr}}. \end{cases}$$

This should be compared with the right-hand side of the equation for it in Example 2.1.1.

Let $\text{fsta} \in D_{\text{int int} \rightarrow \text{int}}$; then for each $p \in D_{\text{ipr}}$

$$\text{case}_{\text{int}}^{\text{ipr}}(p, \{\text{Pair} \rightarrow \text{fsta}\}) = \begin{cases} \text{fsta}(k, \ell) & \text{if } p \neq \perp_{\text{ipr}} \\ & \text{and } p = f_{\text{Pair}}(k, \ell), \\ \perp_{\text{int}} & \text{if } p = \perp_{\text{ipr}}. \end{cases}$$

This should be compared with the right-hand side of the equation for fst in Example 2.1.1.

Now if $b \in D_{\text{bool}}$ and $p_1, p_2 \in D_{\text{ipr}}$ then

$$\text{case}_{\text{ipr}}^{\text{bool}}(b, \{\text{True} \rightarrow p_1, \text{False} \rightarrow p_2\}) = \begin{cases} p_1 & \text{if } b = T, \\ p_2 & \text{if } b = F, \\ \perp_{\text{ipr}} & \text{if } b = \perp_{\text{bool}}. \end{cases}$$

Thus if $\text{pfb} \in D_{\text{int ipr} \rightarrow \text{ipr}}$ and $\text{it} \in D_{\text{ipr} \rightarrow \text{ipr}}$ then

$$\begin{aligned} \text{case}_{\text{ipr}}^{\text{bool}}(\text{eq}(n, 0), \{\text{True} \rightarrow p, \text{False} \rightarrow \text{pfb}(\text{sub}(n, 1), \text{it}(p))\}) \\ = \begin{cases} p & \text{if } n = 0, \\ \text{pfb}(n - 1, \text{it}(p)) & \text{if } n \in \mathbb{Z} \setminus \{0\}, \\ \perp_{\text{ipr}} & \text{if } n = \perp_{\text{int}}, \end{cases} \end{aligned}$$

for all $n \in D_{\text{int}}$, $p \in D_{\text{ipr}}$. This should be compared with the right-hand side of the equation for pfb in Example 2.1.1.

Proof Here (D_S, f_N) is the initial completion of the Σ -algebra (Y_S, q_N) associated with (Y_B, q_K) and Y_S , where (Y_B, q_K) is a minimal monotone regular extension of (X_B, p_K) and Y_S is the full monotone functional extension of Y_B . Note that by Proposition 4.3.7 $D_{\text{int}} = Y_{\text{int}}$ and $D_{\text{bool}} = Y_{\text{bool}}$, and this implies that the Case 2

integer operators can be used unchanged in Case 2'. For example, the mapping Add is the same in Cases 2 and 2', and in Case 2 the element $\text{add} \in Y_{\text{int}_2 \rightarrow \text{int}}$ satisfies

$$g_{\{\text{int}_2 \rightarrow \text{int}, \text{int}_2\}}(\text{add} \triangleleft (m, n)) = \text{Add}(m, n)$$

for all $m, n \in Y_{\text{int}}$; thus, regarding add as an element of $D_{\text{int}_2 \rightarrow \text{int}}$, it satisfies

$$f_{\{\text{int}_2 \rightarrow \text{int}, \text{int}_2\}}(\text{add} \triangleleft (m, n)) = \text{Add}(m, n)$$

for all $m, n \in D_{\text{int}}$, and so $\text{add} \in D_{\text{int}_2 \rightarrow \text{int}}$ is still associated with Add .

The case operators must now be considered. Let $\theta \in B_f$ and $\beta \in B$; then the mapping $\text{Case}_\beta^\theta : D_\theta \times D_\diamond^{K_{\theta, \beta}} \rightarrow D_\beta$ is clearly an extension of the corresponding mapping from $Y_\theta \times Y_\diamond^{K_{\theta, \beta}}$ to Y_β . Thus, regarding $\text{case}_\beta^\theta \in Y_{L \rightarrow \beta}$ (with $L = \theta \cdot K_{\theta, \beta}$) as an element of $D_{L \rightarrow \beta}$, it follows that

$$f_{\{L \rightarrow \beta, L\}}(\text{case}_\beta^\theta \triangleleft (u \triangleleft c)) = \text{Case}_\beta^\theta(u, c)$$

for all $u \in Y_\theta$, $c \in Y_\diamond^{K_{\theta, \beta}}$. Therefore this actually holds for all $u \in D_\theta$, $c \in D_\diamond^{K_{\theta, \beta}}$, since the proof of Proposition 5.4.4 shows that the mapping Case_β^θ is continuous. \square

5.5 Notes

Each functional type has the form $L \rightarrow \beta$ with $\beta \in B$, i.e., the result is always a ground type. It is possible to consider types of the form $L \rightarrow \tau$ with $\tau \in S \setminus B$ a functional type. However, there is then a problem of non-unique representation since, for example, if $\tau = \tau_1 \cdots \tau_n \rightarrow \beta$ then $\sigma_1 \cdots \sigma_m \rightarrow \tau$ and $\sigma_1 \cdots \sigma_m \tau_1 \cdots \tau_n \rightarrow \beta$ are (in any sensible interpretation) the same type.

What is here called a functional extension more-or-less corresponds to what in Mitchell (1990) is referred to as a *type frame*. The full functional extension then corresponds to the *full set-theoretic function hierarchy* and the full continuous functional extension to the *full continuous hierarchy*.

Chapter 6 Equations

The rest of the study will work within the framework introduced in Chapter 5. This means that (D_S, f_N) is a monotone regular functional extension of (X_B, p_K) which supports both integer and case operators.

For the three basic Cases recall that (D_S, f_N) is the functional Σ -algebra associated with (D_B, f_K) and D_S , where D_S is a functional extension of D_B supporting partial application and which was chosen as follows:

Case 1 (with (D_B, f_K) any monotone regular extension of (X_B, p_K)) Here D_S is the full functional extension of D_B , which trivially supports partial application. It is also clear that (D_S, f_N) supports both integer and case operators.

Case 2 (with (D_B, f_K) a minimal monotone regular extension of (X_B, p_K)) Here D_S is the full monotone functional extension of D_B , which by Proposition 5.1.5 supports partial application. Moreover, by Propositions 5.3.1 and 5.3.3 (D_S, f_N) supports both integer and case operators.

Case 3 (with (D_B, f_K) the initial completion of a minimal monotone regular extension of (X_B, p_K)) Here D_S is the full continuous functional extension of D_B , which by Proposition 5.1.6 supports partial application. Moreover, Propositions 5.3.2 and 5.3.4 imply that (D_S, f_N) supports both integer and case operators.

The present chapter explains what is meant by equations and what their solutions are. An algorithm for computing values will also be presented.

In Section 6.1 the basic terms algebras are introduced: This involves choosing for each S -typed set I disjoint from P a functionally $I \cup P$ -free Σ -algebra (F_S^I, \odot_N^I) which is an extension of the ground term algebra (F_B, \odot_K) (with P the S -typed set specifying the names and types of the primitive functions). If $c \in D_\circ^I$ then $\llbracket \cdot \rrbracket_S^c$ will be used to denote the unique Σ -homomorphism from (F_S^I, \odot_N^I) to (D_S, f_N) such that $\llbracket \xi \rrbracket_S^c = c(\xi)$ for each $\xi \in I$ and $\llbracket \zeta \rrbracket_S^c = p(\zeta)$ for each $\zeta \in P$ (with $p \in D_\circ^P$ the assignment which determines the meaning of the primitive functions). The homomorphism $\llbracket \cdot \rrbracket_S^c$ is regarded as giving a meaning, relative to the assignment c , to the terms in the family F_S^I : In other words, for each $s \in F_\sigma^I$ the value $\llbracket s \rrbracket_S^c$ is interpreted as the meaning of the term s , given that ξ has already been assigned the meaning $c(\xi)$ for each $\xi \in I$. In Lemma 6.1.1 it is shown that this Σ -homomorphism is an extension of the Λ -homomorphism $\llbracket \cdot \rrbracket_B$, i.e., $\llbracket s \rrbracket_\beta^c = \llbracket s \rrbracket_\beta$ for all $s \in F_\beta$, $\beta \in B$

Now although the choice has been made to work formally with the Σ -algebra (F_S^I, \odot_N^I) , continual use will be made of the fact that the associated $\Sigma^{I \cup P}$ -algebra is initial. This leads, in particular, to the unique representation for the elements in the family F_S^I given in Proposition 6.1.1, and which forms the basis of much of the subsequent discussion.

Let I be an S -typed set disjoint from P . In Section 6.1 a family ${}^H F_B^I \subset F_B^I$ is defined in terms of the trace R_B of (D_B, f_K) by putting ${}^H F_\beta^I = \{s \in F_\beta^I : \Omega_\beta^I(s) \in R_\beta\}$ for each $\beta \in B$, where $\Omega_B^I : (F_B^I, \odot_K^I) \rightarrow (F_B^b, \odot_K^b)$ is a certain Λ -homomorphism

whose definition only depends on Σ and I (and not on (D_S, f_N)). In fact, it is also the case that $HF_\beta^I = \{s \in F_\beta^I : \varrho_\beta^I(s) \neq \perp_\beta\}$ for each $\beta \in B$, where ϱ_S^I is the unique Σ -homomorphism from (F_S^I, \odot_N^I) to (D_S, f_N) such that $\varrho_\xi^I(\xi) = \perp_\xi$ for each $\xi \in I \cup P$. The family HF_B^I plays an important role in the algorithm to be defined in Section 6.5 because it has the property that $\llbracket s \rrbracket_\beta^c \neq \perp_\beta$ for all $s \in HF_\beta^I$, $c \in D_\diamond^I$.

At the end of Section 6.1 a functionally J -free Σ -algebra is explicitly constructed for each S -typed set J ; this is essentially a term algebra as defined in Section 2.4 and is an extension of the explicit term algebra introduced in Section 3.1. These algebras (with sets J of the form $I \cup P$) then provide the basis for a rudimentary functional programming language.

Equations are introduced in Section 6.2. Let I be a non-empty S -typed set disjoint from P . Consider $\xi \in I$ of type $\sigma = L \rightarrow \beta$ and assume that L is disjoint from I and P . Then $F_\beta^{I \cup L}$ is regarded as the set of equations (better, as the set of right-hand sides of equations) for ξ and this set will be denoted by $\text{Eq}(\xi, I)$. The set of solutions $\text{Sol}(s)$ of an equation $s \in \text{Eq}(\xi, I)$ is the set of assignments $c \in D_\diamond^I$ for which

$$f_{\triangleleft\{\sigma, L\}}(c(\xi) \triangleleft b) = \llbracket s \rrbracket_\beta^{c \oplus b}$$

for all $b \in D_\diamond^L$ if $L \neq \emptyset$, and for which $c(\xi) = \llbracket s \rrbracket_\beta^c$ if $L = \emptyset$.

A system of equations then consists of a family of equations, one for each name in I , i.e., a family of the form s_I , where $s_\xi \in \text{Eq}(\xi, I)$ for each $\xi \in I$, and the set of all such families is denoted by $\text{Eq}(I)$. If $\alpha = s_I \in \text{Eq}(I)$ is a system of equations then an assignment $c \in D_\diamond^I$ is said to be a solution of the equations α if $c \in \text{Sol}(s_\xi)$ for each $\xi \in I$; the set of solutions will be denoted by $\text{Sol}(\alpha)$.

Starting in Section 6.3 equations are considered as replacement rules and from this point of view an algorithm is developed for computing values. In Section 6.3 itself it is explained how in general equations can be used as replacement rules in order to make valid assertions about their solutions. Then in Section 6.4 the problem of computing values is looked at, which means the following: Let $\alpha \in \text{Eq}(I)$ be a system of equations for which $\text{Sol}(\alpha) \neq \emptyset$ (and in general it is possible for a system of equations to have no solution) and consider the family F_B^α with $F_\beta^\alpha \subset F_B^I$ defined by

$$F_\beta^\alpha = \{s \in F_\beta^I : \text{there exists } x \in X_\beta \text{ such that } \llbracket s \rrbracket_\beta^c = x \text{ for all } c \in \text{Sol}(\alpha)\}$$

for each $\beta \in B$. (In particular, $F_B \subset F_B^\alpha$, since if $s \in F_\beta$ then Lemma 6.1.1 implies that $\llbracket s \rrbracket_\beta^c = \llbracket s \rrbracket_\beta$ for all $c \in D_\diamond^I$ and so in particular for $c \in \text{Sol}(\alpha)$.) It is then asked for which elements $s \in F_\beta^\alpha$ is it possible to compute the value of s , i.e., the unique element $x \in X_\beta$ such that $\llbracket s \rrbracket_\beta^c = x$ for all $c \in \text{Sol}(\alpha)$?

In Section 6.5 the basic algorithm for computing values is introduced. This is defined in terms of a Λ -homomorphism $\Phi_B : (F_B^I, \odot_K^I) \rightarrow (F_B^I, \odot_K^I)$. The homomorphism Φ_B depends, of course, on the system of equations $\alpha \in \text{Eq}(I)$ under consideration. It also depends on (D_B, f_K) via the trace R_B , and this dependence enters because the family HF_B^I defined in Section 6.1 is involved in the definition of Φ_B .

6.1 The basic term algebras

This section introduces the basic terms algebras. More precisely, this will mean that for each S -typed set I disjoint from P a suitable functionally $I \cup P$ -free Σ -algebra (F_S^I, \odot_N^I) is chosen in order to give a meaning to ‘expressions’ or ‘terms’ involving the names in I . (Recall that P is the S -typed set specifying the names and types of the primitive functions).

The reasons for working with functionally free algebras were explained in Section 5.3, and continual use will be made of the fact that the $\Sigma^{I \cup P}$ -algebra associated with (F_S^I, \odot_N^I) is initial.

Assume (without any real loss of generality) that the ground term algebra (F_B, \odot_K) is disjoint from every S -typed set. By Proposition 5.3.6 it can then be supposed that for each S -typed set I disjoint from P a functionally $I \cup P$ -free Σ -algebra (F_S^I, \odot_N^I) has been chosen which is an extension of (F_B, \odot_K) . These Σ -algebras are referred to as the *basic term algebras*. Later in the section a specific choice of these algebras is made to serve as a basis for an explicit functional programming language.

For each $c \in D_\diamond^I$ denote by $\llbracket \cdot \rrbracket_S^c$ the unique Σ -homomorphism from (F_S^I, \odot_N^I) to the functional Σ -algebra (D_S, f_N) such that $\llbracket \xi \rrbracket_\xi^c = c(\xi)$ for each $\xi \in I$ and $\llbracket \zeta \rrbracket_\zeta^c = p(\zeta)$ for each $\zeta \in P$ (with $p \in D_\diamond^P$ the assignment which determines the meaning of the primitive functions). As already stated several times before, $\llbracket \cdot \rrbracket_S^c$ is considered as giving a meaning, relative to the assignment c , to the terms in the family F_S^I : In other words, for each $s \in F_\sigma^I$ the ‘value’ $\llbracket s \rrbracket_\sigma^c$ is interpreted as being the meaning of the term s , given that ξ has already been assigned the meaning $c(\xi)$ for each $\xi \in I$.

Note that the unique Λ -isomorphism $\llbracket \cdot \rrbracket_B$ from (F_B, \odot_K) to (X_B, p_K) is also the unique Λ -homomorphism from (F_B, \odot_K) to (D_B, f_K) .

Lemma 6.1.1 *For each $c \in D_\diamond^I$ the Σ -homomorphism $\llbracket \cdot \rrbracket_S^c$ is an extension of the Λ -homomorphism $\llbracket \cdot \rrbracket_B$, i.e., $\llbracket s \rrbracket_\beta^c = \llbracket s \rrbracket_\beta$ for all $s \in F_\beta$, $\beta \in B$.*

Proof For each $\beta \in B$ let $\llbracket \cdot \rrbracket'_\beta$ be the restriction of $\llbracket \cdot \rrbracket_\beta^c$ to F_β . Then $\llbracket \cdot \rrbracket'_\beta$ is also a Λ -homomorphism from (F_B, \odot_K) to (D_B, f_K) and thus $\llbracket \cdot \rrbracket'_\beta = \llbracket \cdot \rrbracket_B$, since $\llbracket \cdot \rrbracket_B$ is the unique such homomorphism, i.e., $\llbracket \cdot \rrbracket_S^c$ is an extension of $\llbracket \cdot \rrbracket_B$. \square

Let I be an S -typed set disjoint from P which is considered to be fixed in what follows. It is useful to make the following classification:

- For each $\beta \in B$ denote by κF_β^I the set of all elements in F_β^I having the form $\odot_\kappa^I(b)$ with $\kappa \in K$ of type $L \rightarrow \beta$ for some $L \in \mathcal{F}_S$ and with $b \in (F_\diamond^I)^L$.
- For each $\sigma = L \rightarrow \beta \in S$ denote by ${}_A F_\sigma^I$ the set of all elements in F_σ^I having the form $\odot_{\triangleleft\{\tau, J\}}^I(\xi \triangleleft b)$ with $\xi \in I \cup P$ having functional type $\tau = J \cup L \rightarrow \beta$ for some $J \in \mathcal{F}_S^o$ disjoint from L and with $b \in (F_\diamond^I)^J$.

If J is an S -typed set then as usual put $J_\sigma = \{\eta \in J : \langle \eta \rangle = \sigma\}$ for each $\sigma \in S$. The unique representations appearing in the next result will play a decisive role.

Proposition 6.1.1 (1) For each $\beta \in B$ the sets κF_β^I , AF_β^I and I_β form a partition of F_β^I . Furthermore, for each $\beta \in B$ the following statements hold:

- If $s \in \kappa F_\beta^I$ then there exists a unique $\kappa \in K$ having type $L \rightarrow \beta$ for some $L \in \mathcal{F}_S$ and a unique element $b \in (F_\diamond^I)^L$ such that $s = \odot_\kappa^I(b)$.
- If $s \in AF_\beta^I$ then there exists a unique $\xi \in I \cup P$ having functional type $\sigma = L \rightarrow \beta$ for some $L \in \mathcal{F}_S^o$ and a unique element $b \in (F_\diamond^I)^L$ such that $s = \odot_{\triangleleft\{\sigma, L\}}^I(\xi \triangleleft b)$.

(2) For each $\sigma \in S \setminus B$ the sets AF_σ^I , I_σ and P_σ form a partition of F_σ^I . Moreover, if $\sigma = L \rightarrow \beta \in S \setminus B$ and $s \in AF_\sigma^I$ then there exists a unique $\xi \in I \cup P$ having type $\tau = J \cup L \rightarrow \beta$ for some $J \in \mathcal{F}_S^o$ disjoint from L and a unique assignment $b \in (F_\diamond^I)^J$ such that $s = \odot_{\triangleleft\{\tau, J\}}^I(\xi \triangleleft b)$.

Proof This follows since Proposition 5.3.4 states that the $\Sigma^{I \cup P}$ -algebra associated with (F_S^I, \odot_N^I) is initial, and so in particular regular. (Note also that $P_\beta = \emptyset$ for each $\beta \in B$.) \square

Because of Proposition 6.1.1 it is convenient to work with the operator names from the $\Sigma^{I \cup P}$ -algebra associated with (F_S^I, \odot_N^I) . This means that if $\xi \in I \cup P$ is of type $\sigma = L \rightarrow \beta$ and J is a non-empty subset of L then $\odot_{\xi\{J\}}^I$ will be used to denote the mapping from $(F_\diamond^I)^J$ to $F_{L \setminus J \rightarrow \beta}^I$ given by

$$\odot_{\xi\{J\}}^I(b) = \odot_{\triangleleft\{\sigma, J\}}^I(\xi \triangleleft b)$$

for each $b \in (F_\diamond^I)^J$, and in the same way $\odot_{\xi\{\emptyset\}}^I$ will be used to denote the mapping from \mathbb{I} to F_σ^I given by $\odot_{\xi\{\emptyset\}}^I(\varepsilon) = \xi$.

Moreover, if $\xi \in I \cup P$ is of type $\sigma = L \rightarrow \beta$ then it is also convenient to allow \odot_ξ^I as a more legible alternative to the notation $\odot_{\xi\{L\}}^I$; thus $\odot_\xi^I : (F_\diamond^J)^L \rightarrow F_\beta^I$ is the mapping defined by $\odot_\xi^J(\varepsilon) = \xi$ when $L = \emptyset$ and by

$$\odot_\xi^J(b) = \odot_{\triangleleft\{\sigma, L\}}^J(\xi \triangleleft b)$$

for each $b \in (F_\diamond^J)^L$ when $L \neq \emptyset$. One reason for introducing this last notation is because if $\beta \in B$ and $s \in F_\beta^I$ then Proposition 6.1.1 (1) implies that there exists a unique element $\lambda \in K \cup I \cup P$ having type $L \rightarrow \beta$ for some $L \in \mathcal{F}_S$ and a unique assignment $b \in (F_\diamond^I)^L$ such that $s = \odot_\lambda^I(b)$.

Note that if $\xi \in I$ is of type $\sigma = L \rightarrow \beta$ with $L \neq \emptyset$ and $b \in (F_\diamond^I)^L$ then

$$\llbracket \odot_\xi^I(b) \rrbracket_\beta^c = f_{\triangleleft\{\sigma, L\}}(c(\xi) \triangleleft (\llbracket b \rrbracket_\diamond^c)^L)$$

for each $c \in D_\diamond^I$. In the same way, if $\zeta \in P$ is of type $\sigma = L \rightarrow \beta$ and $b \in (F_\diamond^I)^L$ then

$$\llbracket \odot_\zeta^I(b) \rrbracket_\beta^c = f_{\triangleleft\{\sigma, L\}}(p(\zeta) \triangleleft (\llbracket b \rrbracket_\diamond^c)^L)$$

for each $c \in D_\diamond^I$. In the rest of the study Λ -algebra (F_B^I, \odot_K^I) will usually play a more important role than the Σ -algebra (F_S^I, \odot_N^I) itself, and then the following fact is often useful:

Lemma 6.1.2 F_B^I is the only invariant family \hat{F}_B^I in the Λ -algebra (F_B^I, \odot_K^I) such that $AF_\beta^I \cup I_\beta \subset \hat{F}_\beta^I$ for each $\beta \in B$.

Proof Let \hat{F}_B^I be invariant in (F_B^I, \odot_K^I) and such that $AF_\beta^I \cup I_\beta \subset \hat{F}_\beta^I$ for each $\beta \in B$. For each $\sigma \in S \setminus B$ put $\hat{F}_\sigma^I = F_\sigma^I$; then the family \hat{F}_S^I is clearly invariant in the $\Sigma^{I \cup P}$ -algebra $(F_S^I, \odot_{N^{I \cup P}}^I)$. Hence $\hat{F}_S^I = F_S^I$, since this $\Sigma^{I \cup P}$ -algebra is initial and thus minimal. In particular $\hat{F}_B^I = F_B^I$. \square

Recall how the bottomed ground term algebra (F_B^b, \odot_K^b) was introduced in Section 3.3. This is an initial bottomed extension of (F_B, \odot_K) , (but with the bottom element of F_β^b being denoted by b_β rather than \perp_β). The unique bottomed homomorphism from (F_B^b, \odot_K^b) to (D_B, f_K) (i.e., the unique homomorphism with $\llbracket b_\beta \rrbracket_\beta^\perp = \perp_\beta$ for each $\beta \in B$) is denoted by $\llbracket \cdot \rrbracket_B^\perp$. By Proposition 2.5.3 $\llbracket \cdot \rrbracket_B^\perp$ is an extension of $\llbracket \cdot \rrbracket_B$, i.e., $\llbracket s \rrbracket_\beta^\perp = \llbracket s \rrbracket_\beta$ for all $s \in F_\beta$, $\beta \in B$.

The Λ -algebra (F_B^b, \odot_K^b) can be extended in a trivial way to a bottomed Σ -algebra (F_S^b, \odot_N^b) : Let F_σ^b consist of a single element b_σ for each $\sigma \in S \setminus B$ (which is considered to be the bottom element of F_σ^b), and if $\sigma = L \rightarrow \beta$ and J is a non-empty subset of L then define $\odot_{\triangleleft\{\sigma, J\}}^b : (F_\sigma^b)^J \rightarrow F_{L \setminus J \rightarrow \beta}^b$ simply to be the mapping given by

$$\odot_{\triangleleft\{\sigma, J\}}^b(c) = b_{L \setminus J \rightarrow \beta}$$

for all $c \in (F_\sigma^b)^J$.

Lemma 6.1.3 (F_S^b, \odot_N^b) is a functional Σ -algebra.

Proof This follows since if $L \rightarrow \beta$ is a functional type, J and J' are non-empty disjoint subsets of L and $s \in F_{L \rightarrow \beta}^b$ then both the terms $\odot_{\triangleleft\{L \rightarrow \beta, J \cup J'\}}^b(s \triangleleft (c \oplus c'))$ and $\odot_{\triangleleft\{L \setminus J \rightarrow \beta, J'\}}^b(\odot_{\triangleleft\{L \rightarrow \beta, J\}}^b(s \triangleleft c) \triangleleft c')$ are equal to $b_{L \setminus (J \cup J') \rightarrow \beta}$ for each $c \in (F_\sigma^b)^J$, $c' \in (F_\sigma^b)^{J'}$. \square

The Λ -homomorphism $\llbracket \cdot \rrbracket_B^\perp$ will also be extended to a family $\llbracket \cdot \rrbracket_S^\perp$ by letting $\llbracket \cdot \rrbracket_\sigma^\perp : F_\sigma^b \rightarrow D_\sigma$ be the mapping with $\llbracket b_\sigma \rrbracket_\sigma^\perp = \perp_\sigma$ for each $\sigma \in S \setminus B$.

Lemma 6.1.4 $\llbracket \cdot \rrbracket_S^\perp$ is a Σ -homomorphism, and in fact it is the unique bottomed Σ -homomorphism from (F_S^b, \odot_N^b) to (D_S, f_N) .

Proof Consider a functional type $\sigma = L \rightarrow \beta$ and a non-empty subset J of L ; put $\tau = L \setminus J \rightarrow \beta$. Then for all $s \in F_\sigma^b$, $c \in (F_\sigma^b)^J$

$$\begin{aligned} f_{\triangleleft\{\sigma, J\}}(\llbracket s \triangleleft c \rrbracket_\sigma^\perp)^{\sigma \cdot J} &= f_{\triangleleft\{\sigma, J\}}(\llbracket s \rrbracket_\sigma^\perp \triangleleft (\llbracket c \rrbracket_\sigma^\perp)^J) \\ &= f_{\triangleleft\{\sigma, J\}}(\perp_\sigma \triangleleft (\llbracket c \rrbracket_\sigma^\perp)^J) = \perp_\tau = \llbracket b_\tau \rrbracket_\tau^\perp = \llbracket \odot_{\triangleleft\{\sigma, J\}}^b(s \triangleleft c) \rrbracket_\tau^\perp. \end{aligned}$$

On the other hand, if $\kappa \in K$ is of type $L \rightarrow \beta$ then $\llbracket \odot_\kappa^b(c) \rrbracket_\beta^\perp = f_\kappa(\llbracket c \rrbracket_\sigma^\perp)^L$ holds for all $c \in (F_\sigma^b)^L$ because $\llbracket \cdot \rrbracket_B^\perp$ is a Λ -homomorphism. This implies that $\llbracket \cdot \rrbracket_S^\perp$

is a Σ -homomorphism, which by definition is bottomed. The uniqueness now follows from the fact that $\llbracket \cdot \rrbracket_B^\perp$ is the unique bottomed Λ -homomorphism from $(F_B^\flat, \odot_K^\flat)$ to (D_B, f_K) . \square

Next, a particular Σ -homomorphism $\Omega_S^I : (F_S^I, \odot_N^I) \rightarrow (F_S^\flat, \odot_N^\flat)$ will be introduced, but before doing this note the following simple fact:

Lemma 6.1.5 *Let $\pi_S : (F_S^I, \odot_N^I) \rightarrow (F_S^\flat, \odot_N^\flat)$ be any Σ -homomorphism. Then:*

- (1) $\pi_\beta(s) = \flat_\beta$ for all $s \in AF_\beta^I$, $\beta \in B$.
- (2) $\pi_\beta(s) = s$ for all ground terms $s \in F_\beta$, $\beta \in B$.

Proof (1) This follows since if $\xi \in I \cup P$ is of functional type $\sigma = L \rightarrow \beta$ and $b \in (F_\diamond^I)^L$ then $\pi_\beta(\odot_{\triangleleft\{\sigma, L\}}^I(\xi \triangleleft b))$ has the form $\odot_{\triangleleft\{\sigma, L\}}^\flat(a)$ for some $a \in (F_\diamond^\flat)^{\sigma \cdot L}$, which is by definition is equal to \flat_β .

(2) This is a special case of Lemma 3.2.1. \square

Now (F_S^I, \odot_N^I) is functionally $I \cup P$ -free and $(F_S^\flat, \odot_N^\flat)$ is a functional Σ -algebra; there therefore exists a unique Σ -homomorphism $\Omega_S^I : (F_S^I, \odot_N^I) \rightarrow (F_S^\flat, \odot_N^\flat)$ such that $\Omega_\xi^I(\xi) = \flat_\xi$ for each $\xi \in I \cup P$. By Lemma 6.1.5 it then follows automatically that $\Omega_\beta^I(s) = \flat_\beta$ for all $s \in AF_\beta^I$ and $\Omega_\beta^I(s) = s$ for all ground terms $s \in F_\beta$, $\beta \in B$. In addition, Ω_S^I has the following property:

Lemma 6.1.6 $\{s \in F_\beta^I : \Omega_\beta^I(s) \in F_\beta\} = F_\beta$ for each $\beta \in B$.

Proof Lemma 6.1.5 (2) implies that $\Omega_\beta^I(s) \in F_\beta$ for all $s \in F_\beta$, $\beta \in B$. The proof of the converse is essentially the same as the proof of Proposition 3.3.3. Consider the family \hat{F}_B^I with $\hat{F}_B^I \subset F_B^I$ given by

$$\hat{F}_\beta^I = F_\beta \cup \{s \in F_\beta^I \setminus F_\beta : \Omega_\beta^I(s) \in F_\beta^\flat \setminus F_\beta\}$$

for each $\beta \in B$. If $\kappa \in K$ is of type $\emptyset \rightarrow \beta$ then $\odot_\kappa^I(\varepsilon) = \odot_\kappa(\varepsilon) \in F_\beta \subset \hat{F}_\beta^I$. Now let $\kappa \in K$ be of type $L \rightarrow \beta$ with $L \neq \emptyset$ and let $a \in (\hat{F}_\diamond^I)^L$, i.e., $a(\eta) \in \hat{F}_\eta^I$ for each $\eta \in L$. Suppose $\Omega_\beta^I(\odot_\kappa^I(a)) = \odot_\kappa^\flat((\Omega_\diamond^I)^L(a)) \in F_\beta$. Then by Proposition 3.2.6 $(\Omega_\diamond^I)^L(a) \in F_\diamond^L$ (since $(F_B^\flat, \odot_K^\flat)$ is an initial bottomed extension of (F_B, \odot_K)), i.e., $\Omega_\eta^I(a(\eta)) \in F_\eta$ for each $\eta \in L$. Then by assumption $a(\eta) \in F_\eta$ for each $\eta \in L$ and thus $\odot_\kappa^I(a) = \odot_\kappa(a) \in F_\beta$. This shows that $\odot_\kappa^I(a) \in \hat{F}_\beta^I$, and hence that the family \hat{F}_B^I is invariant in (F_B^I, \odot_K^I) . But by definition $I_\beta \subset \hat{F}_\beta^I$ and by Lemma 6.1.5 (2) $AF_\beta^I \subset \hat{F}_\beta^I$ for each $\beta \in B$. Therefore by Lemma 6.1.2 $\hat{F}_B^I = F_B^I$, which of course implies that $\Omega_\beta^I(s) \in F_\beta^\flat \setminus F_\beta$ for all $s \in F_\beta^I \setminus F_\beta$, $\beta \in B$. \square

It is important to note that Ω_S^I has nothing to do with the Σ -algebra (D_S, f_N) . In fact, in the sense in which a functionally $I \cup P$ -free Σ -algebra is unique, it can be said that Ω_S^I is uniquely determined by the signature Σ and the S -typed sets I and P .

For the rest of the study denote by ϱ_S^I the unique Σ -homomorphism from (F_S^I, \odot_N^I) to (D_S, f_N) such that $\varrho_\xi^I(\xi) = \perp_\xi$ for each $\xi \in I \cup P$.

Proposition 6.1.2 $\llbracket \cdot \rrbracket_S^\perp \Omega_S^I = \varrho_S^I$, i.e., $\llbracket \Omega^I(s) \rrbracket_\sigma^\perp = \varrho_\sigma^I(s)$ for all $s \in F_\sigma^I$, $\sigma \in S$.

Proof By Proposition 2.2.1 $\omega_S^I = \llbracket \cdot \rrbracket_S^\perp \Omega_S^I$ a Σ -homomorphism from (F_S^I, \odot_N^I) to (D_S, f_N) and $\omega_\xi^I(\xi) = \llbracket \Omega^I(\xi) \rrbracket_\xi^\perp = \llbracket \flat_\xi \rrbracket_\xi^\perp = \perp_\xi$ for each $\xi \in I \cup P$. Thus by the uniqueness of ϱ_S^I it follows that $\omega_S^I = \varrho_S^I$. \square

Now define a family HF_B^I with $HF_B^I \subset F_B^I$ by putting

$$HF_\beta^I = \{s \in F_\beta^I : \Omega_\beta^I(s) \in R_\beta\}$$

for each $\beta \in B$, where R_β is the trace of (D_B, f_K) , (and recall that this is defined by $R_\beta = \{s \in F_\beta^b : \llbracket s \rrbracket_\beta^\perp \neq \perp_\beta\}$ for each $\beta \in B$). Thus HF_B^I depends on (D_B, f_K) (but not on the whole of (D_S, f_N)), and it only depends on (D_B, f_K) via the trace R_B . Of course, Proposition 6.1.2 implies that also $HF_\beta^I = \{s \in F_\beta^I : \varrho_\beta^I(s) \neq \perp_\beta\}$ for each $\beta \in B$.

