5
Recursion

5.1 Introduction

Section 4.6 has introduced operators for describing unbounded processes. The collection of unbounded processes that can be described using the syntax presented so far is, however, still very limited. For example, it is not possible to describe the process given as a transition system in Figure 5.1: it executes an action a followed by a b any number of times before continuing with the execution of c. To extend the expressiveness of the algebraic framework developed up to this point, so that it is possible to describe more realistic processes, this chapter investigates the notion of recursion.

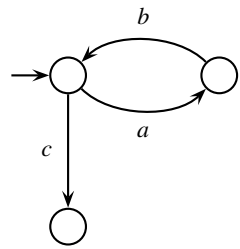

Fig. 5.1. Transition system that cannot be described by a process term so far.

Example 5.1.1 (Coffee machine) Consider a simple coffee machine that operates as follows: after the insertion of a quarter by a client, a cup of coffee is dispensed. A description of such a coffee machine (called *SCM*) could be given as follows:

$$SCM = quarter.coffee.1,$$

where the atomic action *quarter* represents the insertion of a quarter and the atomic action *coffee* represents the handing out of coffee. An objection to

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

this description is that after one cup of coffee has been dispensed no coffee is handed out ever again. A better description therefore would be the following equation:

$$SCM = quarter.coffee.SCM.$$

In this equation, the object to be described, namely the simple coffee machine, is represented by a variable, namely SCM, that occurs not only as the left-hand side of the equation but again in the right-hand side of the equation. Such an equation is called a *recursive* equation and a variable such as SCM is called a *recursion variable*.

A set of recursive equations (that satisfies some simple requirements to be detailed below) is called a *recursive specification*. In the remainder of this chapter, an extension of the process theory BSP(A) with recursive specifications is presented. Recursion is needed in any practical algebraic framework. It turns out that with recursive specifications, in principle, any process can be described.

5.2 Recursive specifications

Definition 5.2.1 (Recursive specification) Let Σ be a signature and let V_R be a set of recursion variables. A *recursive equation* over Σ and V_R is an equation of the form $X = t$ where X is a recursion variable from V_R and t is a term over the signature Σ in which no other variables than those from V_R occur. A *recursive specification* E over Σ and V_R is a set of recursive equations that contains precisely one recursive equation $X = t$ over Σ and V_R for each recursion variable X from V_R. To denote the set of all recursion variables in specification E notation $V_R(E)$ is used.

The reader familiar with fixed point equations may recognize that recursive specifications are nothing but sets of fixed point equations. However, in the context of process algebra, the terminology recursive specification is preferred.

Observe from the above definition that recursion variables can be used in terms (see Definition 2.2.3) just like ordinary variables. However, recursion variables are not the same as ordinary variables. Intuitively, a recursion variable is meant to specify some specific process of interest, whereas ordinary variables are meant to denote arbitrary processes. To emphasize the special role of recursion variables, they are usually written with capitals. In the remainder, the role of recursion variables is investigated in some more detail. The signature and/or the set of recursion variables are often left implicit when they are clear from the context.

Example 5.2.2 (Recursive specifications) Let A be the set of actions that includes a, b, c, and d. The following are examples of recursive specifications:

(i) $E_1 = \{X = a.(b.1 + c.d.0)\}$. The only recursion variable is X. Observe that X does not appear in the right-hand side of the recursive equation. Nevertheless, although no recursion is present, the equation still is a recursive equation according to Definition 5.2.1.

(ii) $E_2 = \{X = a.(b.X + c.0)\}$. Again, X is the only recursion variable. It does occur in the right-hand side of the recursive equation.

(iii) $E_3 = \{X = a.Y, Y = b.X + a.Y\}$. This recursive specification has two recursion variables, namely X and Y.

(iv) $E_4 = \{X_0 = a.X_1, X_{i+1} = a.X_{i+2} + b.X_i \mid i \in \mathbf{N}\}$. This recursive specification has an infinite number of recursion variables, namely $X_0, X_1, \ldots$.

Example 5.2.3 (Recursive specifications) Intuitively, the transition system of Figure 5.1 can be specified by recursion variable X of the recursive specification $E = \{X = a.Y + c.0, Y = b.X\}$. Alternatively, the process can be specified by recursion variable Z in the recursive specification $E' = \{Z = a.b.Z + c.0\}$. Of course, it remains to formalize the claims that these recursive specifications represent the transition system of Figure 5.1. In other words, an operational semantics for recursive specifications in terms of transition systems is needed. Such a semantics is given in Section 5.4.

In deriving equalities between terms in a process theory with recursion, it seems natural to use the equations of recursive specifications as ordinary axioms. Actually, under certain assumptions, they are axioms. A recursive specification can be seen as an extension of some given process theory. The recursion variables should then be interpreted as *constants* in the signature of this process theory. This corresponds to the intuition that a recursion variable specifies a specific process in a model of the theory. Recall that the construction of a model of an equational theory from some algebra requires that constants from the signature of the theory are mapped onto precisely one element in the domain of the algebra (Definitions 2.3.6 (Validity) and 2.3.8 (Model)). Let E be some recursive specification. Table 5.1 gives the process theory BSP(A) extended with E.

Example 5.2.4 (Derivations) Consider the recursive specification $E = \{X = a.Y, Y = b.X\}$. Theory (BSP + E)(A) can be used to derive equation $X = a.(b.X)$ as

$$(\mathrm{BSP} + E)(A) \vdash X = a.Y = a.(b.X).$$

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

$(\mathrm{BSP}+E)(A)$
$\mathrm{BSP}(A)$;
constant: $(X)_{X\in V_R(E)}$;
$(X=t)_{X=t\in E}$ R

Table 5.1. The process theory $(\mathrm{BSP}+E)(A)$.

This result can, in turn, be used in for example the following derivation. For readability, redundant parentheses are omitted.

$$(\mathrm{BSP}+E)(A)\vdash X=a.b.X=a.b.a.b.X.$$

Notation 5.2.5 (Recursion variables) Sometimes, it is desirable to emphasize the fact that recursion variables should be interpreted as constants, or it is necessary to explicitly provide the recursive specification in which a recursion variable occurs. Notation $\mu X.E$ denotes recursion variable X interpreted as a constant as defined by recursive specification E.

Notation 5.2.6 (Process theories with recursion) Table 5.1 shows the extension of a process theory with a single recursive specification. It is convenient to have a notation for a process theory extended with all recursive specifications of potential interest. In such a case, a subscript rec is used. For example, process theory $\mathrm{BSP}(A)$ extended with arbitrary recursive specifications of interest is denoted $\mathrm{BSP}_{\mathrm{rec}}(A)$. The set of all recursive specifications of interest is denoted *Rec*.

Exercises

5.2.1 A man is playing a game of Russian roulette. After pulling the trigger of the gun either a bang or a click follows. This game is repeated until the man dies due to the occurrence of a bang. Describe the process of playing this game with a recursive specification. Use the following atomic actions in the description:

- t for pulling the trigger;
- b for hearing a bang;
- c for hearing a click;
- d for the act of dying.

In this description, you can assume that the man is alive when he starts playing.

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

5.2.2 A more complicated vending machine than the one described in Example 5.1.1 (Coffee machine) charges 25 cents for coffee and 20 cents for hot chocolate. The machine accepts the following coins: 5 cents, 10 cents, and 20 cents. Give a recursive specification for this vending machine. In order to do this, several questions must be answered. For instance: Is it allowed to insert too much money? Can coins be inserted in arbitrary order? Can coins be inserted simultaneously? Can the machine ever terminate? Pay attention to the moments at which choices are made.

5.2.3 Consider a vending machine for refreshing drinks (coffee is not considered to be such a drink). This machine accepts 5 cent, 10 cent, and 20 cent coins only. The customer has to pay 25 cents for a refreshment. When a wrong coin is inserted, it is rejected by the machine. Sometimes the machine does not accept a good coin. Describe this vending machine using the following atomic actions:

- 5 for inserting a 5 cent coin;
- 10 for inserting a 10 cent coin;
- 20 for inserting a 20 cent coin;
- *re* for rejecting a wrong coin;
- *a* for accepting a coin;
- *na* for not accepting a good coin;
- *r* for returning a refreshment.

5.3 Solutions of recursive specifications

An operational semantics for a process theory with recursive specifications is, as before, based on a model built from transition systems. That is, it is an algebra of transition systems modulo bisimilarity. The semantics of a recursive specification in such a model is then a set of processes from the domain of this model, one for each recursion variable in the specification, that validates the equations in the specification when interpreted in the model. Such a set of processes is called a *solution* of the recursive specification. Before turning to the construction of a term model for $\mathrm{BSP}(A)$ with recursion in the next section, the concept of solutions is investigated in some more detail.

Definition 5.3.1 (Solution) Let T be a process theory with signature Σ, let E be a recursive specification over Σ and recursion variables V_R, and let $\mathbb{M}$ be a model of T with respect to interpretation ι. The signature consisting of Σ and the variables in V_R as additional constants is referred to as the *extended*

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

signature. Let κ be an interpretation of this extended signature into algebra $\mathbb{M}$ that is identical to ι for elements of Σ. Interpretation κ is a *solution* of recursive specification E in $\mathbb{M}$ if and only if the equations from the recursive specification are valid in the model under interpretation κ: $\mathbb{M}, \kappa \models X = t$ for any equation $X = t \in E$.

Most of the time, one of the recursion variables in E represents the process of interest. In such cases, a process p in the domain of $\mathbb{M}$ is called a solution of X if and only if there exists a solution κ of E such that $\kappa(X) = p$.

As an aside, recall that a recursive specification can be seen as a set of fixed point equations. Consequently, solutions of recursive specifications are in fact fixed points.

Example 5.3.2 (Solutions) Consider again recursive specification $E' = \{Z = a.b.Z + c.0\}$ of Example 5.2.3 (Recursive specifications). It has been mentioned that the transition system of Figure 5.1 is a solution for Z. Based on Definition 5.3.1, it is only possible to talk about solutions in the context of some basic equational theory and a model for this theory. Process theory BSP(A) is our basic equational theory. For now, let us assume the existence of some operational model for BSP(A) with as its domain equivalence classes of arbitrary transition systems (assuming bisimilarity as the equivalence). Under this assumption, it is possible to prove the claim that the (equivalence class corresponding to the) transition system of Figure 5.1 is a solution for Z by showing that the transition system for the left-hand side of the equation $Z = a.b.Z+c.0$ equals (i.e., is bisimilar to) the transition system for the right-hand side of this equation. The transition system for Z is the transition system of Figure 5.1; the transition system for $a.b.Z + c.0$ can intuitively be constructed via the semantics of BSP(A)-terms (see Section 4.4) and substitution of the transition system for Z at the appropriate point, fusing the two deadlock states. The result is shown in Figure 5.2. It is not difficult to see that the two transition systems of Figures 5.1 and 5.2 are indeed bisimilar. The construction of a bisimulation is left as Exercise 5.3.1.