Proposition 6.1.3 $F_B \subset HF_B^I \subset \kappa F_B^I$. Moreover, if (D_B, f_K) is the flat extension of (X_B, p_K) then $HF_B^I = F_B$, and if (D_B, f_K) is a fully regular bottomed extension of (X_B, p_K) then $HF_B^I = \kappa F_B^I$.

Proof The first statement follows from Proposition 6.1.1 and Lemma 6.1.5, because $F_\beta \subset R_\beta \subset F_\beta^b \setminus \{\flat_\beta\}$ for each $\beta \in B$. The rest then follows from Lemma 6.1.6, since if (D_B, f_K) is the flat extension of (X_B, p_K) then by Lemma 3.3.2 $R_B = F_B$, and if (D_B, f_K) is a fully regular bottomed extension then Proposition 3.3.2 (2) implies that $R_\beta = F_\beta^b \setminus \{\flat_\beta\}$ for each $\beta \in B$. \square

The first statement in Proposition 6.1.3 implies that $HF_\beta^I = F_\beta$ for each primitive type $\beta \in B$ (since then $\kappa F_\beta^I = F_\beta$). Thus in particular $HF_{\text{int}}^I = F_{\text{int}}$.

In the algorithm for computing values to be presented in Section 6.5 the family HF_B^I will play an important role, and the reason is because of Proposition 6.1.4 below. Note that, except for the construction of the initial completion in Chapter 4, the proof of this result is the first time where the assumption that the bottomed extension (D_B, f_K) is monotone is really necessary.

Proposition 6.1.4 If $s \in HF_\beta^I$ then $\llbracket s \rrbracket_\beta^c \neq \perp_\beta$ for all $c \in D_\diamond^I$.

Proof The following lemma allows the assumption that (D_B, f_K) is monotone to be used. Note that the family ϱ_B^I (obtained by omitting the mappings $\{\varrho_\sigma^I : \sigma \in S \setminus B\}$ from the family ϱ_S^I) is then a Λ -homomorphism from (F_B^I, \odot_K^I) to (D_B, f_K) .

Lemma 6.1.7 F_B^I is the only invariant family in (F_B^I, \odot_K^I) containing U_B , where $U_B = \{s \in F_B^I : \varrho_\beta^I(s) = \perp_\beta\}$ for each $\beta \in B$; in other words, the Λ -homomorphism ϱ_B^I is fat bottomed.

Proof If $\xi \in I \cup P$ has functional type $\sigma = L \rightarrow \beta$ then

$$\varrho_\beta^I(\odot_\xi^I(b)) = f_{\triangleleft\{\sigma, L\}}(\perp^I(\xi) \triangleleft (\varrho_\sigma^I)^L(b)) = f_{\triangleleft\{\sigma, L\}}(\perp_\sigma \triangleleft (\varrho_\sigma^I)^L(b)) = \perp_\beta$$

for all $b \in (F_\sigma^I)^L$. Moreover, by definition $\varrho_\beta^I(\xi) = \perp_\beta$ for each $\xi \in I_\beta$, and hence $AF_B^I \cup I_\beta \subset U_\beta$ for each $\beta \in B$. Thus by Lemma 6.1.2 F_B^I is the only invariant family in (F_B^I, \odot_K^I) containing U_B . \square

Proof of Proposition 6.1.4 If $s \in HF_\beta^I$ then by Proposition 6.1.2 $\varrho_\beta^I(s) \neq \perp_\beta$. But by Lemma 6.1.7 the Λ -homomorphism ϱ_B^I is fat bottomed and by Proposition 3.4.6 (D_B, f_K) is strongly monotone. Therefore $\pi_\beta(s) \neq \perp_\beta$ for every Λ -homomorphism $\pi_B : (F_B^I, \odot_K^I) \rightarrow (D_B, f_K)$, and so in particular $\llbracket s \rrbracket_\beta^c \neq \perp_\beta$ for each $c \in D_\sigma^I$. \square

The final topic considered in this section is the explicit construction of a functionally J -free Σ -algebra $(\check{E}_S^J, \check{\square}_N^J)$ for each S -typed set J . Each of these algebras involves a term algebra as defined in Section 2.4 and is an extension of the term algebra (E_B, \square_K) introduced in Section 3.1. If I is an S -typed set disjoint from P then the functionally $I \cup P$ -free Σ -algebra $(\check{E}_S^{I \cup P}, \check{\square}_N^{I \cup P})$ will be denoted just by (E_S^I, \square_N^I) . These algebras will be used as the basis for an explicit functional programming language.

Consider the S -typed set J to be fixed. The functionally J -free Σ -algebra $(\check{E}_S^J, \check{\square}_N^J)$ will be obtained with the help of Proposition 5.3.3, which means the first task is to construct an initial Σ^J -algebra $(\check{E}_S^J, \check{\square}_{N^J}^J)$ adapted to J .

Recall that in Section 3.1 (E_B, \square_K) was defined to be the standard term Λ -algebra defined by some family of enumerations i_K (which means that if κ is of type $L \rightarrow \beta$ then i_κ is a bijective mapping from $[m]$ to the set L , where $m = |L|$ is the cardinality of L). This family i_K must now be extended to a family of enumerations $i_{N^J}^J$ for the signature Σ^J : For each $\kappa \in K$ let $i_\kappa^J = i_\kappa$, and for each $\xi \in J$ of type $L \rightarrow \beta$ and each subset V of L choose a bijective mapping $i_{\xi\{V\}}^J : [m] \rightarrow V$, where $m = |V|$. Without loss of generality it can be assumed that K is disjoint from J , and hence a term algebra specifier $\Gamma^J : N^J \rightarrow K \cup J$ can then be defined by letting

$$\Gamma^J(\nu) = \begin{cases} \kappa & \text{if } \nu = \kappa \in K, \\ \xi & \text{if } \nu = \xi\{V\} \in N^J \setminus K. \end{cases}$$

Let $(\check{E}_S^J, \check{\square}_{N^J}^J)$ be the term Σ^J -algebra specified by Γ^J and the family of enumerations $i_{N^J}^J$. Thus $\check{E}_\sigma^J \subset (K \cup J)^*$ for each $\sigma \in S$ and the family \check{E}_S^J can be regarded as being defined by the following rules:

- (i) If $\kappa \in K$ is of type $\emptyset \rightarrow \beta$ then the list consisting of the single component κ is an element of \check{E}_β^J .
- (ii) If $\kappa \in K$ is of type $L \rightarrow \beta$ with $L \neq \emptyset$ and $e_j \in \check{E}_{\beta_j}^J$ for $j = 1, \dots, m$, where $\beta_j = \langle i_\kappa(j) \rangle$ and $m = |L|$, then $\kappa e_1 \cdots e_m$ is an element of \check{E}_β^J .
- (iii) If $\xi \in J$ is of type σ then the list consisting of the single component ξ is an element of \check{E}_σ^J .
- (iv) If $\xi \in J$ is of type $L \rightarrow \beta$, V is a non-empty subset of L and $e_j \in \check{E}_{\sigma_j}^J$ for $j = 1, \dots, m$, where $\sigma_j = \langle i_{\xi\{V\}}^J(j) \rangle$ and $m = |V|$, then $\xi e_1 \cdots e_m$ is an element of \check{E}_σ^J with $\sigma = L \setminus V \rightarrow \beta$.
- (v) The only elements in \check{E}_σ^J are those which can be obtained using (i), (ii), (iii) and (iv).

Proposition 6.1.5 $(\check{E}_S^J, \check{\square}_{NJ}^J)$ is an initial Σ^J -algebra adapted to J . Moreover, it is an extension of (E_B, \square_K) .

Proof If $\sigma = L \rightarrow \beta$ and $\xi\{V\} \in N_\sigma^J$ then the sets V and L are disjoint and ξ is of type $V \cup L \rightarrow \beta$. This implies that the restriction of Γ^J to N_σ^J is injective for each $\sigma \in S$ and hence by Proposition 2.4.5 $(\check{E}_S^J, \check{\square}_{NJ}^J)$ is an initial Σ^J -algebra. Moreover, by definition $\check{\square}_{\xi\{\emptyset\}}^J(\varepsilon) = \xi$ for each $\xi \in J$ (recalling that each set Z is considered to be a subset of Z^* by identifying $z \in Z$ with the list whose single component is equal to z) and therefore $(\check{E}_S^J, \check{\square}_{NJ}^J)$ is adapted to J . Finally, Proposition 2.5.5 implies that $(\check{E}_S^J, \check{\square}_{NJ}^J)$ is an extension of (E_B, \square_K) , since Σ^J is an extension of the signature Λ and the family of enumerations i_{NJ}^J is an extension of the family i_K . \square

Now let $(\check{E}_S^J, \check{\square}_N^J)$ be the functional Σ -algebra associated with $(\check{E}_S^J, \check{\square}_{NJ}^J)$. Then by Proposition 5.3.3 $(\check{E}_S^J, \check{\square}_N^J)$ is functionally J -free, and clearly it is also an extension of (E_B, \square_K) .

As already stated, these algebras (at least for sets J of the form $I \cup P$) will provide the basis for a rudimentary functional programming language.

Some care must be taken with the operations in the Σ -algebra $(\check{E}_S^J, \check{\square}_N^J)$: Suppose, for example, that $\xi \in J$ is of type $L \rightarrow \beta$, where $L = \{\eta_1, \eta_2, \eta_3, \eta_4\}$ and for $j = 1, 2, 3, 4$ let $e_j \in \check{E}_{\eta_j}^J$. Consider $V = \{\eta_2, \eta_4\}$ and suppose that $i_{\xi\{L\}}^J(j) = \eta_j$ for each j and that $i_{\xi\{V\}}^J(1) = \eta_2$ and $i_{\xi\{V\}}^J(2) = \eta_4$. Then $\xi e_2 e_4$ is an element of \check{E}_τ^J with $\tau = L \setminus V \rightarrow \beta$ and so there is the following element

$$\odot_{\triangleleft\{\tau, L \setminus V\}}^J(\xi e_2 e_4 \triangleleft \{\eta_1 \rightarrow e_1, \eta_3 \rightarrow e_3\})$$

of \check{E}_β^J . However, this term has the unique representation $\xi e_1 e_2 e_3 e_4$ (and not $\xi e_2 e_4 e_1 e_3$). Moreover, this problem cannot be avoided by a ‘better’ choice of the

enumerations, since for any choice of $i_{\xi\{L\}}^J$ there will be subsets V of L for which the order is then not ‘correct’. Of course, the problem does not arise in the usual functional programming languages, because they restrict partial application to subsets V which are compatible with the enumeration on L (more precisely, to initial segments of L).

If $\xi \in J$ is of type $L \rightarrow \beta$ with $L = \sigma_1 \cdots \sigma_m$ (and so $[m]$ is the underlying set of the S -typed set L) then $i_{\xi\{L\}}^J : [m] \rightarrow [m]$ will always be chosen to be the identity mapping. Moreover, if V is a non-empty subset of L then $i_{\xi\{V\}}^J$ will be taken to be the induced enumeration (i.e., so that the elements in V occur in the same relative order as they occur in L).

The S -typed sets J which occur in what follows are usually of the form $I \cup P$, and so the enumerations corresponding to each name of a primitive function have to be decided upon. The names of the integer operators were already dealt with above (since each is of type $\text{int int} \rightarrow \beta$ with β either int or bool). It thus remains to consider a name $\zeta = \text{case}_{\beta}^{\theta}$ with $\beta \in B$ and $\theta \in B_f$, and here an enumeration of the elements of V for each non-empty subset V of $\theta \cdot K_{\theta, \beta}$ has to be chosen. However, in all the examples to be given the only terms which occur involving ζ are of the form $\odot_{\zeta\{V\}}^J(s \triangleleft a)$ with $V = \theta \cdot K_{\theta, \beta}$ (i.e., total applications), and instead of fixing an enumeration of the set $\theta \cdot K_{\theta, \beta}$ once and for all, it is preferable rather to write such a term more ‘legibly’ as

$$\text{case } s \text{ of } \{ \kappa_1 \rightarrow a(\kappa_1), \dots, \kappa_m \rightarrow a(\kappa_m) \}$$

where $\kappa_1, \dots, \kappa_m$ is any enumeration of the elements of the set $K_{\theta, \beta} = K_{\theta}$ (which can be chosen arbitrarily by the author of the term). It is clear that θ and β , which have been omitted here from the name $\text{case}_{\beta}^{\theta}$, can always be uniquely determined from the context.

Finally, when working with explicit examples of terms it is sensible to allow the use of brackets if this increases the legibility. The bracketing that will be employed here is essentially that used in *Haskell*.

As already stated, if I is an S -typed set disjoint from P then $(\check{E}_S^{I \cup P}, \check{\square}_N^{I \cup P})$ will be denoted just by (E_S^I, \square_N^I) , so (E_S^I, \square_N^I) is the functionally $I \cup P$ -free Σ -algebra corresponding to (F_S^I, \odot_N^I) . Example 6.1.1 below gives a first indication of how these term algebras will be involved in the equations from Chapter 1.

Example 6.1.1 Let (D_S, f_N) be as in one of the three basic Cases. Let $I = \{\text{it}, \text{ita}, \text{fst}, \text{fsta}, \text{pfb}, \text{fib}\}$ be the S -typed set in which

it is of type $\text{ipr} \rightarrow \text{ipr}$, ita of type $\text{int int} \rightarrow \text{ipr}$,
 fst of type $\text{ipr} \rightarrow \text{int}$, fsta of type $\text{int int} \rightarrow \text{int}$,
 pfb of type $\text{int ipr} \rightarrow \text{ipr}$ and fib of type $\text{int} \rightarrow \text{int}$.

Let L be the S -typed set consisting of the element p of type ipr . Then the following are elements of $E_{\text{ipr}}^{I \cup L}$ and $E_{\text{int}}^{I \cup L}$

$\text{case } p \text{ of } \{\text{Pair} \rightarrow \text{ita}\}$
 $\text{case } p \text{ of } \{\text{Pair} \rightarrow \text{fsta}\}$

which will be denoted respectively by e_{it} and e_{fst} . These correspond to the right-hand sides of the equations for it and fst in Chapter 1.

Let $p \in D_{\text{ipr}}$ and consider an assignment $c \in D_{\circ}^{I \cup L}$ with $c(p) = p$. Then

$$\begin{aligned} \llbracket e_{\text{it}} \rrbracket_{\text{ipr}}^c &= \text{case}_{\text{ipr}}^{\text{ipr}}(\llbracket p \rrbracket_{\text{ipr}}^c, \{\text{Pair} \rightarrow \llbracket \text{ita} \rrbracket_{\text{ita}}^c\}) \\ &= \text{case}_{\text{ipr}}^{\text{ipr}}(p, \{\text{Pair} \rightarrow c_{\text{ita}}\}) \\ &= \begin{cases} c_{\text{ita}}(m, n) & \text{if } p \neq \perp_{\text{ipr}} \text{ and } p = f_{\text{Pair}}(m, n), \\ \perp_{\text{ipr}} & \text{if } p = \perp_{\text{ipr}}. \end{cases} \end{aligned}$$

Moreover, it follows that

$$\begin{aligned} \llbracket e_{\text{fst}} \rrbracket_{\text{int}}^c &= \text{case}_{\text{int}}^{\text{ipr}}(\llbracket p \rrbracket_{\text{ipr}}^c, \{\text{Pair} \rightarrow \llbracket \text{fsta} \rrbracket_{\text{fsta}}^c\}) \\ &= \text{case}_{\text{int}}^{\text{ipr}}(p, \{\text{Pair} \rightarrow c_{\text{fsta}}\}) \\ &= \begin{cases} c_{\text{fsta}}(m, n) & \text{if } p \neq \perp_{\text{ipr}} \text{ and } p = f_{\text{Pair}}(m, n), \\ \perp_{\text{int}} & \text{if } p = \perp_{\text{ipr}}, \end{cases} \end{aligned}$$

where, to increase the legibility, c_{η} is used as an alternative notation for $c(\eta)$. In particular, if (D_B, f_K) is the flat extension of (X_B, p_K) then

$$\begin{aligned} \llbracket e_{\text{int}} \rrbracket_{\text{ipr}}^c &= \begin{cases} c_{\text{ita}}(m, n) & \text{if } p = (m, n) \text{ with } m, n \in \mathbb{Z}, \\ \perp_{\text{ipr}} & \text{if } p = \perp_{\text{ipr}}, \end{cases} \\ \llbracket e_{\text{fst}} \rrbracket_{\text{int}}^c &= \begin{cases} c_{\text{fsta}}(m, n) & \text{if } p = (m, n) \text{ with } m, n \in \mathbb{Z}, \\ \perp_{\text{int}} & \text{if } p = \perp_{\text{ipr}}. \end{cases} \end{aligned}$$

(*Example 6.1.1 is continued on the next page.*)

Example 6.1.1 (continued) Next let L be the S -typed set consisting of the elements m and n of type int . Then m is an element $e_{\text{fst}a}$ of E_{int}^{IUL} and $\text{Pair } n$ ($\text{add } m \ n$) an element e_{ita} of E_{ipr}^{IUL} . These terms correspond to the right-hand sides of the equations for $\text{fst}a$ and ita in Chapter 1.

Let $m, n \in D_{\text{int}}$ and $c \in D_{\circ}^{IUL}$ be an assignment with $c(m) = m$ and $c(n) = n$. Then

$$\llbracket e_{\text{ita}} \rrbracket_{\text{ipr}}^c = f_{\text{Pair}}(\llbracket n \rrbracket_{\text{int}}^c, \llbracket \text{add } m \ n \rrbracket_{\text{int}}^c) = f_{\text{Pair}}(n, \text{add}(m, n)).$$

In particular, if (D_B, f_K) is the flat extension of (X_B, p_K) then

$$\llbracket e_{\text{ita}} \rrbracket_{\text{ipr}}^c = \begin{cases} (n, m + n) & \text{if } m, n \in \mathbb{Z}, \\ \perp_{\text{ipr}} & \text{otherwise.} \end{cases}$$

Now let L be the S -typed set consisting of the element n of type int . The following is then an element e_{fib} of E_{int}^{IUL} :

$$\text{fst } (\text{pfb } n \ (\text{Pair } 0 \ 1))$$

which corresponds to the right-hand side of the equation for fib in Chapter 1.

Let $n \in D_{\text{int}}$ and $c \in D_{\circ}^{IUL}$ be an assignment with $c(n) = n$. Then

$$\begin{aligned} \llbracket e_{\text{fib}} \rrbracket_{\text{int}}^c &= c_{\text{fst}}(\llbracket \text{pfb } n \ \text{Pair } 0 \ 1 \rrbracket_{\text{ipr}}^c) \\ &= c_{\text{fst}}(c_{\text{pfb}}(\llbracket n \rrbracket_{\text{int}}^c, \llbracket \text{Pair } 0 \ 1 \rrbracket_{\text{ipr}}^c)) \\ &= c_{\text{fst}}(c_{\text{pfb}}(c_n, f_{\text{Pair}}(\llbracket 0 \rrbracket_{\text{int}}^c, \llbracket 1 \rrbracket_{\text{int}}^c))) \\ &= c_{\text{fst}}(c_{\text{pfb}}(n, f_{\text{Pair}}(0, 1))) = c_{\text{fst}}(c_{\text{pfb}}(n, (0, 1))) \end{aligned}$$

Finally, consider the S -typed set L consisting of the elements n and p , with n of type int and p of type ipr . Then the following element e_{pfb} of E_{ipr}^{IUL}

$$\text{case } (\text{eq } n \ 0) \text{ of } \{ \text{True} \rightarrow p, \text{False} \rightarrow \text{pfb } (\text{sub } n \ 1) \ (\text{it } p) \}$$

corresponds to the right-hand side of the equation for ipr in Chapter 1.

Let $n \in D_{\text{int}}$, $p \in D_{\text{ipr}}$ and let $c \in D_{\circ}^{IUL}$ be an assignment with $c(n) = n$ and $c(p) = p$. Then

$$\llbracket \text{eq } n \ 0 \rrbracket_{\text{bool}}^c = \text{eq}(n, 0) = \begin{cases} T & \text{if } n = 0, \\ F & \text{if } n \in \mathbb{Z} \setminus \{0\}, \\ \perp_{\text{bool}} & \text{if } n = \perp_{\text{int}}. \end{cases}$$

(*Example 6.1.1 is continued on the next page.*)

Example 6.1.1 (continued) Thus, putting $z = \llbracket \text{eq } n \ 0 \rrbracket_{\text{bool}}^c$, it follows that

$$\begin{aligned} & \llbracket e_{\text{pfb}} \rrbracket_{\text{ipr}}^c \\ &= \text{case}_{\text{ipr}}^{\text{bool}}(z, \{ \text{True} \rightarrow \llbracket p \rrbracket_{\text{ipr}}^c, \text{False} \rightarrow \llbracket \text{pfb sub } n \ 1 \ \text{it } p \rrbracket_{\text{ipr}}^c \}) \\ &= \text{case}_{\text{ipr}}^{\text{bool}}(\text{eq}(n, 0), \{ \text{True} \rightarrow p, \text{False} \rightarrow c_{\text{pfb}}(\text{sub}(n, 1), c_{\text{it}}(p)) \}) \\ &= \begin{cases} p & \text{if } n = 0, \\ c_{\text{pfb}}(n - 1, c_{\text{it}}(p)) & \text{if } n \in \mathbb{Z} \setminus \{0\}, \\ \perp_{\text{ipr}} & \text{if } n = \perp_{\text{int}}. \end{cases} \end{aligned}$$

The meanings of the terms occurring in this example should be compared with the right-hand sides of the equations in Example 2.1.1.

6.2 Equations and their solutions

This section explains what is meant by equations and their solutions. A non-empty S -typed set I will be called *global* if I is disjoint from P and if L is disjoint from $I \cup P$ whenever $\xi \in I$ is of type $L \rightarrow \beta$. For the whole of the section let I be a global S -typed set.

In general, what is meant by an equation for the names in I is a pair (s_1, s_2) , where $s_1, s_2 \in F_\beta^{I \cup V}$ for some $\beta \in B$ and where V is some S -typed set disjoint from I and P . Here s_1 should be thought of as the left-hand and s_2 as the right-hand side of the equation. A solution of this equation is then any assignment $c \in D_\diamond^I$ for which

$$\llbracket s_1 \rrbracket_\beta^{c \oplus b} = \llbracket s_2 \rrbracket_\beta^{c \oplus b}$$

for all $b \in D_\diamond^V$. Given a system (i.e., a family) of such equations, the solutions of the system are those assignments which are solutions of each of the equations in the family.

However, this notion of an equation is much too general. Note that in these equations there is really no difference in the role played by the left-hand and the right-hand side, whereas in the equations in Chapter 1 this is not the case, since there the left-hand sides of the equations clearly have a very special form. Thus the elements which can occur as the left-hand side of an equation will now be restricted, and this involves what will be called an abstractor.

Until further notice let ξ be an element of I of type $\sigma = L \rightarrow \beta$ (and so L is disjoint from I and P). Define an element $a_\xi \in F_\beta^{I \cup L}$, called the ξ -*abstractor*, by putting

$$a_\xi = \odot_\xi^{I \cup L}(a_L),$$

where $a_L \in (F_\diamond^{I \cup L})^L$ is given by $a_L(\eta) = \eta$ (with η here as an element of $F_\eta^{I \cup L}$) for each $\eta \in L$. Thus if $L = \emptyset$ then $a_\xi = \xi$, and if $L \neq \emptyset$ then

$$a_\xi = \odot_{\mathfrak{A}\{\sigma, L\}}^{I \cup L}(\xi \triangleleft a_L).$$

Of course, if the ‘genuine’ term algebra $(E_S^{I \cup L}, \square_N^{I \cup L})$ is being used then

$$a_\xi = \xi \eta_1 \cdots \eta_m$$

where $m = |L|$ and η_1, \dots, η_m is the enumeration of the elements of L which comes with the definition of $E_S^{I \cup L}$.

Lemma 6.2.1 *If $L \neq \emptyset$ then $\llbracket a_\xi \rrbracket_\beta^{c \oplus b} = f_{\mathfrak{A}\{\sigma, L\}}(c(\xi) \triangleleft b)$ for all $c \in D_\diamond^I$ and all $b \in D_\diamond^L$. Moreover, if $L = \emptyset$ then $\llbracket a_\xi \rrbracket_\beta^c = c(\xi)$ for all $c \in D_\diamond^I$.*

Proof This holds trivially if $L = \emptyset$, and if $L \neq \emptyset$ then

$$\llbracket a_\xi \rrbracket_\beta^{c \oplus b} = \llbracket \odot_\xi^{I \cup L}(a_L) \rrbracket_\beta^{c \oplus b} = f_{\mathfrak{A}\{\sigma, L\}}(c(\xi) \triangleleft (\llbracket a_L \rrbracket_\diamond^{c \oplus b})^L) = f_{\mathfrak{A}\{\sigma, L\}}(c(\xi) \triangleleft b)$$

since $(\llbracket a_L \rrbracket_\diamond^{c \oplus b})^L(\eta) = \llbracket a_L(\eta) \rrbracket_\eta^{c \oplus b} = \llbracket \eta \rrbracket_\eta^{c \oplus b} = b(\eta)$ for each $\eta \in L$, and therefore $(\llbracket a_L \rrbracket_\diamond^{c \oplus b})^L = b$. \square

The set $F_\beta^{I \cup L}$ will be denoted by $\text{Eq}(\xi, I)$. Thus to each element $s \in \text{Eq}(\xi, I)$ there corresponds the equation (a_ξ, s) , and with this correspondence s can also be regarded as being an *equation* for ξ .

If $s \in \text{Eq}(\xi, I)$ is an equation for ξ then $\text{Sol}(s)$ will denote the set of solutions, i.e., $\text{Sol}(s)$ is the set of assignments $c \in D_\diamond^I$ for which

$$\llbracket a_\xi \rrbracket_\beta^{c \oplus b} = \llbracket s \rrbracket_\beta^{c \oplus b}$$

for all $b \in D_\diamond^L$. Because of the special form of the left-hand side a_ξ Lemma 6.2.1 implies that if $L \neq \emptyset$ then $\text{Sol}(s)$ is the set of assignments $c \in D_\diamond^I$ for which

$$f_{\triangleleft\{\sigma, L\}}(c(\xi) \triangleleft b) = \llbracket s \rrbracket_\beta^{c \oplus b}$$

for all $b \in D_\diamond^L$, and if $L = \emptyset$ then then $\text{Sol}(s)$ is just the set of assignments $c \in D_\diamond^I$ for which $c(\xi) = \llbracket s \rrbracket_\beta^c$.

Now a system of equations is just a family of equations, one for each name in I . More precisely, denote by $\text{Eq}(I)$ the set of all families s_I , where $s_\xi \in \text{Eq}(\xi, I)$ for each $\xi \in I$. Each element of $\text{Eq}(I)$ will be regarded as a *system of equations* for the names in I . Note that the definition of the set $\text{Eq}(I)$ has nothing to do with the Σ -algebra (D_S, f_N) . It really only depends on the the signature Σ and the S -typed set I .

If $\alpha = s_I \in \text{Eq}(I)$ is a system of equations then an assignment $c \in D_\diamond^I$ is said to be a *solution* of the equations α if $c \in \text{Sol}(s_\xi)$ for each $\xi \in I$; the set of such solutions will be denoted by $\text{Sol}(\alpha)$. An assignment $c \in D_\diamond^I$ being a solution thus means that

$$f_{\triangleleft\{\sigma, L\}}(c(\xi) \triangleleft b) = \llbracket s_\xi \rrbracket_\beta^{c \oplus b}$$

must hold for all $b \in D_\diamond^L$ whenever $\xi \in I$ is of type $\sigma = L \rightarrow \beta$ with $L \neq \emptyset$ and that $c(\xi) = \llbracket s_\xi \rrbracket_\beta^c$ whenever ξ is of type β . (Of course, if (D_S, f_N) is as in one of the three basic Cases then $f_{\triangleleft\{\sigma, L\}}(c(\xi) \triangleleft b)$ is just equal to $c(\xi)(b)$.)

Although the definition of $\text{Eq}(I)$ has nothing to do with (D_S, f_N) , the set $\text{Sol}(\alpha)$ is a subset of D_\diamond^I , and for this trivial reason it thus depends on (D_S, f_N) . In fact, the dependence is perhaps more subtle than would at first be expected. Examples 6.2.3 and 6.2.4 towards the end of the section indicate the typical kind of behaviour which occurs.

In what follows consider the explicit programming language obtained using the ‘genuine’ term algebras. An equation $e \in \text{Eq}(I)$ for ξ will here be written as

$$a_\xi = \hat{e}$$

where a_ξ is the ξ -abstractor (and, as already noted, a_ξ just has the form $\xi \eta_1 \cdots \eta_m$), and \hat{e} is the ‘legible’ notation for the term e introduced in Section 6.1 (i.e., with the particular notation for the case operators and permitting the use of brackets). For a system of equations this notation will be used for each single equation. Moreover, since in practice I will be finite, such a system will be represented as a list, with each single equation written on a separate line. (This involves, of course, choosing some enumeration of the elements of I .)

To prepare for the examples there a further point which has to be dealt with. In all of these examples each element $\xi \in I$ has a type of the form $\sigma_1 \cdots \sigma_m \rightarrow \beta$. Now

the underlying set of the S -typed set $\sigma_1 \cdots \sigma_m$ is just $[m]$, which means that the left-hand side of the equation for ξ should always have the form

$$\xi \ 1 \ 2 \ \cdots \ m$$

with the ‘names’ $1 \ 2, \dots, m$ also occurring as the ‘local variables’ on the right-hand side of the equation. This is not very satisfactory, in particular since these ‘names’ give no information about their types. Moreover, the problem then arises of representing the ‘name’ $k \in [m]$ in a way which distinguishes it from the constructor name \underline{k} of type `int` (i.e., from the element \underline{k} of *Int*).

To avoid these problems it is convenient to replace each $j \in [m]$ in the equation for ξ by some more suggestive alias η_j . The left-hand side of the equation then becomes

$$\xi \ \eta_1 \ \eta_2 \ \cdots \ \eta_m$$

with the right-hand side changed accordingly. In any example the choice of the aliases used in an equation can of course be determined from the modified left-hand side.

It should be emphasised that this aliasing device is simply a method of making explicit examples more readable, and there is no reason compelling its use. (The procedure also has a more elegant interpretation, which the interested reader is left to work out.)

The system of equations introduced in Chapter 1 is treated in Example 6.2.1. These equations are first presented without aliasing, just to show how unreadable they are. In this version it is necessary to distinguish each element of $[m]$ from the corresponding element of *Int*, which is done by enclosing the elements of $[m]$ in single quotes (so 1 will be represented as ‘1’, 2 as ‘2’, and so on). In all the remaining examples in these notes aliasing is employed without comment.

Example 6.2.1 As in Example 6.1.1 let

$$I = \{it, ita, fst, fsta, pfb, fib\}$$

be the S -typed set with

it of type $ipr \rightarrow ipr$, ita of type $int\ int \rightarrow ipr$,
 fst of type $ipr \rightarrow int$, $fsta$ of type $int\ int \rightarrow int$,
 pfb of type $int\ ipr \rightarrow ipr$ and fib of type $int \rightarrow int$.

Consider that following terms (each being an element of $E_{\beta}^{I \cup \gamma}$ for some list of types $\gamma \in S^*$ and some $\beta \in B$):

$$\begin{aligned} e_{it} &= \text{case '1' of \{Pair -> ita\}} \in E_{ipr}^{I \cup ipr}, \\ e_{ita} &= \text{Pair '2' (add '1' '2')} \in E_{ipr}^{I \cup int\ ipr}, \\ e_{fst} &= \text{case '1' of \{Pair -> fsta\}} \in E_{int}^{I \cup ipr}, \\ e_{fsta} &= \text{'1'} \in E_{int}^{I \cup int\ int}, \\ e_{pfb} &= \text{case (eq '1' 0) of \{True -> '2', \\ &\quad \text{False -> pfb (sub '1' 1) (it p)\}} \in E_{ipr}^{I \cup int\ ipr}, \\ e_{fib} &= \text{fst (pfb '1' (Pair 0 1))} \in E_{int}^{I \cup int}. \end{aligned}$$

Then $\alpha = e_I$ is an element of $\text{Eq}(I)$ which should be written as

$$\begin{aligned} it\ '1' &= \text{case '1' of \{Pair -> ita\}} \\ ita\ '1'\ '2' &= \text{Pair '2' (add '1' '2')} \\ fst\ '1' &= \text{case '1' of \{Pair -> fsta\}} \\ fsta\ '1'\ '2' &= \text{'1'} \\ pfb\ '1'\ '2' &= \text{case (eq '1' 0) of \{True -> '2', \\ &\quad \text{False -> pfb (sub '1' 1) (it p)\}} \\ fib\ '1' &= \text{fst (pfb '1' (Pair 0 1))} \end{aligned}$$

(Example 6.2.1 is continued on the next page.)

Example 6.2.1 (continued) However, with an appropriate choice of aliases, α can be written more ‘legibly’ as

```

it p = case p of {Pair -> ita}
ita m n = Pair n (add m n)
fst p = case p of {Pair -> fsta}
fsta m n = m
pfb n p = case (eq n 0) of {True -> p,
                             False -> pfb (sub n 1) (it p)}
fib n = fst (pfb n (Pair 0 1))

```

Again let (D_S, f_N) be as in one of the three basic Cases. Exactly the same calculations as in Example 6.1.1 show that an assignment $c \in D_\circ^I$ is in $\text{Sol}(\alpha)$ if and only if for all $m, n \in D_{\text{int}}$, $p \in D_{\text{ipr}}$

$$c_{\text{it}}(p) = \begin{cases} c_{\text{ita}}(m, n) & \text{if } p \neq \perp_{\text{ipr}} \text{ and } p = f_{\text{Pair}}(m, n), \\ \perp_{\text{ipr}} & \text{if } p = \perp_{\text{ipr}}, \end{cases}$$

$$c_{\text{ita}}(m, n) = f_{\text{Pair}}(n, \text{add}(m, n)),$$

$$c_{\text{fst}}(p) = \begin{cases} c_{\text{fsta}}(m, n) & \text{if } p \neq \perp_{\text{ipr}} \text{ and } p = f_{\text{Pair}}(m, n), \\ \perp_{\text{int}} & \text{if } p = \perp_{\text{ipr}}, \end{cases}$$

$$c_{\text{fsta}}(m, n) = m,$$

$$c_{\text{pfb}}(n, p) = \begin{cases} p & \text{if } n = 0, \\ c_{\text{pfb}}(n - 1, c_{\text{it}}(p)) & \text{if } n \in \mathbb{Z} \setminus \{0\}, \\ \perp_{\text{ipr}} & \text{if } n = \perp_{\text{int}}, \end{cases}$$

$$c_{\text{fib}}(n) = c_{\text{fst}}(c_{\text{pfb}}(n, (0, 1))).$$

The special case of Example 6.2.1 in which (D_B, f_K) is the flat extension of (X_B, p_K) is looked at in Example 6.2.2 on the following page: This corresponds to the situation considered in the third interpretation of the original *Haskell* equations.

Example 6.2.2 As in Example 6.2.1 let $\alpha \in \text{Eq}(I)$ be the system of equations written 'legibly' as

```

it p = case p of {Pair -> ita}
ita m n = Pair n (add m n)
fst p = case p of {Pair -> fsta}
fsta p = m
pfb n p = case (eq n 0) of {True -> p,
                             False -> pfb (sub n 1) (it p)}
fib n = fst (pfb n (Pair 0 1))

```

Suppose now, however, that (D_B, f_K) is the flat extension of (X_B, p_K) . Then $c \in \text{Sol}(\alpha)$ if and only if for all $m, n \in D_{\text{int}}$, $p \in D_{\text{ipr}}$

$$c_{\text{it}}(p) = \begin{cases} c_{\text{ita}}(m, n) & \text{if } p = (m, n) \text{ with } m, n \in \mathbb{Z}, \\ \perp_{\text{ipr}} & \text{if } p = \perp_{\text{ipr}}, \end{cases}$$

$$c_{\text{ita}}(m, n) = \begin{cases} (n, m + n) & \text{if } p = (m, n) \text{ with } m, n \in \mathbb{Z}, \\ \perp_{\text{ipr}} & \text{if } p = \perp_{\text{ipr}}, \end{cases}$$

$$c_{\text{fst}}(p) = \begin{cases} c_{\text{fsta}}(m, n) & \text{if } p = (m, n) \text{ with } m, n \in \mathbb{Z}, \\ \perp_{\text{int}} & \text{if } p = \perp_{\text{ipr}}, \end{cases}$$

$$c_{\text{fsta}}(m, n) = m,$$

$$c_{\text{pfb}}(n, p) = \begin{cases} p & \text{if } n = 0, \\ c_{\text{pfb}}(n - 1, c_{\text{it}}(p)) & \text{if } n \in \mathbb{Z} \setminus \{0\}, \\ \perp_{\text{ipr}} & \text{if } n = \perp_{\text{int}}, \end{cases}$$

$$c_{\text{fib}}(n) = c_{\text{fst}}(c_{\text{pfb}}(n, (0, 1))).$$

In particular, if $c \in \text{Sol}(\alpha)$ then for all $m, n \in D_{\text{int}}$, $p \in D_{\text{ipr}}$

$$c_{\text{it}}(p) = \begin{cases} (n, m + n) & \text{if } p = (m, n) \text{ with } m, n \in \mathbb{Z}, \\ \perp_{\text{ipr}} & \text{if } p = \perp_{\text{ipr}}, \end{cases}$$

$$c_{\text{fst}}(p) = \begin{cases} m & \text{if } p = (m, n) \text{ with } m, n \in \mathbb{Z}, \\ \perp_{\text{int}} & \text{if } p = \perp_{\text{ipr}}. \end{cases}$$

This is essentially the situation first encountered in the third interpretation of the original *Haskell* equations.

As already mentioned, Examples 6.2.3 and 6.2.4 below show that the dependence of the solutions on (D_S, f_N) is perhaps more subtle than would at first be expected.

Example 6.2.3 Let

$$I = \{ \text{fst}, \text{fsta}, \text{inf}, \text{one} \}$$

be the S -typed set with

`fst` of type `ipr → int`, `fsta` of type `int int → int`,
`inf` and `one` of type `int`.

Consider the system of equations $\alpha \in \text{Eq}(I)$ given by

```
fst p = case p of {Pair -> fsta}
fsta m n = m
inf = add inf 1
one = fst (Pair 1 inf)
```

Let (D_S, f_N) be as in one of the three basic Cases. Note that $c_{\text{inf}} = \perp_{\text{int}}$ for all $c \in \text{Sol}(\alpha)$. From this it follows that for all $c \in \text{Sol}(\alpha)$

$$c_{\text{one}} = \begin{cases} 1 & \text{if } f_{\text{Pair}}(1, \perp_{\text{int}}) \neq \perp_{\text{ipr}}, \\ \perp_{\text{int}} & \text{otherwise.} \end{cases}$$

Therefore if (D_B, f_K) is the flat extension of (X_B, p_K) then $c_{\text{one}} = \perp_{\text{int}}$ for all $c \in \text{Sol}(\alpha)$. On the other hand, if (D_B, f_K) is a fully regular extension of (X_B, p_K) then $c_{\text{one}} = 1$ for all $c \in \text{Sol}(\alpha)$. Note that these equations do have a solution (i.e., $\text{Sol}(\alpha) \neq \emptyset$) and this fact does not depend on (D_B, f_K) .

Example 6.2.4 Let (D_S, f_N) be as in one of the three basic Cases and

$$I = \{\text{hd}, \text{hda}, \text{ones}, \text{one}\}$$

be the S -typed set with

`hd` of type `ilst` \rightarrow `int`, `hda` of type `int ilst` \rightarrow `int`,
`ones` of type `ilst` and `one` of type `int`.

Consider the system of equations $\alpha \in \text{Eq}(I)$ given by

```
hd xs = case xs of {Nil -> hd xs, Cons -> hda}
hda x xs = x
ones = Cons 1 ones
one = hd ones
```

Then $c_{\text{ones}} = f_{\text{cons}}(1, c_{\text{ones}})$ for all $c \in \text{Sol}(\alpha)$ and from this it follows that

$$c_{\text{one}} = \begin{cases} 1 & \text{if } c_{\text{ones}} \neq \perp_{\text{ilst}}, \\ \perp_{\text{int}} & \text{otherwise.} \end{cases}$$

Thus if (D_B, f_K) is the flat extension of (X_B, p_K) then $c_{\text{one}} = \perp_{\text{int}}$ for all $c \in \text{Sol}(\alpha)$ (since here $c_{\text{ones}} = \perp_{\text{ilst}}$ for each $c \in \text{Sol}(\alpha)$). Moreover, if (D_B, f_K) is a fully regular extension of (X_B, p_K) then $c_{\text{one}} = 1$ for all $c \in \text{Sol}(\alpha)$ (since here $c_{\text{ones}} \neq \perp_{\text{ilst}}$ for each $c \in \text{Sol}(\alpha)$).

Nevertheless, the situation here is quite different to that in Example 6.2.3. For simplicity suppose that (D_S, f_N) is as in Case 1. Then the present system of equations has a solution if (D_B, f_K) is the flat extension of (X_B, p_K) , but whether or not there are solutions for a fully regular extension depends on the extension.

In particular, if (D_B, f_K) is an initial extension of (X_B, p_K) then it is easy to see that there are no solutions to the equations, since the equation for `ones` cannot be satisfied by any finite list. However, if the fully regular extension is obtained as an initial completion (as in Section 4.3) then there will be a solution, which can be considered as an infinite list in which each component is equal to 1.

In Example 6.2.5 a simple system of equations is presented which gives a more typical instance of the ‘case’ construction than that occurring in the original equations. This example illustrates, moreover, how ‘genuine’ partial functional applications are often required (i.e., partial applications with some but not all parameters applied).

Example 6.2.5 Let $I = \{\text{map}, \text{mapa}\}$ be the S -typed set with

`map` of type $(\text{int} \rightarrow \text{int}) \text{ilst} \rightarrow \text{ilst}$,

`mapa` of type $(\text{int} \rightarrow \text{int}) \text{int ilst} \rightarrow \text{ilst}$.

Let $\alpha \in \text{Eq}(I)$ be the system of equations written ‘legibly’ as

`map f ns = case ns of {Nil -> Nil, Cons -> mapa f }`

`mapa f m ms = Cons (f m) (map f ms)`

Note that the term `mapa f` occurring in the right-hand side of the equation for `map` denotes a ‘genuine’ partial application.

As usual let (D_S, f_N) be as in one of the three basic Cases, and let $c \in D_\diamond^I$; then a simple calculation shows that $c \in \text{Sol}(\alpha)$ if and only if

$$c_{\text{map}}(h, \ell) = \begin{cases} \varepsilon & \text{if } \ell = \varepsilon, \\ c_{\text{mapa}}(h, m', \ell') & \text{if } \ell \neq \perp_{\text{ilst}} \text{ and } \ell = f_{\text{cons}}(m', \ell'), \\ \perp_{\text{ilst}} & \text{if } \ell = \perp_{\text{ilst}}, \end{cases}$$

$$c_{\text{mapa}}(h, m, \ell) = f_{\text{cons}}(h(m), c_{\text{map}}(h, \ell)),$$

for all $h \in D_{\text{int} \rightarrow \text{int}}$, $m \in D_{\text{int}}$ and $\ell \in D_{\text{ilst}}$. In particular, if $c \in \text{Sol}(\alpha)$ then it follows that

$$c_{\text{map}}(h, \ell) = \begin{cases} \varepsilon & \text{if } \ell = \varepsilon, \\ f_{\text{cons}}(h(m), c_{\text{map}}(h, \ell')) & \text{if } \ell \neq \perp_{\text{ilst}} \\ & \text{and } \ell = f_{\text{cons}}(m, \ell'), \\ \perp_{\text{ilst}} & \text{if } \ell = \perp_{\text{ilst}}, \end{cases}$$

for all $h \in D_{\text{int} \rightarrow \text{int}}$, $\ell \in D_{\text{ilst}}$.

Note that Example 6.2.5 also provides a typical system of equations which is not first order: A family $\alpha \in \text{Eq}(I)$ is called a *first order* system of equations if for each $\xi \in I$ the type of ξ is first order, recalling that an element of S is said to be a first order type if it has the form $L \rightarrow \beta$ with $\langle \eta \rangle \in B$ for each $\eta \in L$.

6.3 Replacement rules

For the whole of the section let I be a global S -typed set and $\alpha = s_I \in \text{Eq}(I)$ be a system of equations.

The intention here is to find rules which allow valid assertions to be made about the solutions of the equations α . The simplest kind of situation in which such rules are needed arises when trying to ‘compute values’: Let $s \in F_\beta^I$ for some $\beta \in B$ and let $x \in X_\beta$; the question is then whether it be asserted that $\llbracket s \rrbracket_\beta^c = x$ for all $c \in \text{Sol}(\alpha)$, i.e., that s has the ‘value’ x independent of the solution $c \in \text{Sol}(\alpha)$?

Another kind of situation arises when attempting to verify that the solutions of the equations agree with some given partial functions. Let $\xi \in I$ be of type $\sigma = L \rightarrow \beta$, let A be a subset of D_\diamond^L and let $h : A \rightarrow D_\beta$ be a mapping; the question is then whether it can be asserted that $f_{\triangleleft\{\sigma, L\}}(c(\xi) \triangleleft b) = h(b)$ for all $c \in \text{Sol}(\alpha)$ and all $b \in A$? Note by Lemma 6.2.1 this assertion states that $\llbracket a_\xi \rrbracket_\beta^{c\oplus b} = h(b)$ for all $c \in \text{Sol}(\alpha)$ and all $b \in A$, where $a_\xi \in F_\beta^{I \cup L}$ is the ξ -abstractor.

To arrive at valid assertions of the type described above the following situation will be considered: Let U be a fixed S -typed set disjoint from I and P . Then, given a term $s \in F_\sigma^{I \cup U}$ and $b \in D_\diamond^U$, the aim is to find rules which allow s to be replaced with an new (and hopefully simpler) term $s' \in F_\sigma^{I \cup U}$ such that

$$\llbracket s' \rrbracket_\sigma^{c\oplus b} = \llbracket s \rrbracket_\sigma^{c\oplus b}$$

for all $c \in \text{Sol}(\alpha)$.

Four replacement rules will be introduced. However, only the second rule actually involves the equations α being considered, and only the third and the fourth rule depend on which Σ -algebra (D_S, f_N) is being employed.

Replacement rule 1: The first replacement rule is:

(R1) Let $\nu \in N$ be of type $L \rightarrow \sigma$ and let $a, a' \in (F_\diamond^{I \cup U})^L$. Then

$$\llbracket \odot_\nu^{I \cup U}(a') \rrbracket_\sigma^{c\oplus b} = \llbracket \odot_\nu^{I \cup U}(a) \rrbracket_\sigma^{c\oplus b}$$

for any $c \in D_\diamond^I$, $b \in D_\diamond^U$ such that $(\llbracket a' \rrbracket_\diamond^{c\oplus b})^L = (\llbracket a \rrbracket_\diamond^{c\oplus b})^L$, i.e., such that

$$\llbracket a'(\eta) \rrbracket_\eta^{c\oplus b} = \llbracket a(\eta) \rrbracket_\eta^{c\oplus b}$$

for each $\eta \in L$.

Lemma 6.3.1 *The replacement rule (R1) is valid.*

Proof This is clear, since

$$\begin{aligned} \llbracket \odot_\nu^{I \cup U}(a') \rrbracket_\sigma^{c\oplus b} &= f_\nu((\llbracket a' \rrbracket_\diamond^{c\oplus b})^L) \\ &= f_\nu((\llbracket a \rrbracket_\diamond^{c\oplus b})^L) = \llbracket \odot_\nu^{I \cup U}(a) \rrbracket_\sigma^{b\oplus c} . \quad \square \end{aligned}$$

The replacement rule (R1), at least when it is applied recursively, is a form of what is called *referential transparency*. Because of Proposition 6.1.1, only the following two special cases of this rule are really needed:

(R1¹) Let $\kappa \in K$ be of type $L \rightarrow \beta$ and let $a, a' \in (F_{\diamond}^{I \cup U})^L$. Then

$$\llbracket \odot_{\kappa}^{I \cup U}(a') \rrbracket_{\beta}^{c \oplus b} = \llbracket \odot_{\kappa}^{I \cup U}(a) \rrbracket_{\beta}^{c \oplus b}$$

for any $c \in D_{\diamond}^I$, $b \in D_{\diamond}^U$ such that $(\llbracket a' \rrbracket_{\diamond}^{c \oplus b})^L = (\llbracket a \rrbracket_{\diamond}^{c \oplus b})^L$.

(R1²) Let $\xi \in I \cup P$ be of type $\sigma = L \rightarrow \beta$, J be a non-empty subset of L , and let $a, a' \in (F_{\diamond}^{I \cup U})^J$. Then

$$\llbracket \odot_{\xi\{J\}}^{I \cup U}(a') \rrbracket_{L \setminus J \rightarrow \beta}^{c \oplus b} = \llbracket \odot_{\xi\{J\}}^{I \cup U}(a) \rrbracket_{L \setminus J \rightarrow \beta}^{c \oplus b}$$

for any $c \in D_{\diamond}^I$, $b \in D_{\diamond}^U$ such that $(\llbracket a' \rrbracket_{\diamond}^{c \oplus b})^J = (\llbracket a \rrbracket_{\diamond}^{c \oplus b})^J$.

Replacement rule 2: The second replacement rule involves substitution. Roughly speaking it says that a term which matches the left-hand side of an equation can be replaced with the corresponding right-hand side.

Let L be an S -typed set disjoint from I and P and $a \in (F_{\diamond}^{I \cup U})^L$ be an assignment (which just means that $a(\eta) \in F_{\eta}^{I \cup U}$ for each $\eta \in L$). Then, since $(F_S^{I \cup U}, \odot_N^{I \cup U})$ is functionally $I \cup L \cup P$ -free and $(F_S^{I \cup U}, \odot_N^{I \cup U})$ is a functional Σ -algebra, there exists a unique Σ -homomorphism $\delta_S^a : (F_S^{I \cup U}, \odot_N^{I \cup U}) \rightarrow (F_S^{I \cup U}, \odot_N^{I \cup U})$ such that $\delta_S^a(\xi) = \xi$ for each $\xi \in I \cup P$ and $\delta_S^a(\eta) = a(\eta)$ for each $\eta \in L$.

For each $s \in F_{\sigma}^{I \cup L}$ the (perhaps) more suggestive notation

$$s[a/L]$$

will also be used for the element $\delta_S^a(s)$ of $F_{\sigma}^{I \cup U}$, this being interpreted as the term obtained by replacing for each $\eta \in L$ each occurrence of η in s by $a(\eta)$. Moreover, when working with examples it is convenient to allow the more explicit notation

$$s[a(\eta_1)/\eta_1, \dots, a(\eta_m)/\eta_m]$$

where η_1, \dots, η_m is some enumeration of the elements of L . The second rule is:

(R2) Let $\xi \in I$ be of type $L \rightarrow \beta$ and $a \in (F_{\diamond}^{I \cup U})^L$; then

$$\llbracket \odot_{\xi}^{I \cup U}(a) \rrbracket_{\beta}^{c \oplus b} = \llbracket s_{\xi}[a/L] \rrbracket_{\beta}^{c \oplus b}$$

for each $c \in \text{Sol}(\alpha)$ and all $b \in D_{\diamond}^U$ (where s_{ξ} is here the right-hand side of the equation for ξ). In particular, if $L = \emptyset$, in which case ξ is of type β and $a = \varepsilon$, then $\llbracket \xi \rrbracket_{\beta}^{c \oplus b} = \llbracket s_{\xi}[\varepsilon/\emptyset] \rrbracket_{\beta}^{c \oplus b}$ for each $c \in \text{Sol}(\alpha)$ and all $b \in D_{\diamond}^U$.

The following fact is required in order to show that rule (R2) is valid:

Lemma 6.3.2 *Let $a \in (F_{\diamond}^{I \cup U})^L$, $c \in D_{\diamond}^I$, and $b \in D_{\diamond}^U$, and put $d = (\llbracket a \rrbracket_{\diamond}^{c \oplus b})^L$, i.e., $d \in D_{\diamond}^L$ is the assignment given by $d(\eta) = \llbracket a(\eta) \rrbracket_{\eta}^{c \oplus b}$ for each $\eta \in L$. Then*

$$\llbracket s[a/L] \rrbracket_{\sigma}^{c \oplus b} = \llbracket s \rrbracket_{\sigma}^{c \oplus d}$$

for all $s \in F_{\sigma}^{I \cup L}$, $\sigma \in S$.

Proof Put $\pi_S = \llbracket \cdot \rrbracket_S^{c \oplus b} \delta_S^a$. Then by Proposition 2.2.1 π_S is a Σ -homomorphism from $(F_S^{I \cup L}, \odot_N^{I \cup L})$ to (D_S, f_N) and $\pi_{\xi}(\xi) = \llbracket \delta_{\xi}^a(\xi) \rrbracket_{\xi}^{c \oplus b} = \llbracket \xi \rrbracket_{\xi}^{c \oplus b} = c(\xi)$ for each $\xi \in I$, and in the same way it follows that $\pi_{\zeta}(\zeta) = p(\zeta)$ for each $\zeta \in P$. Moreover, $\pi_{\eta}(\eta) = \llbracket \delta_{\eta}^a(\eta) \rrbracket_{\eta}^{c \oplus b} = \llbracket a(\eta) \rrbracket_{\eta}^{c \oplus b} = d(\eta)$ for each $\eta \in L$. But $\llbracket \cdot \rrbracket_S^{c \oplus d}$ is the unique such Σ -homomorphism, and hence $\pi_S = \llbracket \cdot \rrbracket_S^{c \oplus d}$, i.e.,

$$\llbracket s[a/I] \rrbracket_{\sigma}^{c \oplus b} = \llbracket \delta_{\sigma}^a(s) \rrbracket_{\sigma}^{c \oplus b} = \llbracket s \rrbracket_{\sigma}^{c \oplus d}$$

for all $s \in F_{\sigma}^{I \cup L}$, $\sigma \in S$. \square

Lemma 6.3.3 *The replacement rule (R2) is valid.*

Proof Let $a \in (F_{\diamond}^{I \cup U})^L$, $c \in \text{Sol}(\alpha)$ and $b \in D_{\diamond}^U$ and put $d = (\llbracket a \rrbracket_{\diamond}^{c \oplus b})^L$; thus by Lemma 6.3.2 $\llbracket s_{\xi}[a/L] \rrbracket_{\beta}^{c \oplus b} = \llbracket s_{\xi} \rrbracket_{\beta}^{c \oplus d}$. If $L \neq \emptyset$ then by Lemma 6.2.1

$$\begin{aligned} \llbracket s_{\xi}[a/L] \rrbracket_{\beta}^{c \oplus b} &= \llbracket s_{\xi} \rrbracket_{\beta}^{c \oplus d} = f_{\triangleleft\{\sigma, L\}}(c(\xi) \triangleleft d) \\ &= f_{\triangleleft\{\sigma, L\}}(c(\xi) \triangleleft (\llbracket a \rrbracket_{\diamond}^{c \oplus b})^L) = \llbracket \odot_{\xi}^{I \cup U}(a) \rrbracket_{\beta}^{c \oplus b}. \end{aligned}$$

On the other hand, if $L = \emptyset$ then, again by Lemma 6.2.1,

$$\llbracket s_{\xi}[\varepsilon/\emptyset] \rrbracket_{\beta}^{c \oplus b} = \llbracket s_{\xi} \rrbracket_{\beta}^{c \oplus d} = c(\xi) = \llbracket \xi \rrbracket_{\beta}^{c \oplus b}. \quad \square$$

Replacement rule 3: The third replacement rule involves the integer operators.

Let $\zeta \in P$ be the name of an integer operator (and so ζ is of type $\text{int}_2 \rightarrow \beta$ with β either int or bool , recalling that int_2 is being used to denote the list int int).

Lemma 6.3.4 *If $n_1, n_2 \in \mathbb{Z}$ then $f_{\triangleleft\{\text{int}_2 \rightarrow \beta, \text{int}_2\}}(p(\zeta) \triangleleft (n_1, n_2))$ is an element of X_{β} , and so in particular it has a unique representation in the form $\llbracket s \rrbracket_{\beta}$ with $s \in F_{\beta}$.*

Proof Let $\hat{p}(\zeta) : D_{\text{int}} \times D_{\text{int}} \rightarrow D_{\beta}$ be the mapping given by

$$\hat{p}(\zeta)(n_1, n_2) = f_{\triangleleft\{\text{int}_2 \rightarrow \beta, \text{int}_2\}}(p(\zeta) \triangleleft (n_1, n_2))$$

for all $n_1, n_2 \in D_{\text{int}}$ (which just means that $\hat{p}(\text{add}) = \text{Add}$, $\hat{p}(\text{sub}) = \text{Sub}$ and so on). Then $\hat{p}(\zeta)(n_1, n_2) \in X_{\beta}$ for all $n_1, n_2 \in \mathbb{Z}$, since $\hat{p}(\zeta)$ is the strict extension of a mapping from $\mathbb{Z} \times \mathbb{Z}$ to X_{β} . \square

Example 6.3.1 Let $\alpha = e_I$ be as in Example 6.2.1 and let $U = \{n\}$ with n of type `int`. Then, for example,

$$e_{\text{fib}}[n/n] = \text{fst}(\text{pfb } n \text{ (Pair 0 1)})$$

$$\begin{aligned} e_{\text{fst}}[p/\text{pfb } n \text{ (Pair 0 1)}] \\ = \text{case}(\text{pfb } n \text{ (Pair 0 1)}) \text{ of } \{\text{Pair} \rightarrow \text{fst}\} \end{aligned}$$

$$e_{\text{pfb}}[n/n, p/\text{Pair 0 1}] = e$$

where e is the following element of E_{ipr}^{IUU} :

$$\begin{aligned} \text{case}(\text{eq } n \text{ 0}) \text{ of } \{\text{True} \rightarrow \text{Pair 0 1}, \\ \text{False} \rightarrow \text{pfb}(\text{sub } n \text{ 1}) \text{ (it (Pair 0 1))}\} \end{aligned}$$

Moreover, by rule (R2)

$$\begin{aligned} \llbracket \text{fib } n \rrbracket_{\text{int}}^{c \oplus b} &= \llbracket \text{fst}(\text{pfb } n \text{ (Pair 0 1)}) \rrbracket_{\text{int}}^{c \oplus b} \\ &= \llbracket \text{case}(\text{pfb } n \text{ (Pair 0 1)}) \text{ of } \{\text{Pair} \rightarrow \text{fst}\} \rrbracket_{\text{int}}^{c \oplus b} \end{aligned}$$

and

$$\llbracket \text{pfb } n \text{ (Pair 0 1)} \rrbracket_{\text{ipr}}^{c \oplus b} = \llbracket e \rrbracket_{\text{ipr}}^{c \oplus b}$$

for all $c \in \text{Sol}(\alpha)$ and all $b \in D_{\diamond}^U$.

The third replacement rule is:

(R3) Let $s_1, s_2 \in F_{\text{int}}^{IUU}$, let $c \in D_{\diamond}^I$, $b \in D_{\diamond}^U$ and suppose for each $j = 1, 2$ that $\llbracket s_j \rrbracket_{\text{int}}^{c \oplus b} = n_j \in \mathbb{Z}$. Then

$$\llbracket \odot_{\zeta}^{IUU}(s_1, s_2) \rrbracket_{\beta}^{c \oplus b} = \llbracket s_0 \rrbracket_{\beta}^{c \oplus b},$$

where s_0 is the unique element of F_{β} with

$$\llbracket s_0 \rrbracket_{\beta} = f_{\triangleleft\{\text{int}_2 \rightarrow \beta, \text{int}_2\}}(p(\zeta) \triangleleft (n_1, n_2)).$$

Lemma 6.3.5 *The replacement rule (R3) is valid.*

Proof This is clear, since for any $c \in D_{\diamond}^I$

$$\begin{aligned} \llbracket \odot_{\zeta}^{IUU}(s_1, s_2) \rrbracket_{\beta}^{c \oplus b} &= f_{\triangleleft\{\text{int}_2 \rightarrow \beta, \text{int}_2\}}(p(\zeta) \triangleleft (\llbracket s_1 \rrbracket_{\text{int}}^{c \oplus b}, \llbracket s_2 \rrbracket_{\text{int}}^{c \oplus b})) \\ &= f_{\triangleleft\{\text{int}_2 \rightarrow \beta, \text{int}_2\}}(p(\zeta) \triangleleft (n_1, n_2)) = \llbracket s_0 \rrbracket_{\beta} \end{aligned}$$

and by Lemma 6.1.1 $\llbracket s_0 \rrbracket_{\beta} = \llbracket s_0 \rrbracket_{\beta}^d$ for each $d \in D_{\diamond}^{IUU}$. \square

There is a particular case of rule (R3) which is especially useful when ‘computing values’, and for this the following is needed:

Lemma 6.3.6 *There exists a unique mapping $\text{Eval}_\zeta : F_{\text{int}} \times F_{\text{int}} \rightarrow F_\beta$ such that*

$$\llbracket \text{Eval}_\zeta(s_1, s_2) \rrbracket_\beta = f_{\triangleleft\{\text{int}_2 \rightarrow \beta, \text{int}_2\}}(p(\zeta) \triangleleft (\llbracket s_1 \rrbracket_{\text{int}}, \llbracket s_2 \rrbracket_{\text{int}}))$$

for all $s_1, s_2 \in F_{\text{int}}$.

Proof This follows from the fact that the mapping $\llbracket \cdot \rrbracket_{\text{int}} : F_{\text{int}} \rightarrow X_{\text{int}} = \mathbb{Z}$ is a bijection. \square

Note that if $\hat{p}(\zeta) : D_{\text{int}} \times D_{\text{int}} \rightarrow D_\beta$ is the mapping introduced in the proof of Lemma 6.3.4 (and hence $\hat{p}(\text{add}) = \text{Add}$, $\hat{p}(\text{sub}) = \text{Sub}$ and so on) then

$$\llbracket \text{Eval}_\zeta(s_1, s_2) \rrbracket_\beta = \hat{p}(\zeta)(\llbracket s_1 \rrbracket_{\text{int}}, \llbracket s_2 \rrbracket_{\text{int}})$$

for all $s_1, s_2 \in F_{\text{int}}$. This means that in the special case of the ‘genuine’ term algebras the mapping $\text{Eval}_\zeta : E_{\text{int}} \times E_{\text{int}} \rightarrow E_\beta$ is just the ‘syntactic’ version of the operator whose name is ζ . (For example, $\text{Eval}_{\text{add}}(10, 4) = 14$ and $\text{Eval}_{\text{eq}}(10, 4) = \text{False}$.)

The particular case of replacement rule (R3) is:

(R3') Let $s_1, s_2 \in F_{\text{int}}$. Then

$$\llbracket \odot_\zeta^{I \cup U}(s_1, s_2) \rrbracket_\beta^{c \oplus b} = \llbracket \text{Eval}_\zeta(s_1, s_2) \rrbracket_\beta^{c \oplus b}$$

for all $c \in D_\diamond^I$, $b \in D_\diamond^U$.

This is valid since Lemma 6.1.1 implies that $\llbracket s_j \rrbracket_{\text{int}}^{c \oplus b} = \llbracket s_j \rrbracket_{\text{int}} \in \mathbb{Z}$ for all $c \in D_\diamond^I$, $b \in D_\diamond^U$.

Replacement rule 4: The final replacement rule involves the case operators.

Let $\zeta = \text{case}_\beta^\theta$ for some $\beta \in B$, $\theta \in B_f$. Let $a \in (F_\diamond^{I \cup U})^{K_\theta, \beta}$ and $s \in F_\theta^{I \cup U}$ with $s = \odot_\kappa^{I \cup U}(a')$, where $\kappa \in K$ is of type $L' \rightarrow \theta$ for some $L' \in \mathcal{F}_B$ and where $a' \in (F_\diamond^{I \cup U})^{L'}$. Then the fourth rule is:

(R4) Let $c \in D_\diamond^I$, $b \in D_\diamond^U$ and suppose $\llbracket s \rrbracket_\theta^{c \oplus b} \neq \perp_\theta$. Then

$$\llbracket \odot_\zeta^{I \cup U}(s \triangleleft a) \rrbracket_\beta^{c \oplus b} = \llbracket \odot_{\triangleleft\{L' \rightarrow \beta, L'\}}^{I \cup U}(a(\kappa) \triangleleft a') \rrbracket_\beta^{c \oplus b}.$$

Note that if $L' = \emptyset$ (i.e., if κ is of type $\emptyset \rightarrow \theta$) then $s = \odot_\kappa^{I \cup U}(\varepsilon)$ and in this case the condition that $\llbracket s \rrbracket_\theta^{c \oplus b} \neq \perp_\theta$ is automatically satisfied, since here $\llbracket s \rrbracket_\theta^{c \oplus b} = f_\kappa(\varepsilon)$.

Lemma 6.3.7 *The replacement rule (R4) is valid.*

Proof Here $\llbracket s \rrbracket_\theta^{c\oplus b} = \llbracket \odot_\kappa^{I\cup U}(a') \rrbracket_\theta^{c\oplus b} = f_\kappa((\llbracket a' \rrbracket_\diamond^{c\oplus b})^{L'}) \neq \perp_\theta$ and thus

$$\begin{aligned}
\llbracket \odot_\zeta^{I\cup U}(s \triangleleft a) \rrbracket_\beta^{c\oplus b} &= f_{\triangleleft\{L \rightarrow \beta, L\}}(p(\zeta) \triangleleft (\llbracket s \triangleleft a \rrbracket_\diamond^{c\oplus b})^{\theta \cdot K_{\theta, \beta}}) \\
&= f_{\triangleleft\{L \rightarrow \beta, L\}}(\text{case}_\beta^\theta \triangleleft (\llbracket s \rrbracket_\theta^{c\oplus b} \triangleleft (\llbracket a \rrbracket_\diamond^{c\oplus b})^{K_{\theta, \beta}})) \\
&= f_{\triangleleft\{L \rightarrow \beta, L\}}(\text{case}_\beta^\theta \triangleleft (f_\kappa((\llbracket a' \rrbracket_\diamond^{c\oplus b})^{L'}) \triangleleft (\llbracket a \rrbracket_\diamond^{c\oplus b})^{K_{\theta, \beta}})) \\
&= f_{\triangleleft\{L' \rightarrow \beta, L'\}}((\llbracket a \rrbracket_\diamond^{c\oplus b})^{K_{\theta, \beta}}(\kappa) \triangleleft (\llbracket a' \rrbracket_\diamond^{c\oplus b})^{L'}) \\
&= f_{\triangleleft\{L' \rightarrow \beta, L'\}}(\llbracket a(\kappa) \rrbracket_{L' \rightarrow \beta}^{c\oplus b} \triangleleft (\llbracket a' \rrbracket_\diamond^{c\oplus b})^{L'}) \\
&= \llbracket \odot_{\triangleleft\{L' \rightarrow \beta, L'\}}^{I\cup U}(a(\kappa) \triangleleft a') \rrbracket_\beta^{c\oplus b},
\end{aligned}$$

where $L = \theta \cdot K_{\theta, \beta}$. \square

If (D_B, f_K) is a fully regular bottomed extension of (X_B, p_K) (so in particular if it is initial) then the condition in rule (R4) (that $\llbracket s \rrbracket_\theta^{c\oplus b} \neq \perp_\theta$) is always satisfied for any $c \in D_\diamond^I$, $b \in D_\diamond^U$, since here

$$\llbracket s \rrbracket_\theta^d = \llbracket \odot_\kappa^{I\cup U}(a') \rrbracket_\theta^d = f_\kappa((\llbracket a' \rrbracket_\diamond^d)^{L'}) \neq \perp_\theta$$

for all $d \in D_\diamond^{I\cup U}$. Therefore if (D_B, f_K) is fully regular and s has the form $\odot_\kappa^{I\cup U}(a')$ for some $\kappa \in K_\theta$ then this rule can always be applied to the element $\odot_\zeta^{I\cup U}(s \triangleleft a)$.