At this point it is useful to consider the special role of recursion variables in a bit more detail. It has already been mentioned that in the context of an equational theory, they should be considered as constants. Nevertheless, they are called recursion *variables*, and, intuitively, they also behave like variables in some aspects. Considering the above definition of a solution sheds some light on these two faces of recursion variables. Recall that validity (Definition 2.3.6) requires a *fixed* interpretation for the constants and operators of an equational theory in the model under consideration, and an *arbitrary* interpretation

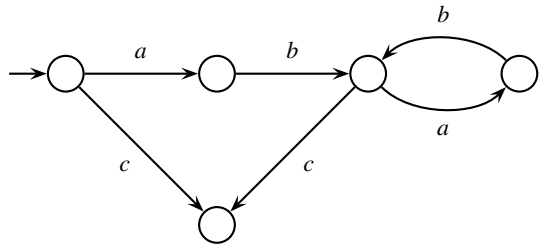

Fig. 5.2. A transition system for $a.b.Z + c.0$ using the transition system of Figure 5.1 for Z.

for all the normal variables in a term. Definition 5.3.1 (Solution) shows that a recursion variable can be interpreted *in any way that satisfies the recursive specification* defining the recursion variable. In that sense, recursion variables are in fact *constrained variables* that behave like variables in the context of a recursive specification but turn into constants when added to the signature of an equational theory.

A question that arises from the introduction of recursion is how to deal with equality between recursion variables, possibly from different recursive specifications. That is, when do two recursion variables specify the same process?

Example 5.3.3 (Equivalence of recursion variables) Consider the recursive specifications $E_1 = \{X_1 = a.X_1\}$ and $E_2 = \{X_2 = a.a.X_2\}$. It can be shown that any solution of X_1 is also a solution of X_2. Assume that BSP(A) is the basic equational theory, and that $\mathbb{M}$ with domain $\mathbf{M}$ is a model of BSP(A) with interpretation ι. Suppose that process p taken from $\mathbf{M}$ is a solution for recursion variable X_1, i.e., there exists an interpretation κ_1 of the signature of BSP(A) plus recursion variable X_1 consistent with interpretation ι such that $\kappa_1(X_1) = p$. According to Definitions 5.3.1 (Solution) and 2.3.6 (Validity), it follows that $p =_{\mathbf{M}} \kappa_1(X_1) =_{\mathbf{M}} \kappa_1(a.)(\kappa_1(X_1)) =_{\mathbf{M}} \kappa_1(a.)(p) =_{\mathbf{M}} \iota(a.)(p)$, i.e., $p =_{\mathbf{M}} \iota(a.)(p)$. Let κ_2 be the interpretation of the signature of BSP(A) consistent with ι and of recursion variable X_2 with $\kappa_2(X_2) = p$. It can be shown that κ_2 specifies a solution of X_2. Using the previous result and the assumption, it follows that $\kappa_2(X_2) =_{\mathbf{M}} p =_{\mathbf{M}} \iota(a.)(p) =_{\mathbf{M}} \iota(a.)(\iota(a.)(p)) =_{\mathbf{M}} \kappa_2(a.)(\kappa_2(a.)(\kappa_2(X_2)))$, i.e., $\mathbb{M}, \kappa_2 \models X_2 = a.a.X_2$. This shows that κ_2, and thus p, is indeed a solution of X_2. Since p was chosen as an arbitrary solution of X_1, this proves that any solution of X_1 is also a solution of X_2.

Using equational reasoning, the fact that any solution of recursion variable X_1 is a solution of X_2 can be achieved by deriving the equation defining X_2 in

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

recursive specification E_2 with all occurrences of X_2 replaced by X_1 from the equation defining X_1 in E_1 in the equational theory $(\mathrm{BSP} + E_1)(A)$:

$$(\mathrm{BSP} + E_1)(A) \vdash X_1 = a.X_1 = a.a.X_1.$$

Note that this derivation does not assume that recursive specification E_2 is part of the equational theory. It is therefore not allowed to use the equation of E_2 as an axiom in the derivation. Despite the fact that E_2 is not a part of the equational theory, the above derivation proves a property of recursion variable X_2 of E_2, namely that any solution of X_1 is also a solution of X_2.

The results derived thus far in this example do not yet mean that X_1 and X_2 have the same solutions. It can still be the case that there are solutions of X_2 that are not solutions of X_1. In fact, an attempt to prove $(\mathrm{BSP}+E_2)(A)\vdash X_2 = a.X_2$, which would imply that any solution of X_2 is also a solution of X_1, will not be successful.

As an aside, the above example shows that equational theory $\mathrm{BSP}(A)$ allows the derivation of a so-called *conditional* equation. The reasoning proves that $\mathrm{BSP}(A) \vdash X = a.X \Rightarrow X = a.a.X$, for any action a and (recursion) variable X. Note that, from the validity point of view, variable X can be interpreted as a normal variable in this conditional equation. The equation has to be valid for *any* interpretation of X, which emphasizes the special role of recursion variables. This book does not study equational theories with conditional equations and axioms in detail. The interested reader is referred to, e.g., (Hussman, 1985).

An important consequence of the reasoning in the above example is the following useful theorem, stating that two recursive specifications E_1 and E_2 specify the same solutions if and only if there is a one-to-one mapping between the recursion variables from E_1 and E_2 such that both E_1 can be rewritten into E_2 with its variables replaced by the corresponding ones in E_1 and E_2 can be rewritten into E_1 with its variables substituted by those in E_2. The theorem uses the notion of substitution of variables of Definition 2.2.6 (Substitution) overloaded to recursion variables (which are strictly speaking constants in the context of an equational theory).

Theorem 5.3.4 (Equivalence of recursive specifications) Let E_1 and E_2 be two recursive specifications over the signature of theory $\mathrm{BSP}(A)$. These two specifications define the same solutions in any model of $\mathrm{BSP}(A)$ if and only if there is a bijective substitution $\sigma : V_R(E_1) \to V_R(E_2)$ with inverse σ^{-1} such that, for all $X_2 = t_2 \in E_2$, $(\mathrm{BSP} + E_1)(A) \vdash \sigma^{-1}(X_2) = t_2[\sigma^{-1}]$ and for all $X_1 = t_1 \in E_1$, $(\mathrm{BSP} + E_2)(A) \vdash \sigma(X_1) = t_1[\sigma]$.

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

Example 5.3.3 (Equivalence of recursion variables) illustrates some of the subtleties of recursive specifications and their solutions. The main idea is that a recursive specification *defines* a process that cannot be expressed as a closed term. (Recall that the major purpose of recursive specifications is to extend the expressiveness of basic process theories.) That is, in principle, a recursive specification is usually expected to have a single solution. However, the definitions of recursive specifications and their solutions allow that a recursive specification has no solutions at all in some model of the basic process theory under consideration, or that it has two or more different solutions. This is illustrated by the following examples.

Example 5.3.5 (No solutions) Consider the following recursive specification: $E = \{X = a.X\}$. In the term model for process theory BSP(A) given in Section 4.4, namely the algebra of transition systems $\mathbb{P}(\mathrm{BSP}(A))_{/\leftrightarrows}$, this recursive specification has no solutions. This can be seen as follows.

Suppose that some process p from the domain of $\mathbb{P}(\mathrm{BSP}(A))_{/\leftrightarrows}$ is a solution of X. The construction of the term model as a quotient algebra (see Definition 2.3.18) implies that there exists a closed BSP(A)-term t such that p is the equivalence class of t under bisimilarity: $p = [t]_{\leftrightarrows}$. In other words, t is a representative of the equivalence class p. From Definition 5.3.1 (Solution), it follows that there must be an interpretation κ such that $\kappa(X) = p$. The construction of the term model $\mathbb{P}(\mathrm{BSP}(A))_{/\leftrightarrows}$ and the interpretation of recursion variable X as a constant imply that X must be a representative of equivalence class p, and thus that X and t are representatives of the same equivalence class.

Consider term t again. As any closed BSP(A)-term is also a closed (BSP + PR)(A)-term, by Theorem 4.5.4 (Bounded depth), there exists a natural number n such that $(\mathrm{BSP}+\mathrm{PR})(A) \vdash \pi_k(t) = t$ for all $k \geq n$. Thus, the soundness result of Theorem 4.5.8 implies that $\pi_k(t) \leftrightarrows t$ for all $k \geq n$.

Next, recall the n-fold action prefix of Notation 4.6.6. It is not hard to prove that $(\mathrm{BSP} + \mathrm{PR} + E)(A) \vdash \pi_m(X) = a^m 0$ for any natural number m. Since term t and constant X are representatives of the same equivalence class under bisimilarity in the term model $\mathbb{P}(\mathrm{BSP}(A))_{/\leftrightarrows}$, it follows that $\pi_m(t) \leftrightarrows a^m 0$ for any natural number m.

Combining the facts derived in the last two paragraphs means that it must be the case that for all natural numbers $k, l \geq n$, $a^k 0 \leftrightarrows \pi_k(t) \leftrightarrows t \leftrightarrows \pi_l(t) \leftrightarrows a^l 0$. However, it is easily shown that the process terms $a^k 0$ and $a^l 0$ are not bisimilar for the cases that $k \neq l$. Hence, a contradiction results, which implies that there cannot be a process p in the algebra $\mathbb{P}(\mathrm{BSP}(A))_{/\leftrightarrows}$ that is a solution of E. Thus, this example shows that the algebra $\mathbb{P}(\mathrm{BSP}(A))_{/\leftrightarrows}$ cannot be turned into a model for $(\mathrm{BSP} + E)(A)$.

The observation made at the end of the above example relates solutions to models, and can be generalized as follows.

Theorem 5.3.6 (Solutions vs. models) A model $\mathbb{M}$ of BSP(A) has a solution for a recursive specification E if and only if $\mathbb{M}$ is also a model of the extended theory (BSP + E)(A).

Proof Given the definition of theory (BSP + E)(A) in Table 5.1, the result immediately follows from Definitions 5.3.1 (Solution) and 2.3.8 (Model). □

Example 5.3.7 (Multiple solutions) Consider recursive specification $\{X = X\}$. For any process theory and any model $\mathbb{M}$ of that theory, each element p of the domain $\mathbf{M}$ of that model is a solution for X. If $\mathbf{M}$ has at least two elements, the recursive specification has at least two different solutions. Thus, it does not uniquely define a process.

The first of the above examples indicates that there are recursive specifications that in some models of some given basic process theory do not have a solution at all. In some sense, this is a desirable result, as the goal of introducing recursion is to increase expressiveness. The second example indicates that in any model with two or more elements there is more than one solution for some recursive specifications. When constructing a model for a process theory with recursion, it is necessary to take these intricacies into account. It should also be investigated under what conditions a recursive specification precisely defines one unique process. Section 5.5 addresses these issues in more detail. First, the next section presents a term model for BSP(A) with recursive specifications.

Exercises

5.3.1 Give a bisimulation relation between the two transition systems of Figures 5.1 and 5.2.

5.3.2 Consider Example 5.3.2 (Solutions). Along the lines of this example, provide a solution for recursion variable *SCM* defined in the recursive specification of Example 5.1.1 (Coffee machine), including a correctness argument.

5.3.3 Find in the term model $\mathbb{P}(\mathrm{BSP}(A))_{/\leftrightarrow}$ two different solutions of the recursive equation $X = X + a.0$.

5.3.4 Does the term model $\mathbb{P}(\mathrm{BSP}(A))_{/\leftrightarrow}$ have a solution of the recursive equation $X = a.X + X$?

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

5.3.5 Show that any solution of X from $\{X = Y, Y = a.X\}$ is also a solution of Z from $\{Z = a.W, W = a.Z\}$.

5.4 The term model

This section presents a term model for BSP(A) with recursion. Recall Notations 5.2.5 (Recursion variables) and 5.2.6 (Process theories with recursion). Since recursion variables are interpreted as constants in the process theory BSP(A) with recursion, that is, $\mathrm{BSP}_{\mathrm{rec}}(A)$, note that $\mathcal{C}(\mathrm{BSP}_{\mathrm{rec}}(A))$ denotes all closed terms over the BSP(A) signature extended with all recursion variables of potential interest.