Example 6.3.2 Let α be as in Example 6.2.1 and consider the sequence $\{e_n\}_{0 \leq n \leq 20}$ of elements of E_{ipr}^I listed below:

```

pfb 2 (Pair 0 1)
case (eq 2 0) of {True -> Pair 0 1,
                 False -> pfb (sub 2 1) (it (Pair 0 1))}
case False of {True -> Pair 0 1,
               False -> pfb (sub 2 1) (it (Pair 0 1))}
pfb (sub 2 1) (it (Pair 0 1))
case (eq (sub 2 1) 0) of {True -> it (Pair 0 1),
                        False -> pfb (sub (sub 2 1) 1) (it (it (Pair 0 1)))}
case (eq 1 0) of {True -> it (Pair 0 1),
                 False -> pfb (sub (sub 2 1) 1) (it (it (Pair 0 1)))}
case False of {True -> it (Pair 0 1),
               False -> pfb (sub (sub 2 1)) 1 (it (it (Pair 0 1)))}
pfb (sub (sub 2 1) 1) (it (it (Pair 0 1)))
case (eq (sub (sub 2 1) 1) 0) of {True -> it (it (Pair 0 1)),
                                False -> pfb (sub (sub (sub 2 1) 1) 1)
                                             (it (it (it (Pair 0 1))))}
case (eq (sub 1 1) 0) of {True -> it (it (Pair 0 1)),
                         False -> pfb (sub (sub (sub 2 1) 1) 1)
                                       (it (it (it (Pair 0 1))))}
case (eq 0 0) of {True -> it (it (Pair 0 1)),
                 False -> pfb (sub (sub (sub 2 1) 1) 1)
                               (it (it (it (Pair 0 1))))}
case True of {True -> it (it (Pair 0 1)),
              False -> pfb (sub (sub (sub 2 1) 1) 1)
                           (it (it (it (Pair 0 1))))}
it (it (Pair 0 1))
case (it (Pair 0 1)) of {Pair -> ita}
case (case (Pair 0 1) of {Pair -> ita}) of {Pair -> ita}
case (ita 0 1) of {Pair -> ita}

```

(Example 6.3.2 is continued on the next page.)

Example 6.3.2 (continued)

```

case (Pair 1 (add 0 1)) of {Pair -> ita}
case (Pair 1 1) of {Pair -> ita}
ita 1 1
Pair 1 (add 1 1)
Pair 1 2

```

Then $\llbracket e_{n+1} \rrbracket_{\text{ipr}}^c = \llbracket e_n \rrbracket_{\text{ipr}}^c$ for $n = 0, \dots, 19$ for each $c \in \text{Sol}(\alpha)$. (Each of these assertions can be justified using one of the four rules introduced in this section.) Thus for each $c \in \text{Sol}(\alpha)$

$$\begin{aligned} \llbracket \text{pfb 2 (Pair 0 1)} \rrbracket_{\text{ipr}}^c &= \llbracket e_0 \rrbracket_{\text{ipr}}^c \\ &= \llbracket e_{20} \rrbracket_{\text{ipr}}^c = \llbracket \text{Pair 1 2} \rrbracket_{\text{ipr}}^c = (1, 2). \end{aligned}$$

Example 6.3.3 Again let α be as in Example 6.2.1. The following assertions can be justified using the four replacement rules. Let $U = \{\mathbf{n}, \mathbf{p}\}$, with \mathbf{n} of type `int` and \mathbf{p} of type `ipr`. Put $e = \text{pfb (sub n 1) (it p)}$. Then

$$\llbracket \text{pfb n p} \rrbracket_{\text{ipr}}^{c \oplus b} = \llbracket \text{case (eq n 0) of \{True -> p, False -> e\}} \rrbracket_{\text{ipr}}^{c \oplus b}$$

for all $c \in \text{Sol}(\alpha)$ and all $b \in D_\diamond^U$, and

$$\begin{aligned} &\llbracket \text{case (eq n 0) of \{True -> p, False -> e\}} \rrbracket_{\text{ipr}}^{c \oplus b} \\ &= \llbracket \text{case True of \{True -> p, False -> e\}} \rrbracket_{\text{ipr}}^{c \oplus b} = \llbracket \mathbf{p} \rrbracket_{\text{ipr}}^{c \oplus b} \end{aligned}$$

for all $c \in \text{Sol}(\alpha)$ and all $b \in D_\diamond^U$ with $b_{\mathbf{n}} = 0$. Moreover

$$\begin{aligned} &\llbracket \text{case (eq n 0) of \{True -> p, False -> e\}} \rrbracket_{\text{ipr}}^{c \oplus b} \\ &= \llbracket \text{case False of \{True -> p, False -> e\}} \rrbracket_{\text{ipr}}^{c \oplus b} = \llbracket \mathbf{e} \rrbracket_{\text{ipr}}^{c \oplus b} \end{aligned}$$

for all $c \in \text{Sol}(\alpha)$ and all $b \in D_\diamond^U$ with $b_{\mathbf{n}} \in \mathbb{Z} \setminus \{0\}$.

(Example 6.3.3 is continued on the next page.)

Example 6.3.3 (continued) Thus

$$c_{\text{pfb}}(b_n, b_p) = \llbracket \text{pfb } n \text{ p} \rrbracket_{\text{ipr}}^{c \oplus b} = \llbracket \text{p} \rrbracket_{\text{ipr}}^{c \oplus b} = b_p$$

for all $c \in \text{Sol}(\alpha)$ and all $b \in D_{\diamond}^U$ with $b_n = 0$, and

$$\begin{aligned} c_{\text{pfb}}(b_n, b_p) &= \llbracket \text{pfb } n \text{ p} \rrbracket_{\text{ipr}}^{c \oplus b} = \llbracket \text{pfb (sub } n \text{ 1) (it p)} \rrbracket_{\text{ipr}}^{c \oplus b} \\ &= c_{\text{pfb}}(\text{sub}(b_n, 1), c_{\text{it}}(b_p)) = c_{\text{pfb}}(b_n - 1, c_{\text{it}}(b_p)) \end{aligned}$$

for all $c \in \text{Sol}(\alpha)$ and all $b \in D_{\diamond}^U$ with $b_n \in \mathbb{Z} \setminus \{0\}$. This implies that if $c \in \text{Sol}(\alpha)$ and $p \in D_{\text{ipr}}$ then $c_{\text{pfb}}(0, p) = p$ and for all $n \in \mathbb{Z} \setminus \{0\}$

$$c_{\text{pfb}}(n, p) = c_{\text{pfb}}(n - 1, c_{\text{it}}(p)) .$$

Now let $U' = \{\mathbf{a}, \mathbf{b}\}$ with \mathbf{a} and \mathbf{b} of type `int`. Using the replacement rules it then follows that

$$\begin{aligned} &\llbracket \text{it (Pair a b)} \rrbracket_{\text{ipr}}^{c \oplus b} \\ &= \llbracket \text{case (Pair a b) of \{Pair -> ita\}} \rrbracket_{\text{ipr}}^{c \oplus b} \\ &= \llbracket \text{ita a b} \rrbracket_{\text{ipr}}^{c \oplus b} = \llbracket \text{Pair b (add a b)} \rrbracket_{\text{ipr}}^{c \oplus b} \end{aligned}$$

for all $c \in \text{Sol}(\alpha)$ and all $b \in D_{\diamond}^{U'}$ with $b_a, b_b \in \mathbb{Z}$. Thus

$$\begin{aligned} c_{\text{it}}((b_a, b_b)) &= \llbracket \text{it (Pair a b)} \rrbracket_{\text{ipr}}^{c \oplus b} \\ &= \llbracket \text{Pair b (add a b)} \rrbracket_{\text{ipr}}^{c \oplus b} = (b_b, b_a + b_b) \end{aligned}$$

for all $c \in \text{Sol}(\alpha)$ and all $b \in D_{\diamond}^{U'}$ with $b_a, b_b \in \mathbb{Z}$, and hence

$$c_{\text{it}}((a, b)) = (b, a + b)$$

for all $c \in \text{Sol}(\alpha)$ and all $a, b \in \mathbb{Z}$. This implies that if $c \in \text{Sol}(\alpha)$ and $m \in \mathbb{N}$ then $c_{\text{pfb}}(0, (f_m, f_{m+1})) = (f_m, f_{m+1})$ and

$$\begin{aligned} c_{\text{pfb}}(n, (f_m, f_{m+1})) &= c_{\text{pfb}}(n - 1, c_{\text{it}}((f_m, f_{m+1}))) \\ &= c_{\text{pfb}}(n - 1, (f_{m+1}, f_m + f_{m+1})) = c_{\text{pfb}}(n - 1, (f_{m+1}, f_{m+2})) \end{aligned}$$

for all $n > 1$. Thus using induction on n it follows that

$$c_{\text{pfb}}(n, (f_m, f_{m+1})) = (f_{m+n}, f_{m+n+1})$$

for all $m, n \in \mathbb{N}$. In particular, this means that for all $n \in \mathbb{N}$

$$c_{\text{pfb}}(n, (0, 1)) = (f_n, f_{n+1}) .$$

6.4 Computing values

Very roughly speaking, ‘computing values’ means something like the following: Given a system of equations $\alpha \in \text{Eq}(I)$ and an element $s \in F_\sigma^I$, the value $\llbracket s \rrbracket_\sigma^c \in D_\sigma$ should be determined for some, or for all, of the solutions $c \in \text{Sol}(\alpha)$. Now the standpoint will be taken that the only way of directly representing an element from one of the sets in the family D_S is with the help of the ground algebra (F_B, \odot_K) . This means that the only hope is to compute values which lie in the family X_B . (The restriction here to ground types can hardly be avoided, since there is no way of directly representing an element in the set D_σ when $\sigma \in S \setminus B$ is a functional type. The restriction to the family X_B , i.e., the exclusion of values in the family $\{D_\beta \setminus X_\beta : \beta \in B\}$, could be avoided by making use of a suitable extension of the ground algebra (F_B, \odot_K) , but to keep things as simple as possible this possibility will not be considered here.)

A further restriction will be imposed. This is based on the plausible assumption that a simple explicit algorithm will be unable to distinguish between different solutions in $\text{Sol}(\alpha)$, and hence only values $\llbracket s \rrbracket_\beta^c$ which are independent of $c \in \text{Sol}(\alpha)$ can be computed. (Even if this assumption is not acceptable in general, the argument in Chapter 1 shows that it will be certainly satisfied by an algorithm based on interpreting equations as replacement rules.)

In what follows let I be a global S -typed set and let $\alpha \in \text{Eq}(I)$ be a system of equations with $\text{Sol}(\alpha) \neq \emptyset$. The above restrictions suggest looking at the family F_B^α with $F_B^\alpha \subset F_B^I$ defined for each $\beta \in B$ by

$$F_\beta^\alpha = \{s \in F_\beta^I : \text{there exists } x \in X_\beta \text{ such that } \llbracket s \rrbracket_\beta^c = x \text{ for all } c \in \text{Sol}(\alpha)\}.$$

In particular, $F_B \subset F_B^\alpha$, since if $s \in F_\beta$ then by Lemma 6.1.1 $\llbracket s \rrbracket_\beta^c = \llbracket s \rrbracket_\beta$ for all $c \in D_\sigma^I$ (and so in particular for all $c \in \text{Sol}(\alpha)$). Example 6.4.1 on the next page illustrates this definition using the equations first considered in Chapter 1.

The problem of ‘computing values’ can now be formulated as the following question: For which elements $s \in F_B^\alpha$ is it possible to compute the value of s , i.e., the unique element $x \in X_\beta$ such that $\llbracket s \rrbracket_\beta^c = x$ for all $c \in \text{Sol}(\alpha)$?

It is important to note that, although $F_B^\alpha \subset F_B^I$ and (D_S, f_N) is not involved in the definition of F_B^I , the family F_B^α will in general depend on this Σ -algebra. Example 6.4.1 shows in particular how F_B^α can depend on the Λ -algebra (D_B, f_K) .

In what follows the Λ -algebra (F_B^I, \odot_K^I) will play a more important role than the whole of the Σ -algebra (F_S^I, \odot_N^I) . This means that if $c \in D_\sigma^I$ is an assignment then it is usually only the Λ -homomorphism $\llbracket \cdot \rrbracket_B^c$ from (F_B^I, \odot_K^I) to (D_B, f_K) which will be of interest (and not the whole Σ -homomorphism $\llbracket \cdot \rrbracket_S^c$).

It is convenient to state the problem of ‘computing values’ in a slightly different form, and for this the following simple fact is needed:

Lemma 6.4.1 *The family F_B^α is invariant in the Λ -algebra (F_B^I, \odot_K^I) .*

Example 6.4.1 Again let $\alpha \in \text{Eq}(I)$ be as in Example 6.2.1.

Let $e \in E_{\text{int}}^I$ be the term `fib 4`; then $e \in E_{\text{int}}^\alpha$, since (as already stated in Section 1.2) $\llbracket e \rrbracket_{\text{int}}^c = 3$ for all $c \in \text{Sol}(\alpha)$, and this is independent of (D_B, f_K) and D_S .

Now let $e' \in E_{\text{int}}^I$ be the term `fst (Pair 1 (fib -1))`. Then

$$\llbracket e' \rrbracket_{\text{int}}^c = \begin{cases} 1 & \text{if } f_{\text{Pair}}(1, c_{\text{fib}}(-1)) \neq \perp_{\text{ipr}}, \\ \perp_{\text{int}} & \text{otherwise,} \end{cases}$$

for all $c \in \text{Sol}(\alpha)$. Thus if (D_B, f_K) is a fully regular extension of (X_B, p_K) then $\llbracket e' \rrbracket_{\text{int}}^c = 1$ for all $c \in \text{Sol}(\alpha)$, and hence in this case $e' \in E_{\text{int}}^\alpha$.

On the other hand, if (D_S, f_N) is as in Case 1 with (D_B, f_K) the flat extension of (X_B, p_K) then there exists a solution $c \in \text{Sol}(\alpha)$ with $c_{\text{fib}}(-1) = \perp_{\text{int}}$ and then $\llbracket e' \rrbracket_{\text{int}}^c = \perp_{\text{int}}$. This implies that here $e' \notin E_{\text{int}}^\alpha$.

Of course, this example is in essence the same as Example 6.2.3.

Proof If $\kappa \in K$ is of type $\emptyset \rightarrow \beta$ then $\odot_\kappa^I(\varepsilon) \in F_\beta^\alpha$, since $\llbracket \odot_\kappa^I(\varepsilon) \rrbracket_\beta^c = f_\kappa(\varepsilon)$ for all $c \in D_\emptyset^I$ and $f_\kappa(\varepsilon) = p_\kappa(\varepsilon) \in X_\beta$. Suppose then that $\kappa \in K$ is of type $L \rightarrow \beta$ with $L \neq \emptyset$ and let $b \in (F_\emptyset^\alpha)^L$; then for each $\eta \in L$ there exists $x_\eta \in X_\eta$ such that $\llbracket b(\eta) \rrbracket_\eta^c = x_\eta$ for all $c \in \text{Sol}(\alpha)$. Let $b' \in X_\emptyset^L$ be the assignment given by $b'(\eta) = x_\eta$ for each $\eta \in L$; then for all $c \in \text{Sol}(\alpha)$

$$\llbracket \odot_\kappa^I(b) \rrbracket_\beta^c = f_\kappa(\llbracket b \rrbracket_\emptyset^c) = f_\kappa(b'),$$

and $f_\kappa(b') = p_\kappa(b') \in X_\beta$; thus $\odot_\kappa^I(b) \in F_\beta^\alpha$. This means that the family F_B^α is invariant in (F_B^I, \odot_K^I) . \square

For each $\kappa \in K$ of type $L \rightarrow \beta$ let \odot_κ^α denote the restriction of \odot_κ^I to $(F_\emptyset^\alpha)^L$. By Lemma 6.4.1 $(F_B^\alpha, \odot_K^\alpha)$ is then a Λ -algebra which is an extension of the ground term algebra (F_B, \odot_K) .

Now for each $\beta \in B$ define a mapping $\llbracket \cdot \rrbracket_\beta^\alpha : F_\beta^\alpha \rightarrow X_\beta$ by putting $\llbracket s \rrbracket_\beta^\alpha = x$, where x is the (unique) element of X_β such that $\llbracket s \rrbracket_\beta^c = x$ for all $c \in \text{Sol}(\alpha)$. In other words, $\llbracket s \rrbracket_\beta^\alpha$ is defined so that $\llbracket s \rrbracket_\beta^\alpha = \llbracket s \rrbracket_\beta^c$ for all $c \in \text{Sol}(\alpha)$.

Lemma 6.4.2 *The family $\llbracket \cdot \rrbracket_B^\alpha$ is a Λ -homomorphism from $(F_B^\alpha, \odot_K^\alpha)$ to (X_B, p_K) which extends the Λ -isomorphism $\llbracket \cdot \rrbracket_B : (F_B, \odot_K) \rightarrow (X_B, p_K)$ (i.e., $\llbracket s \rrbracket_\beta^\alpha = \llbracket s \rrbracket_\beta$ for all $s \in F_\beta$).*

Proof This is what the proof of Lemma 6.4.1 shows. \square

The problem of ‘computing values’ can now be stated in terms of the following question: For which elements $s \in F_\beta^\alpha$ is it possible to compute the unique ground element $s' \in F_\beta$ such that $\llbracket s' \rrbracket_\beta = \llbracket s \rrbracket_\beta^\alpha$?

To answer this question some kind of algorithm is needed which will manipulate the elements from the sets in the family F_B^I : In response to the ‘input’ s it should either produce the ‘output’ s' (whenever possible) or produce no output at all. Such an algorithm is presented in Section 6.5, and is given in terms of a Λ -homomorphism from the Λ -algebra (F_B^I, \odot_K^I) to itself.

In what follows let $\Phi_B : (F_B^I, \odot_K^I) \rightarrow (F_B^I, \odot_K^I)$ be a Λ -homomorphism. Some of the elementary properties of such a homomorphism will now be looked at with a view to seeing how Φ_B could form the basis of an algorithm for ‘computing values’.

Lemma 6.4.3 $\Phi_\beta(s) = s$ for each ground term $s \in F_\beta$.

Proof For each $\beta \in B$ let Φ'_β be the restriction of Φ_β to the ground terms F_β ; then $\Phi'_\beta : (F_B, \odot_K) \rightarrow (F_B^I, \odot_K^I)$ is a Λ -homomorphism. But a Λ -homomorphism $\text{id}'_\beta : (F_B, \odot_K) \rightarrow (F_B^I, \odot_K^I)$ can also be defined by letting $\text{id}'_\beta(s) = s$ for all $s \in F_\beta$. Hence by Proposition 2.2.3 (1) $\Phi'_\beta = \text{id}'_\beta$, since (F_B, \odot_K) is minimal, i.e., $\Phi_\beta(s) = s$ for each $s \in F_\beta$. \square

The iterates $\{\Phi_B^n\}_{n \geq 0}$ of Φ_B will also be needed: For each $\beta \in B$ the mappings $\Phi_\beta^n : F_\beta^I \rightarrow F_\beta^I$, $n \geq 0$, are defined by letting $\Phi_\beta^0(s) = s$ for each $s \in F_\beta^I$, $\Phi_\beta^1 = \Phi_\beta$ and (for $n > 0$) $\Phi_\beta^n = \Phi_\beta \Phi_\beta^{n-1}$. By Proposition 2.2.1 Φ_β^n is a Λ -homomorphism for each $n \geq 0$. Now for each $\beta \in B$ let

$$F_\beta^\Phi = \{s \in F_\beta^I : \Phi_\beta^n(s) \in F_\beta \text{ for some } n \geq 0\}.$$

Lemma 6.4.4 The family F_B^Φ is invariant in the Λ -algebra (F_B^I, \odot_K^I) .

Proof This is clear, since Φ_B is a Λ -homomorphism. \square

For each $\kappa \in K$ of type $L \rightarrow \beta$ let \odot_κ^Φ denote the restriction of \odot_κ^I to $(F_\circ^\Phi)^L$. By Lemma 6.4.4 (F_B^Φ, \odot_K^Φ) is then a Λ -algebra which is an extension of the ground term algebra (F_B, \odot_K) .

By Lemma 6.4.3 $\Phi_\beta(s) = s$ for each $s \in F_\beta$ and hence a mapping $\Phi_\beta^\infty : F_\beta^\Phi \rightarrow F_\beta$ can be defined by putting $\Phi_\beta^\infty(s) = \Phi_\beta^n(s)$, where $n \geq 0$ is chosen so that $\Phi_\beta^n(s) \in F_\beta$.

Lemma 6.4.5 $\Phi_B^\infty : (F_B^\Phi, \odot_K^\Phi) \rightarrow (F_B, \odot_K)$ is a Λ -homomorphism.

Proof This again follows from the fact that Φ_B is a Λ -homomorphism. \square

The homomorphism Φ_B is now said to be α -consistent if

$$\llbracket \Phi_\beta(s) \rrbracket_\beta^c = \llbracket s \rrbracket_\beta^c$$

for all $s \in F_\beta^I$, $\beta \in B$, and for all $c \in \text{Sol}(\alpha)$. It is important to note that the α -consistency of Φ_B depends on the Σ -algebra (D_S, f_N) .

Proposition 6.4.1 *Suppose Φ_B is α -consistent; then $F_B^\Phi \subset F_B^\alpha$. Moreover, $\Phi_\beta^\infty(s)$ is the (unique) ground term which denotes $\llbracket s \rrbracket_\beta^\alpha$ for all $s \in F_\beta^\Phi$, $\beta \in B$, i.e.,*

$$\llbracket s \rrbracket_\beta^\alpha = \llbracket \Phi_\beta^\infty(s) \rrbracket_\beta$$

for all $s \in F_\beta^\Phi$, $\beta \in B$.

Proof Let $s \in F_\beta^\Phi$; then for each $n > 0$ it follows that

$$\llbracket \Phi_\beta^n(s) \rrbracket_\beta^c = \llbracket \Phi_\beta(\Phi_\beta^{n-1}(s)) \rrbracket_\beta^c = \llbracket \Phi_\beta^{n-1}(s) \rrbracket_\beta^c$$

and thus $\llbracket \Phi_\beta^\infty(s) \rrbracket_\beta^c = \llbracket \Phi_\beta^0(s) \rrbracket_\beta^c = \llbracket s \rrbracket_\beta^c$ for each $c \in \text{Sol}(\alpha)$. But $\Phi_\beta^\infty(s) \in F_\beta$ and so by Lemma 6.1.1 $\llbracket \Phi_\beta^\infty(s) \rrbracket_\beta^{c'} = \llbracket \Phi_\beta^\infty(s) \rrbracket_\beta$ for all $c' \in D_\diamond^I$. Hence

$$\llbracket s \rrbracket_\beta^c = \llbracket \Phi_\beta^\infty(s) \rrbracket_\beta$$

for all $c \in \text{Sol}(\alpha)$. This shows that $s \in F_\beta^\alpha$ and also that $\llbracket s \rrbracket_\beta^\alpha = \llbracket \Phi_\beta^\infty(s) \rrbracket_\beta$. \square

If Φ_B is α -consistent then by Proposition 6.4.1 the iterates $\{\Phi_\beta^n\}_{n \geq 0}$ of Φ_β can be used to compute the element denoting $\llbracket s \rrbracket_\beta^\alpha$ for each $s \in F_\beta^\Phi$. Of course, the iterates give no information when starting with an element $s \in F_\beta^\alpha \setminus F_\beta^\Phi$.

Consider for a moment the identity Λ -homomorphism $\text{id}_B : (F_B^I, \odot_K^I) \rightarrow (F_B^I, \odot_K^I)$ (given by $\text{id}_\beta(s) = s$ for all $s \in F_\beta^I$). In this case $(F_B^{\text{id}}, \odot_K^{\text{id}}) = (F_B, \odot_K)$, and id_B^∞ is just the identity Λ -homomorphism from $(F_B^{\text{id}}, \odot_K^{\text{id}}) = (F_B, \odot_K)$ to itself. Moreover, id_B is always trivially α -consistent. But, of course, applying Proposition 6.4.1 to id_B does not really provide any information at all.

The best to be hoped for is an α -consistent Λ -homomorphism Φ_B with $F_B^\Phi = F_B^\alpha$ and such a homomorphism Φ_B will be called α -complete. If Φ_B is α -complete then Proposition 6.4.1 implies that $\Phi_\beta^\infty(s)$ is the ground element denoting $\llbracket s \rrbracket_\beta^\alpha$ for all $s \in F_\beta^\alpha$. (Note, however, that the iterates $\{\Phi_\beta^n\}_{n \geq 0}$ cannot be used to effectively decide that a given term $s \in F_\beta^I$ is an element of $F_\beta^I \setminus F_\beta^\alpha$.)

In Section 6.5 an α -consistent Λ -homomorphism Φ_B is constructed for each system of equations $\alpha \in \text{Eq}(I)$. Then in Chapter 7 it is shown that in Case 3 Φ_B is α -complete.

Finally, note that the definition of Φ_B being α -consistent still makes sense when $\text{Sol}(\alpha) = \emptyset$ (although in this case every Λ -homomorphism is trivially α -consistent). Thus in the next section, where an α -consistent Λ -homomorphism Φ_B is constructed, it is not necessary to assume that $\text{Sol}(\alpha) \neq \emptyset$.

6.5 An algorithm for computing values

Again let I be a global S -typed set and let $\alpha = s_I \in \text{Eq}(I)$ be a system of equations. In this section an α -consistent Λ -homomorphism

$$\Phi_B : (F_B^I, \odot_K^I) \rightarrow (F_B^I, \odot_K^I)$$

will be defined which, as explained in Section 6.4, will form the basis of an algorithm for computing values.

The Λ -homomorphism Φ_B will depend, of course, on the system of equations α . Moreover, Φ_B will also depend on (D_S, f_N) , because its definition involves choosing what is called a *support system*: This is a family N_B^I with $F_B \subset N_B^I \subset \kappa F_B^I$ such that $\llbracket s \rrbracket_\beta^c \neq \perp_\beta$ for all $s \in N_\beta^I$, $\beta \in B$ and for all $c \in D_\circ^I$.

The family F_B provides a somewhat trivial example of a support system (since by Lemma 6.1.1 $\llbracket s \rrbracket_\beta^c = \llbracket s \rrbracket_\beta \in X_\beta$ for all $s \in F_\beta$, $c \in D_\circ^I$). However, except when (D_B, f_K) is the flat extension of (X_B, p_K) this is not very effective. A much better choice is the family HF_B^I which was introduced in Section 6.1. Recall that

$$HF_\beta^I = \{s \in F_\beta^I : \Omega_\beta^I(s) \in R_\beta\}$$

for each $\beta \in B$, where R_B is the trace of (D_B, f_K) and $\Omega_S^I : (F_S^I, \odot_N^I) \rightarrow (F_S^b, \odot_N^b)$ is the unique Σ -homomorphism such that $\Omega_\xi^I(\xi) = b_\xi$ for each $\xi \in I \cup P$, and that by Proposition 6.1.4 HF_B^I is a support system.

In fact, this is really the only support system of interest (since it turns out to be the ‘correct’ one). Nevertheless, in the present section the results are first formulated in terms of an arbitrary support system.

Note that HF_B^I only depends on (D_B, f_K) . However, whether or not a given family N_B^I is a support system can in general depend on the whole of (D_S, f_N) and not just on (D_B, f_K) .

If L is an S -typed set disjoint from I and P and $a \in (F_\circ^I)^L$ is an assignment then for each term $s \in F_\sigma^{I \cup L}$ there is the element $s[a/L]$ of F_σ^I defined in Section 6.3 (here with $U = \emptyset$). More precisely, this means that

$$s[a/L] = \delta_\sigma^a(s)$$

where $\delta_S^a : (F_S^{I \cup L}, \odot_N^{I \cup L}) \rightarrow (F_S^I, \odot_N^I)$ is the unique Σ -homomorphism such that $\delta_\xi^a(\xi) = \xi$ for each $\xi \in I \cup P$ and $\delta_\eta^a(\eta) = a(\eta)$ for each $\eta \in L$. If $L = \emptyset$ then it is easy to see that δ_S^\emptyset is just the identity homomorphism $\text{id}_S : (F_S^I, \odot_N^I) \rightarrow (F_S^I, \odot_N^I)$.

The following result is stated in terms of a concept weaker than that of a support system: The family N_B^I is said to be a *pre-support system* if just $F_B \subset N_B^I \subset \kappa F_B^I$. (This notion has nothing to do with (D_S, f_N) .)

Note that if N_B^I is a support system and \check{N}_B^I a pre-support system with $\check{N}_B^I \subset N_B^I$ then \check{N}_B^I is also a support system.

Proposition 6.5.1 *Let N_B^I be a pre-support system. Then there exists a unique Λ -homomorphism $\Phi_B : (F_B^I, \odot_K^I) \rightarrow (F_B^I, \odot_K^I)$ such that the following hold:*

(i) *Let $\xi \in I$ be of type $L \rightarrow \beta$; then for all $a \in (F_\diamond^I)^L$*

$$\Phi_\beta(\odot_\xi^I(a)) = s_\xi[a/L].$$

In particular, $\Phi_\beta(\xi) = s_\xi$ if $L = \emptyset$ (in which case ξ is of type β).

(ii) *Let $\zeta = \text{case}_\beta^\theta$ for some $\beta \in B$, $\theta \in B_f$; then*

$$\Phi_\beta(\odot_\zeta^I(s \triangleleft a)) = \begin{cases} \odot_\zeta^I(\Phi_\theta(s) \triangleleft a) & \text{if } s \notin N_\theta^I, \\ \odot_{\triangleleft\{L' \rightarrow \beta, L'\}}^I(a(\kappa) \triangleleft a') & \text{if } s = \odot_\kappa^I(a') \in N_\theta^I \\ & \text{with } \kappa \in K \text{ of type } L' \rightarrow \theta, \end{cases}$$

for all $a \in (F_\diamond^I)^{K_{\theta, \beta}}$, $s \in F_\theta^I$. (Note that, since $N_\theta^I \subset \kappa F_\theta^I$, s does have a unique representation of the form $\odot_\kappa^I(a')$ for some $\kappa \in K_\theta$ whenever $s \in N_\theta^I$.)

(iii) *Let $\zeta \in P$ be the name of an integer operator (so ζ is of type $\text{int}_2 \rightarrow \beta$ with β either int or bool) and let $s_1, s_2 \in F_{\text{int}}^I$. Then*

$$\Phi_\beta(\odot_\zeta^I(s_1, s_2)) = \begin{cases} \text{Eval}_\zeta(s_1, s_2) & \text{if } s_1, s_2 \in F_{\text{int}}, \\ \odot_\zeta^I(\Phi_{\text{int}}(s_1), \Phi_{\text{int}}(s_2)) & \text{otherwise,} \end{cases}$$

where $\text{Eval}_\zeta : F_{\text{int}} \times F_{\text{int}} \rightarrow F_\beta$ is the unique mapping such that

$$\llbracket \text{Eval}_\zeta(s_1, s_2) \rrbracket_\beta = f_{\triangleleft\{\text{int}_2 \rightarrow \beta, \text{int}_2\}}(p(\zeta) \triangleleft (\llbracket s_1 \rrbracket_{\text{int}}, \llbracket s_2 \rrbracket_{\text{int}}))$$

for all $s_1, s_2 \in F_{\text{int}}$.

Proof Let ℓ_S be the family of mappings obtained by applying Lemma 2.3.4 to the $\Sigma^{I \cup P}$ -algebra $(F_S^I, \odot_{N^{I \cup P}}^I)$. (This $\Sigma^{I \cup P}$ -algebra is initial and so minimal and regular.) Thus ℓ_S is the unique family with $\ell_\sigma : F_\sigma^I \rightarrow \mathbb{N}$ for each $\sigma \in S$ such that:

(i) If $\kappa \in K$ is of type $\emptyset \rightarrow \beta$ then $\ell_\beta(\odot_\kappa^I(\varepsilon)) = 0$.

(ii) If $\kappa \in K$ is of type $L \rightarrow \beta$ with $L \neq \emptyset$ and $a \in (F_\diamond^I)^L$ then

$$\ell_\beta(\odot_\kappa^I(a)) = 1 + \max\{\ell_\eta(a(\eta)) : \eta \in L\}.$$

(iii) $\ell_\xi(\xi) = 0$ for each $\xi \in I \cup P$.

(iv) If $\xi \in I \cup P$ is of type $L \rightarrow \beta$ and J is a non-empty subset of L then

$$\ell_{L \setminus J \rightarrow \beta}(\odot_{\xi\{J\}}^I(a)) = 1 + \max\{\ell_\eta(a(\eta)) : \eta \in J\}$$

for all $a \in (F_\diamond^I)^J$.

$\Phi_\beta(s)$ will be defined by induction on $\ell_\beta(s)$, making use of the unique representations given in Proposition 6.1.1 (1). Suppose first that $s \in F_\beta^I$ with $\ell_\beta(s) = 1$. Then either

$s = \odot_{\kappa}^I(\varepsilon)$ for some $\kappa \in K$ of type $\emptyset \rightarrow \beta$, in which case $\Phi_{\beta}(s)$ is defined to be s , or $s = \xi$ for some $\xi \in I$ of type β and here $\Phi_{\beta}(s)$ is defined to be s_{ξ} .

Now let $k > 1$ and suppose for each $\theta \in B$ that $\Phi_{\theta}(t)$ has already been defined for all $t \in F_{\theta}^I$ with $\ell_{\theta}(t) < k$. Consider $s \in F_{\beta}^I$ with $\ell_{\beta}(s) = k$; there are three cases:

1. The element s has the form $\odot_{\kappa}^I(a)$, where $\kappa \in K$ is of type $L \rightarrow \beta$ with $L \neq \emptyset$ and $a \in (F_{\circ}^I)^L$. Then, since $\ell_{\eta}(a(\eta)) < k$ for each $\eta \in L$, $\Phi_{\beta}(s)$ can be defined to be $\odot_{\kappa}^I(\Phi_{\circ}^L(a))$.
2. The element s has the form $\odot_{\xi}^I(a)$ with $\xi \in I$ of type $L \rightarrow \beta$ with $L \neq \emptyset$ and $a \in (F_{\circ}^I)^L$. Here $\Phi_{\beta}(\odot_{\xi}^I(a))$ can be defined directly using condition (i).
3. The element s has the form $\odot_{\zeta}^I(a)$, where $\zeta \in P$ is of type $L \rightarrow \beta$ with $L \neq \emptyset$ and $a \in (F_{\circ}^I)^L$. But $\ell_{\eta}(a(\eta)) < k$ for each $\eta \in L$, and so the appropriate instance of the conditions (ii) and (iii) can be used to actually define $\Phi_{\beta}(\odot_{\zeta}^I(a))$.

The uniqueness of Φ_B also follows using induction on $\ell_{\beta}(s)$. \square

The Λ -homomorphism $\Phi_B : (F_B^I, \odot_K^I) \rightarrow (F_B^I, \odot_K^I)$ given by Proposition 6.5.1 will be called the Λ -homomorphism defined by α and the pre-support system N_B^I .

Proposition 6.5.2 *Suppose N_B^I is a support system. Then the Λ -homomorphism Φ_B defined by α and N_B^I is α -consistent, i.e., $\llbracket \Phi_{\beta}(s) \rrbracket_{\beta}^c = \llbracket s \rrbracket_{\beta}^c$ for all $s \in F_{\beta}^I$, $\beta \in B$ and all $c \in \text{Sol}(\alpha)$.*

Proof The family of mappings ℓ_s introduced in Proposition 6.5.1 will be needed again, as well as the unique representations given in Proposition 6.1.1 (1). Put

$$G_{\beta} = \{s \in F_{\beta}^I : \llbracket \Phi_{\beta}(s) \rrbracket_{\beta}^c = \llbracket s \rrbracket_{\beta}^c \text{ for all } c \in \text{Sol}(\alpha)\}$$

for each $\beta \in B$; the task is thus to show that $G_B = F_B^I$. First consider $s \in F_{\beta}^I$ with $\ell_{\beta}(s) = 1$. If $s = \odot_{\kappa}^I(\varepsilon)$ for some $\kappa \in K$ of type $\emptyset \rightarrow \beta$ then $s \in G_{\beta}$ holds trivially because $\Phi_{\beta}(s) = s$. On the other hand, if $s = \xi$ for some $\xi \in I$ of type β then $\Phi_{\beta}(s) = s_{\xi}$, and thus for each $c \in \text{Sol}(\alpha)$

$$\llbracket \Phi_{\beta}(s) \rrbracket_{\beta}^c = \llbracket s_{\xi} \rrbracket_{\beta}^c = c(\xi) = \llbracket \xi \rrbracket_{\beta}^c = \llbracket s \rrbracket_{\beta}^c;$$

hence again $s \in G_{\beta}$.

Now let $k > 1$ and suppose for each $\theta \in B$ it is known that $t \in G_{\theta}$ for all $t \in F_{\theta}^I$ with $\ell_{\theta}(t) < k$. Consider $s \in F_{\beta}^I$ with $\ell_{\beta}(s) = k$; there are four cases:

1. The element s has the form $\odot_{\kappa}^I(a)$, where $\kappa \in K$ is of type $L \rightarrow \beta$ with $L \neq \emptyset$ and $a \in (F_{\circ}^I)^L$. Let $c \in \text{Sol}(\alpha)$, then $\llbracket \Phi_{\eta}(a(\eta)) \rrbracket_{\eta}^c = \llbracket a(\eta) \rrbracket_{\eta}^c$ for each $\eta \in L$, since $\ell_{\eta}(a(\eta)) < k$, and hence $(\llbracket \Phi_{\circ}^L(a) \rrbracket_{\circ}^c)^L = (\llbracket a \rrbracket_{\circ}^c)^L$. Thus

$$\begin{aligned} \llbracket \Phi_{\beta}(s) \rrbracket_{\beta}^c &= \llbracket \odot_{\kappa}^I(\Phi_{\circ}^L(a)) \rrbracket_{\beta}^c = f_{\kappa}((\llbracket \Phi_{\circ}^L(a) \rrbracket_{\circ}^c)^L) \\ &= f_{\kappa}((\llbracket a \rrbracket_{\circ}^c)^L) = \llbracket \odot_{\kappa}^I(a) \rrbracket_{\beta}^c = \llbracket s \rrbracket_{\beta}^c \end{aligned}$$

for all $c \in \text{Sol}(\alpha)$, i.e., $s \in G_{\beta}$. (This is really just an instance of rule (R1).)

2. The element s has the form $\odot_{\xi}^I(a)$ with $\xi \in I$ of type $L \rightarrow \beta$ for some $L \neq \emptyset$ and with $a \in (F_{\diamond}^I)^L$. Then by replacement rule (R2)

$$\llbracket \Phi_{\beta}(s) \rrbracket_{\beta}^c = \llbracket s_{\xi}[a/L] \rrbracket_{\beta}^c = \llbracket \odot_{\xi}^I(a) \rrbracket_{\beta}^c = \llbracket s \rrbracket_{\beta}^c$$

for all $c \in \text{Sol}(\alpha)$, i.e., $s \in G_{\beta}$.

3. The element s has the form $\odot_{\zeta}^I(s' \triangleleft a)$, where $\zeta = \text{case}_{\beta}^{\theta}$ with $\beta \in B$ and $\theta \in B_{\neq}$ and where $s' \in F_{\diamond}^I$ and $a \in (F_{\diamond}^I)^{K_{\theta, \beta}}$.

Assume first that $s' \notin N_{\theta}^I$. Then $\llbracket \Phi_{\theta}(s') \rrbracket_{\theta}^c = \llbracket s' \rrbracket_{\theta}^c$ for each $c \in \text{Sol}(\alpha)$, since $\ell_{\theta}(s') < k$, and therefore by replacement rule (R1)

$$\llbracket \Phi_{\beta}(s) \rrbracket_{\beta}^c = \llbracket \odot_{\zeta}^I(\Phi_{\theta}(s') \triangleleft a) \rrbracket_{\beta}^c = \llbracket \odot_{\zeta}^I(s' \triangleleft a) \rrbracket_{\beta}^c = \llbracket s \rrbracket_{\beta}^c$$

for all $c \in \text{Sol}(\alpha)$, which again implies that $s \in G_{\beta}$.

Assume next that $s' \in N_{\theta}^I$ and so $\llbracket s' \rrbracket_{\theta}^c \neq \perp_{\theta}$ for each $c \in D_{\diamond}^I$. Moreover, since $N_{\theta}^I \subset \kappa F_{\theta}^I$, s' has the form $\odot_{\kappa}^I(a')$ with $\kappa \in K$ of type $L' \rightarrow \theta$ for some $L' \in \mathcal{F}_B$ and with $a' \in (F_{\diamond}^I)^{L'}$. Therefore by replacement rule (R4)

$$\llbracket s \rrbracket_{\beta}^c = \llbracket \odot_{\zeta}^I(s' \triangleleft a) \rrbracket_{\beta}^c = \odot_{\triangleleft\{L' \rightarrow \beta, L'\}}^I(a(\kappa) \triangleleft a') = \llbracket \Phi_{\beta}(s) \rrbracket_{\beta}^c$$

for all $c \in D_{\diamond}^I$, and thus in particular for all $c \in \text{Sol}(\alpha)$, i.e., $s \in G_{\beta}$.

4. The element s has the form $\odot_{\zeta}^I(s_1, s_2)$ with $\zeta \in P$ the name of an integer operator (so ζ is of type $\text{int}_2 \rightarrow \beta$ with β either int or bool) and $s_1, s_2 \in F_{\text{int}}^I$.

Assume first that both s_1 and s_2 are ground terms; then by replacement rule (R3')

$$\llbracket \Phi_{\beta}(s) \rrbracket_{\beta}^c = \llbracket \text{Eval}_{\zeta}(s_1, s_2) \rrbracket_{\beta}^c = \llbracket \odot_{\zeta}^I(s_1, s_2) \rrbracket_{\beta}^c = \llbracket s \rrbracket_{\beta}^c$$

for all $c \in D_{\diamond}^I$, and thus in particular for all $c \in \text{Sol}(\alpha)$, i.e., $s \in G_{\beta}$.

Finally, assume that not both of s_1 and s_2 are ground terms. Then, since $\ell_{\text{int}}(s_j) < k$ for each $j = 1, 2$, it follows that $\llbracket \Phi_{\text{int}}(s_j) \rrbracket_{\text{int}}^c = \llbracket s_j \rrbracket_{\text{int}}^c$ for all $c \in \text{Sol}(\alpha)$, and thus by replacement rule (R1)

$$\llbracket \Phi_{\beta}(s) \rrbracket_{\beta}^c = \llbracket \odot_{\zeta}^I(\Phi_{\text{int}}(s_1), \Phi_{\text{int}}(s_2)) \rrbracket_{\beta}^c = \llbracket \odot_{\zeta}^I(s_1, s_2) \rrbracket_{\beta}^c = \llbracket s \rrbracket_{\beta}^c$$

for all $c \in \text{Sol}(\alpha)$, and so once again $s \in G_{\beta}$.

Therefore by induction on k it follows that $G_B = F_B^I$, and this shows that Φ_B is α -consistent. \square

If $\Phi_B : (F_B^I, \odot_K^I) \rightarrow (F_B^I, \odot_K^I)$ is a Λ -homomorphism then, as in Section 6.4, put

$$F_{\beta}^{\Phi} = \{s \in F_{\beta}^I : \Phi_{\beta}^n(s) \in F_{\beta} \text{ for some } n \geq 0\}$$

for each $\beta \in B$, and define a mapping $\Phi_{\beta}^{\infty} : F_{\beta}^{\Phi} \rightarrow F_{\beta}$ by letting $\Phi_{\beta}^{\infty}(s) = \Phi_{\beta}^n(s)$, where $n \geq 0$ is chosen so that $\Phi_{\beta}^n(s) \in F_{\beta}$. If $s \in F_{\beta}^{\Phi}$ then $\Phi_{\beta}^{\infty}(s)$ (or perhaps better $\llbracket \Phi_{\beta}^{\infty}(s) \rrbracket_{\beta}$) can be considered as the ‘value’ of s computed by Φ_B . The next result says, very roughly, that the ‘values’ computed by the Λ -homomorphism defined by α and N_B^I do not depend on which pre-support system N_B^I is used.

Proposition 6.5.3 *Let N_B^I and \check{N}_B^I be pre-support systems and let Φ_B (resp. $\check{\Phi}_B$) be the Λ -homomorphism defined by α and N_B^I (resp. defined by α and \check{N}_B^I). Then $\Phi_\beta^\infty(s) = \check{\Phi}_\beta^\infty(s)$ for each $s \in F_\beta^\Phi \cap F_\beta^{\check{\Phi}}$.*

Proof Let (D_S, f_N) be as in Case 3 with, in addition, (D_B, f_K) an initial completion of an initial bottomed extension of (X_B, p_K) . Then κF_B^I and hence also both of N_B^I and \check{N}_B^I are support systems. Therefore by Proposition 6.5.2 Φ_B and $\check{\Phi}_B$ are both α -consistent. Moreover, Proposition 7.1.1 will show that the set $\text{Sol}(\alpha)$ is non-empty. Thus by Proposition 6.4.1 it follows that

$$\llbracket \Phi_\beta^\infty(s) \rrbracket_\beta = \llbracket s \rrbracket_\beta^\alpha = \llbracket \check{\Phi}_\beta^\infty(s) \rrbracket_\beta$$

for each $s \in F_\beta^\Phi \cap F_\beta^{\check{\Phi}}$, and hence that $\Phi_\beta^\infty(s) = \check{\Phi}_\beta^\infty(s)$. \square

Of course, Proposition 6.5.3 does not say anything about the relationship between the families F_B^Φ and $F_B^{\check{\Phi}}$. In fact, using the techniques employed in Chapter 7 it is possible to show that $F_B^{\check{\Phi}}$ is smaller than F_B^Φ whenever \check{N}_B^I is smaller than N_B^I (i.e., $F_B^{\check{\Phi}} \subset F_B^\Phi$ whenever $\check{N}_B^I \subset N_B^I$).

Let Φ_B be the Λ -homomorphism defined by α and some support system N_B^I . If $\text{Sol}(\alpha) \neq \emptyset$ then Proposition 6.5.2 and Proposition 6.4.1 imply that $F_B^\Phi \subset F_B^\alpha$ and $\llbracket s \rrbracket_\beta^\alpha = \llbracket \Phi_\beta^\infty(s) \rrbracket_\beta$ for all $s \in F_\beta^\Phi$, $\beta \in B$, i.e., $\Phi_\beta^\infty(s)$ is the (unique) ground term which denotes $\llbracket s \rrbracket_\beta^\alpha$. Note however the requirement here that $\text{Sol}(\alpha) \neq \emptyset$. If $\text{Sol}(\alpha) = \emptyset$ then no conclusions can be made about an element $s \in F_\beta^\Phi$. (This point is illustrated in Example 6.5.3, which is essentially the situation considered in Example 4.3.4.)

The Λ -homomorphism defined by α and the support system HF_B^I will now be called simply *the Λ -homomorphism defined by α* . As already mentioned, this is really the only case of interest. Starting on the next page some examples are given illustrating how this Λ -homomorphism works in practice.

Example 6.5.1 Again let $\alpha \in \text{Eq}(I)$ be as in Example 6.2.1 the following system of equations:

```

it p = case p of {Pair -> ita}
ita m n = Pair n (add m n)
fst p = case p of {Pair -> fsta}
fsta m n = m
pfb n p = case (eq n 0) of {True -> p,
                           False -> pfb (sub n 1) (it p)}
fib n = fst (pfb n (Pair 0 1))

```

Let Φ_B be the Λ -homomorphism defined by α and $e \in E_{\text{ipr}}^I$ be the term

$$\text{pfb } 2 \text{ (Pair } 0 \text{ } 1)$$

If (D_B, f_K) is the flat extension of (X_B, p_K) (and so ${}_{H}E_B^I = E_B$) then the iterates $\Phi_{\text{ipr}}^n(e)$, $0 \leq n \leq 20$, are those listed in Example 6.3.2:

```

pfb 2 (Pair 0 1)
case (eq 2 0) of {True -> Pair 0 1,
                 False -> pfb (sub 2 1) (it (Pair 0 1))}
case False of {True -> Pair 0 1,
               False -> pfb (sub 2 1) (it (Pair 0 1))}
pfb (sub 2 1) (it (Pair 0 1))
:
case (Pair 1 (add 0 1)) of {Pair -> ita}
case (Pair 1 1) of {Pair -> ita}
ita 1 1
Pair 1 (add 1 1)
Pair 1 2

```

Thus $\Phi_{\text{ilst}}^n(e) = \text{Pair } 1 \text{ } 2$ for all $n \geq 21$.

Example 6.5.2 Consider the S -typed set

$$I = \{ \text{map}, \text{mapa}, \text{plus}, \text{hd}, \text{hda}, \text{ulist} \}$$

with

map of type $(\text{int} \rightarrow \text{int}) \text{ilst} \rightarrow \text{ilst}$,
 mapa of type $(\text{int} \rightarrow \text{int}) \text{int ilst} \rightarrow \text{ilst}$,
 plus of type $\text{int int} \rightarrow \text{int}$,
 hd of type $\text{ilst} \rightarrow \text{int}$, hda of type $\text{int ilst} \rightarrow \text{int}$,
 ulist of type int .

Let $\alpha \in \text{Eq}(I)$ be the following system of equations:

$\text{map } f \text{ ns} = \text{case ns of } \{ \text{Nil} \rightarrow \text{Nil}, \text{Cons} \rightarrow \text{mapa } f \}$
 $\text{mapa } f \text{ m ms} = \text{Cons } (f \text{ m}) (\text{map } f \text{ ms})$
 $\text{plus } m \text{ n} = \text{add } m \text{ n}$
 $\text{hd } ms = \text{case ms of } \{ \text{Nil} \rightarrow \text{ulist}, \text{Cons} \rightarrow \text{hda} \}$
 $\text{hda } n \text{ ns} = n$
 $\text{ulist} = \text{ulist}$

Let Φ_B be the Λ -homomorphism defined by α and $e \in E_{\text{ilst}}^I$ be the term

$$\text{map } (\text{plus } 1) (\text{Cons } 1 (\text{Cons } 2 \text{ Nil}))$$

Then for any monotone regular extension (D_B, f_K) of (X_B, p_K) the iterates $\Phi_{\text{ilst}}^n(e)$, for $0 \leq n \leq 8$, are those listed below:

$\text{map } (\text{plus } 1) (\text{Cons } 1 (\text{Cons } 2 \text{ Nil}))$
 $\text{case } (\text{Cons } 1 (\text{Cons } 2 \text{ Nil})) \text{ of}$
 $\quad \{ \text{Nil} \rightarrow \text{Nil}, \text{Cons} \rightarrow \text{mapa } (\text{plus } 1) \}$
 $\text{mapa } (\text{plus } 1) 1 (\text{Cons } 2 \text{ Nil})$
 $\text{Cons } (\text{plus } 1 1) (\text{map } (\text{plus } 1) (\text{Cons } 2 \text{ Nil}))$
 $\text{Cons } (\text{add } 1 1) (\text{case } (\text{Cons } 2 \text{ Nil}) \text{ of}$
 $\quad \{ \text{Nil} \rightarrow \text{Nil}, \text{Cons} \rightarrow \text{mapa } (\text{plus } 1) \})$
 $\text{Cons } 2 (\text{mapa } (\text{plus } 1) 2 \text{ Nil})$
 $\text{Cons } 2 (\text{Cons } (\text{plus } 1 2) (\text{mapa } (\text{plus } 1) \text{ Nil}))$

(Example 6.5.2 is continued on the next page.)

Example 6.5.2 (continued)

```
Cons 2 (Cons (add 1 2) (case Nil of
                  {Nil -> Nil, Cons -> mapa (plus 1)}))
Cons 2 (Cons 3 Nil)
```

Therefore for all $n \geq 9$

$$\Phi_{\text{11st}}^n(e) = \text{Cons 2 (Cons 3 Nil)}$$

Now consider the term $e' \in E_{\text{int}}^I$ given by

```
hd (map (plus 1) (Cons 1 (Cons 2 Nil)))
```

If (D_B, f_K) is here a fully regular extension of (X_B, p_K) (so $HE_B^I = KE_B^I$) then the iterates $\Phi_{\text{11st}}^n(e')$, for $0 \leq n \leq 8$, are those listed below:

```
hd (map (plus 1) (Cons 1 (Cons 2 Nil)))
case (map (plus 1) (Cons 1 (Cons 2 Nil))) of
    {Nil -> uelist, Cons -> hda}
case (case (Cons 1 (Cons 2 Nil)) of
    {Nil -> Nil, Cons -> mapa (plus 1)}) of
    {Nil -> uelist, Cons -> hda}
case (mapa (plus 1) 1 (Cons 2 Nil)) of
    {Nil -> uelist, Cons -> hda}
case (Cons (plus 1 1) (map (plus 1) (Cons 2 Nil))) of
    {Nil -> uelist, Cons -> hda}
hda (plus 1 1) (map (plus 1) (Cons 2 Nil))
plus 1 1
add 1 1
2
```

Thus $\Phi_{\text{int}}^n(e') = 2$ for all $n \geq 9$.

(Example 6.5.2 is continued on the next page.)

Example 6.5.2 (continued) Suppose now (D_B, f_K) is the flat extension of (X_B, p_K) (so $HE_B^I = E_B$). In this case the iterates $\Phi_{\text{int}}^n(e')$, $0 \leq n \leq 10$, are the following:

```

hd (map (plus 1) (Cons 1 (Cons 2 Nil)))
case (case (Cons 1 (Cons 2 Nil)) of
      {Nil -> Nil, Cons -> mapa (plus 1)}) of
      {Nil -> u-list, Cons -> hda}
case (mapa (plus 1) 1 (Cons 2 Nil)) of
      {Nil -> u-list, Cons -> hda}
case (Cons (plus 1 1) (map (plus 1) (Cons 2 Nil))) of
      {Nil -> u-list, Cons -> hda}
case (Cons (add 1 1) (case (Cons 2 Nil) of
      {Nil -> Nil, Cons -> mapa (plus 1)})) of
      {Nil -> u-list, Cons -> hda}
case (Cons 2 (mapa (plus 1) 2 Nil)) of
      {Nil -> u-list, Cons -> hda}
case (Cons 2 (Cons (plus 1 2) (mapa (plus 1) Nil))) of
      {Nil -> u-list, Cons -> hda}
case (Cons 2 (Cons (add 1 2) (case Nil of
      {Nil -> Nil, Cons -> mapa (plus 1)}))) of
      {Nil -> u-list, Cons -> hda}
case (Cons 2 (Cons 3 Nil)) of {Nil -> u-list, Cons -> hda}
hda 2 (Cons 3 Nil)
2

```

Thus again $\Phi_{\text{int}}^n(e') = 2$ for all $n \geq 11$ (although this time it takes longer to get to the 'answer').

Example 6.5.3 The following is very similar to the situation looked at in Example 6.2.3.

Let $I = \{\text{fst}, \text{fsta}, \text{inf}\}$ be the S -typed set with

fst of type $\text{ipr} \rightarrow \text{int}$, fsta of type $\text{int int} \rightarrow \text{int}$,
 inf of type int .

Consider the following system of equations $\alpha \in \text{Eq}(I)$:

$\text{fst } p = \text{case } p \text{ of } \{\text{Pair } \rightarrow \text{fsta}\}$
 $\text{fsta } m \ n = m$
 $\text{inf} = \text{inf}$

Let Φ_B be the Λ -homomorphism defined by α and $e \in E_{\text{int}}^I$ be the term

$\text{fst } (\text{Pair } 1 \ \text{inf})$

If (D_B, f_K) is the flat extension of (X_B, p_K) (and hence $HE_B^I = E_B$) then $\Phi_{\text{int}}^n(e) = \text{case } (\text{Pair } 1 \ \text{inf}) \text{ of } \{\text{Pair } \rightarrow \text{fsta}\}$ for all $n \geq 1$. Thus in this case $e \notin E_{\text{int}}^\Phi$.

On the other hand, if (D_B, f_K) is a fully regular extension (in which case $HE_B^I = KE_B^I$) then $\Phi_{\text{int}}^1(e) = \text{case } (\text{Pair } 1 \ \text{inf}) \text{ of } \{\text{Pair } \rightarrow \text{fsta}\}$, $\Phi_{\text{int}}^2(e) = \text{fsta } 1 \ \text{inf}$ and thus $\Phi_{\text{int}}^n(e) = 1$ for all $n \geq 3$. This means that here $e \in E_{\text{int}}^\Phi$ and $\Phi_{\text{int}}^\infty(e) = 1$.

Next let $I = \{\text{hd}, \text{hda}, \text{ones}\}$ be the S -typed set with

hd of type $\text{ilst} \rightarrow \text{int}$, hda of type $\text{int ilst} \rightarrow \text{int}$,
 ones of type ilst .

Consider the following system of equations $\alpha \in \text{Eq}(I)$:

$\text{hd } xs = \text{case } xs \text{ of } \{\text{Nil } \rightarrow \text{hd } xs, \text{Cons } \rightarrow \text{hda}\}$
 $\text{hda } x \ xs = x$
 $\text{ones} = \text{Cons } 1 \ \text{ones}$

Let Φ_B be the Λ -homomorphism defined by α and $e \in E_{\text{int}}^I$ be the term

$\text{hd } \text{ones}$

(Example 6.5.3 is continued on the next page.)

Example 6.5.3 (continued) Now if (D_B, f_K) is a fully regular extension of (X_B, p_K) (and so $HE_B^I = KE_B^I$) then the iterates $\Phi_{\text{int}}^n(e)$, $0 \leq n \leq 4$, are those listed below:

```

hd ones
case ones of {Nil -> hd ones, Cons -> hda}
case (Cons 1 ones) of {Nil -> hd ones, Cons -> hda}
hda 1 ones
1

```

Therefore $\Phi_{\text{int}}^n(e) = 1$ for all $n \geq 4$. This means that here $e \in E_{\text{int}}^\Phi$ and $\Phi_{\text{int}}^\infty(e) = 1$. (However, although $e \in E_{\text{int}}^\Phi$, it is still possible for $\text{Sol}(\alpha)$ to be empty.)

However, if (D_B, f_K) is the flat extension of (X_B, p_K) (and so $HE_B^I = E_B$) then the iterates $\Phi_{\text{int}}^n(e)$, $n \geq 0$, are the following:

```

hd ones
case ones of {Nil -> hd ones, Cons -> hda}
case (Cons 1 ones) of {Nil -> hd ones, Cons -> hda}
case (Cons 1 (Cons 1 ones)) of {Nil -> hd ones, Cons -> hda}
case (Cons 1 (Cons 1 (Cons 1 ones))) of
                                {Nil -> hd ones, Cons -> hda}
case (Cons 1 (Cons 1 (Cons 1 (Cons 1 ones)))) of
                                {Nil -> hd ones, Cons -> hda}
                                :

```

Thus in this case $e \notin E_{\text{int}}^\Phi$.

6.6 Notes

As already mentioned in the Preface, the approach to equations presented here is an instance of the ‘initial algebra semantics’ philosophy propagated by the ADJ group, for example in the papers Goguen, Thatcher, Wagner and Wright (1977) and Goguen, Thatcher and Wagner (1978). Note that in this framework the bottomed extension only really influences the solutions of equations via the ‘case’ operators.

The algorithm presented in Section 6.5 can be seen as giving an operational semantics to the rudimentary programming language: The meaning of a term is the meaning of the ground term computed by the algorithm. In many accounts, for example Chapter 9 of Winskel (1993), the operational semantics consists of a list of rules for evaluating terms. These rules are essentially the special cases of the four rules given in Sections 6.3 in which the auxiliary set U is empty. The algorithm in Section 6.5 is also based on the special cases of these rules; in addition it also specifies which rule to apply when more than one could be applied to a given term.

The algorithm could of course be used to actually implement the language. The reader interested in how functional programming languages are implemented in practice is recommended to consult Peyton Jones (1987) and Peyton Jones and Lester (1991).

Chapter 7 Completeness of the algorithm

The main result of this chapter (Proposition 7.1.1) is that in Case 3 each system of equations $\alpha \in \text{Eq}(I)$ has a solution, and that the Λ -homomorphism Φ_B defined by α (i.e., defined by α and the support system HF_B^I) is always α -complete.

In Proposition 7.1.3 a result is presented which applies to both Case 3 and Case 2, and which in Case 3 immediately implies that Φ_B is α -complete. Note, however, that in Case 2 it is possible for the system of equations not to have a solution.

The proofs of these results rely heavily on a technique which goes under the name of the *method of logical relations*.

In Section 7.4 it is shown that if (D_S, f_N) is any monotone regular functional extension of (X_B, p_K) supporting both integer and case operators then, although the assumptions required in the rest of the chapter do not necessarily hold, the algorithm can still almost always be used to compute values. This involves what are called *uniform values*.

7.1 Solutions as fixed-points

Throughout the section (D_S, f_N) is at least a monotone regular functional extension of (X_B, p_K) which supports both integer and case operators, although most of the results will require (D_S, f_N) to be as in Case 2 or Case 3. Let $\alpha = s_I \in \text{Eq}(I)$ be a system of equations and let Φ_B be the Λ -homomorphism defined by α . Recall from Section 6.4 that the family $F_B^\Phi \subset F_B^I$ is defined by

$$F_B^\Phi = \{s \in F_B^I : \Phi_B^n(s) \in F_B \text{ for some } n \geq 0\}$$

for each $\beta \in B$, and if $\text{Sol}(\alpha)$ is non-empty then the family $F_B^\alpha \subset F_B^I$ is defined by

$$F_B^\alpha = \{s \in F_B^I : \text{there exists } x \in X_B \text{ such that } \llbracket s \rrbracket_\beta^c = x \text{ for all } c \in \text{Sol}(\alpha)\}$$

for each $\beta \in B$. The following is the main result of this study:

Proposition 7.1.1 *In Case 3 the system of equations α always has a solution, and the Λ -homomorphism Φ_B is α -complete, i.e., $F_B^\Phi = F_B^\alpha$.*

Proof This will take up the rest of the present and the whole of the following two sections. \square

As noted in Chapter 6, in Case 2 it is possible for the system of equations not to have a solution. There is, however, a property which holds in both Case 2 and Case 3, and which in Case 3 immediately implies that Φ_B is α -complete. This will be introduced later in the section (in Proposition 7.1.3).

The starting point for both the proof of Proposition 7.1.1 and the analysis of Case 2 is to represent the set of solutions $\text{Sol}(\alpha)$ as the set of fixed-points of a certain mapping $U_\alpha : D_\circ^I \rightarrow D_\circ^I$. In order to define U_α the following facts are needed:

Lemma 7.1.1 *Let J be a non-empty S -typed set disjoint from P .*

(1) *Suppose D_σ is a poset for each $\sigma \in S$ and that the mapping f_ν is monotone for each $\nu \in N$. Then for each $\sigma \in S$ and each $s \in F_\sigma^J$ the mapping $c \mapsto \llbracket s \rrbracket_\sigma^c$ from D_σ^J to D_σ is monotone.*

(2) *Suppose D_σ is a complete poset for each $\sigma \in S$ and that the mapping f_ν is continuous for each $\nu \in N$. Then for each $\sigma \in S$ and each $s \in F_\sigma^J$ the mapping $c \mapsto \llbracket s \rrbracket_\sigma^c$ from D_σ^J to D_σ is continuous.*

Proof (2) For each $s \in F_\sigma^J$ define a mapping $h_s : D_\sigma^J \rightarrow D_\sigma$ by letting $h_s(c) = \llbracket s \rrbracket_\sigma^c$ for each $c \in D_\sigma^J$, and for each $\sigma \in S$ let \dot{F}_σ^J be the set of elements $s \in F_\sigma^J$ for which the mapping h_s is continuous. It is enough to show that the family \dot{F}_σ^J is invariant in (F_S^J, \odot_N^J) and that $\xi \in \dot{F}_\xi^J$ for each $\xi \in J \cup P$, since then by Proposition 5.3.7 $\dot{F}_S^J = F_S^J$.

Let $\kappa \in K$ be of type $\varepsilon \rightarrow \beta$ and put $s = \odot_\kappa^J(\varepsilon)$. Then

$$h_s(c) = \llbracket \odot_\kappa^J(\varepsilon) \rrbracket_\beta^c = f_\kappa(\varepsilon)$$

for each $c \in D_\sigma^J$. Hence h_s is a constant and so in particular a continuous mapping.

Next consider $\kappa \in K$ having type $L \rightarrow \beta$ with $L \in \mathcal{F}_B^o$, let $a \in (\dot{F}_\sigma^J)^L$ and put $s = \odot_\kappa^J(a)$. Then for each $c \in D_\sigma^J$

$$h_s(c) = \llbracket \odot_\kappa^J(a) \rrbracket_\beta^c = f_\kappa(\llbracket a \rrbracket_\sigma^c)^L = f_\kappa(\gamma(c)),$$

where $\gamma : D_\sigma^J \rightarrow D_\sigma^L$ is the mapping given for each $c \in D_\sigma^J$, $\xi \in L$ by

$$\gamma(c)(\xi) = \llbracket a(\xi) \rrbracket_\xi^c = h_{a(\xi)}(c).$$

But by assumption $h_{a(\xi)}$ is continuous for each $\xi \in L$ and thus by Proposition 4.2.4 γ is continuous. Hence $h_s = f_\kappa \gamma$, as the composition of two continuous mappings, is itself continuous, i.e., $h_s \in \dot{F}_\beta^J$.

Now let $\sigma = L \rightarrow \beta$ be a functional type and L' be a non-empty subset of L ; let $a \in (\dot{F}_\sigma^J)^{\sigma \cdot L'}$ and put $s = \odot_{\downarrow\{\sigma, L'\}}^J(a)$. Then for all $c \in D_\sigma^J$

$$h_s(c) = \llbracket \odot_{\downarrow\{\sigma, L'\}}^J(a) \rrbracket_{L \setminus L' \rightarrow \beta}^c = f_{\downarrow\{\sigma, L'\}}(\llbracket a \rrbracket_\sigma^c)^{L'} = f_{\downarrow\{\sigma, L'\}}(\gamma(c)),$$

where here $\gamma : D_\sigma^J \rightarrow D_\sigma^{\sigma \cdot L'}$ is the mapping given for each $c \in D_\sigma^J$, $\xi \in \sigma \cdot L'$ by

$$\gamma(c)(\xi) = \llbracket a(\xi) \rrbracket_\xi^c = h_{a(\xi)}(c).$$

But by assumption $h_{a(\xi)}$ is continuous for each $\xi \in \sigma \cdot L'$ and so exactly as above it follows that h_s is continuous, i.e., $h_s \in \dot{F}_{L \setminus L' \rightarrow \beta}^J$.

Finally, consider $\xi \in J \cup P$. If $\xi \in P$ then $h_\xi(c) = \llbracket \xi \rrbracket_\xi^c = p(\xi)$ for each $c \in D_\sigma^J$; this is a constant mapping and so in particular it is continuous. On the other hand, if $\xi \in J$ then $h_\xi(c) = \llbracket \xi \rrbracket_\xi^c = c(\xi)$ for each $c \in D_\sigma^J$; this is a projection mapping which, as was noted in Section 4.2, is also continuous. Thus in both cases $h_\xi \in \dot{F}_\xi^J$.