Definition 5.4.1 (Term algebra) The *term algebra* for $\mathrm{BSP}_{\mathrm{rec}}(A)$ is the algebra $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A)) = (\mathcal{C}(\mathrm{BSP}_{\mathrm{rec}}(A)), +, (a._-)_{a\in A}, (\mu X.E)_{E\in Rec, X\in V_R(E)}, 0, 1)$.

Notation 5.4.2 (Solutions of terms) Let E be a recursive specification over a set of recursion variables V_R. For convenience, the notation $\mu X.E$ for some variable X in V_R is generalized to $\mu t.E$ for some arbitrary term t in $\mathcal{C}((\mathrm{BSP} + E)(A))$. An inductive definition of this notation is given by:

(i) $\mu 1.E = 1$;
(ii) $\mu 0.E = 0$;
(iii) for any recursion variable $X \in V_R$, $\mu(\mu X.E).E = \mu X.E$;
(iv) for any $a \in A$ and $t \in \mathcal{C}((\mathrm{BSP} + E)(A))$, $\mu(a.t).E = a.(\mu t.E)$;
(v) for any $s, t \in \mathcal{C}((\mathrm{BSP} + E)(A))$, $\mu(s + t).E = \mu s.E + \mu t.E$.

The termination predicate and the ternary transition relation in the transition-system space underlying the model under construction are defined via the term deduction system in Table 5.2, which extends the term deduction systems in Tables 4.2 and 4.4. Essentially, the new deduction rules express that a solution for a recursion variable in a recursive specification behaves as the right-hand side of its defining recursive equation.

Proposition 5.4.3 (Congruence) Bisimilarity is a congruence on term algebra $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))$.

Proof The property follows immediately from the format of the deduction rules given in Tables 4.2, 4.4, and 5.2, and Theorem 3.2.7 (Congruence theorem). □

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

$TDS(\mathrm{BSP}_{\mathrm{rec}}(A))$

$TDS(\mathrm{BSP}(A))$;

constant: $(\mu X.E)_{E \in Rec, X \in V_R(E)}$;

y;

$$\frac{\mu t.E\downarrow}{\mu X.E\downarrow} \qquad \frac{\mu t.E \xrightarrow{a} y}{\mu X.E \xrightarrow{a} y}$$

Table 5.2. Deduction rules for recursion (with $a \in A$ and $X = t \in E$).

Definition 5.4.4 (Term model of $\mathrm{BSP}_{\mathrm{rec}}(A)$) The term model of $\mathrm{BSP}_{\mathrm{rec}}(A)$ is the quotient algebra $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrow}$.

Theorem 5.4.5 (Soundness of $\mathrm{BSP}_{\mathrm{rec}}(A)$) Theory $\mathrm{BSP}_{\mathrm{rec}}(A)$ is a sound axiomatization of $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrow}$, i.e., $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrow} \models \mathrm{BSP}_{\mathrm{rec}}(A)$.

Proof According to Definition 2.3.8 (Model), it must be shown that, for each axiom $s = t$ of $\mathrm{BSP}_{\mathrm{rec}}(A)$, $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrow} \models s = t$. The proof for the axioms of $\mathrm{BSP}(A)$ carries over directly from the proof of Theorem 4.4.7 (Soundness of $\mathrm{BSP}(A)$) and Exercise 4.4.4. The proof for the axioms that are new in $\mathrm{BSP}_{\mathrm{rec}}(A)$, i.e., all the *recursive equations* in all recursive specifications of interest, follows in a straightforward way from the deduction rules in Table 5.2. If $X = t$ is some recursive equation in recursive specification E, then $R = \{(\mu X.E, \mu t.E)\} \cup \{(p, p) \mid p \in \mathcal{C}(\mathrm{BSP}_{\mathrm{rec}}(A))\}$ is the bisimulation relation showing the desired result. □

Example 5.4.6 (Solutions in term models) Solutions of recursive equations over the $\mathrm{BSP}(A)$ signature in the term model of $\mathrm{BSP}_{\mathrm{rec}}(A)$ are equivalence classes of closed $\mathrm{BSP}_{\mathrm{rec}}(A)$-terms under bisimilarity. Consider again recursive specification $E = \{X = a.Y + c.0, Y = b.X\}$ of Example 5.2.3. The solution of X in the term model of $\mathrm{BSP}_{\mathrm{rec}}(A)$ is $[\mu X.E]_{\leftrightarrow}$. The solution of Z of recursive specification $E' = \{Z = a.b.Z + c.0\}$ of the same example is $[\mu Z.E']_{\leftrightarrow}$. It is easy to show that $\mu X.E \leftrightarrow \mu Z.E'$. Figure 5.3 shows the transition systems for $\mu X.E$ and $\mu Z.E'$. Hence, $[\mu X.E]_{\leftrightarrow} = [\mu Z.E']_{\leftrightarrow}$, which means that X and Z specify the same process in the term model of $\mathrm{BSP}_{\mathrm{rec}}(A)$. These results confirm the claims made in Example 5.2.3.

For reasons of brevity, it is often said that a closed $\mathrm{BSP}_{\mathrm{rec}}(A)$-term is a solution of some recursive specification over the $\mathrm{BSP}(A)$ signature, meaning that the corresponding equivalence class under bisimilarity is a solution.

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

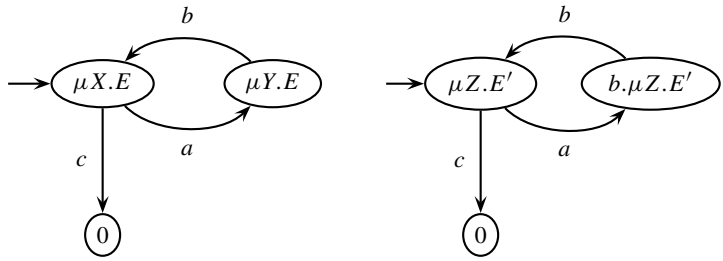

Fig. 5.3. Transition systems for $\mu X.E$ and $\mu Z.E'$ of Example 5.4.6.

It is interesting to observe that the term model $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrows}$ defines specific solutions for every recursive specification in the theory $\mathrm{BSP}_{\mathrm{rec}}(A)$, even for the ones that have multiple solutions. Recall that, following Definition 2.3.18 (Quotient algebra), the standard interpretation of constants and functions of theory $\mathrm{BSP}_{\mathrm{rec}}(A)$ in the term model $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrows}$ maps all constants, including the constants corresponding to recursion variables, onto their equivalence classes under bisimilarity, and each operator onto the corresponding function in the term model. For the recursive specification $\{X = X\}$, for example, which has all processes in $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrows}$ as solutions, the standard interpretation in the term model maps X onto (the equivalence class of) 0, because the term deduction system of Table 5.2 does not allow the derivation of any steps or the successful termination of X. This observation is consistent with the fact that any model of a process theory, and therefore also any model of $\mathrm{BSP}_{\mathrm{rec}}(A)$, is always accompanied with an interpretation that maps every closed term, and hence every recursion variable in the signature of $\mathrm{BSP}_{\mathrm{rec}}(A)$, to precisely one process in the model. These observations once again emphasize the dual role of recursion variables, as constrained variables when considering solutions in some given model of the basic theory under consideration and as constants when considering a model of the basic theory extended with recursion.

It is not possible to give a general ground-completeness result for $\mathrm{BSP}_{\mathrm{rec}}(A)$ and the term model introduced in this section, because in general it is not guaranteed that the recursive equations capture all equalities between recursion variables that are valid in the term model. The following example illustrates this fact.

Example 5.4.7 (Ground-completeness of $\mathrm{BSP}_{\mathrm{rec}}(A)$) Consider again Example 5.3.3 (Equivalence of recursion variables). Assume that the set of recursive specifications *Rec* equals $\{E_1, E_2\}$. It has been mentioned in Example 5.3.3

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

that it is not possible to prove that the two recursion variables X_1 and X_2 always have the same solutions. Thus, in particular, it is not possible to prove that $\mathrm{BSP}_{\mathrm{rec}}(A) \vdash X_1 = X_2$. However, in the term model constructed in this section, the equality $X_1 = X_2$ is valid, i.e., $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrow} \models X_1 = X_2$. The bisimulation relation proving this is the relation $\{(X_1, X_2), (X_1, a.X_2)\}$. This simple example proves that theory $\mathrm{BSP}_{\mathrm{rec}}(A)$ is in general not ground-complete for the term model. Note that for specific instances of the set of recursive specifications *Rec* it may still be possible to obtain a ground-completeness result.

Given the results obtained so far, it is possible to prove that the extension of $\mathrm{BSP}(A)$ with recursion is conservative. In other words, using the extended theory, it is not possible to derive any new equalities between closed $\mathrm{BSP}(A)$-terms. Note that it is not possible to prove an elimination result; that is, it is in general not possible to eliminate recursion variables from closed $\mathrm{BSP}_{\mathrm{rec}}(A)$-terms resulting in $\mathrm{BSP}(A)$-terms. This last observation should not be very surprising given the fact that recursion has been introduced in order to extend the expressiveness of the process theory $\mathrm{BSP}(A)$. It is also possible to show that $\mathrm{BSP}_{\mathrm{rec}}(A)$ is a conservative ground-extension of theory $\mathrm{MPT}_{\mathrm{rec}}(A)$, i.e., the minimal theory $\mathrm{MPT}(A)$ of Section 4.2 extended with recursion.

Theorem 5.4.8 (Conservative ground-extension) The theory $\mathrm{BSP}_{\mathrm{rec}}(A)$ is a conservative ground-extension of theories $\mathrm{BSP}(A)$ and $\mathrm{MPT}_{\mathrm{rec}}(A)$.

Proof The desired conservativity result for theory $\mathrm{BSP}(A)$ follows immediately from Theorem 3.2.21 (Conservativity), the fact that $\mathrm{BSP}(A)$ is a ground-complete axiomatization of the term model $\mathbb{P}(\mathrm{BSP}(A))_{/\leftrightarrow}$ (Theorem 4.4.12), the fact that $\mathrm{BSP}_{\mathrm{rec}}(A)$ is a sound axiomatization of $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrow}$, and the fact that $\mathit{TDS}(\mathrm{BSP}_{\mathrm{rec}}(A))$ is an operational conservative extension of $\mathit{TDS}(\mathrm{BSP}(A))$ (Table 4.4). This last fact is a direct consequence of the format of the deduction rules in Tables 4.2, 4.4 and 5.2, and Theorem 3.2.19 (Operational conservative extension).

For theory $\mathrm{MPT}_{\mathrm{rec}}(A)$, it is not possible to give a similar proof, because theory $\mathrm{MPT}_{\mathrm{rec}}(A)$ is not a ground-complete axiomatization of the underlying term model (see Example 5.4.7 and Exercise 5.4.3). Instead, the proof goes along the lines of Theorem 4.4.1. The signature and axioms of $\mathrm{BSP}_{\mathrm{rec}}(A)$ include the signature and axioms of $\mathrm{MPT}_{\mathrm{rec}}(A)$, implying that $\mathrm{BSP}_{\mathrm{rec}}(A)$ is a (ground-)extension of $\mathrm{MPT}_{\mathrm{rec}}(A)$. Furthermore, since no axioms of $\mathrm{BSP}_{\mathrm{rec}}(A)$ can introduce a completely new or entirely eliminate a 1 constant, any derivation in $\mathrm{BSP}_{\mathrm{rec}}(A)$ showing the equality of two closed $\mathrm{MPT}_{\mathrm{rec}}(A)$-terms is in

fact a derivation in $\mathrm{MPT}_{\mathrm{rec}}(A)$, which yields the desired conservativity result. □

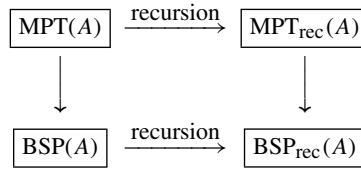

Fig. 5.4. Conservativity results for $\mathrm{MPT}_{\mathrm{rec}}(A)$ and $\mathrm{BSP}_{\mathrm{rec}}(A)$.

Figure 5.4 shows the conservativity results for theory $\mathrm{BSP}_{\mathrm{rec}}(A)$. It also shows that $\mathrm{MPT}_{\mathrm{rec}}(A)$ is a conservative ground-extension of $\mathrm{MPT}(A)$. The proof is left as an exercise.

Exercises

5.4.1 For each of the following terms, give the transition system associated with it by means of the deduction rules presented in this section.

(a) $\mu X.\{X = a.Y, Y = b.X\}$,
(b) $\mu X.\{X = 1 + a.Y, Y = b.X\}$,
(c) $\mu(a.X).\{X = a.b.X + b.Y + 0, Y = X + a.(Y + X)\}$,
(d) $\mu(b.X + 1).\{X = a.X + Y, Y = a.Y\}$.

5.4.2 Assume the standard interpretation of constants and functions of the-ory $\mathrm{BSP}_{\mathrm{rec}}(A)$ in its term model $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrows}$, i.e., all constants are mapped onto their equivalence classes under bisimilarity, and each operator is mapped onto the corresponding function in the term model following Definition 2.3.18 (Quotient algebra).
Check that $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrows} \models$

(a) $\mu X.\{X = X\} = 0$,
(b) $\mu X.\{X = a.0\} = a.0$,
(c) $\mu X.\{X = a.X + Y, Y = a.Y\} = \mu Z.\{Z = a.Z\}$,
(d) $\mu X.\{X = X + a.0\} = a.0$, and
(e) $\mu X.\{X = X + a.X\} = \mu Y.\{Y = a.Y\}$.

5.4.3 Develop the theory $\mathrm{MPT}_{\mathrm{rec}}(A)$ by extending theory $\mathrm{MPT}(A)$ with recursion. Define the equational theory $\mathrm{MPT}_{\mathrm{rec}}(A)$, provide a term model, prove soundness, and prove a conservativity result. Argue that there are no general elimination and ground-completeness results.

5.5 Recursion principles

It is a common situation that there is a need to add recursion to some given basic process theory without recursion. It is then interesting to know to what extent it is possible to build upon existing models of that basic theory when developing an operational semantics of the theory with recursion. The models of interest are those in which recursive specifications have precisely one solution. However, Example 5.3.7 (Multiple solutions) shows that it is in general impossible to find interesting models in which all recursive specifications have precisely one solution. Furthermore, Example 5.3.5 (No solutions) gives a recursive specification over the signature of theory $\mathrm{BSP}(A)$ that has no solutions in the term model $\mathbb{P}(\mathrm{BSP}(A))_{/\leftrightarrows}$.

These examples illustrate that it is interesting to study which models of process theories allow an easy extension of a theory with recursion. Example 5.3.7 (Multiple solutions) shows that the goal of precisely one solution for each recursive specification cannot be achieved without at least some restrictions. Section 5.4 has already shown the construction of a term model for a theory with recursion in such a way that every recursive specification has a solution. However, this term model does not address the issue that some recursive specifications have multiple solutions. To investigate these issues in a generic setting, this section introduces a number of so-called *recursion principles*. A recursion principle specifies characteristics of models and/or recursive specifications. A very elementary recursion principle is the *Recursive Definition Principle* (RDP), stating that, given a basic theory with some model, any recursive specification over the syntax of that theory has *at least one* solution in the given model.

Definition 5.5.1 (RDP) Assume some equational theory and some model of this theory. The *Recursive Definition Principle* (RDP) is the following assumption: every recursive specification over the signature of the theory has a solution in the given model.

The notion of validity for ordinary equations (see Definition 2.3.6) carries over to recursion principles. Given a model $\mathbb{M}$ of some equational theory and some recursion principle R, $\mathbb{M} \models R$ denotes that R is valid in model $\mathbb{M}$. Since a recursion principle must be valid in the algebra that is considered as a potential model for the process theory at hand, the introduction of a principle restricts the allowed models of the process theory.

It is interesting to consider the recursion principle RDP in a bit more detail in the context of process theories $\mathrm{BSP}(A)$ and $\mathrm{BSP}_{\mathrm{rec}}(A)$ and their term models.

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

An immediate consequence of Example 5.3.5 (No solutions) is that RDP is not valid in the term model of theory BSP(A), i.e., the algebra $\mathbb{P}(\mathrm{BSP}(A))_{/\leftrightarrow}$.

Theorem 5.5.2 (Invalidity of RDP **for** $\mathbb{P}(\mathrm{BSP}(A))_{/\leftrightarrow}$**)** Principle RDP is not valid in the term model of BSP(A), i.e., $\mathbb{P}(\mathrm{BSP}(A))_{/\leftrightarrow} \not\models \mathrm{RDP}$.

In the term model of theory $\mathrm{BSP}_{\mathrm{rec}}(A)$, however, which is also a model of theory BSP(A), RDP is valid.

Theorem 5.5.3 (Validity of RDP **in** $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrow}$**)** Recursion principle RDP is valid in the term model of $\mathrm{BSP}_{\mathrm{rec}}(A)$, i.e., $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrow} \models \mathrm{RDP}$.

Proof The validity of RDP follows immediately from the construction of the model $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrow}$. For any recursive specification E and recursion variable X in E, the equivalence class $[\mu X.E]_{/\leftrightarrow}$ is a solution for X. Hence, each recursive specification has a solution. □

Theorem 5.5.3 is not really surprising. Theory $\mathrm{BSP}_{\mathrm{rec}}(A)$ and its model $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrow}$ have been constructed in such a way that every recursive specification has at least one solution. In fact, given the construction of theory $\mathrm{BSP}_{\mathrm{rec}}(A)$, this observation straightforwardly yields the following result.

Theorem 5.5.4 ($\mathrm{BSP}_{\mathrm{rec}}(A)$ **and** RDP**)** An arbitrary algebra $\mathbb{M}$ is a model of $\mathrm{BSP}_{\mathrm{rec}}(A)$ if and only if it validates theory BSP(A) and recursion principle RDP, i.e., $\mathbb{M} \models \mathrm{BSP}_{\mathrm{rec}}(A)$ if and only if $\mathbb{M} \models \mathrm{BSP}(A)$ and $\mathbb{M} \models \mathrm{RDP}$.

As already mentioned before, the construction of theory $\mathrm{BSP}_{\mathrm{rec}}(A)$ does not enforce that a recursive specification has only one solution; recursive specifications may have multiple solutions in some given model. One solution to this problem would be to simply disallow all such models. However, as a direct consequence of Example 5.3.7 (Multiple solutions), this only leaves models in which all processes are equal, so-called *one-point models*. This is clearly undesirable. Other solutions are to restrict the recursive specifications that are allowed or considered relevant, or to select a unique default solution of a recursive specification in case it has more than one. The latter is done in for example CCS, where the notion of a least fixed point is used to select solutions (Milner, 1980; Milner, 1989). Also the deduction rules for recursion underlying a term model for a theory with recursion, as for example given in Table 5.2, generate one particular solution, as explained in the previous section and further illustrated by Exercise 5.4.2. However, choosing this generated solution as the default solution does not allow the interpretation of recursion variables as constrained variables, and is furthermore model-dependent. Therefore, a

restriction of the set of the allowed recursive specifications is considered. This can be done independent of any particular model. To this end, the notion of *guardedness* of recursive specifications is introduced. The basic idea is that guarded recursive specifications have precisely one solution.

Definition 5.5.5 (Guardedness, part 1) Let s be a $\mathrm{BSP}_{\mathrm{rec}}(A)$-term. An occurrence of a (normal or recursion) variable x in term s is called *guarded* if and only if it occurs in the operand of an action-prefix operator, i.e., s has a subterm of the form $a.t$ for some $a \in A$ and $\mathrm{BSP}_{\mathrm{rec}}(A)$-term t, and this x occurs in t. Term s is called *completely guarded* if and only if all occurrences of all variables in s are guarded. A *recursive specification* E is called *completely guarded* if and only if all right-hand sides of all recursive equations of E are completely guarded.

Example 5.5.6 (Guardedness)

(i) Let $t_1 \equiv a.X + Y + c.(b.0 + X)$. In this term, both occurrences of recursion variable X are guarded: the first one because it is the operand of the action-prefix operator $a._-$; the second one because it occurs in the operand of the action-prefix operator $c._-$. The occurrence of Y is unguarded. Therefore, t_1 is not completely guarded.

(ii) Let $t_2 \equiv a.X + Y + a.(b.0 + Y)$. In this term, the occurrence of X is guarded and Y occurs both guarded and unguarded. As a result, t_2 is not completely guarded.

(iii) Let $t_3 \equiv a.(X + Y)$. The occurrences of X and Y are guarded and hence t_3 is completely guarded.

(iv) Recursive specification $E_1 = \{X_1 = a.X_1, Y_1 = a.X_1\}$ is completely guarded.

(v) Recursive specification $E_2 = \{X_2 = a.X_2, Y_2 = X_2\}$ is not completely guarded because X_2 occurs unguarded in equation $Y_2 = X_2$.

A closer look at the last two examples reveals that the notion of guardedness presented in Definition 5.5.5 is too restrictive. Recursive specification E_1 is completely guarded. Recall that a guarded specification is supposed to have a single solution in some appropriate model of the basic theory $\mathrm{BSP}(A)$. Assuming that this is the case, recursion variables X_1 and Y_1 in E_1 each determine precisely one process in that model. However, recursive specification E_2 is not completely guarded, suggesting that recursion variables X_2 and Y_2 do not determine precisely one process each. It can be shown that the latter is not true by proving that the two recursive specifications are equivalent in the sense of Theorem 5.3.4 (Equivalence of recursive specifications), meaning that

all solutions of X_1 and Y_1 in some model of BSP(A) are also solutions of X_2 and Y_2 and vice versa. In other words, the two recursive specifications specify precisely the same processes.

Example 5.5.7 (Guardedness anomaly) Consider again recursive specifications $E_1 = \{X_1 = a.X_1, Y_1 = a.X_1\}$ and $E_2 = \{X_2 = a.X_2, Y_2 = X_2\}$ of Example 5.5.6. In order to obtain that all solutions of X_1 and Y_1 are also solutions of X_2 and Y_2 in some given model of BSP(A), it suffices to show that the recursive equations from E_2 with the occurrences of X_2 and Y_2 replaced by X_1 and Y_1, respectively, are derivable from the process theory $(\mathrm{BSP} + E_1)(A)$ (Theorem 5.3.4). This is easily achieved as follows:

$$(\mathrm{BSP} + E_1)(A) \vdash X_1 = a.X_1$$

and

$$(\mathrm{BSP} + E_1)(A) \vdash Y_1 = a.X_1 = X_1.$$

Similarly, the fact that all solutions of X_2 and Y_2 are solutions of X_1 and Y_1, can be seen as follows:

$$(\mathrm{BSP} + E_2)(A) \vdash X_2 = a.X_2$$

and

$$(\mathrm{BSP} + E_2)(A) \vdash Y_2 = X_2 = a.X_2.$$

But this means that X_1 and Y_1 determine unique processes if and only if X_2 and Y_2 determine unique processes, showing that the notion of guardedness introduced in Definition 5.5.5 (Guardedness, part 1) is not suitable for the intended purpose.