This shows that the family \hat{F}_S^J is invariant in (F_S^J, \odot_N^J) and that $\xi \in \hat{F}_\xi^J$ for each $\xi \in J \cup P$. Therefore $\hat{F}_S^J = F_S^J$.

(1) This is more-or-less the same as (2) (but somewhat simpler). \square

Until further notice suppose that (D_S, f_N) is either as in Case 2 or as in Case 3. Then Proposition 5.2.1 (resp. Proposition 5.2.2) implies that in Case 2 (resp. in Case 3) (D_S, f_N) satisfies the hypotheses of Lemma 7.1.1 (1) (resp. those of Lemma 7.1.1 (2)). Consider $\xi \in I$ of type $\sigma = L \rightarrow \beta$ with $L \neq \emptyset$; then by Lemma 7.1.1 (and the fact that for each $c \in D_\diamond^I$ the mapping $b \mapsto c \oplus b$ from D_\diamond^L to $D_\diamond^{I \cup L}$ is monotone (resp. continuous)) a mapping $U_\xi : D_\diamond^I \rightarrow D_\sigma$ can be defined by letting

$$U_\xi(c)(b) = \llbracket s_\xi \rrbracket_\beta^{c \oplus b}$$

for all $c \in D_\diamond^I$, $b \in D_\diamond^L$. Moreover, if $\xi \in I$ is of type $\beta \in B$ then a mapping $U_\xi : D_\diamond^I \rightarrow D_\beta$ can be defined by just letting $U_\xi(c) = \llbracket s_\xi \rrbracket_\beta^c$ for each $c \in D_\diamond^I$. Combining these mappings then results in the mapping $U_\alpha : D_\diamond^I \rightarrow D_\diamond^I$ defined by

$$U_\alpha(c)(\xi) = U_\xi(c)$$

for all $c \in D_\diamond^I$, $\xi \in I$.

Proposition 7.1.2 *The set of solutions $\text{Sol}(\alpha)$ of the system of equations α is exactly the set of fixed-points of the mapping $U_\alpha : D_\diamond^I \rightarrow D_\diamond^I$. In other words,*

$$\text{Sol}(\alpha) = \{c \in D_\diamond^I : U_\alpha(c) = c\}.$$

Proof By definition $U_\alpha(c) = c$ if and only if $U_\xi(c) = c(\xi)$ for each $\xi \in I$, and clearly $U_\xi(c) = c(\xi)$ if and only if $c \in \text{Sol}(s_\xi)$. Thus $U_\alpha(c) = c$ if and only if $c \in \text{Sol}(\alpha)$. \square

Lemma 7.1.2 *The mapping $U_\alpha : D_\diamond^I \rightarrow D_\diamond^I$ is monotone.*

Proof Let $c, c' \in D_\diamond^I$ with $c \sqsubseteq_\diamond^I c'$. Consider $\xi \in I$ of type $L \rightarrow \beta$ and suppose first that $L \neq \emptyset$. If $b \in D_\diamond^L$ then it is clear that $c \oplus b \sqsubseteq_\diamond^{I \cup L} c' \oplus b$ and hence by Lemma 7.1.1 $\llbracket s_\xi \rrbracket_\beta^{c \oplus b} \sqsubseteq_\beta \llbracket s_\xi \rrbracket_\beta^{c' \oplus b}$. It thus follows that

$$U_\xi(c)(b) = \llbracket s_\xi \rrbracket_\beta^{c \oplus b} \sqsubseteq_\beta \llbracket s_\xi \rrbracket_\beta^{c' \oplus b} = U_\xi(c')(b)$$

for all $b \in D_\diamond^L$, i.e., $U_\xi(c) \sqsubseteq_\xi U_\xi(c')$. In the same way, if ξ is of type $\beta \in B$ then

$$U_\xi(c) = \llbracket s_\xi \rrbracket_\beta^c \sqsubseteq_\beta \llbracket s_\xi \rrbracket_\beta^{c'} = U_\xi(c').$$

This holds for each $\xi \in I$ and therefore $U_\alpha(c) \sqsubseteq_\diamond^I U_\alpha(c')$. \square

Lemma 7.1.3 *In Case 3 the mapping $U_\alpha : D_\diamond^I \rightarrow D_\diamond^I$ is continuous.*

Proof It was already shown in Lemma 7.1.2 that U_α is monotone. Let $A \in \mathbf{d}(D_\diamond^I)$ with $c' = \bigsqcup A$ and consider $\xi \in I$ of type $L \rightarrow \beta$ with $L \neq \emptyset$. If $b \in D_\diamond^L$ then, putting $A_b = \{c \oplus b : c \in A\}$, it is easily checked that $A_b \in \mathbf{d}(D_\diamond^{I \cup L})$ and $c' \oplus b = \bigsqcup A_b$, and it thus follows from Lemma 7.1.1 (2) that

$$U_\xi(c')(b) = \llbracket s_\xi \rrbracket_\beta^{c' \oplus b} = \bigsqcup \{ \llbracket s_\xi \rrbracket_\beta^{c \oplus b} : c \in A \} = \bigsqcup \{ U_\xi(c)(b) : c \in A \}$$

for all $b \in D_\diamond^L$. Hence by the second statement in Proposition 4.2.5

$$U_\xi(c') = \bigsqcup \{ U_\xi(c) : c \in A \}.$$

On the other hand, if $\xi \in I$ is of type $\beta \in B$ then by Lemma 7.1.1 (2)

$$U_\xi(c') = \llbracket s_\xi \rrbracket_\beta^{c'} = \bigsqcup \{ \llbracket s_\xi \rrbracket_\beta^c : c \in A \} = \bigsqcup \{ U_\xi(c) : c \in A \}.$$

Therefore for each $\xi \in I$ it follows that

$$U_\alpha(c')(\xi) = U_\xi(c') = \bigsqcup \{ U_\xi(c) : c \in A \} = \bigsqcup \{ U_\alpha(c)(\xi) : c \in A \},$$

and hence Lemma 4.2.1 implies that $U_\alpha(c') = \bigsqcup \{ U_\alpha(c) : c \in A \}$. This shows that U_α is continuous. \square

Propositions 4.2.9 and 7.1.2 and Lemma 7.1.3 imply that in Case 3 the set $\text{Sol}(\alpha)$ is non-empty, since then D_\diamond^I is a bottomed complete poset. This shows that the first statement in Proposition 7.1.1 holds.

Recall from Section 6.1 that ϱ_S^I denotes the unique Σ -homomorphism from (F_S^I, \odot_N^I) to (D_S, f_N) such that $\varrho_\xi^I(\xi) = \perp_\xi$ for each $\xi \in I \cup P$. The next result is really a generalisation of Lemma 6.1.7:

Lemma 7.1.4 $\varrho_\beta^I(s) \sqsubseteq_\beta \pi_\beta(s)$ for all $s \in F_\beta^I$, $\beta \in B$ for any Λ -homomorphism $\pi_B : (F_B^I, \odot_K^I) \rightarrow (D_B, f_K)$.

Proof For each $\beta \in B$ put $\dot{F}_\beta^I = \{s \in F_\beta^I : \varrho_\beta^I(s) \sqsubseteq_\beta \pi_\beta(s)\}$. Now if $\kappa \in K$ is of type $L \rightarrow \beta$ and $a \in (\dot{F}_\beta^I)^L$ then

$$\varrho_\beta^I(\odot_\kappa^I(a)) = f_\kappa((\varrho_\beta^I)^L(a)) \sqsubseteq_\beta f_\kappa(\pi_\beta^L(a)) = \pi_\beta(\odot_\kappa^I(a))$$

(since the mapping f_κ is monotone), and thus $\odot_\kappa^I(a) \in \dot{F}_\beta^I$. This implies that the family \dot{F}_β^I is invariant in (F_B^I, \odot_K^I) . But in the proof of Lemma 6.1.7 it was shown that $\varrho_\beta^I(s) = \perp_\beta$ for each $s \in AF_\beta^I \cup I_\beta$, and hence $AF_\beta^I \cup I_\beta \subset \dot{F}_\beta^I$ for each $\beta \in B$. Therefore by Lemma 6.1.2 $\dot{F}_\beta^I = F_\beta^I$. \square

Lemma 7.1.5 $\varrho_\beta^I(s) \sqsubseteq_\beta \varrho_\beta^I(\Phi_\beta(s))$ for all $s \in F_\beta^I$, $\beta \in B$.

Proof This follows immediately from Lemma 7.1.4 and Proposition 2.2.1. \square

By Lemma 7.1.5 the sequence $\{\varrho_\beta^I(\Phi_\beta^m(s))\}_{m \geq 0}$ is monotone for each $s \in F_\beta^I$, $\beta \in B$.

Now define a sequence $\{k_n\}_{n \geq 0}$ of elements of D_\diamond^I by starting with $k_0 = \perp^I$ (given by $\perp^I(\xi) = \perp_\xi$ for each $\xi \in I$) and letting $k_{n+1} = U_\alpha(k_n)$ for each $n \geq 0$. The proof of Proposition 4.2.9, together with Lemma 7.1.2, shows that the sequence $\{k_n\}_{n \geq 0}$ is monotone, i.e., $k_n \sqsubseteq_\diamond^I k_{n+1}$ for all $n \geq 0$.

Proposition 7.1.3 *Assume in Case 2 that I is finite. Then for each $\beta \in B$*

$$F_\beta^\Phi = \{s \in F_\beta^I : \llbracket s \rrbracket_\beta^{k_m} \in X_\beta \text{ for some } m \geq 0\}.$$

Proof This will be a corollary of Proposition 7.1.4 below. \square

It should be emphasised that Proposition 7.1.3 holds even if $\text{Sol}(\alpha) = \emptyset$ (which is certainly possible in Case 2). Proposition 7.1.1 now follows more-or-less immediately:

Proof of Proposition 7.1.1 The subset $\{k_n : n \geq 0\}$ of D_\diamond^I is directed (since the sequence $\{k_n\}_{n \geq 0}$ is monotone), and the proof of Proposition 4.2.9 shows that its least upper bound k_∞ is the least fixed-point of U_α . In particular, by Proposition 7.1.2 $k_\infty \in \text{Sol}(\alpha)$. Now by Propositions 6.4.1 and 6.5.2 $F_B^\Phi \subset F_B^\alpha$, and so it only remains to show that $F_B^\alpha \subset F_B^\Phi$. Let $s \in F_B^\alpha$, and hence in particular $\llbracket s \rrbracket_\beta^{k_\infty} = x \in X_\beta$. But Proposition 4.1.3 (2) states that the set $\{u \in D_\beta : u \sqsubseteq_\beta x\}$ is finite. Thus by Lemma 7.1.1 (2) $\llbracket s \rrbracket_\beta^{k_n} = x$ for some $n \geq 0$, and therefore by Proposition 7.1.3 $s \in F_\beta^\Phi$. \square

Consider now the following situation, which certainly includes Case 3 (and will later also be used indirectly to deal with Case 2): Let (D_S, f_N) be a monotone regular functional extension of (X_B, p_K) (with as usual the bottom element of D_σ being denoted by \perp_σ). Then (D_S, f_N) is said to be *continuous* if D_σ is a complete poset with bottom element \perp_σ for each $\sigma \in S$, f_ν is a continuous mapping for each $\nu \in N$ and \sqsubseteq_B is an ordering associated with (D_B, f_K) (where \sqsubseteq_σ denotes the partial order on D_σ for each $\sigma \in S$). In particular, the functional Σ -algebra in Case 3 is such an extension. In what follows fix a continuous monotone regular functional extension (D_S, f_N) of (X_B, p_K) which supports both integer and case operators. Suppose also that a continuous mapping $U_\alpha : D_\diamond^I \rightarrow D_\diamond^I$ is given such that:

- (i) If $\xi \in I$ is of type $\sigma = L \rightarrow \beta$ with $L \neq \emptyset$ then

$$f_{\triangleleft\{\sigma, L\}}(U_\alpha(c)(\xi) \triangleleft b) = \llbracket s_\xi \rrbracket_\beta^{c \oplus b}$$

for all $c \in D_\diamond^I$, $b \in D_\diamond^L$.

- (ii) If $\xi \in I$ is of type $\beta \in B$ then $U_\alpha(c)(\xi) = \llbracket s_\xi \rrbracket_\beta^c$ for each $c \in D_\diamond^I$.

Of course, in Case 3 these two conditions actually define the mapping U_α , but in general this is not true. Note that each fixed-point of U_α will be a solution of the equations. In Case 3 the converse also holds, although again in general this is not true.

As above define a sequence $\{k_n\}_{n \geq 0}$ of elements of D_\circ^I by starting with $k_0 = \perp^I$ and letting $k_{n+1} = U_\alpha(k_n)$ for each $n \geq 0$. The proof of Proposition 4.2.9 again shows that the sequence $\{k_n\}_{n \geq 0}$ is monotone and the proof now also shows that the least upper bound k_∞ of the directed set $\{k_n : n \geq 0\}$ is the least fixed-point of the mapping U_α . In particular, k_∞ is a solution of the equations (and in Case 3 it is the least solution).

Lemma 7.1.4 is clearly still valid in the present set-up, and thus by Lemma 7.1.5 the sequence $\{\varrho_\beta^I(\Phi_\beta^m(s))\}_{m \geq 0}$ is again monotone for each $s \in F_\beta^I$, $\beta \in B$.

Proposition 7.1.4 *For all $s \in F_\beta^I$, $\beta \in B$*

$$\sqcup \{ \varrho_\beta^I(\Phi_\beta^m(s)) : m \geq 0 \} = \llbracket s \rrbracket_\beta^{k_\infty} .$$

In particular, $F_\beta^\Phi = \{ s \in F_\beta^I : \llbracket s \rrbracket_\beta^{k_m} \in X_\beta \text{ for some } m \geq 0 \}$ for each $\beta \in B$.

Proof The proof of the first statement is given in Section 7.3. The second statement then follows from the fact that $\{u \in D_\beta : u \sqsubseteq_\beta x\}$ is finite for each $x \in X_\beta$ (i.e., from Proposition 4.1.3 (2)). This, together with Lemma 7.1.1, implies that if $s \in F_\beta^I$ then $\llbracket s \rrbracket_\beta^{k_\infty} \in X_\beta$ if and only if $\llbracket s \rrbracket_\beta^{k_m} \in X_\beta$ for some $m \geq 0$. In the same way $\sqcup \{ \varrho_\beta^I(\Phi_\beta^m(s)) : m \geq 0 \} \in X_\beta$ if and only if $\varrho_\beta^I(\Phi_\beta^m(s)) \in X_\beta$ for some $m \geq 0$, which is the case if and only if $s \in F_\beta^\Phi$, since from Proposition 3.3.3, Lemma 6.1.6 and Proposition 6.1.2 it follows that $\{t \in F_\beta^I : \varrho_\beta^I(t) \in X_\beta\} = F_\beta$ for each $\beta \in B$. \square

For each $m \geq 0$ the element $\varrho_\beta^I(\Phi_\beta^m(s))$ of D_β can be considered as the partial information obtained about the ‘value’ of s after m applications of the mapping Φ_β . Proposition 7.1.4 thus says, at least in Case 3, that this partial information converges to the value of s with respect to the least solution k_∞ of the equations.

Of course, Proposition 7.1.4 immediately gives Proposition 7.1.3 in Case 3.

It remains to show that Proposition 7.1.3 also holds in Case 2 (and recall here it is assumed that I is finite). For this it is necessary to look at Case 2'. Thus in what follows let (Y_B, q_K) be a minimal monotone regular extension of (X_B, p_K) , let Y_S be the full monotone functional extension of Y_B and let (Y_S, q_N) be the Σ -algebra associated with (Y_B, q_K) and Y_S . Moreover, let (D_S, f_N) be the initial completion of (Y_S, q_N) . In particular, (D_S, f_N) is a continuous monotone regular functional extension of (X_B, p_K) which by Proposition 5.4.5 supports both integer and case operators.

Let ϱ_S^I be as in Case 2 the unique Σ -homomorphism from (F_S^I, \odot_N^I) to (Y_S, q_N) such that $\varrho_\xi^I(\xi) = \perp_\xi$ for each $\xi \in I \cup P$. Then it is clear that ϱ_S^I is also the corresponding unique Σ -homomorphism from (F_S^I, \odot_N^I) to (D_S, f_N) (considering ϱ_σ^I as a mapping from F_σ^I to D_σ for each $\sigma \in S$).

Denote by U_α^o the mapping defined above in terms of (Y_S, q_N) , i.e., $U_\alpha^o : Y_\circ^I \rightarrow Y_\circ^I$ is given by $U_\alpha^o(c)(\xi) = U_\xi^o(c)$ for all $c \in Y_\circ^I$, $\xi \in I$, where if $\xi \in I$ of type $\sigma = L \rightarrow \beta$

with $L \neq \emptyset$ then $U_\xi^o : Y_\diamond^I \rightarrow Y_\sigma$ is defined by

$$U_\xi^o(c)(b) = \llbracket s_\xi \rrbracket_\beta^{c \oplus b}$$

for all $c \in Y_\diamond^I$, $b \in Y_\diamond^L$, and if $\xi \in I$ is of type $\beta \in B$ then $U_\xi^o : Y_\diamond^I \rightarrow Y_\beta$ is just given by $U_\xi^o(c) = \llbracket s_\xi \rrbracket_\beta^c$ for each $c \in Y_\diamond^I$.

Lemma 7.1.6 (1) For each $\xi \in I$ the mapping $U_\xi^o : Y_\diamond^I \rightarrow Y_\sigma$ extends uniquely to a continuous mapping $U_\xi : D_\diamond^I \rightarrow D_\xi$.

(2) Let $U_\alpha : D_\diamond^I \rightarrow D_\diamond^I$ be the mapping given by $U_\alpha(c)(\xi) = U_\xi(c)$ for all $c \in D_\diamond^I$, $\xi \in I$. Then U_α is the unique continuous extension of the mapping $U_\alpha^o : Y_\diamond^I \rightarrow Y_\diamond^I$.

(3) Let $\xi \in I$ be of type $\sigma = L \rightarrow \beta$ with $L \neq \emptyset$; then

$$f_{\triangleleft\{\sigma, L\}}(U_\xi(c) \triangleleft b) = \llbracket s_\xi \rrbracket_\beta^{c \oplus b}$$

for all $c \in D_\diamond^I$, $b \in D_\diamond^L$.

(4) Let $\xi \in I$ be of type $\beta \in B$; then $U_\xi(c) = \llbracket s_\xi \rrbracket_\beta^c$ for each $c \in D_\diamond^I$.

Proof (1) The mapping $U_\xi^o : Y_\diamond^I \rightarrow Y_\sigma$ is still monotone considered as a mapping from Y_\diamond^I to D_σ ; moreover, by Proposition 4.3.6 D_\diamond^I is an initial completion of Y_\diamond^I (and it is here the assumption that I is finite is needed). Thus by Proposition 4.3.3 U_ξ^o extends uniquely to a continuous mapping $U_\xi : D_\diamond^I \rightarrow D_\xi$.

(2) By Proposition 4.2.4 U_α is continuous and by definition it is an extension of U_α^o ; it is thus the unique continuous extension of U_α^o .

(3) The mapping $h_\xi^o : Y_\diamond^I \times Y_\diamond^L \rightarrow Y_\beta$ given by $h_\xi^o(c, b) = \llbracket s_\xi \rrbracket_\beta^{c \oplus b}$ for all $c \in Y_\diamond^I$, $b \in Y_\diamond^L$ is monotone and so by Proposition 4.3.6 it extends uniquely to a continuous mapping $h_\xi : D_\diamond^I \times D_\diamond^L \rightarrow D_\beta$. But by Lemma 7.1.1 (2) the mapping $(c, b) \mapsto \llbracket s_\xi \rrbracket_\beta^{c \oplus b}$ from $D_\diamond^I \times D_\diamond^L$ to D_β is continuous and hence $h_\xi(c, b) = \llbracket s_\xi \rrbracket_\beta^{c \oplus b}$ for all $c \in D_\diamond^I$, $b \in D_\diamond^L$. On the other hand,

$$h_\xi^o(c, b) = \llbracket s_\xi \rrbracket_\beta^{c \oplus b} = U_\xi^o(c)(b) = q_{\triangleleft\{\sigma, L\}}(U_\xi^o(c) \triangleleft b)$$

for all $c \in Y_\diamond^I$, $b \in Y_\diamond^L$, and therefore the mapping $(c, b) \mapsto f_{\triangleleft\{\sigma, L\}}(U_\xi(c) \triangleleft b)$ is also a continuous extension of h_ξ^o . Thus $f_{\triangleleft\{\sigma, L\}}(U_\xi(c) \triangleleft b) = \llbracket s_\xi \rrbracket_\beta^{c \oplus b}$ for all $c \in D_\diamond^I$, $b \in D_\diamond^L$.

(4) This is contained in the proof of (3). \square

By Lemma 7.1.6 (3) and (4) the continuous mapping $U_\alpha : D_\diamond^I \rightarrow D_\diamond^I$ satisfies the hypotheses of Proposition 7.1.4. Hence

$$\sqcup \{ \varrho_\beta^I(\Phi_\beta^m(s)) : m \geq 0 \} = \llbracket s \rrbracket_\beta^{k_\infty},$$

and in particular $F_\beta^\Phi = \{ s \in F_\beta^I : \llbracket s \rrbracket_\beta^{k_m} \in X_\beta \text{ for some } m \geq 0 \}$ for each $\beta \in B$. But $k_0 = \perp^I \in Y_\diamond^I$ and if $k_n \in Y_\diamond^I$ then also $k_{n+1} = U_\alpha(k_n) = U_\alpha^o(k_n) \in Y_\diamond^I$.

Therefore $\{k_n\}_{n \geq 0}$ is a sequence of elements from Y_\diamond^I , and so it must in fact be the sequence which occurs in Case 2. Moreover, as was already noted, there is really no difference between the Σ -homomorphism ϱ_S^I in Case 2 and that in Case 2'. Thus the statement that $F_\beta^\Phi = \{s \in F_\beta^I : \llbracket s \rrbracket_\beta^{k_m} \in X_\beta \text{ for some } m \geq 0\}$ for each $\beta \in B$ is still valid in the Σ -algebra (Y_S, q_N) , which shows that Proposition 7.1.3 also holds in Case 2.

In fact, in Case 2 a stronger statement holds than that in Proposition 7.1.3 (recalling that, as noted in the proof of the second statement in Proposition 7.1.4, $s \in F_\beta^\Phi$ if and only if $\varrho_\beta^I(\Phi_\beta^m(s)) \in X_\beta$ for some $m \geq 0$):

Proposition 7.1.5 *Let $s \in F_\beta^I$ with $\beta \in B$; then for each $m \geq 0$ there exists $n \geq 0$ such that $\varrho_\beta^I(\Phi_\beta^m(s)) \sqsubseteq_\beta \llbracket s \rrbracket_\beta^{k_n}$ and $\llbracket s \rrbracket_\beta^{k_m} \sqsubseteq_\beta \varrho_\beta^I(\Phi_\beta^n(s))$.*

Proof As established above, $\{\varrho_\beta^I(\Phi_\beta^m(s)) : m \geq 0\}$ and $\{\llbracket s \rrbracket_\beta^{k_m} : m \geq 0\}$ are directed subsets of Y_\diamond^I , and by Proposition 7.1.4 they have the same least upper bound (in D_\diamond^I). Thus by Proposition 4.3.3 they are mutually cofinal, since D_\diamond^I is an initial completion of Y_\diamond^I . But this is clearly equivalent to the statement to be proved. \square

7.2 Logical relations and functionally free algebras

The preparations for the proof of Proposition 7.1.4 involve looking at how a concept called the method of logical relations fits in with the notion of a functionally free algebra. The results in this direction considered here (in particular Proposition 7.2.3) will then be applied in the next section to prove Proposition 7.1.4.

Throughout the section let (D_S, f_N) and (Y_S, q_N) be fixed, but arbitrary, functional Σ -algebras. The results will be applied in the next section with (D_S, f_N) a continuous monotone regular functional extension of (X_B, p_K) , but the only assumption needed here is that it is a functional Σ -algebra.

If D and Y are sets then a relation on $D \times Y$ is just a subset of $D \times Y$. However, infix notation will always be employed, and so $u \preceq y$ will be used to mean that the pair (u, y) belongs to the relation \preceq .

Let \preceq_B be a family of relations, with \preceq_β a relation on the set $D_\beta \times Y_\beta$ for each $\beta \in B$. This family is considered to be fixed for the whole of the section. For each B -typed set L there is then a relation \preceq_\diamond^L on the set $D_\diamond^L \times Y_\diamond^L$ given as usual by $c \preceq_\diamond^L a$ if and only if $c(\eta) \preceq_\eta a(\eta)$ for each $\eta \in L$. The single assumption that will be made about the family \preceq_B is the following:

(A) If $\kappa \in K$ is of type $L \rightarrow \beta$ and $c \in D_\diamond^L$, $a \in Y_\diamond^L$ are such that $c \preceq_\diamond^L a$ then

$$f_\kappa(c) \preceq_\beta q_\kappa(a).$$

In particular, this means that $f_\kappa(\varepsilon) \preceq_\beta q_\kappa(\varepsilon)$ whenever κ is of type $\emptyset \rightarrow \beta$.

The method of logical relations involves extending the family \preceq_B in an appropriate way to a family \preceq_S , with \preceq_σ a relation on the set $D_\sigma \times Y_\sigma$ for each $\sigma \in S$. Note that if \preceq_S is any such family then for each S -typed set L there is the corresponding relation \preceq_\diamond^L defined on the set $D_\diamond^L \times Y_\diamond^L$ (in the same way as above).

Proposition 7.2.1 *The family of relations \preceq_B can be extended uniquely to a family of relations \preceq_S , with \preceq_σ a relation on $D_\sigma \times Y_\sigma$ for each $\sigma \in S$, in such a way that if $\sigma = L \rightarrow \beta$ is a functional type and $u \in D_\sigma$, $s \in Y_\sigma$, then $u \preceq_\sigma s$ if and only if*

$$f_{\lambda\{\sigma, L\}}(u \triangleleft c) \preceq_\beta q_{\lambda\{\sigma, L\}}(s \triangleleft a)$$

for all $c \in D_\diamond^L$ and all $a \in Y_\diamond^L$ such that $c \preceq_\diamond^L a$.

Proof Let $|\cdot| : S \rightarrow \mathbb{N}$ be the mapping given in Lemma 5.1.1 with $|\beta| = 0$ for each $\beta \in B$ and such that $|L \rightarrow \beta| = 1 + \sum_{\eta \in L} |\langle \eta \rangle|$ for each functional type $L \rightarrow \beta$. The family \preceq_S will be defined by induction on $|\sigma|$: If $|\sigma| = 0$ then $\sigma \in B$ and so \preceq_σ is a member of the original family \preceq_B . Thus suppose that $k > 0$ and that \preceq_τ has already been defined for all $\tau \in S$ with $|\tau| \leq k$. Consider $\sigma \in S$ with $|\sigma| = k$. Then

$\sigma = L \rightarrow \beta$ is a functional type and $|\langle \eta \rangle| < k$ for each $\eta \in L$. This means that \preceq_σ can be defined by stipulating that $u \preceq_\sigma s$ if and only if

$$f_{\triangleleft\{\sigma, L\}}(u \triangleleft c) \preceq_\beta q_{\triangleleft\{\sigma, L\}}(s \triangleleft a)$$

for all $c \in D_\diamond^L$ and all $a \in Y_\diamond^L$ such that $c(\eta) \preceq_\eta a(\eta)$ for each $\eta \in L$. The family \preceq_S obtained in this way then has the required property by construction. The uniqueness also follows by induction on $|\sigma|$. \square

Now for each S -typed set I fix a functionally I -free Σ -algebra (Z_S^I, r_N^I) . If $c \in D_\diamond^I$ then π_ξ^c will denote the unique Σ -homomorphism from (Z_S^I, r_N^I) to (D_S, f_N) such that $\pi_\xi^c(\xi) = c(\xi)$ for each $\xi \in I$. Moreover, if $a \in Y_\diamond^I$ then δ_ξ^a will denote the unique Σ -homomorphism from (Z_S^I, r_N^I) to (Y_S, q_N) such that $\delta_\xi^a(\xi) = a(\xi)$ for each $\xi \in I$.

If I is an S -typed set then $c \in D_\diamond^I$ will be called *compatible with* $a \in Y_\diamond^I$ if

$$\pi_\sigma^c(s) \preceq_\sigma \delta_\sigma^a(s)$$

for all $s \in Z_\sigma^I$, $\sigma \in S$.

Proposition 7.2.2 *Let I be an S -typed set and let $c \in D_\diamond^I$, $a \in Y_\diamond^I$. Then c is compatible with a if and only if $c \preceq_\diamond^I a$.*

Proof Since $c(\xi) = \pi_\xi^c(\xi)$ and $a(\xi) = \delta_\xi^a(\xi)$ for each $\xi \in I$, it follows that $c \preceq_\diamond^I a$ whenever c is compatible with a . Conversely, suppose that $c \preceq_\diamond^I a$. Then, since the assignments c and a are fixed throughout the proof, it is convenient to just write π_S instead of π_ξ^c and δ_S instead of δ_ξ^a . Define a family $G_S \subset Z_S^I$ by putting

$$G_\sigma = \{s \in Z_\sigma^I : \pi_\sigma(s) \preceq_\sigma \delta_\sigma(s)\}$$

for each $\sigma \in S$. Thus by assumption $\xi \in G_\xi$ for each $\xi \in I$. It will be shown that the family G_S is invariant in the Σ^I -algebra $(Z_S^I, r_{N^I}^I)$ associated with (Z_S^I, r_N^I) .

Note that if L is an S -typed set and $b \in G_\diamond^L$ then $b(\eta) \in G_\eta$ for each $\eta \in L$ and hence $\pi_\eta(b(\eta))_\eta \preceq_\eta \delta_\eta(b(\eta))$, and this means that $\pi_\diamond^L(b) \preceq_\diamond^L \delta_\diamond^L(b)$.

First consider $\kappa \in K$ of type $L \rightarrow \beta$ and let $b \in G_\diamond^L$. Then $\pi_\diamond^L(b) \preceq_\diamond^L \delta_\diamond^L(b)$ and therefore by assumption (A)

$$\pi_\beta(r_\kappa^I(b)) = f_\kappa(\pi_\diamond^L(b)) \preceq_\beta q_\kappa(\delta_\diamond^L(b)) = \delta_\beta(r_\kappa^I(b)),$$

which shows that $r_\kappa^I(b) \in G_\beta$. In particular, if κ is of type $\emptyset \rightarrow \beta$ then $r_\kappa^I(\varepsilon) \in G_\beta$.

Next consider $\xi \in I$ of type $\sigma = L \rightarrow \beta$, let J be a subset of L and $b \in G_\diamond^J$. The main step in the proof is then to show that $r_{\xi\{J\}}^I(b) \in G_{L \setminus J \rightarrow \beta}$, and for this it can be assumed that $J \neq \emptyset$, since $r_{\xi\{\emptyset\}}^I(\varepsilon) = \xi \in G_\xi$ holds by assumption. The case with J a proper subset of L will be dealt with first, and here put $\tau = L \setminus J \rightarrow \beta$. By the definition of \preceq_τ it must be shown that

$$f_{\triangleleft\{\tau, L \setminus J\}}(\pi_\tau(r_{\xi\{J\}}^I(b)) \triangleleft c') \preceq_\beta q_{\triangleleft\{\tau, L \setminus J\}}(\delta_\tau(r_{\xi\{J\}}^I(b)) \triangleleft a')$$

whenever $c' \in D_\diamond^{L \setminus J}$ and $a' \in Y_\diamond^{L \setminus J}$ are such that $c' \preceq_\diamond^{L \setminus J} a'$. But (as noted above) $\pi_\diamond^I(b) \preceq_\diamond^J \delta_\diamond^J(b)$, and hence also $\pi_\diamond^J(b) \oplus c' \preceq_\diamond^L \delta_\diamond^J(b) \oplus a'$. Thus, since $\pi_\xi(\xi) \preceq_\xi \delta_\xi(\xi)$ and since (D_S, f_N) and (Y_S, q_N) are both functional Σ -algebras, it follows that

$$\begin{aligned}
f_{\triangleleft\{\tau, L \setminus J\}}(\pi_\tau(r_{\xi\{J\}}^I(b)) \triangleleft c') &= f_{\triangleleft\{\tau, L \setminus J\}}(\pi_\tau(r_{\triangleleft\{\sigma, J\}}^I(\xi \triangleleft b)) \triangleleft c') \\
&= f_{\triangleleft\{\tau, L \setminus J\}}(f_{\triangleleft\{\sigma, J\}}(\pi_\sigma(\xi) \triangleleft \pi_\sigma^J(b)) \triangleleft c') \\
&= f_{\triangleleft\{\sigma, L\}}(\pi_\sigma(\xi) \triangleleft (\pi_\sigma^J(b) \oplus c')) \preceq_\beta q_{\triangleleft\{\sigma, L\}}(\delta_\sigma(\xi) \triangleleft (\delta_\sigma^J(b) \oplus a')) \\
&= q_{\triangleleft\{\tau, L \setminus J\}}(q_{\triangleleft\{\sigma, J\}}(\delta_\sigma(\xi) \triangleleft \delta_\sigma^J(b)) \triangleleft a') \\
&= q_{\triangleleft\{\tau, L \setminus J\}}(\delta_\tau(r_{\triangleleft\{\sigma, J\}}^I(\xi \triangleleft b)) \triangleleft a') = q_{\triangleleft\{\tau, L \setminus J\}}(\delta_\tau(r_{\xi\{J\}}^I(b)) \triangleleft a')
\end{aligned}$$

and this shows that $r_{\xi\{J\}}^I(b) \in G_\tau$. Finally there is the case $J = L$: Here it must be shown that $\pi_\beta(r_{\xi\{L\}}^I(b)) \preceq_\beta \delta_\beta(r_{\xi\{L\}}^I(b))$ and this follows since

$$\begin{aligned}
\pi_\beta(r_{\xi\{L\}}^I(b)) &= \pi_\beta(r_{\triangleleft\{\sigma, L\}}^I(\xi \triangleleft b)) = f_{\triangleleft\{\sigma, L\}}(\pi_\sigma(\xi) \triangleleft \pi_\sigma^L(b)) \\
&\preceq_\beta q_{\triangleleft\{\sigma, L\}}(\delta_\sigma(\xi) \triangleleft \delta_\sigma^L(b)) = \delta_\beta(r_{\triangleleft\{\sigma, L\}}^I(\xi \triangleleft b)) = \delta_\beta(r_{\xi\{L\}}^I(b)).
\end{aligned}$$

The family G_S is thus invariant in the initial Σ^I -algebra (Z_S^I, r_{NI}^I) and therefore $G_S = Z_S^I$, i.e., c is compatible with a . \square

Now fix an S -typed set I and an assignment $i \in Y_\diamond^I$. (This is denoted by i because in the application in the next section (Y_S, q_N) will contain I and i will be given by $i(\xi) = \xi$ for each $\xi \in I$.) The proof of Proposition 7.1.4 there involves showing that certain assignments $k \in D_\diamond^I$ are compatible with this particular assignment i . Of course, Proposition 7.2.2 says that k is compatible with i if and only if $k \preceq_\diamond^I i$. However, the main technique needed for establishing compatibility turns out to be Proposition 7.2.3 below.