The reason for the anomaly observed in the above example is that the definition of guardedness is a purely syntactical one. It does not take into account that syntactically different process terms can be semantically equivalent. In the following definition, a notion of guardedness is defined that also includes E_2 of the above example as a guarded recursive specification; the definition builds upon the notion 'completely guarded' as defined in Definition 5.5.5 (Guardedness, part 1).

Definition 5.5.8 (Guardedness, part 2) Let s be a $\mathrm{BSP}_{\mathrm{rec}}(A)$-term. Term s is *guarded* if and only if there exists a $\mathrm{BSP}_{\mathrm{rec}}(A)$-term t that is completely guarded and derivably equal to s, i.e., $\mathrm{BSP}_{\mathrm{rec}}(A) \vdash s = t$. A recursive specification E is *guarded* if and only if it can be rewritten into a completely guarded recursive specification F, i.e., there exists a completely guarded recursive specification F with $V_R(E) = V_R(F)$ and for all $X = t \in F$, $(\mathrm{BSP} + E)(A) \vdash X = t$.

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

Example 5.5.9 (Guarded recursive specifications) Consider again recursive specification $E_2 = \{X_2 = a.X_2, Y_2 = X_2\}$. Although this recursive specification is not completely guarded, the derivations

$$(\mathrm{BSP} + E_2)(A) \vdash X_2 = a.X_2$$

and

$$(\mathrm{BSP} + E_2)(A) \vdash Y_2 = X_2 = a.X_2$$

show that it can be rewritten into the completely guarded recursive specification $F = \{X_2 = a.X_2, Y_2 = a.X_2\}$, indicating that it is guarded.

Note that, although the guardedness notions in the above definitions are defined in the context of theory $\mathrm{BSP}(A)$, they can be generalized to arbitrary process theories in a straightforward way.

The question remains whether the notion of guardedness defined in Definition 5.5.8 is satisfactory. To investigate this issue, a new recursion principle is introduced that disallows models in which guarded recursive specifications have more than one solution. This principle is called the *Recursive Specification Principle*.

Definition 5.5.10 (RSP) Assume some equational theory and some model of this theory. The *Recursive Specification Principle* (RSP) is the following assumption: a guarded recursive specification over the signature of the theory has at most one solution in the model.

In the context of process theory $\mathrm{BSP}(A)$ and its models $\mathbb{P}(\mathrm{BSP}(A))_{/\leftrightarrows}$ and $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrows}$, the following theorem holds.

Theorem 5.5.11 (Validity of RSP) Principle RSP is valid in both the term model of $\mathrm{BSP}(A)$ and the term model of $\mathrm{BSP}_{\mathrm{rec}}(A)$, i.e., $\mathbb{P}(\mathrm{BSP}(A))_{/\leftrightarrows} \models \mathrm{RSP}$ and $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrows} \models \mathrm{RSP}$.

Proof The theorem is a consequence of Theorems 5.5.29 (Relation between the recursion principles) and 5.5.23 (Validity of AIP^-), proven later in this section. Note that these theorems are only proven for theories with projection operators. However, since extensions of $\mathrm{BSP}(A)$ and $\mathrm{BSP}_{\mathrm{rec}}(A)$ with projection operators are ground-conservative, any element in the term models of $\mathrm{BSP}(A)$ and $\mathrm{BSP}_{\mathrm{rec}}(A)$ is also an element of the term models of $(\mathrm{BSP}+\mathrm{PR})(A)$ and $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$. Hence, if a recursive specification over the signature without projection operators has at most one solution in the term models of the theories with projection, it has also at most one solution in the term models of the theories without projection. □

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

Through the introduction of the recursion principles RSP and RDP, it has been shown that, in the term model $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrows}$, every *guarded* recursive specification over the signature of theory $\mathrm{BSP}(A)$ has at most one solution and that *every* recursive specification over the $\mathrm{BSP}(A)$-signature has at least one solution. Together, this gives that *every guarded* recursive specification has *precisely one* solution. Thus, the notion of guardedness introduced in Definition 5.5.8 is meaningful indeed.

So how can recursion principles be used to reason about the equivalence of recursive specifications? In general, recursion principles may affect the notion of derivability, as defined in Definition 2.2.8. Thus, recursion principles may be used to derive equations between terms of some process theory that cannot be derived from the basic axioms of the theory, in particular equations involving recursion variables. Recursion principle RDP does not affect derivability. However, RSP does have an effect. The detailed redefinition of derivability is omitted, in this case and also in the remainder when other recursion principles are introduced. The use of recursion principles in the derivation of equations with recursion variables is illustrated through examples, as for instance Example 5.5.12 that follows below.

To clarify that a recursion principle is assumed to hold in derivations, its acronym is added to the name of the process theory being used. For example, the notation $(\mathrm{BSP}_{\mathrm{rec}} + \mathrm{RSP})(A)$ indicates that the theory $\mathrm{BSP}_{\mathrm{rec}}(A)$ is being used for derivations while assuming RSP to hold.

Example 5.5.12 (Recursion principle RSP**)** Consider again recursive specifications $E_1 = \{X_1 = a.X_1\}$ and $E_2 = \{X_2 = a.a.X_2\}$ of Example 5.3.3 (Equivalence of recursion variables). On intuitive grounds, it is expected that X_1 and X_2 define the same process. How can this (correct) intuition be captured in the equational theory? Observe that it is necessary to consider theory $(\mathrm{BSP} + E_1 + E_2)(A)$, or an extension of this theory. Only when E_1 and E_2 are part of the equational theory, the theory has a means to reason about the recursive specifications, and in particular about the equivalence of X_1 and X_2. As already mentioned in Example 5.3.3, an attempt to prove the equivalence of X_1 and X_2 using only the basic axioms of $(\mathrm{BSP}+E_1+E_2)(A)$ will not be successful. However, assuming recursion principle RSP, the desired equivalence can be proven.

First, it is shown that any solution of X_1 is a solution of X_2 by deriving the equation for X_2 with all occurrences of X_2 replaced by occurrences of X_1 from the process theory $(\mathrm{BSP} + E_1)(A)$ as shown before in Example 5.3.3:

$$(\mathrm{BSP} + E_1)(A) \vdash X_1 = a.X_1 = a.a.X_1.$$

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

Second, assume the validity of principle RSP. This means that only models of theory $(\mathrm{BSP}+E_1+E_2)(A)$ that satisfy this principle, such as the term models $\mathbb{P}((\mathrm{BSP}+E_1+E_2)(A))_{/\leftrightarrows}$ or $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrows}$, are considered. RSP, the fact that both E_1 and E_2 are guarded, and the fact that every model of $(\mathrm{BSP}+E_1+E_2)(A)$ contains solutions for E_1 and E_2 by definition (see Theorem 5.3.6 (Solutions vs. models)) imply that both X_1 and X_2 have precisely one solution in any model considered. Since any solution for X_1 is a solution for X_2, the solutions for X_1 and X_2 must be the same. This shows that X_1 and X_2 are derivably equivalent in the process theory $(\mathrm{BSP}+E_1+E_2+\mathrm{RSP})(A)$:

$$(\mathrm{BSP}+E_1+E_2+\mathrm{RSP})(A)\vdash X_1=X_2.$$

Note that the reasoning leading to this conclusion is completely model-independent. It illustrates how the recursion principle RSP can be used to derive equations involving recursion variables.

Example 5.5.13 (Recursion principle RSP) Consider the two recursive specifications

$$E=\left\{\left.\begin{array}{l} X_0=a.X_1,\\ X_{n+1}=a.X_{n+2}+b.X_n\end{array}\right|\ n\in\mathbf{N}\right\}$$

and

$$F=\left\{\left.\begin{array}{l} Y_{i,0}=a.Y_{i,1},\\ Y_{i,j+1}=a.Y_{i,j+2}+b.Y_{i+1,j}\end{array}\right|\ i,j\in\mathbf{N}\right\}.$$

These recursive specifications are guarded. Hence, each of them has precisely one solution in the term model $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrows}$. Using the principle RSP, it can be shown that $(\mathrm{BSP}+E+F+\mathrm{RSP})(A)\vdash X_n=Y_{i,n}$ for any natural numbers i and n. This shows that the recursive specification F where variables have a double index can be simplified to the recursive specification E where variables have only one index.

One way to arrive at the desired result is to show that the equations of F are derivable from the equations of E after replacing the occurrences of the $Y_{i,j}$ by occurrences of X_j. Hence, it has to be shown that $(\mathrm{BSP}+E)(A)\vdash X_0=a.X_1$ and, for any $j\in\mathbf{N}$, $(\mathrm{BSP}+E)(A)\vdash X_{j+1}=a.X_{j+2}+b.X_j$. Obviously, this is the case. This result implies that any solution of X_j is also a solution of $Y_{i,j}$, for any $i\in\mathbf{N}$. Furthermore, both E and F have precisely one solution in any model of theory $(\mathrm{BSP}+E+F+\mathrm{RSP})(A)$, leading to the desired result that $(\mathrm{BSP}+E+F+\mathrm{RSP})(A)\vdash X_n=Y_{i,n}$ for any $i,n\in\mathbf{N}$.

A reasoning where the equations of E are shown to be derivable from the equations of F also leads to the desired result. In this approach, the occurrences of variables X_n in E can be replaced by occurrences of $Y_{0,n}$ (or any other $Y_{i,n}$). Hence, it has to be shown that $(\mathrm{BSP}+F)(A)\vdash Y_{0,0}=a.Y_{0,1}$

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

and, for any $n \in \mathbf{N}$, $(\mathrm{BSP} + F)(A) \vdash Y_{0,n+1} = a.Y_{0,n+2} + b.Y_{0,n}$. This is much more elaborate than the above proof as it requires additional proofs of $(\mathrm{BSP} + F)(A) \vdash Y_{i,j} = Y_{i+1,j}$ for any $i, j \in \mathbf{N}$.

The above examples show that the principle RSP and the notion of guardedness are meaningful and useful. Guardedness of a recursive specification over the signature of $\mathrm{BSP}(A)$ implies that it has a unique solution in the term model $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrow}$, and RSP allows equational reasoning about guarded recursive specifications. It turns out that guardedness is also exactly the notion needed to characterize recursive specifications with unique solutions when considering the theory $\mathrm{BSP}_{\mathrm{rec}}(A)$ and its term model $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrow}$. Every unguarded recursive specification over the signature of $\mathrm{BSP}(A)$ with a finite or infinite but countable number of equations has multiple solutions in $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrow}$ when A is not empty. The proof for finite unguarded recursive specifications is as follows.

Proposition 5.5.14 (Guardedness) Every finite unguarded recursive specification over the signature of theory $\mathrm{BSP}(A)$ has multiple solutions in the model $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrow}$ when A is not empty.

Proof Suppose E is an unguarded recursive specification over the signature of $\mathrm{BSP}(A)$ with a finite number of equations. As E is unguarded, then either there is a variable X in $V_R(E)$ that occurs unguarded in the equation of X or there is a number $n \geq 1$ and a sequence of variables $X_0, \ldots, X_n$ in $V_R(E)$ such that X_{i+1} occurs unguarded in the equation of X_i $(i < n)$ and X_0 occurs unguarded in the equation of X_n (i.e., there is a 'cycle of unguardedness'). Consider the second case. The first case can be treated similarly.
Let p be an arbitrary closed $\mathrm{BSP}(A)$-term. Consider the recursive specification E_p that is obtained from E by adding an extra summand p to each of the equations of $X_0, \ldots, X_n$, so, if $X_i = t_i$ is the equation in E, then $X_i = t_i + p$ becomes the equation in E_p.
Let $X_i = t_i + p$ and $X_{i+1} = t_{i+1} + p$ (for $i < n$) be equations in E_p. Recall that X_{i+1} occurs unguarded in t_i, which means that t_i is derivably equal to a term of the form $s + X_{i+1}$, for some term s. This implies that $(\mathrm{BSP} + E_p)(A) \vdash X_i = t_i + p = s + X_{i+1} + p = s + X_{i+1} + X_{i+1} + p = t_i + X_{i+1} + p$, i.e., X_i has an X_{i+1} summand. It follows that $(\mathrm{BSP} + E_p)(A) \vdash X_i = t_i + p = t_i + X_{i+1} + p = t_i + (t_{i+1} + p) + p = t_i + t_{i+1} + p = t_i + X_{i+1} = s + X_{i+1} + X_{i+1} = s + X_{i+1} = t_i$, i.e., $(\mathrm{BSP} + E_p)(A) \vdash X_i = t_i$. In a similar way, it is possible to show that $(\mathrm{BSP} + E_p)(A) \vdash X_n = t_n$. Thus, it follows that $(\mathrm{BSP} + E_p)(A) \vdash E$, and that any solution of E_p is also a solution of E. When A is not empty there are

infinitely many different instantiations that can be chosen for p (e.g., choosing $1, a.1, a.a.1, \ldots$ for some $a \in A$).

Next, consider the deduction system of Table 5.2. It implies that every state of the transition system defined by any recursion constant $\mu X.E$ is a subterm of one of the right-hand sides of the equations in E. As E has only finitely many equations, there are finitely many such subterms. The deduction rules imply that, as a consequence, the number of states of any transition system of a recursion variable of a finite recursive specification is finite. Note that the deduction system of Table 5.2 also shows that processes $\mu X_i.E_p$ $(i \leq n)$ among others allow the behavior of p. Only finitely many of these behaviors can already be present in each of the $\mu X_i.E$, as the corresponding transition systems have only finitely many states. Since there are infinitely many choices for p and any solution for $\mu X_i.E_p$ is also a solution for $\mu X_i.E$, infinitely many different solutions are obtained for each of the $\mu X_i.E$ and thus for E. □

The above property assumes that the action set is not empty. It is not difficult to give an unguarded recursive specification over the signature of BSP(A) with only one solution if the action set can be empty; see Exercise 5.5.3.

It seems that there are no general conditions independent of the basic process theory and the precise model that guarantee that guarded recursive specifications are always precisely the recursive specifications with one solution. In Chapter 10, the so-called inaccessible process is introduced. This process can be used to construct unguarded recursive specifications with only a single solution, even in the presence of actions (see Exercise 10.3.2). There are also models in which guarded recursive specifications have multiple solutions. In the initial algebra $\mathbb{I}(\mathrm{BSP}_{\mathrm{rec}}(A))$ of theory $\mathrm{BSP}_{\mathrm{rec}}(A)$, see Definition 2.3.19 (Initial algebra), in general many guarded recursive specifications have multiple solutions. Consider for example the two guarded recursive specifications $E_1 = \{X = a.X\}$ and $E_2 = \{Y = a.Y\}$, for some $a \in A$. Because theory $\mathrm{BSP}_{\mathrm{rec}}(A)$ – without assuming RSP – has no means to derive equalities between X and Y, the initial algebra $\mathbb{I}(\mathrm{BSP}_{\mathrm{rec}}(A))$ contains two equivalence classes $[\mu X.E_1]_{\vdash}$ and $[\mu Y.E_2]_{\vdash}$ that are both solutions of both X and Y. Furthermore, Example 4.5.2 in (Usenko, 2002) introduces an algebra that can be used to construct models for theories including sequential composition, as presented later in this book, in which every guarded recursive specification with at least one recursion variable in the right-hand side of one of its equations has infinitely many solutions. However, these models capture no longer the intuitive semantics of process theories and their operators. They exploit the large freedom in constructing models for equational theories, but have no meaning in process specification and modeling. In conclusion, the discussed examples

show that guardedness does not capture uniqueness of solutions of recursive specifications under all possible circumstances. However, all the examples are exceptional or counterintuitive cases. In practice, guardedness is an appropriate and commonly used way to capture the uniqueness of solutions of recursive specifications, in a model-independent way.

An interesting observation is that principle RDP is stronger than strictly necessary for the purpose of capturing all recursive specifications with precisely one solution: it suffices to assume that *guarded* recursive specifications have a solution. This weaker principle is called the *Restricted Recursive Definition Principle*.

Definition 5.5.15 (RDP$^-$) Assume some equational theory and some model of this theory. The *Restricted Recursive Definition Principle* (RDP$^-$) is the following assumption: every guarded recursive specification over the signature of the theory has a solution in the model of that theory.

Theorem 5.5.16 ((In-)validity of RDP$^-$**)** Assuming BSP(A) as the basic process theory, principle RDP$^-$ is valid in $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrow}$ but it is not valid in $\mathbb{P}(\mathrm{BSP}(A))_{/\leftrightarrow}$, i.e., $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrow} \models \mathrm{RDP}^-$ and $\mathbb{P}(\mathrm{BSP}(A))_{/\leftrightarrow} \not\models \mathrm{RDP}^-$.

Proof The validity of RDP$^-$ in term model $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrow}$ is a direct consequence of the facts that RDP is valid in $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrow}$ (Theorem 5.5.3) and that RDP implies RDP$^-$. The invalidity of RDP$^-$ in model $\mathbb{P}(\mathrm{BSP}(A))_{/\leftrightarrow}$ follows from the same example that proved the invalidity of the principle RDP for this term model (Example 5.3.5 (No solutions)). □

Theorems 5.5.2 (Invalidity of RDP), 5.5.3 (Validity of RDP), and 5.5.16 ((In-)validity of RDP$^-$) show that both RDP and RDP$^-$ are invalid in the term model $\mathbb{P}(\mathrm{BSP}(A))_{/\leftrightarrow}$ and valid in the term model $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrow}$. It is also possible to construct models for BSP(A) that satisfy RDP$^-$ but not RDP, for example the term model $\mathbb{P}(\mathrm{BSP}(A))_{/\leftrightarrow}$ extended with bisimilarity equivalence classes for all constants $\mu X.E$ for arbitrary recursion variables X and *guarded* recursive specifications E.

The validity of the recursion principles RSP and RDP$^-$ in the term model $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrow}$ reconfirms that every guarded recursive specification over the signature of BSP(A) has precisely one solution in $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrow}$.

Recall that the proof of Theorem 5.5.11, claiming the validity of principle RSP in the term models of BSP(A) and $\mathrm{BSP}_{\mathrm{rec}}(A)$, was in effect postponed. The given proof uses the validity of another recursion principle, namely the

Restricted Approximation Induction Principle (AIP$^-$). This recursion principle is based on the idea that a solution of a recursive specification, which is a potentially infinite process, can be approximated by its series of finite projections, using the projection operators introduced in Section 4.5. Each such a finite projection can be considered as an approximation of the solution itself. In computing science and mathematics in general, and in process algebra in particular, it is quite common to describe an infinite object in terms of a series of approximations. The essence of AIP$^-$ is that two processes – for example, but not necessarily, two solutions of some recursive specification(s) – can be considered equal if all their finite projections are equal. However, as the name already suggests, AIP$^-$ restricts this essential idea to some extent. It is in fact a restricted version of a more general recursion principle that precisely captures this idea, namely the *Approximation Induction Principle* (AIP). Note that the principles AIP and AIP$^-$ explicitly refer to projections of processes. Hence, it is required that the basic theory under consideration contains projection operators. Therefore, in the following, the basic theories considered are $(\mathrm{BSP}+\mathrm{PR})(A)$ and $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$ instead of $\mathrm{BSP}(A)$ and $\mathrm{BSP}_{\mathrm{rec}}(A)$.

Definition 5.5.17 (AIP) The *Approximation Induction Principle* (AIP) is the following assumption: for arbitrary $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$-terms s and t, if $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A) \vdash \pi_n(s) = \pi_n(t)$ for all natural numbers $n \in \mathbf{N}$, then $((\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}+\mathrm{AIP})(A) \vdash s = t$.

An equivalent formulation of AIP is obtained by also allowing the use of the principle AIP in the derivation of $\pi_n(s) = \pi_n(t)$, i.e., $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A) \vdash \pi_n(s) = \pi_n(t)$ can be replaced by $((\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}+\mathrm{AIP})(A) \vdash \pi_n(s) = \pi_n(t)$. Furthermore, note that the above definition is formulated in terms of the theory $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$. However, the principle can be defined for arbitrary process theories that contain projection operators, also theories without recursion. In the remainder, it is used without further notice for other theories as well.

Example 5.5.18 (AIP) Consider again recursive specifications $\{X_1 = a.X_1\}$ and $\{X_2 = a.a.X_2\}$. Using AIP, it can be shown that X_1 and X_2 denote the same process, i.e., it can be shown that $((\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}+\mathrm{AIP})(A) \vdash X_1 = X_2$.

Recall Notation 4.6.6 (n-fold action prefix). Using natural induction, it is not difficult to show that, for any natural number $n \in \mathbf{N}$,

$$(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A) \vdash \pi_n(X_1) = a^n 0$$

and

$$(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A) \vdash \pi_n(X_2) = a^n 0.$$

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

Thus, for any natural number $n \in \mathbf{N}$,

$$(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash \pi_n(X_1) = a^n 0 = \pi_n(X_2).$$

Using AIP gives

$$((\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}} + \mathrm{AIP})(A) \vdash X_1 = X_2.$$

Unfortunately, the principle AIP is not valid in the term model of theory $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A)$. It does hold in the term model of $(\mathrm{BSP} + \mathrm{PR})(A)$, but that model is not very interesting because not all recursive specifications have a solution in that model. The proof of the following theorem uses a new notation, $\sum$, which generalizes the choice operator $+$ to a choice between an arbitrary number of processes.

Notation 5.5.19 (Generalized choice) Let $I \subset \mathbf{N}$ be some finite index set of natural numbers, $j \in \mathbf{N} \setminus I$ a fresh index not in I, and t_i, for all $i \in I \cup \{j\}$, arbitrary terms in some process theory containing the choice operator $+$.

$$\sum_{i \in \emptyset} t_i \equiv 0 \text{ and } \sum_{i \in I \cup \{j\}} t_i \equiv t_j + \sum_{i \in I} t_i.$$

At this point, the above notation is simply an abbreviation. In Chapter 10, a true choice quantifier, i.e., the quantifier corresponding to the binary operator $+$ with 0 as its identity element, is added as an operator in process theories.

Theorem 5.5.20 ((In-)validity of AIP) Principle AIP is valid in $\mathbb{P}((\mathrm{BSP} + \mathrm{PR})(A))_{/\leftrightarrows}$ and it is not valid in $\mathbb{P}((\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A))_{/\leftrightarrows}$. That is, $\mathbb{P}((\mathrm{BSP} + \mathrm{PR})(A))_{/\leftrightarrows} \models \mathrm{AIP}$ and $\mathbb{P}((\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A))_{/\leftrightarrows} \not\models \mathrm{AIP}$.