Let U be an S -typed set disjoint from I . An assignment $k \in D_\diamond^I$ is then said to be U -compatible with i if whenever $a \in Y_\diamond^U$, $c \in D_\diamond^U$ are such that $c \preceq_\diamond^U a$ then

$$\pi_\sigma^{k \oplus c}(s) \preceq_\sigma \delta_\sigma^{i \oplus a}(s)$$

for all $s \in Z_\sigma^{I \cup U}$, $\sigma \in S$. In particular, this means that k is compatible if and only if it is \emptyset -compatible with i .

Some care must be taken here: If the set Y_\diamond^U is empty then any assignment $k \in D_\diamond^I$ is trivially U -compatible with i . Moreover, Y_\diamond^U will be empty if and only if $Y_\eta = \emptyset$ for some $\eta \in U$, and this is quite possible when, for example, (Y_S, q_N) is some kind of term algebra.

Proposition 7.2.3 *An assignment $k \in D_\diamond^I$ is compatible with i if and only if it is U -compatible with i for all S -typed sets U disjoint from I .*

Proof If k is U -compatible with i for all S -typed sets U such that $U \cap I = \emptyset$ then in particular it is \emptyset -compatible and thus compatible with i .

Suppose conversely that k is compatible with i , and let U be an S -typed set disjoint from I . Consider $a \in Y_{\diamond}^U$, $c \in D_{\diamond}^U$ with $c \preceq_{\diamond}^U a$. Then

$$(k \oplus c)(\eta) = c(\eta) \preceq_{\eta} a(\eta) = (i \oplus a)(\eta)$$

for each $\eta \in U$ and, since k is compatible with i ,

$$(k \oplus c)(\xi) = k(\xi) \preceq_{\xi} i(\xi) = (i \oplus a)(\xi)$$

for each $\xi \in I$, i.e., $(k \oplus c)(\xi) \preceq_{\xi} (i \oplus a)(\xi)$ for all $\xi \in I \cup U$. Proposition 7.2.2 (applied to the S -typed set $I \cup U$ and the assignments $k \oplus c$ and $i \oplus a$) thus implies that $\pi_{\sigma}^{k \oplus c}(s) \preceq_{\sigma} \delta_{\sigma}^{i \oplus a}(s)$ for all $s \in Z_{\sigma}^{I \cup U}$, $\sigma \in S$, and therefore that k is U -compatible with i . \square

7.3 An application of the method of logical relations

Throughout the section (D_S, f_N) is a continuous monotone regular functional extension of (X_B, p_K) supporting both integer and case operators, and k_∞ is the least fixed-point of a continuous mapping $U_\alpha : D_\diamond^I \rightarrow D_\diamond^I$ satisfying the two conditions:

(i) If $\xi \in I$ is of type $\sigma = L \rightarrow \beta$ with $L \neq \emptyset$ then

$$f_{\triangleleft\{\sigma, L\}}(U_\alpha(c)(\xi) \triangleleft b) = \llbracket s_\xi \rrbracket_\beta^{c \oplus b}$$

for all $c \in D_\diamond^I$, $b \in D_\diamond^L$.

(ii) If $\xi \in I$ is of type $\beta \in B$ then $U_\alpha(c)(\xi) = \llbracket s_\xi \rrbracket_\beta^c$ for each $c \in D_\diamond^I$.

This section is taken up with the proof of Proposition 7.1.4, which states that

$$\bigsqcup \{ \varrho_\beta^I(\Phi_\beta^m(s)) : m \geq 0 \} = \llbracket s \rrbracket_\beta^{k_\infty}$$

for all $s \in F_\beta^I$, $\beta \in B$. The proof relies heavily on Proposition 7.2.3.

Recall that the proof of Proposition 4.2.9 shows that k_∞ is the least upper bound of the directed set $\{k_n : n \geq 0\}$, where $k_0 = \perp^I$ and $k_{n+1} = U_\alpha(k_n)$ for each $n \geq 0$ (and that this set is directed because the sequence $\{k_n\}_{n \geq 0}$ is monotone).

For each $\beta \in B$ let $v_\beta^\Phi : F_\beta^I \rightarrow D_\beta$ be the mapping given for each $s \in F_\beta^I$ by

$$v_\beta^\Phi(s) = \bigsqcup \{ \varrho_\beta^I(\Phi_\beta^m(s)) : m \geq 0 \}.$$

Thus what needs to be shown is that $v_\beta^\Phi(s) = \llbracket s \rrbracket_\beta^{k_\infty}$ for all $s \in F_\beta^I$, $\beta \in B$.

Lemma 7.3.1 (1) v_B^Φ is a Λ -homomorphism from (F_B^I, \odot_K^I) to (D_B, f_K) .

(2) $v_\beta^\Phi(\Phi_\beta(s)) = v_\beta^\Phi(s)$ for all $s \in F_\beta^I$, $\beta \in B$.

(3) $v_\beta^\Phi(s) \sqsubseteq_\beta \llbracket s \rrbracket_\beta^{k_\infty}$ for all $s \in F_\beta^I$, $\beta \in B$.

Proof (1) Let $\kappa \in K$ be of type $L \rightarrow \beta$ and let $b \in (F_\diamond^I)^L$. Then for each $m \geq 0$

$$\varrho_\beta^I(\Phi_\beta^m(\odot_\kappa^I(b))) = \varrho_\beta^I(\odot_\kappa^I((\Phi_\diamond^m)^L(b))) = f_\kappa((\varrho_\diamond^I)^L((\Phi_\diamond^m)^L(b)))$$

(since ϱ_B^I and Φ_B^m are both Λ -homomorphisms), and by Lemma 4.2.1

$$\begin{aligned} & (\bigsqcup \{ (\varrho_\diamond^I)^L((\Phi_\diamond^m)^L(b)) : m \geq 0 \}) (\eta) \\ &= \bigsqcup \{ (\varrho_\diamond^I)^L((\Phi_\diamond^m)^L(b)) (\eta) : m \geq 0 \} \\ &= \bigsqcup \{ \varrho_\eta^I(\Phi_\eta^m(b(\eta))) : m \geq 0 \} = v_\eta^\Phi(b(\eta)) \end{aligned}$$

for each $\eta \in L$, i.e., $\bigsqcup \{ (\varrho_\diamond^I)^L((\Phi_\diamond^m)^L(b)) : m \geq 0 \} = (v_\diamond^\Phi)^L(b)$. Therefore

$$\begin{aligned}
v_\beta^\Phi(\odot_\kappa^I(b)) &= \bigsqcup \{ \varrho_\beta^I(\Phi_\beta^m(\odot_\kappa^I(b))) : m \geq 0 \} \\
&= \bigsqcup \{ f_\kappa((\varrho_\circ^I)^L((\Phi_\circ^m)^L(b))) : m \geq 0 \} \\
&= f_\kappa(\bigsqcup \{ (\varrho_\circ^I)^L((\Phi_\circ^m)^L(b)) : m \geq 0 \}) = f_\kappa((v_\circ^\Phi)^L(b)),
\end{aligned}$$

and this implies that $v_B^\Phi : (F_B^I, \odot_K^I) \rightarrow (D_B, f_K)$ is a Λ -homomorphism.

(2) This holds immediately by the definition of v_β^Φ .

(3) Let $s \in F_\beta^I$ for some $\beta \in B$. Then by Lemma 7.1.4 and Proposition 6.5.2

$$\varrho_\beta^I(\Phi_\beta^m(s)) \sqsubseteq_\beta \llbracket \Phi_\beta^m(s) \rrbracket_\beta^{k_\infty} = \llbracket s \rrbracket_\beta^{k_\infty},$$

since $k_\infty \in \text{Sol}(\alpha)$, and thus also $v_\beta^\Phi(s) \sqsubseteq_\beta \llbracket s \rrbracket_\beta^{k_\infty}$. \square

In particular, Lemma 7.3.1 (3) already shows that ‘one half’ of Proposition 7.1.4 holds. The proof of the ‘other half’ (i.e., that $\llbracket s \rrbracket_\beta^{k_\infty} \sqsubseteq_\beta v_\beta^\Phi(s)$ for all $s \in F_\beta^I$, $\beta \in B$) uses the method of logical relations introduced in Section 7.2.

The family of relations \preceq_B which will provide the key to the proof of Proposition 7.1.4 is specified in the next result, which lists the important properties of this family:

Proposition 7.3.1 *For each $\beta \in B$ let \preceq_β be the relation on $D_\beta \times F_\beta^I$ defined by stipulating that $u \preceq_\beta s$ if and only if $u \sqsubseteq_\beta v_\beta^\Phi(s)$. Then the family \preceq_B has the following properties:*

(P1) $\perp_\beta \preceq_\beta s$ for all $s \in F_\beta^I$, $\beta \in B$.

(P2) Let $\kappa \in K$ be of type $L \rightarrow \beta$, let $c \in D_\circ^L$ with $f_\kappa(c) \neq \perp_\beta$ and $a \in (F_\circ^I)^L$. Then $f_\kappa(c) \preceq_\beta \odot_\kappa^I(a)$ if and only if $c \preceq_\circ^L a$, i.e., if and only if $c(\eta) \preceq_\eta a(\eta)$ for each $\eta \in L$. In particular, if κ is of type $\emptyset \rightarrow \beta$ then $f_\kappa(\varepsilon) \preceq_\beta \odot_\kappa^I(\varepsilon)$.

(P3) Let $\zeta \in P$ be the name of an integer operator (so ζ is of type $\text{int}_2 \rightarrow \beta$ with β either int or bool) and for $j = 1, 2$ let $n_j \in \mathbb{Z}$, $t_j \in F_{\text{int}}$ with $n_j \preceq_{\text{int}} t_j$. Then $f_{\triangleleft\{\text{int}_2 \rightarrow \beta, \text{int}_2\}}(p(\zeta) \triangleleft(n_1, n_2)) \preceq_\beta \text{Eval}_\zeta(t_1, t_2)$.

(P4) If $u \in D_\beta$ and $s \in F_\beta^I$ then $u \preceq_\beta \Phi_\beta(s)$ if and only if $u \preceq_\beta s$.

(P5) If $u \in D_\beta$ and $s \in F_\beta^I$ with $\perp_\beta \neq u \preceq_\beta s$ then there exists $m \geq 0$ such that $\Phi_\beta^m(s) \in \text{HF}_\beta^I$.

(P6) If $s \in F_\beta^I$ and $\{u_m\}_{m \geq 0}$ is a monotone sequence of elements from D_β with $u_m \preceq_\beta s$ for each $m \geq 0$ then $\bigsqcup \{u_m : m \geq 0\} \preceq_\beta s$.

Proof Let $\kappa \in K$ be of type $L \rightarrow \beta$, let $c \in D_\beta^L$ with $f_\kappa(c) \neq \perp_\beta$ and $a \in (F_\beta^I)^L$. Then by Lemma 7.3.1 (1) $v_\beta^\Phi(\odot_\kappa^I(c)) = f_\kappa((v_\beta^\Phi)^L(a))$ and thus $f_\kappa(c) \preceq_\beta \odot_\kappa^I(c)$ if and only if $f_\kappa(c) \sqsubseteq_\beta f_\kappa((v_\beta^\Phi)^L(a))$. But by the definition of an associated ordering this holds if and only if $c \sqsubseteq_\beta^L (v_\beta^\Phi)^L(a)$, i.e., if and only if $c \preceq_\beta^L a$. Hence (P2) holds. Now by Lemma 7.3.1 (2) $v_\beta^\Phi(\Phi_\beta(s)) = v_\beta^\Phi(s)$ for all $s \in F_\beta^I$, $\beta \in B$, which implies (P4) also holds. Recall by Proposition 6.1.2 that

$$HF_\beta^I = \{s \in F_\beta^I : \varrho_\beta^I(s) \neq \perp_\beta\}$$

for each $\beta \in B$. Therefore if $u \in D_\beta$ and $s \in F_\beta^I$ with $\perp_\beta \neq u \preceq_\beta s$ then in particular $v_\beta^\Phi(s) \neq \perp_\beta$, which means that $\varrho_\beta^I(\Phi_\beta(s)) \neq \perp_\beta$ and hence that $\Phi_\beta(s) \in HF_\beta^I$ for some $m \geq 0$, i.e., (P5) holds. Finally, (P1) and (P6) hold trivially, and (P3) holds more-or-less by the definition of Eval_ζ (since if $n \in \mathbb{Z}$ and $t \in F_{\text{int}}$ with $n \preceq_{\text{int}} t$ then $n \sqsubseteq_{\text{int}} v_{\text{int}}^\Phi(t) = \llbracket t \rrbracket_{\text{int}}$, and this is only possible if $\llbracket t \rrbracket_{\text{int}} = n$). \square

It should be clear from the formulation of Proposition 7.3.1 that the results of Section 7.2 will be applied taking the functional Σ -algebra (Y_S, q_N) there to be (F_S^I, \odot_N^I) . Note then that condition (P2) is just the assumption (A) made in Section 7.2. The ‘other half’ of Proposition 7.1.4 is now an immediate corollary of the following result:

Proposition 7.3.2 *For each $\beta \in B$ let \preceq_β be a relation on the set $D_\beta \times F_\beta^I$, and suppose the family \preceq_B has the properties (P1), (P2), (P3), (P4), (P5) and (P6). Then $\llbracket s \rrbracket_\beta^{k_\infty} \preceq_\beta s$ for all $s \in F_\beta^I$, $\beta \in B$.*

Proof First note a couple of facts which follow from the properties enjoyed by the family \preceq_B .

Lemma 7.3.2 *Let $\kappa \in K$ be of type $L \rightarrow \beta$.*

(1) *If $c \in D_\beta^L$ and $a \in (F_\beta^I)^L$ are such that $c \preceq_\beta^L a$ then $f_\kappa(c) \preceq_\beta \odot_\kappa^I(a)$.*

(2) *Let $c \in D_\beta^L$ with $f_\kappa(c) \neq \perp_\beta$, let $s \in F_\beta^I$ with $f_\kappa(c) \preceq_\beta s$ and let $m \geq 0$ be such that $\Phi_\beta^m(s) \in HF_\beta^I$ (the existence of such an m being guaranteed by (P5)). Then there exists a unique $a \in (F_\beta^I)^L$ such that $\Phi_\beta^m(s) = \odot_\kappa^I(a)$, and then $c \preceq_\beta^L a$.*

Proof (1) If $f_\kappa(c) = \perp_\beta$ then this follows immediately from (P1), and if $f_\kappa(c) \neq \perp_\beta$ then it is just a part of (P2).

(2) By Proposition 6.1.1 there exists a unique $a \in (F_\beta^I)^L$ such that $\Phi_\beta^m(s) = \odot_\kappa^I(a)$, since $\Phi_\beta^m(s) \in HF_\beta^I \subset \kappa F_\beta^I$. Then $f_\kappa(c) \preceq_\beta \Phi_\beta^m(s) = \odot_\kappa^I(a)$ (by (P4)), and hence (P2) implies that $c \preceq_\beta^L a$. \square

Now by Proposition 7.2.1 the family of relations \preceq_B can be extended uniquely to a family of relations \preceq_S , with \preceq_σ a relation on $D_\sigma \times F_\sigma^I$ for each $\sigma \in S$, in such a way that if $\sigma = L \rightarrow \beta \in S \setminus B$ and $u \in D_\sigma$, $s \in F_\sigma^I$, then $u \preceq_\sigma s$ if and only if

$$f_{\triangleleft\{\sigma, L\}}(u \triangleleft c) \preceq_\beta \odot_{\triangleleft\{\sigma, L\}}^I(s \triangleleft a)$$

for all $c \in D_\beta^L$ and all $a \in (F_\beta^I)^L$ such that $c \preceq_\beta^L a$.

Lemma 7.3.3 $\perp_\sigma \preceq_\sigma s$ for all $s \in F_\sigma^I$, $\sigma \in S$.

Proof If $\sigma \in B$ then this is just (P1), so let $\sigma = L \rightarrow \beta$ be a functional type and consider $c \in D_\sigma^L$, $a \in (F_\sigma^I)^L$ with $c \preceq_\sigma^L a$. Then by (P1)

$$f_{\triangleleft\{\sigma,L\}}(\perp_\sigma \triangleleft c) = \perp_\beta \preceq_\beta \odot_{\triangleleft\{\sigma,L\}}^I(s \triangleleft a)$$

from which $\perp_\sigma \preceq_\sigma s$ follows by the definition of \preceq_σ . \square

Proposition 7.2.3 will be applied with the S -typed set I occurring there being replaced by $I \cup P$. If J is an S -typed set disjoint from P then (F_S^J, \odot_N^J) will play the role of the functionally free algebra $(Z_S^{J \cup P}, r_N^{J \cup P})$ occurring in Section 7.2. Moreover, if $b \in (F_\sigma^J)^{J \cup P}$ then π_σ^b will again be used to denote the unique Σ -homomorphism from (F_S^J, \odot_N^J) to (D_S, f_N) such that $\pi_\sigma^b(\xi) = b(\xi)$ for each $\xi \in J \cup P$. (Thus if $c \in D_\sigma^J$ then in fact $\llbracket \cdot \rrbracket_\sigma^c = \pi_\sigma^{c \oplus p}$.) In the same way, if $a \in (F_\sigma^I)^{J \cup P}$ then δ_σ^a denotes the unique Σ -homomorphism from (F_S^J, \odot_N^J) to (F_S^I, \odot_N^I) such that $\delta_\sigma^a(\xi) = a(\xi)$ for each $\xi \in J \cup P$.

Consider the assignment $i \in (F_\sigma^I)^{I \cup P}$ given by $i(\xi) = \xi$ for each $\xi \in I \cup P$. Then clearly $\delta_\sigma^i : (F_S^I, \odot_N^I) \rightarrow (F_S^I, \odot_N^I)$ is just the identity homomorphism, i.e., $\delta_\sigma^i(s) = s$ for all $s \in F_\sigma^I$, $\sigma \in S$. The following lemma indicates how the notions introduced in Section 7.2 could (and will) be used to prove Proposition 7.3.2:

Lemma 7.3.4 Suppose $k_n \oplus p$ is compatible with the assignment i for each $n \in \mathbb{N}$. Then $\llbracket s \rrbracket_\beta^{k_\infty} \preceq_\beta s$ for all $s \in F_\beta^I$, $\beta \in B$.

Proof By definition, $k_n \oplus p$ being compatible with i means that $\pi_\sigma^{k_n \oplus p}(s) \preceq_\sigma \delta_\sigma^i(s)$, and thus $\llbracket s \rrbracket_\sigma^{k_n} \preceq_\sigma s$, for all $s \in F_\sigma^I$, $\sigma \in S$. In particular, if $k_n \oplus p$ is compatible with i then $\llbracket s \rrbracket_\beta^{k_n} \preceq_\beta s$ for all $s \in F_\beta^I$, $\beta \in B$, and if this holds for all $n \in \mathbb{N}$ then by (P6) and Lemma 7.1.1 (2) $\llbracket s \rrbracket_\beta^{k_\infty} \preceq_\beta s$ for all $s \in F_\beta^I$, $\beta \in B$. \square

Lemma 7.3.5 $p(\zeta) \preceq_\zeta \zeta$ for each $\zeta \in P$.

Proof Suppose first that $\zeta \in P$ is the name of an integer operator (so ζ is of type $\text{int}_2 \rightarrow \beta$ with β either int or bool). Then for any $n_j \in D_{\text{int}}$, $s_j \in F_{\text{int}}^I$ with $n_j \preceq_{\text{int}} s_j$ for $j = 1, 2$ it must be shown that

$$f_{\triangleleft\{\text{int}_2 \rightarrow \beta, \text{int}_2\}}(p(\zeta) \triangleleft (n_1, n_2)) \preceq_\beta \odot_\zeta^I(s_1, s_2).$$

By (P1) this holds immediately if $n_1 = \perp_{\text{int}}$ or $n_2 = \perp_{\text{int}}$, since then by definition $f_{\triangleleft\{\text{int}_2 \rightarrow \beta, \text{int}_2\}}(p(\zeta) \triangleleft (n_1, n_2)) = \perp_\beta$. It can thus be assumed in what follows that $n_1, n_2 \in D_{\text{int}} \setminus \{\perp_{\text{int}}\} = \mathbb{Z}$. Now $\perp_{\text{int}} \neq n_j \preceq_{\text{int}} s_j$, thus by (P5) there exists $m_j \geq 0$ with $\Phi_{\text{int}}^{m_j}(s_j) \in {}_H F_{\text{int}}^I = F_{\text{int}}$. Let

$$m = \min \{ k \geq 0 : \Phi_{\text{int}}^k(s_j) \in F_{\text{int}} \text{ for } j = 1, 2 \};$$

then $\Phi_\beta^{m+1}(\odot_\zeta^I(s_1, s_2)) = \text{Eval}_\zeta(t_1, t_2)$ with $t_j = \Phi_{\text{int}}^m(s_j)$ for $j = 1, 2$. But by (P4)

$n_j \preceq_{\text{int}} t_j$ for $j = 1, 2$ and therefore by (P3) it follows that

$$f_{\triangleleft\{\text{int}_2 \rightarrow \beta, \text{int}_2\}}(p(\zeta) \triangleleft (n_1, n_2)) \preceq_{\beta} \text{Eval}_{\zeta}(t_1, t_2) = \Phi_{\beta}^{m+1}(\odot_{\zeta}^I(s_1, s_2)) .$$

Hence, again using (P4), $f_{\triangleleft\{\text{int}_2 \rightarrow \beta, \text{int}_2\}}(p(\zeta) \triangleleft (n_1, n_2)) \preceq_{\beta} \odot_{\zeta}^I(s_1, s_2)$.

Suppose next that $\zeta = \text{case}_{\beta}^{\theta}$ for some $\beta \in B$, $\theta \in B_f$. Consider assignments $c \in D_{\diamond}^{K_{\theta, \beta}}$, $a \in (F_{\diamond}^I)^{K_{\theta, \beta}}$ and elements $u \in D_{\theta}$, $s \in F_{\theta}^I$ with $u \triangleleft c \preceq_{\diamond}^{\theta, K_{\theta, \beta}} s \triangleleft a$, i.e., with $u \preceq_{\theta} s$ and $c \preceq_{\diamond}^{K_{\theta, \beta}} a$. Put $L = \theta \cdot K_{\theta, \beta}$. Then, since $p(\zeta) = \text{case}_{\beta}^{\theta}$, it must be shown that $f_{\triangleleft\{L \rightarrow \beta, L\}}(\text{case}_{\beta}^{\theta} \triangleleft (u \triangleleft c)) \preceq_{\beta} \odot_{\zeta}^I(s \triangleleft a)$. If $u = \perp_{\theta}$ then by (P1)

$$f_{\triangleleft\{L \rightarrow \beta, L\}}(\text{case}_{\beta}^{\theta} \triangleleft (u \triangleleft c)) = \perp_{\beta} \preceq_{\beta} \odot_{\zeta}^I(s \triangleleft a)$$

so in what follows it can be assumed that $u \neq \perp_{\theta}$. Then u has a unique representation of the form $f_{\kappa}(b)$ with $\kappa \in K$ of type $L' \rightarrow \theta$ for some $L' \in \mathcal{F}_B$ and with $b \in D_{\diamond}^{L'}$. Now $\perp_{\theta} \neq u \preceq_{\theta} s$, thus by (P5) there exists $m \geq 0$ with $\Phi_{\theta}^m(s) \in \text{HF}_{\theta}^I$, and by taking the least such m it can be assumed that $\Phi_{\theta}^i(s) \notin \text{HF}_{\theta}^I$ whenever $0 \leq i < m$. Moreover, by Lemma 7.3.2 (2) $\Phi_{\theta}^m(s)$ is of the form $\odot_{\kappa}^I(t)$ with $t \in (F_{\diamond}^I)^{L'}$ and $b \preceq_{\diamond}^{L'} t$, and it therefore follows that

$$\begin{aligned} \Phi_{\beta}^{m+1}(\odot_{\zeta}^I(s \triangleleft a)) &= \Phi_{\beta}(\odot_{\zeta}^I(\Phi_{\theta}^m(s) \triangleleft a)) \\ &= \Phi_{\beta}(\odot_{\zeta}^I(\odot_{\kappa}^I(t) \triangleleft a)) = \odot_{\triangleleft\{L' \rightarrow \beta, L'\}}^I(a(\kappa) \triangleleft t) . \end{aligned}$$

But $b \preceq_{\diamond}^{L'} t$ and by assumption $c(\kappa) \preceq_{L' \rightarrow \beta} a(\kappa)$, and hence

$$\begin{aligned} f_{\triangleleft\{L \rightarrow \beta, L\}}(\text{case}_{\beta}^{\theta} \triangleleft (u \triangleleft c)) &= f_{\triangleleft\{L' \rightarrow \beta, L'\}}(c(\kappa) \triangleleft b) \\ &\preceq_{\beta} \odot_{\triangleleft\{L' \rightarrow \beta, L'\}}^I(a(\kappa) \triangleleft t) = \Phi_{\beta}^{m+1}(\odot_{\zeta}^I(s \triangleleft a)) . \end{aligned}$$

Thus by (P4) $f_{\triangleleft\{L \rightarrow \beta, L\}}(\text{case}_{\beta}^{\theta} \triangleleft (u \triangleleft c)) \preceq_{\beta} \odot_{\zeta}^I(s \triangleleft a)$. \square

Lemma 7.3.6 *Let $k \in D_{\diamond}^I$ and suppose $k \oplus p$ is compatible with the assignment i . Then $U_{\alpha}(k)(\xi) \preceq_{\xi} \xi$ for each $\xi \in I$.*

Proof First note that if L is an S -typed set and $a \in (F_{\diamond}^I)^L$ then, with the notation from Chapter 6, $s[a/L] = \delta_{\sigma}^{i \oplus a}(s)$ for each $s \in F_{\sigma}^{I \cup L}$, $\sigma \in S$.

Put $k' = U_{\alpha}(k)$. Let $\xi \in I$ be of type $\sigma = L \rightarrow \beta$ and assume first that $L \neq \emptyset$. Consider $c \in D_{\diamond}^L$, $a \in (F_{\diamond}^I)^L$ with $c \preceq_{\diamond}^L a$. Now Proposition 7.2.3 implies that $k \oplus p$ is L -compatible with the assignment i , and therefore

$$\begin{aligned} f_{\triangleleft\{\sigma, L\}}(k'(\xi) \triangleleft c) &= f_{\triangleleft\{\sigma, L\}}(U_{\xi}(k) \triangleleft c) \\ &= \llbracket s_{\xi} \rrbracket_{\beta}^{k \oplus c} = \pi_{\beta}^{k \oplus p \oplus c}(s_{\xi}) \preceq_{\beta} \delta_{\beta}^{i \oplus a}(s_{\xi}) = s_{\xi}[a/L] \\ &= \Phi_{\beta}(\odot_{\xi}^I(a)) = \Phi_{\beta}(\odot_{\triangleleft\{\sigma, L\}}^I(\xi \triangleleft a)) . \end{aligned}$$

Thus by (P4) $f_{\triangleleft\{\sigma, L\}}(k'(\xi) \triangleleft c) \preceq_{\beta} \odot_{\triangleleft\{\sigma, L\}}^I(\xi \triangleleft a)$ also holds, which by the definition

of \preceq_σ implies that $k'(\xi) \preceq_\xi \xi$. The case with $L = \emptyset$ is somewhat easier, since here (without making use of Proposition 7.2.3) it follows that

$$k'(\xi) = U_\xi(k) = \llbracket s_\xi \rrbracket_\beta^k = \pi_\beta^{k \oplus p}(s_\xi) \preceq_\beta \delta_\beta^i(s_\xi) = s_\xi = \Phi_\beta(\xi),$$

and so again by (P4) $k'(\xi) \preceq_\xi \xi$. \square

Lemma 7.3.7 *Let $k \in D_\diamond^I$ and suppose $k \oplus p$ is compatible with the assignment i . Then $U_\alpha(k) \oplus p$ is also compatible with i .*

Proof Again put $k' = U_\alpha(k)$. By Lemma 7.3.5 $p(\zeta) \preceq_\zeta \zeta = i(\zeta)$ for each $\zeta \in P$ and by Lemma 7.3.5 $k'(\xi) \preceq_\xi \xi = i(\xi)$ for each $\xi \in I$. But this just means that $k' \oplus p \preceq_\diamond^{I \cup P} i$, and therefore by Proposition 7.2.2 $k' \oplus p$ is compatible with i . \square

Lemma 7.3.8 *Let $\perp^I \in D_\diamond^I$ be the bottom assignment given by $\perp^I(\xi) = \perp_\xi$ for each $\xi \in I$. Then $\perp^I \oplus p$ is compatible with i .*

Proof By Lemma 7.3.3 $\perp^I(\xi) = \perp_\xi \preceq_\xi \xi$ for each $\xi \in I$, and thus by Lemma 7.3.5 and Proposition 7.2.2 it follows, exactly as in the proof of Lemma 7.3.6, that $\perp^I \oplus p$ is compatible with i . \square

Now by definition $k_0 = \perp^I$ and $k_{n+1} = U_\alpha(k_n)$ for each $n \in \mathbb{N}$. Therefore by Lemmas 7.3.8 and 7.3.7 and induction it follows that $k_n \oplus p$ is compatible with the assignment i for each $n \in \mathbb{N}$. This, together with Lemma 7.3.4, completes the proof of Proposition 7.3.2. \square

7.4 Uniform values

In this final section it will be shown how the algorithm can still be used to compute values without directly referring to the framework introduced in the previous chapters.

Suppose just the initial Λ -algebra (X_B, p_K) and a family R_B are given with $R_B \subset F_B^b$ the trace of some monotone regular extension of (X_B, p_K) . Then a Σ -algebra (D_S, f_N) will be called will be called *R_B -based* if (D_S, f_N) is a monotone regular functional extension of (X_B, p_K) supporting both integer and case operators and with R_B the trace of (D_B, f_K) . In particular, there exists an R_B -based Σ -algebra (D_S, f_N) , since by assumption there exists a suitable extension (D_B, f_K) and then (D_S, f_N) can always be taken as in Case 1 with D_S the full functional extension of D_B .

Note that for a given system of equations $\alpha \in \text{Eq}(I)$ the Λ -homomorphism Φ_B defined by α is independent of which R_B -based Σ -algebra is being used: This is because Φ_B depends only on α and HF_B^I , and by definition HF_B^I is determined by R_B .

If $\alpha \in \text{Eq}(I)$ is a system of equations and \mathcal{E} is an R_B -based Σ -algebra then denote by $\text{Sol}_{\mathcal{E}}(\alpha)$ the set of solutions using \mathcal{E} as the functional extension.

Proposition 7.4.1 *There exists an R_B -based Σ -algebra \mathcal{E} such that:*

- (i) *For each $\alpha \in \text{Eq}(I)$ the set $\text{Sol}_{\mathcal{E}}(\alpha)$ is non-empty, i.e., each system of equations has a solution.*
- (ii) *For each system of equations $\alpha \in \text{Eq}(I)$ the Λ -homomorphism Φ_B defined by α is α -complete, i.e., $F_B^{\Phi} = F_B^{\alpha}$, where F_B^{α} is defined in terms of \mathcal{E} .*

Proof By assumption there exists a monotone regular extension (D'_B, f'_K) of (X_B, p_K) having trace R_B . Let (Y_B, q_K) be the minimal bottomed subalgebra of (D'_B, f'_K) , so (Y_B, q_K) is a minimal bottomed extension of (X_B, p_K) . Now by the remark following Lemma 3.3.1 (Y_B, q_K) is regular, and by Proposition 3.4.7 it is monotone. Moreover, R_B is also the trace of (Y_B, q_K) (since there is essentially no difference between the homomorphism $\llbracket \cdot \rrbracket_B^{\perp} : (F_B^b, \odot_K^b) \rightarrow (Y_B, q_K)$ and the corresponding homomorphism from (F_B^b, \odot_K^b) to (D'_B, f'_K)). This shows there exists a minimal monotone regular extension (Y_B, q_K) of (X_B, p_K) having trace R_B .

The ‘canonical’ construction described at the beginning of the chapter can now be applied starting with (Y_B, q_K) : In other words, let (D_B, f_K) be an initial completion of (Y_B, q_K) (with respect to the unique associated ordering), D_S be the full continuous functional extension of D_B and let (D_S, f_N) be the functional Σ -algebra associated with (D_B, f_K) and D_S . Then (D_S, f_N) is a monotone regular functional extension of (X_B, p_K) supporting both integer and case operators, and by Proposition 4.4.2 (D_B, f_K) has the same trace as (Y_B, q_K) , i.e., (D_S, f_N) is an R_B -based Σ -algebra. Moreover, Proposition 7.1.1 implies that $\mathcal{E} = (D_S, f_N)$ satisfies (i) and (ii). \square

In what follows let I be a finite global S -typed set and let $\alpha \in \text{Eq}(I)$ be a system of equations. Define a family \hat{F}_B^α with $\hat{F}_B^\alpha \subset F_B^I$ by for each $\beta \in B$ letting

$$\hat{F}_\beta^\alpha = \{ s \in F_\beta^I : \text{there exists } x \in X_\beta \text{ such that } \llbracket s \rrbracket_\beta^c = x \\ \text{for all } c \in \text{Sol}_\mathcal{E}(\alpha) \text{ for each } R_B\text{-based } \Sigma\text{-algebra } \mathcal{E} \}.$$

If $s \in \hat{F}_\beta^\alpha$ then the element $x \in X_\beta$ occurring in the definition of \hat{F}_β^α will be called the *uniform value* of s . (By the first statement in Proposition 7.4.1 this value x is uniquely determined by s .)