Proof First, it is proven that recursion principle AIP holds in the term model $\mathbb{P}((\mathrm{BSP} + \mathrm{PR})(A))_{/\leftrightarrows}$.
Let p and q be closed $(\mathrm{BSP} + \mathrm{PR})(A)$-terms such that $\pi_n(p) \leftrightarrows \pi_n(q)$, for all natural numbers $n \in \mathbf{N}$. It has to be proven that $p \leftrightarrows q$. From Theorem 4.5.4 (Bounded depth), it follows that there exist natural numbers n_1 and n_2 such that, for all $k_1 \geq n_1$ and $k_2 \geq n_2$, $(\mathrm{BSP} + \mathrm{PR})(A) \vdash \pi_{k_1}(p) = p$ and $(\mathrm{BSP} + \mathrm{PR})(A) \vdash \pi_{k_2}(q) = q$. Hence, for all $k \geq \max(n_1, n_2)$, $\pi_k(p) \leftrightarrows p$ and $\pi_k(q) \leftrightarrows q$. Recall the assumption that $\pi_n(p) \leftrightarrows \pi_n(q)$ for all $n \in \mathbf{N}$. Taking an arbitrary $n' \geq \max(n_1, n_2)$, it follows that $p \leftrightarrows \pi_{n'}(p) \leftrightarrows \pi_{n'}(q) \leftrightarrows q$. Hence, the principle AIP is valid in the model $\mathbb{P}((\mathrm{BSP} + \mathrm{PR})(A))_{/\leftrightarrows}$.
Second, it is shown that principle AIP is not valid in the term model $\mathbb{P}((\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A))_{/\leftrightarrows}$.
Consider the recursive specifications $\{X_n = a^n 0 + X_{n+1} \mid n \in \mathbf{N}\}$ and $\{Y = a.Y\}$. As in Example 5.5.18 (AIP), it can be shown that $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash$

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

$\pi_n(Y) = a^n 0$ for all $n \in \mathbf{N}$. Furthermore, obviously $(\mathrm{BSP} + \mathrm{PR})(A) \vdash a^n 0 = \pi_n(a^n 0)$ for all $n \in \mathbf{N}$.

By induction, it can be shown that, for any natural number $n \in \mathbf{N}$,

$$(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash X_0 = \sum_{i=0}^{n} a^i 0 + X_{n+1}.$$

This results in

$$(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash X_0 = X_0 + a^n 0$$

as follows, using Axiom A3 in the second step:

$$\begin{aligned}(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash X_0 &= \sum_{i=0}^{n} a^i 0 + X_{n+1} \\ &= \sum_{i=0}^{n} a^i 0 + a^n 0 + X_{n+1} \\ &= X_0 + a^n 0.\end{aligned}$$

Consequently, for any natural number $n \in \mathbf{N}$,

$$\begin{aligned}(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash \pi_n(X_0 + Y) &= \pi_n(X_0) + \pi_n(Y) \\ &= \pi_n(X_0) + a^n 0 \\ &= \pi_n(X_0) + \pi_n(a^n 0) \\ &= \pi_n(X_0 + a^n 0) \\ &= \pi_n(X_0).\end{aligned}$$

Applying AIP then gives

$$((\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}} + \mathrm{AIP})(A) \vdash X_0 + Y = X_0.$$

Assume now that AIP is valid in model $\mathbb{P}((\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A))_{/\leftrightarrows}$. This yields that $X_0 + Y \leftrightarrows X_0$. The term $X_0 + Y$, however, exhibits an infinite path of a executions as follows:

$$X_0 + Y \xrightarrow{a} Y \xrightarrow{a} Y \xrightarrow{a} \cdots \xrightarrow{a} Y \xrightarrow{a} \cdots.$$

On the other hand, the process term X_0 only allows finite paths of a executions. Each such path is of the form

$$X_0 \xrightarrow{a} a^n 0 \xrightarrow{a} a^{n-1} 0 \xrightarrow{a} \cdots \xrightarrow{a} 0,$$

for some $n \geq 1$. However, this means that $X_0 + Y$ and X_0 cannot be bisimilar, i.e.,

$$X_0 + Y \not\leftrightarrows X_0.$$

When playing the bisimulation game as explained in Chapter 3, it is clear that the player playing with X_0 can at some point no longer match the moves of the infinite a sequence of process $X_0 + Y$, no matter what execution path of X_0 is chosen in the initial state. This implies a contradiction, and, as a conclusion, principle AIP is not valid in $\mathbb{P}((\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A))_{/\leftrightarrows}$. □

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

The problem with the Approximation Induction Principle is that it is only capable of discriminating processes that differ in a finite initial part of their behavior. The two process terms $X_0 + Y$ and X_0 in the above proof do not differ in any finite initial part of their behaviors. Hence, AIP is too strong: It is not valid in the standard term model of the basic process theory with recursion. Principle AIP$^-$ restricts the applicability of AIP by requiring that one of the process terms under consideration should be guarded, solving the observed problem. This restricted version turns out to be valid in the standard term model. It can also be used, as intended, to prove the validity of RSP in the standard term model.

Definition 5.5.21 (AIP$^-$) The *Restricted Approximation Induction Principle* (AIP$^-$) is the following assumption: for arbitrary $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A)$-terms s and t such that s is guarded, if $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash \pi_n(s) = \pi_n(t)$ for all natural numbers $n \in \mathbf{N}$, then $((\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}} + \mathrm{AIP}^-)(A) \vdash s = t$.

As for AIP, also in this case, the hypothesis $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash \pi_n(s) = \pi_n(t)$ can be replaced by $((\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}} + \mathrm{AIP}^-)(A) \vdash \pi_n(s) = \pi_n(t)$. Also AIP$^-$ can be defined straightforwardly in the context of other process theories.

Example 5.5.22 (AIP$^-$) Consider once more recursive specifications $\{X_1 = a.X_1\}$ and $\{X_2 = a.a.X_2\}$. As explained in Example 5.5.18 (AIP), $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash \pi_n(X_1) = a^n 0$ and $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash \pi_n(X_2) = a^n 0$, and hence $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash \pi_n(X_1) = \pi_n(X_2)$, for all natural numbers $n \in \mathbf{N}$. As X_1 and X_2 are guarded, applying AIP$^-$ yields $((\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}+\mathrm{AIP}^-)(A) \vdash X_1 = X_2$.

Theorem 5.5.23 (Validity of AIP$^-$**)** The principle AIP$^-$ is valid in both term model $\mathbb{P}((\mathrm{BSP}+\mathrm{PR})(A))_{/\leftrightarrows}$ and term model $\mathbb{P}((\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A))_{/\leftrightarrows}$. That is, $\mathbb{P}((\mathrm{BSP} + \mathrm{PR})(A))_{/\leftrightarrows} \models \mathrm{AIP}^-$ and $\mathbb{P}((\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A))_{/\leftrightarrows} \models \mathrm{AIP}^-$.

Proof The fact that $\mathbb{P}((\mathrm{BSP}+\mathrm{PR})(A))_{/\leftrightarrows} \models \mathrm{AIP}^-$ follows immediately from the observation that AIP implies AIP$^-$ and the fact that $\mathbb{P}((\mathrm{BSP} + \mathrm{PR})(A))_{/\leftrightarrows} \models \mathrm{AIP}$ (Theorem 5.5.20).
The second part of the theorem is proven as follows. Let $p, q \in \mathcal{C}((\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A))$ be such that p is guarded and $\pi_n(p) \leftrightarrows \pi_n(q)$ for all natural numbers $n \in \mathbf{N}$. It has to be proven that $p \leftrightarrows q$. By definition of guardedness there exists a completely guarded term $p' \in \mathcal{C}((\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A))$ such that $p \leftrightarrows p'$. It suffices to prove that $p' \leftrightarrows q$. Define the relation R on closed $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A)$-terms as follows:

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

$$R = \{(u, v) \in (\mathcal{C}((\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A)))^2 \mid \pi_n(u) \leftrightarrows \pi_n(v) \text{ for all } n \in \mathbf{N} \text{ and } u \text{ is completely guarded}\}.$$

Clearly, $(p', q) \in R$. It remains to prove that R is a bisimulation relation. Consider an arbitrary pair $(u, v) \in R$. For technical reasons the parts of the proof are presented in a different order than usual.

- Suppose that $v \xrightarrow{a} v'$ for some $a \in A$ and closed term $v' \in \mathcal{C}((\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A))$. It has to be proven that there exists a $u' \in \mathcal{C}((\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A))$ such that $u \xrightarrow{a} u'$ and $(u', v') \in R$.
  Define, for any natural number $m \in \mathbf{N}$,

  $$S_m = \{u^* \in \mathcal{C}((\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A)) \mid u \xrightarrow{a} u^* \text{ and } \pi_m(u^*) \leftrightarrows \pi_m(v')\}.$$

  The set S_m represents all processes that are reachable from u by performing a once and that are bisimilar to v' up to depth m.
  Then, the following observations can be made:

  (i) For any natural number $i \in \mathbf{N}$, $S_i \supseteq S_{i+1}$. This follows from the observation that, for any closed $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A)$-terms p and q and natural number $k \in \mathbf{N}$, $\pi_{k+1}(p) \leftrightarrows \pi_{k+1}(q)$ implies that $\pi_k(p) \leftrightarrows \pi_k(q)$ (see Exercise 5.5.8).

  (ii) For any natural number $i \in \mathbf{N}$, $S_i \neq \emptyset$. To see this, let $i \in \mathbf{N}$ be some natural number. From $v \xrightarrow{a} v'$ and the deduction rules for projection operators in Table 4.6 in Section 4.5, it follows that $\pi_{i+1}(v) \xrightarrow{a} \pi_i(v')$ for any $i \in \mathbf{N}$. From this, and the fact that $\pi_{i+1}(u) \leftrightarrows \pi_{i+1}(v)$ (assumption $(u, v) \in R$), it follows that there exists a closed $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A)$-term u' such that $\pi_{i+1}(u) \xrightarrow{a} u'$ and $u' \leftrightarrows \pi_i(v')$. Inspection of the deduction rules for projection operators reveals that necessarily $u' \equiv \pi_i(u'')$ for some closed $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$-term u'' such that $u \xrightarrow{a} u''$. Hence, $\pi_i(u'') \leftrightarrows \pi_i(v')$ for some u''. Then, by definition of S_i, $u'' \in S_i$. Hence $S_i \neq \emptyset$.

  (iii) For any natural number $i \in \mathbf{N}$, S_i is finite. This follows immediately from the fact that u is completely guarded and the property of the deduction rules that in one transition only a finite number of terms can be reached from a completely guarded term (see Exercise 5.5.9).

  From these observations, it can be concluded that $\bigcap_{i \in \mathbf{N}} S_i \neq \emptyset$. Take some $u' \in \bigcap_{i \in \mathbf{N}} S_i$. Observe that by the definition of the sets S_i necessarily $u \xrightarrow{a} u'$. As $u' \in S_i$ for all natural numbers $i \in \mathbf{N}$, it follows that $\pi_i(u') \leftrightarrows \pi_i(v')$ for all natural numbers $i \in \mathbf{N}$. It follows

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

from the definition of R that $(u', v') \in R$, which completes this case of the proof.