Let Φ_B be the homomorphism defined by α (and note again that this only depends on α and R_B). Propositions 6.4.1 and 6.5.2 imply that $F_B^\Phi \subset \hat{F}_B^\alpha$. Moreover, if $s \in F_B^\Phi$ then $\llbracket \Phi_\beta^\infty(s) \rrbracket_\beta$ is the uniform value of s .

Proposition 7.4.2 *The uniform value of each element of \hat{F}_B^α can be computed using the homomorphism Φ_B , i.e., $F_B^\Phi = \hat{F}_B^\alpha$.*

Proof Let \mathcal{E} be the R_B -based Σ -algebra given in Proposition 7.4.1 for which Φ_B is α -complete, and let F_B^α be the family defined in terms of \mathcal{E} , thus $F_B^\Phi = F_B^\alpha$. But by Propositions 6.4.1 and 6.5.2 $F_B^\Phi \subset \hat{F}_B^\alpha \subset F_B^\alpha$, and therefore $F_B^\Phi = \hat{F}_B^\alpha$. \square

The fact that $F_B^\Phi = \hat{F}_B^\alpha$ means that the homomorphism Φ_B can only compute uniform values. However, in practice this is exactly what is needed: In all reasonable cases a proof that $\llbracket s \rrbracket_\beta^c = x$ for all $c \in \text{Sol}_\mathcal{E}(\alpha)$ for a particular R_B -based Σ -algebra \mathcal{E} will also prove that $\llbracket s \rrbracket_\beta^c = x$ for all $c \in \text{Sol}_\mathcal{E}(\alpha)$ for any R_B -based Σ -algebra \mathcal{E} .

It will be explained below more precisely why this can be the case, but first note that there is a small point where some care is needed with uniform values: It is quite possible for a particular R_B -based Σ -algebra \mathcal{E} that there are no solutions, i.e., that $\text{Sol}_\mathcal{E}(\alpha) = \emptyset$ (cf. Example 6.2.4). Thus in this case, although Φ_B may compute values, these values cannot be regarded as having any real meaning with respect to \mathcal{E} .

To be more explicit in what follows assume that the trace is given by a monotone core type (H_B, \diamond_K) . Thus by Proposition 3.5.4 R_B is here the unique family with $R_B \subset F_B^b$ satisfying the following conditions:

- (i) $b_\beta \notin R_\beta$ for each $\beta \in B$.
- (ii) If $\kappa \in K$ is of type $\emptyset \rightarrow \beta$ for some $\beta \in B$ then $\odot_\kappa^b(\varepsilon) \in R_\beta$.
- (iii) If $\kappa \in K$ is of type $L \rightarrow \beta$ with $L \neq \emptyset$ and $c \in (F_\beta^b)^L$ then $\odot_\kappa^b(c) \in R_\beta$ if and only if $\diamond_\kappa(b) = \natural_\beta$, where $b \in H_\beta^L$ is given by

$$b(\eta) = \begin{cases} \natural_{\beta_j} & \text{if } c(\eta) \in R_\eta, \\ b_\eta & \text{otherwise.} \end{cases}$$

Suppose a proof has been given that $\llbracket s \rrbracket_\beta^c = x$ for all $c \in \text{Sol}_\mathcal{E}(\alpha)$ for some particular R_B -based Σ -algebra \mathcal{E} , and suppose this involves the replacement rules introduced in

Section 6.3. Then the part of the proof making use of these replacement rules will still be valid for any R_B -based Σ -algebra \mathcal{E} provided the rules are applied in a form which only depends on \mathcal{E} via the core type (H_B, \diamond_K) . Of course, the only problem here is with rules (R3) and (R4), since the remaining rules do not depend on \mathcal{E} at all.

First consider rule (R3). Here $\zeta \in P$ is the name of an integer operator (and so ζ is of type $\text{int}_2 \rightarrow \beta$ with β either int or bool) and the rule is:

- (R3) Let $s_1, s_2 \in F_{\text{int}}^{IUU}$, let $c \in D_\diamond^I$, $b \in D_\diamond^U$ and suppose for each $j = 1, 2$ that $\llbracket s_j \rrbracket_{\text{int}}^{c \oplus b} = n_j \in \mathbb{Z}$. Then

$$\llbracket \odot_\zeta^{IUU}(s_1, s_2) \rrbracket_\beta^{c \oplus b} = \llbracket s_0 \rrbracket_\beta^{c \oplus b},$$

where s_0 is the unique element of F_β with

$$\llbracket s_0 \rrbracket_\beta = f_{\triangleleft\{\text{int}_2 \rightarrow \beta, \text{int}_2\}}(p(\zeta) \triangleleft (n_1, n_2)).$$

Thus what is required is a suitable criterion involving at most (H_B, \diamond_K) which ensures that $\llbracket s \rrbracket_{\text{int}}^{c \oplus b} = n \in \mathbb{Z}$ for an assignment $c \in D_\diamond^I$. In fact in this case there is a suitable criterion which does even not depend on (H_B, \diamond_K) , and which is provided by repeated application of the following rules:

- (i) Let $s = \odot_{\underline{n}}^{IUU}(\varepsilon)$ for some $n \in \mathbb{Z}$. Then $\llbracket s \rrbracket_{\text{int}}^{c \oplus b} = n$ for all $c \in D_\diamond^I$, $b \in D_\diamond^U$.
- (ii) Let $s = \eta$, where $\eta \in U$ is of type int and suppose that $b(\eta) = n \in \mathbb{Z}$. Then $\llbracket s \rrbracket_{\text{int}}^{c \oplus b} = n$ for all $c \in D_\diamond^I$.
- (iii) Let $s = \odot_{\eta'}^{IUU}(s_1, s_2)$ with $\eta' \in P$ of type $\text{int}_2 \rightarrow \text{int}$ and $s_1, s_2 \in F_{\text{int}}^{IUU}$. Let $c \in D_\diamond^I$, $b \in D_\diamond^U$ and suppose that $\llbracket s_j \rrbracket_{\text{int}}^{c \oplus b} = n_j \in \mathbb{Z}$ for each $j = 1, 2$. Then $\llbracket s \rrbracket_{\text{int}}^{c \oplus b} = f_{\triangleleft\{\text{int}_2 \rightarrow \beta, \text{int}_2\}}(p(\eta') \triangleleft (n_1, n_2))$.

Now for rule (R4). Here $\zeta = \text{case}_\beta^\theta$ for some $\beta \in B$, $\theta \in B_f$. Let $a \in (F_\diamond^{IUU})^{K_{\theta, \beta}}$ and $s \in F_\theta^{IUU}$ with $s = \odot_\kappa^{IUU}(a')$, where $\kappa \in K$ is of type $L' \rightarrow \theta$ for some $L' \in \mathcal{F}_B$ and $a' \in (F_\diamond^{IUU})^{L'}$. Recall that the fourth rule is:

- (R4) Let $c \in D_\diamond^I$, $b \in D_\diamond^U$ and suppose $\llbracket s \rrbracket_\theta^{c \oplus b} \neq \perp_\theta$. Then

$$\llbracket \odot_\zeta^{IUU}(s \triangleleft a) \rrbracket_\beta^{c \oplus b} = \llbracket \odot_{\triangleleft\{L' \rightarrow \beta, L'\}}^{IUU}(a(\kappa) \triangleleft a') \rrbracket_\beta^{c \oplus b}.$$

Thus what is required here is a suitable criterion involving only (H_B, \diamond_K) which ensures that $\llbracket s_0 \rrbracket_\theta^{c \oplus b} \neq \perp_\theta$ for an assignment $c \in D_\diamond^I$. Repeated application of the following rules provides such a criterion:

- (i) Let $s = \odot_{\kappa}^{IUU}(\varepsilon)$, where $\kappa \in K$ is of type $\emptyset \rightarrow \beta$. Then $\llbracket s \rrbracket_{\beta}^{c\oplus b} \neq \perp_{\beta}$ for all $c \in D_{\diamond}^I$, $b \in D_{\diamond}^U$.
- (ii) Let $s = \odot_{\kappa}^{IUU}(a)$ with $\kappa \in K$ of type $L \rightarrow \beta$, $L \neq \emptyset$, and $a \in (F_{\diamond}^{IUU})^L$. Let $c \in D_{\diamond}^I$, $b \in D_{\diamond}^U$ and suppose that $\diamond_{\kappa}(d) = \natural_{\beta}$, where

$$d(\eta) = \begin{cases} \natural_{\eta} & \text{if } \llbracket a(\eta) \rrbracket_{\eta}^{c\oplus b} \neq \perp_{\eta}, \\ \flat_{\eta} & \text{otherwise.} \end{cases}$$

Then $\llbracket s \rrbracket_{\beta}^{c\oplus b} \neq \perp_{\beta}$.

- (iii) Let $s = \eta$, where $\eta \in U$ is of type $\beta \in B$ and suppose that $b(\eta) \in X_{\beta}$. Then $\llbracket s \rrbracket_{\beta}^{c\oplus b} \neq \perp_{\beta}$ for all $c \in D_{\diamond}^I$.

These rules will usually suffice, unless U contains an element having a functional type. If this is the case then further rules may sometimes be required, and a typical example of such a rule is looked at below.

Let $\sigma = L \rightarrow \beta$, where $L \neq \emptyset$ and $\langle \eta \rangle \in B$ for each $\eta \in L$. An element $h \in D_{\sigma}$ is said to be a *strict extension* if there exists a mapping $g : X_{\diamond}^L \rightarrow X_{\beta}$ such that

$$f_{\triangleleft\{\sigma, L\}}(h \triangleleft a) = \begin{cases} g(a) & \text{if } a \in X_{\diamond}^L, \\ \perp_{\beta} & \text{otherwise.} \end{cases}$$

Then the additional rule is the following:

- (iv) Let $s = \odot_{\xi}^{IUU}(a)$, where $\xi \in U$ is of type σ and $a(\eta) \in U$ for each $\eta \in L$. Suppose that $b(\xi) \in D_{\sigma}$ is a strict extension and that $b(\eta) \in X_{\eta}$ for each $\eta \in L$. Then $\llbracket s \rrbracket_{\beta}^{c\oplus b} \neq \perp_{\beta}$ for all $c \in D_{\diamond}^I$.

In general, some further rules will be needed when U contains an element having a more complicated functional type than that just considered. This topic will not be pursued here; however, it should be more-or-less clear how such rules can be formulated.

If the rules (R1), (R2), (R3) and (R4) are modified so that rules (R3) and (R4) occur in the versions given above then they will be referred to as (H_B, \diamond_K) -replacement rules.

Now a proof that $\llbracket s \rrbracket_{\beta}^c = x$ for all $c \in \text{Sol}_{\mathcal{E}}(\alpha)$ will usually involve nothing but the four replacement rules together with certain properties of the Λ -algebra (X_B, p_K) . Thus, if in such a proof only (H_B, \diamond_K) -replacement rules are needed, then the proof is valid independently of which R_B -based Σ -algebra \mathcal{E} is used. In other words, the proof would show that $s \in \hat{F}_{\beta}^{\alpha}$ and that x is the uniform value of s .

The main property of the Λ -algebra (X_B, p_K) which is needed in such proofs is the following: Let P_B be a family of predicates (i.e., $P_{\beta} : X_{\beta} \rightarrow \mathbb{B}$ for each $\beta \in B$) satisfying the two conditions:

- (i) If $\kappa \in K$ is of type $\emptyset \rightarrow \beta$ then $P_{\beta}(p_{\kappa}(\varepsilon)) = T$.
- (ii) If $\kappa \in K$ is of type $L \rightarrow \beta$ with $L \neq \emptyset$ and $a \in X_{\diamond}^L$ with $P_{\eta}(a(\eta)) = T$ for each $\eta \in L$ then $P_{\beta}(p_{\kappa}(a)) = T$.

Then $P_\beta(x) = T$ for all $x \in X_\beta$, $\beta \in B$. This property holds, of course, because (X_B, p_K) is minimal (and is, in fact, equivalent to (X_B, p_K) being minimal).

In addition to the above property (which is a form of what is called *structural induction*), proofs involving the data type `int` will also employ the usual principle of mathematical induction (i.e., the particular property of the subset \mathbb{N} of $\mathbb{Z} = X_{\text{int}}$).

The examples on the following pages illustrate how such proofs can be applied to the systems of equations occurring in the previous examples.

Example 7.4.1 Let (H_B, \diamond_K) be any monotone core type and let α be as in Example 6.2.1 the following element of $\text{Eq}(I)$:

```

it p = case p of {Pair -> ita}
ita m n = Pair n (add m n)
fst p = case p of {Pair -> fsta}
fsta m n = m
pfb n p = case (eq n 0) of {True -> p,
                             False -> pfb (sub n 1) (it p)}
fib n = fst (pfb n (Pair 0 1))

```

In Example 6.3.3 only (H_B, \diamond_K) -replacement rules were used. This implies that if $c \in \text{Sol}_\mathcal{E}(\alpha)$ for some R_B -based \mathcal{E} then for all $n \in \mathbb{N}$

$$c_{\text{pfb}}(n, (0, 1)) = (f_n, f_{n+1}).$$

Now let $V = \{n\}$ with n of type `int`, let $n \in \mathbb{N}$ and let $b \in D_\diamond^V$ be the unique assignment with $b_n = n$. Then for all $c \in \text{Sol}_\mathcal{E}(\alpha)$

$$\begin{aligned}
\llbracket \text{fib } \underline{n} \rrbracket_{\text{int}}^c &= c_{\text{fib}}(n) = \llbracket \text{fib } n \rrbracket_{\text{int}}^{c \oplus b} \\
&= \llbracket \text{fst (pfb } n \text{ (Pair } 0 \ 1)) \rrbracket_{\text{int}}^{c \oplus b} \\
&= c_{\text{fst}}(c_{\text{pfb}}(b_n, (0, 1))) = c_{\text{fst}}(c_{\text{pfb}}(n, (0, 1))) \\
&= c_{\text{fst}}((f_n, f_{n+1})) = \llbracket \text{fst (Pair } \underline{f_n} \ \underline{f_{n+1}}) \rrbracket_{\text{int}}^c \\
&= \llbracket \text{case (Pair } \underline{f_n} \ \underline{f_{n+1}}) \text{ fsta} \rrbracket_{\text{int}}^c \\
&= \llbracket \text{fsta } \underline{f_n} \ \underline{f_{n+1}} \rrbracket_{\text{int}}^c = \llbracket \underline{f_n} \rrbracket_{\text{int}}^c = f_n.
\end{aligned}$$

This shows that `fib n` is an element of $\hat{E}_{\text{int}}^\alpha$ with uniform value f_n .

Example 7.4.2 Let

$$I = \{\text{map}, \text{mapa}, \text{plus}\}$$

be the S -typed set with

`map` of type $(\text{int} \rightarrow \text{int}) \text{ilst} \rightarrow \text{ilst}$,
`mapa` of type $(\text{int} \rightarrow \text{int}) \text{int} \text{ilst} \rightarrow \text{ilst}$,
`plus` of type $\text{int} \text{int} \rightarrow \text{int}$.

Consider the following system of equations $\alpha \in \text{Eq}(I)$:

`map f ns = case ns of {Nil -> Nil, Cons -> mapa f}`
`mapa f m ms = Cons (f m) (map f ms)`
`plus m n = add m n`

and again let (H_B, \diamond_K) be any monotone core type.

Let $U = \{f, m, ms\}$, with m of type int , ms of type ilst and f of type $\text{int} \rightarrow \text{int}$. Then using only (H_B, \diamond_K) -replacement rules it follows that

$$\begin{aligned} \llbracket \text{map } f \text{ Nil} \rrbracket_{\text{ilst}}^{c \oplus b} &= \llbracket \text{case Nil of \{Nil -> Nil, Cons -> mapa f\}} \rrbracket_{\text{ilst}}^{c \oplus b} \\ &= \llbracket \text{Nil} \rrbracket_{\text{ilst}}^{c \oplus b} \end{aligned}$$

for all $c \in \text{Sol}_{\mathcal{E}}(\alpha)$ and all $b \in D_{\diamond}^U$, as well as

$$\begin{aligned} \llbracket \text{map } f \text{ (Cons } m \text{ ms)} \rrbracket_{\text{ilst}}^{c \oplus b} &= \llbracket \text{case (Cons } m \text{ ms) of \{Nil -> Nil, Cons -> mapa f\}} \rrbracket_{\text{ilst}}^{c \oplus b} \\ &= \llbracket \text{mapa } f \text{ } m \text{ ms} \rrbracket_{\text{ilst}}^{c \oplus b} = \llbracket \text{Cons (f } m \text{) (map } m \text{ ms)} \rrbracket_{\text{ilst}}^{c \oplus b} \end{aligned}$$

for all $c \in \text{Sol}_{\mathcal{E}}(\alpha)$ and all $b \in D_{\diamond}^U$ with $b_m \in \mathbb{Z}$ and $b_{ms} \in \mathbb{Z}^*$. Thus

$$c_{\text{map}}(b_f, \varepsilon) = \llbracket \text{map } f \text{ Nil} \rrbracket_{\text{ilst}}^{c \oplus b} = \llbracket \text{Nil} \rrbracket_{\text{ilst}}^{c \oplus b} = \varepsilon$$

for all $c \in \text{Sol}_{\mathcal{E}}(\alpha)$ and all $b \in D_{\diamond}^U$.

(Example 7.4.2 is continued on the next page.)

Example 7.4.2 (continued) Moreover,

$$\begin{aligned} c_{\text{map}}(b_f, f_{\text{cons}}(b_m, b_{ms})) &= \llbracket \text{map } f \text{ (Cons } m \text{ ms)} \rrbracket_{\text{ilst}}^{c \oplus b} \\ &= \llbracket \text{Cons } (f \ m) \text{ (map } m \text{ ms)} \rrbracket_{\text{ilst}}^{c \oplus b} \\ &= f_{\text{cons}}(b_f(b_m), c_{\text{map}}(b_f, b_{ms})) \end{aligned}$$

for all $c \in \text{Sol}_{\mathcal{E}}(\alpha)$ and all $b \in D_{\circ}^U$ with $b_m \in \mathbb{Z}$ and $b_{ms} \in \mathbb{Z}^*$.

Therefore if $c \in \text{Sol}_{\mathcal{E}}(\alpha)$ and $m \in \mathbb{Z}$, $\ell \in \mathbb{Z}^*$ and $h \in D_{\text{int} \rightarrow \text{int}}$ then $c_{\text{map}}(h, \varepsilon) = \varepsilon$ and

$$c_{\text{map}}(h, f_{\text{cons}}(m, \ell)) = f_{\text{cons}}(h(m), c_{\text{map}}(h, \ell)).$$

From this it follows using structural induction (or induction on the length of the list ℓ) that if $c \in \text{Sol}_{\mathcal{E}}(\alpha)$ then

$$c_{\text{map}}(h, m_1 \cdots m_p) = h(m_1) \cdots h(m_p)$$

for all $m_1 \cdots m_p \in \mathbb{Z}^*$, provided $h \in D_{\text{int} \rightarrow \text{int}}$ is a strict extension.

Now let $m \in \mathbb{Z}$; then for all $c \in \text{Sol}_{\mathcal{E}}(\alpha)$

$$\llbracket \text{plus } \underline{m} \rrbracket_{\text{int} \rightarrow \text{int}}^c(n) = \llbracket \text{plus } \underline{m} \ \underline{n} \rrbracket_{\text{int}}^c = \llbracket \text{add } \underline{m} \ \underline{n} \rrbracket_{\text{int}}^c = m + n$$

for each $n \in \mathbb{Z}$, which implies that $\llbracket \text{plus } \underline{m} \rrbracket_{\text{int} \rightarrow \text{int}}^c \in D_{\text{int} \rightarrow \text{int}}$ is a strict extension. Let $e \in E_{\text{ilst}}$; then $\llbracket e \rrbracket_{\text{ilst}} \in \mathbb{Z}^*$ and

$$\begin{aligned} \llbracket \text{map } (\text{plus } \underline{m}) \ e \rrbracket_{\text{ilst}}^c &= c_{\text{map}}(\llbracket \text{plus } \underline{m} \rrbracket_{\text{int} \rightarrow \text{int}}^c, \llbracket e \rrbracket_{\text{ilst}}^c) \\ &= c_{\text{map}}(\llbracket \text{plus } \underline{m} \rrbracket_{\text{int} \rightarrow \text{int}}^c, \llbracket e \rrbracket_{\text{ilst}}) \end{aligned}$$

for all $c \in \text{Sol}_{\mathcal{E}}(\alpha)$. This shows in particular that $\text{map } (\text{plus } \underline{m}) \ e$ is an element of $\hat{E}_{\text{ilst}}^{\alpha}$ for all $m \in \mathbb{Z}$ and all $e \in E_{\text{ilst}}$.

7.5 Notes

The proof of Proposition 7.1.4 is adapted from the proofs of Lemmas 11.14 and 11.23 in Winskel (1993). For more information regarding the method of logical relations, on which these proofs are based, see Mitchell's survey article Mitchell (1990).

Bibliography

- Banaschewski, B. and E. Nelson (1982), Completions of partially ordered sets, *SIAM J. Computing*, 11, 521–528.
- Bauer, F.L., R. Berghammer *et al* (1985), The Munich Project CIP, Vol. 1: The Wide Spektrum Language CIP-L, *LNCS*, Vol. 183, Springer-Verlag.
- Bird, R. and Wadler, P. (1988), Introduction to functional programming, Prentice-Hall, Hemel Hempstead.
- Birkhoff, G. (1935), On the structure of abstract algebras, *Proc. Camb. Phil. Soc.*, 31, 433–454.
- Birkhoff, G. (1967), Lattice Theory, Third (New) Edition, *American Mathematical Society Colloquium Publications*, Vol. XXV.
- Birkhoff, G. and J.D. Lipson (1970), Heterogeneous algebras, *J. Combinatorial Th.*, 8, 115–133.
- Bourbaki, N. (1970), Algèbre, Chapitre 1, Hermann, Paris.
- Burstall, R.M. and P.J. Landin (1969), Programs and their proofs, in: B. Meltzer and D. Michie, *Machine Intelligence, Vol. 4*, 17–43, Edinburgh Univ. Press.
- Cohn, P.M. (1965), Universal Algebra, Harper and Row, New York.
- Courcelle, B. and M. Nivat (1976), Algebraic families of interpretations, in: *Proc. Annual Symposium on Foundations of Computer Science*, 137–146, Houston.
- Field, A.J. and P.G. Harrison (1988), Functional programming, Addison-Wesley, Wokingham, Berks.
- Glaser, H., Henkin, C. and D. Till (1984), Principles of functional programming, Prentice-Hall, Englewood Cliffs, N.J.
- Goguen, J.A., J.W. Thatcher, E.G. Wagner and J.B. Wright (1977), Initial algebra semantics and continuous algebras, *Journal ACM*, Vol. 24, 68–95.
- Goguen, J.A., J.W. Thatcher and E.G. Wagner (1978), An initial algebra approach to the specification correctness, and implementation of abstract data types, in: R.T. Yeh, *Current Trends in Programming Methodology, Vol. 4, Data Structuring*, 80–149, Prentice-Hall, Englewood Cliffs, N.J.
- Grätzer, G. (1968), Universal Algebra, van Nostrand, Princeton, N.J.

- Henderson, P. (1980), Functional programming, Prentice-Hall, Englewood Cliffs, N.J.
- Higgins, P.J. (1963), Algebras with a scheme of operators, *Mathematische Nachrichten*, 27, 115–132.
- Holyer, I. (1993), Functional programming with Miranda, Pitman, London.
- Huet, G. and D.C. Oppen (1980), Equations and rewrite rules: a survey, in: R.V. Book *Formal Language Theory: Perspectives and Open Problems*, Academic Press.
- Kleene, S.C. (1952), Introduction to Metamathematics, North-Holland, Amsterdam.
- Lukasiewicz, J. (1925), ?, ?
- MacLennan, B.J. (1990), Functional programming, Addison-Wesley, Reading, Mass.
- Maibaum, T.S.E. (1972), The characterization of the derivation trees of context-free sets of terms as regular sets, *Proc. 13th Ann. IEEE Symp. on Switching and Automata Theory*, 224–230.
- Markowsky, G. (1977), Categories of chain-complete posets, *Theoret. Computer Science*, 4, 125–135.
- Markowsky, G. and B. Rosen (1976), Bases for chain-complete posets, *IBM J. Research Develop.*, 20, 138–147.
- Mitchell, J.C. (1990), Type systems for programming languages, Chapter 8 of *Handbook of Theoretical Computer Science*, Elsevier, Amsterdam.
- Morris, F.L. (1973), Advice on structuring compilers and proving them correct, *Proc. Symp. on Principles of Programming Languages*, Boston, 144–152.
- Morris, J.H. (1968), Lambda-calculus models of programming languages, Proj. MAC, Rep. MAC-TR-57, MIT, Cambridge, Mass.
- Paulson, L.C. (1991), ML for the working mathematician, Cambridge University Press.
- Peyton Jones, S.L. (1987), The Implementation of Functional Programming Languages, Prentice-Hall, Hemel Hempstead.
- Peyton Jones, S.L. and D.R. Lester (1991), Implementing functional languages, Prentice Hall, Hemel Hempstead.
- Reade, C. (1989), Elements of functional programming, Addison-Wesley, Reading, Mass.
- Rosen, B.K. (1973), Tree-manipulating systems and Church-Rosser theorems, *Journal of the ACM*, 20, 160–187.

- Scott, D.S. (1976), Data types as lattices, *SIAM J. Computing*, 5, 522–587.
- Scott, D.S. and C. Gunter (1990), Semantic domains, Chapter 12 of *Handbook of Theoretical Computer Science*, Elsevier, Amsterdam.
- Turner, D.A. (1985), Miranda: A non-strict functional language with polymorphic types, in: *Proceedings IFIP International Conference on Functional Programming Languages and Computer Architecture*, LNCS, Vol. 201, Springer-Verlag.
- Whitehead, A.N. (1898), A treatise on universal algebra, Cambridge University Press. Reprinted in 1960 by Hafner, New York.
- Wikström, Å. (1987), Functional programming using standard ML, Prentice-Hall, Hemel Hempstead.
- Winskel, G. (1993), The Formal Semantics of Programming Languages, The MIT Press, Cambridge, Mass.
- Wirsing, M. (1990), Algebraic specification, Chapter 13 of *Handbook of Theoretical Computer Science*, Elsevier, Amsterdam.
- Wright, J.B., E.G. Wagner and J.W. Thatcher (1978), A uniform approach to inductive posets and inductive closure, *Theoret. Computer Science*, 7, 57–77.

Index

abstractor	152	bottomed algebra	61
algebra	10, 28	cored	78
adapted to a typed set	127	fully regular	68
associated	40, 46, 127, 128	initial	63
bottomed	61	minimal	63
containing a typed set	40	monotone	72
disjoint from a typed set	49	regular	67
extension of	49	strongly monotone	72
free	40	structurally monotone	72
functional	111, 120, 121	unambiguous	64
functionally free	111, 125	bottomed algebras	
ground term	58	isomorphic	62
heterogenous	57	bottomed extension	60, 62
initial completion of	105, 122	cored	78
initial	11, 36	flat	60
minimal	33, 41	initial	65
multi-sorted	57	minimal	65
reachable	57	monotone	72
regular	36, 41	regular	12, 67
term	43, 47, 48, 54	bottomed extensions	
term generated	57	conjugate	62
unambiguous	36, 41	bottomed family	62
algebras		bottomed ground term algebra	68
isomorphic	31	bottomed homomorphism	62
application operator	94	bottomed isomorphism	62
partial	96	bottomed poset	85
assignment	24	bottomed set	60
compatible	195, 196	bottomed subalgebra	
associated algebra	40, 46, 127, 128	minimal	63
associated ordering	85	bottom	60
monotone	89	built-in function	111, 133
associated subalgebra	32	cartesian product	24
associated with a mapping	133	case operator	111, 134
base element	74	chain in a poset	102
basic term algebras	141	finite	102
bottom element	60	codomain	23
bottom element of a poset	85		

cofinal directed set	98	extension homomorphism	62
compatible assignment	195, 196	extension of a poset	97
compatible family of relations	74	extension of a signature	49
complete homomorphism	173	extension of a term algebra specifier	50
complete poset	18, 91	extension of an algebra	49
completeness requirement	3	bottomed	60, 62
completion of a poset	97	extension of an enumeration	51
ideal	98, 103	extensionality property	103
initial	97	family of relations	
conjugate bottomed extensions	62	compatible	74
conjugate extension of a poset	97	full compatible	74
consistent homomorphism	173	family	23
constructor name	58	bottomed	62
continuous extension property	98	contained in a family	23
continuous functional extension	116	invariant	31
continuous mapping	18, 91	fat bottomed homomorphism	72
core of bottomed algebra	67	first order functional type	114
core type replacement rules	207, 213	first order system of equations	160
core type	78	first refinement of a weak ordering	88
monotone	79	flat extension	60
cored bottomed algebra	78	free algebra	40
cored bottomed extension	78	full compatible family of relations	74
currying operator	95	full continuous functional extension	111, 117
directed set	18, 91	full continuous hierarchy	138
cofinal	98	full functional extension	110, 115
directed sets		full monotone functional extension	111, 116
mutually cofinal	99	full set-theoretic function hierarchy	138
domain	23	fully regular bottomed algebra	68
domain theory	109	function	23
embedding	91	built-in	111, 133
enumerated signature	28	partial	5
enumeration		primitive	111, 133
extension of	51	strict	5
equation	153	functional algebra	111, 120, 121
equations		functional application operator	
system of	153	partial	111, 118
		total	111, 117

functional extension	110, 115	initial natural number triple	21
continuous	116	integer operator	111, 133
full	110, 115	invariant family	31
full continuous	111, 117	isomorphic algebras	31
full monotone	111, 116	isomorphic bottomed algebras	62
monotone	116	isomorphic posets	91
monotone regular	121	isomorphism	31
trace based	204	bottomed	62
functional signature	111, 120	kernel of a homomorphism	72
functional type	114	labelled tree	54
first order	114	least upper bound	18, 91
higher order	114	magma	57
functionally free algebra	111, 125	mapping	23
functions	124	continuous	18, 91
		monotone	89, 91
global typed set	152	partial	23
ground signature	58	strict	60
ground term algebra	58	total	23
bottomed	68	typed	24
ground type	58	method of logical relations	186
heterogenous algebra	57	minimal algebra	33, 41
higher order functional type	114	minimal bottomed algebra	63
homomorphism	28	minimal bottomed extension	65
bottomed	62	minimal bottomed subalgebra	63
complete	173	minimal subalgebra	33
consistent	173	monotone associated ordering	89
defined by equations	176, 178	monotone bottomed algebra	72
extension	62	monotone bottomed extension	72
fat bottomed	72	monotone core type	79
ideal completion of a poset	98, 103	monotone functional extension	116
ideal of a poset	103	monotone mapping	89, 91
directed	103	monotone regular functional extension	121
principal	103	continuous	190
image	23	extensional	122
initial algebra	11, 36	multi-sorted algebra	57
initial bottomed algebra	63	mutually cofinal directed sets	99
initial bottomed extension	65		
initial completion of a poset	97		
initial completion of algebra	105, 122		

natural number triple	21	regular bottomed extension	12, 67
initial	21	replacement rules	
operator		core type	207
application	94	sensible signature	57
case	111, 134	set of functional types	110, 112
currying	95	set	
integer	111, 133	bottomed	60
partial application	96	directed	18, 91
uncurrying	95	partially ordered	17, 85
operator name	28	preinvariant	53
order isomorphism	91	typed	23
ordering		signature	10, 28
associated with extension	85	enumerated	28
weak	87	extension of	49
partial application operator	96, 111, 118	functional	111, 120
partial function	5	functional	120
partial mapping	23	ground	58
partial order	17, 26, 83	pervasive	35
product	18, 26	sensible	57
partially ordered set	17, 85	single-sorted	35
Peano triple	21	single-sorted signature	35
pervasive signature	35	solution	153
poset	17, 85	standard term algebra	43, 47
bottomed	85	strict extension of a function	207
complete	18, 91	strict function	5
pre-support system	174	strict mapping	60
preinvariant set	53	strongly monotone bottomed algebra	72
primitive function	111, 133	structural induction	208
primitive type	40	subalgebra	31
product partial order	18, 26	associated with invariant family	32
product		minimal	33
cartesian	24	subposet	93
product type	80	proper bottomed	93
proper bottomed subposet	93	support system	174
reachable algebra	57	supports case operators	135
referential transparency	162	supports integer operators	133
regular algebra	36, 41	supports partial application	111, 117
regular bottomed algebra	67	system of equations	153
		first order	160

term algebra specifier	43, 47	unambiguous algebra	36, 41
extension of	50	unambiguous bottomed algebra	64
term algebra	43, 47, 48	uncurrying operator	95
standard	43, 47	uniform value	186, 205
term generated algebra	57	upper bound	18, 91
total mapping	23	least	18, 91
trace based functional extension	204	least	91
trace of bottomed algebra	69	value	
tree		uniform	186, 205
labelled	54	weak ordering	87
tree algebra	54		
type frame	138		
typed mapping	24		
typed set	23		
global	152		
type	28		
functional	114		
ground	58		
of element in a typed set	23		
of operator name	28		
of tree	54		
primitive	40		
typing	23		