- Suppose that $u \xrightarrow{a} u'$ for some action $a \in A$ and closed term $u' \in \mathcal{C}((\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A))$. It has to be proven that there exists a $v' \in \mathcal{C}((\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A))$ such that $v \xrightarrow{a} v'$ and $(u', v') \in R$.
  Define, similar to the previous case, for any natural number $m \in \mathbf{N}$,

$$T_m = \{v^* \in \mathcal{C}((\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A)) \mid v \xrightarrow{a} v^* \text{ and } \pi_m(u') \leftrightarrows \pi_m(v^*)\}.$$

  Observe that the sequence $(T_i)_{i \in \mathbf{N}}$ is decreasing and that all T_i are non-empty. For each $i \in \mathbf{N}$, pick a term $v_i \in T_i$. Then, $v \xrightarrow{a} v_i$ by definition. By the previous item, there are u_i such that $u \xrightarrow{a} u_i$ and $(u_i, v_i) \in R$ for all $i \in \mathbf{N}$. As before, since u is completely guarded, it follows that only a finite number of terms can be reached from u. Therefore, the sequence $(u_i)_{i \in \mathbf{N}}$ must contain at least one element infinitely often. Let u^* be a term that occurs infinitely often in $(u_i)_{i \in \mathbf{N}}$ and k a natural number such that $u^* \equiv u_k$. It is claimed that $(u', v_k) \in R$, thus finishing the proof. To see that this claim indeed holds, one needs to prove that $\pi_n(u') \leftrightarrows \pi_n(v_k)$ for all $n \in \mathbf{N}$. Let n be an arbitrary natural number. Let l be a natural number such that $l \geq n$ and $u^* \equiv u_l$. Then $\pi_n(u') \leftrightarrows \pi_n(v_l)$ follows from the facts that $v_l \in T_l$ and $T_l \subseteq T_n$. Furthermore $(u^*, v_l) \in R$ and $(u^*, v_k) \in R$ are obtained from $(u_l, v_l) \in R$ and $u^* \equiv u_l$, and $(u_k, v_k) \in R$ and $u^* \equiv u_k$. Therefore, $\pi_n(u') \leftrightarrows \pi_n(v_l) \leftrightarrows \pi_n(u^*) \leftrightarrows \pi_n(v_k)$. Thus, $\pi_n(u') \leftrightarrows \pi_n(v_k)$ for all $n \in \mathbf{N}$ and thus $(u', v_k) \in R$.
- Suppose that $u\!\downarrow$. From the term deduction rules for projection operators, see Table 4.6, it follows that $\pi_n(u)\!\downarrow$ for all $n \in \mathbf{N}$. Since $\pi_n(u) \leftrightarrows \pi_n(v)$, it follows that $\pi_n(v)\!\downarrow$ for all $n \in \mathbf{N}$. Suppose that $v\!\not\downarrow$. Then, by the term deduction rules in Table 4.6, $\pi_n(v)\!\not\downarrow$ for all $n \in \mathbf{N}$. This results in a contradiction and therefore $v\!\downarrow$.
- Suppose that $v\!\downarrow$. Using a reasoning similar to the previous case, it can be shown that $u\!\downarrow$, completing the proof.

□

At this point, five recursion principles have been discussed: RDP, RSP, RDP$^-$, AIP, and AIP$^-$. It is interesting to consider the relations between these principles. Obviously, by their definitions, RDP implies RDP$^-$ and AIP implies AIP$^-$. The most interesting relation is the one already hinted at earlier, namely that validity of AIP$^-$ implies validity of RSP. The latter is only true under the assumption that the equational theory under consideration has the

so-called head-normal-form (HNF) property. This is usually the case. The relations between the recursion principles are visualized in Figure 5.5, where an arrow connecting two principles indicates a logical implication. To formally prove these relations, the HNF property needs to be introduced, and some intermediate results are needed.

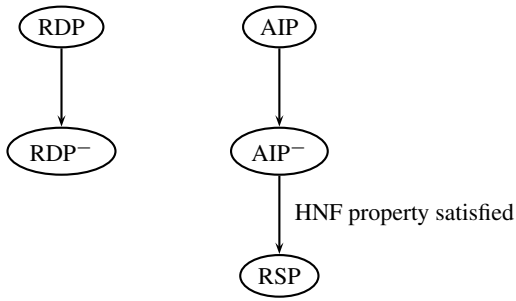

Fig. 5.5. Relation between the recursion principles.

Definition 5.5.24 (Head normal form) Consider an equational theory $T(A)$ that extends BSP(A), as defined in Definition 2.2.14. The set of *head normal forms* of theory $T(A)$ is inductively defined as follows. The constants 0 and 1 are head normal forms; for any action $a \in A$ and $T(A)$-term t, $a.t$ is a head normal form; for any head normal forms s and t, $s + t$ is a head normal form.

Theory $T(A)$ is said to satisfy the *head-normal-form (HNF) property* if and only if every *guarded* $T(A)$-term can be rewritten into a head normal form, i.e., for every guarded $T(A)$-term s, there is a head normal form t such that $T(A) \vdash s = t$.

Head normal forms can always be rewritten into a very specific form.

Proposition 5.5.25 (Head normal forms) Consider a theory $T(A)$ that extends BSP(A). For any head normal form t of $T(A)$, there is a natural number $n \in \mathbf{N}$ such that, for any $i < n$, there are $a_i \in A$ and $T(A)$-terms t_i such that

$$T(A) \vdash t = \sum_{i<n} a_i.t_i(+1),$$

where the $\sum$ notation is the generalized choice notation introduced in Notation 5.5.19 and the notation '$(+s)$' for some term s denotes an optional s-summand which may or may not be present.

Proof Straightforward via induction on the structure of head normal form t. □

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

Proposition 5.5.26 (HNF property) Process theory $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A)$ satisfies the HNF property.

Proof It needs to be proven that every guarded $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$-term s can be rewritten into a head normal form. By Definition 5.5.8 (Guardedness, part 2), it can be assumed without loss of generality that s is completely guarded. The proof goes via induction on the structure of s.

- $s \equiv 1$ or $s \equiv 0$. Term s is a head normal form by definition.
- $s \equiv a.s'$ for some $a \in A$ and $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A)$-term s'. Again, term s is a head normal form by definition.
- $s \equiv s' + s''$ for some completely guarded process terms s' and s''. By induction, there exist head normal forms t' and t'' such that
  $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash s' = t'$
  and
  $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash s'' = t''$.
  Hence, $t'+t''$ is a head normal form such that $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)\vdash s = s' + s'' = t' + t''$, which proves this case.
- $s \equiv \pi_n(s')$ for some natural number $n \in \mathbf{N}$ and some completely guarded process term s'. By induction, there is a head normal form t such that
  $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash s' = t$.
  By Proposition 5.5.25 (Head normal forms), there exists a natural number $m \in \mathbf{N}$ such that for any $i < m$, there are $a_i \in A$ and terms t_i such that
  $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash s' = \sum_{i<m} a_i.t_i(+1)$.
  If $m = 0$, then either $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash s = \pi_n(s') = 0$ or $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash s = \pi_n(s') = 1$. In both cases, s is therefore derivably equal to a head normal form. Assume that $m > 0$.
  $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash$
  $s = \pi_n(s') = \pi_n(\sum_{i<m} a_i.t_i(+1)) = \sum_{i<m} \pi_n(a_i.t_i)(+\pi_n(1))$.
  If $n = 0$, then
  $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash s = \sum_{i<m} \pi_n(a_i.t_i)(+\pi_n(1)) = 0(+1)$.
  If $n > 0$,
  $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash$
  $s = \sum_{i<m} \pi_n(a_i.t_i)(+\pi_n(1)) = \sum_{i<m} a_i.\pi_{n-1}(t_i)(+1)$.
  The last term in the last two derivations is in both cases a head normal form, which completes the proof for this case.

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

- $s \equiv x$ for some (normal or recursion) variable x. Then s is not completely guarded, which means this case cannot occur.

□

The next proposition states that any projection of any solution of a recursion variable specified by a guarded recursive specification can be rewritten into an equivalent closed $(\mathrm{BSP} + \mathrm{PR})(A)$-term, i.e., a term without any variables. The intuition behind this result is that any guarded recursive specification can be rewritten such that the recursion variables in the right-hand sides of the equation in the specifications occur in the scope of any desirable number of nested action-prefix operators, in particular a number larger than the depth of the projection. Recall Definition 2.2.6 (Substitution). Let, for any term t over some arbitrary signature and variable x, t/x denote the substitution that maps x to t and all other variables onto themselves.

Proposition 5.5.27 (Projections of guarded recursive specifications) Let E be a guarded recursive specification and X a recursion variable in $V_R(E)$. For any natural number n, there is a closed $(\mathrm{BSP} + \mathrm{PR})(A)$-term p, such that for any $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A)$-term t satisfying the recursive specification of X, i.e., $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash E[t/X]$, $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash \pi_n(t) = p$.

Proof Assume without loss of generality that recursive specification E is completely guarded.
Let $X = t_X$ be the recursive equation defining X in E. Since E is guarded, based on Propositions 5.5.26 (HNF property) and 5.5.25 (Head normal forms), it follows that

$$(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash t = t_X[t/X] = \sum_{i<m} a_i.t_i[t/X](+1),$$

for some natural number $m \in \mathbf{N}$ and actions a_i and terms t_i for any $i < m$. It follows immediately that

$$(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash \pi_n(t) = \pi_n(\sum_{i<m} a_i.t_i[t/X](+1)),$$

for any natural number $n \in \mathbf{N}$.
Therefore, in the remainder, a slightly more general property is proven, namely that, for any natural number n, and any head normal form $\sum_{i<l} b_i.s_i(+1)$ with $l \in \mathbf{N}$ and, for any $i < l$, actions b_i and $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A)$-terms s_i, such that the s_i contain only recursion variables in $V_R(E)$, there is a closed $(\mathrm{BSP} + \mathrm{PR})(A)$-term q, such that

$$(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash \pi_n(\sum_{i<l} b_i.s_i[t/X](+1)) = q.$$

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

This generalized property can be proven via induction on n.
First, suppose $n = 0$. Then,

$$(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash$$
$$\pi_0(\sum_{i<l} b_i.s_i[t/X](+1)) = \sum_{i<l} \pi_0(b_i.s_i[t/X])(+\pi_0(1)) = 0(+1),$$

which satisfies the base case.
Second, suppose $n > 0$. Then,

$$(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash$$
$$\pi_n(\sum_{i<l} b_i.s_i[t/X](+1)) = \sum_{i<l} b_i.\pi_{n-1}(s_i[t/X])(+1).$$

Recall that the s_i contain only recursion variables from $V_R(E)$ and no other variables, and that E is completely guarded. It follows that all the s_i are guarded because any unguarded variable occurrences in an s_i can be replaced by the right-hand side of the defining equation of that variable. Therefore, by Proposition 5.5.26 (HNF property), each s_i can be rewritten into a head normal form $\sum_{j<k} b'_j.s'_j(+1)$ with $k \in \mathbf{N}$ and, for any $j < k$, actions b'_j and $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A)$-terms s'_j, which contain no other variables than those from $V_R(E)$. Induction yields that for each s_i, there is a closed $(\mathrm{BSP}+\mathrm{PR})(A)$-term q_i, such that

$$(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash$$
$$\pi_{n-1}(s_i[t/X]) = \pi_{n-1}(\sum_{j<k} b'_j.s'_j[t/X](+1)) = q_i.$$

Substituting this in the earlier result yields,

$$(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash \pi_n(\sum_{i<l} b_i.s_i[t/X](+1)) = \sum_{i<l} b_i.q_i(+1),$$

which also completes this case. □

The following theorem is now straightforward to derive. It implies the derivable equality of all the finite projections of any two solutions of a guarded recursive specification.

Theorem 5.5.28 (Projection Theorem) Let E be a guarded recursive specification and X a recursion variable in $V_R(E)$. For any $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A)$-terms s and t satisfying the recursive specification of X and any natural number n, $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash \pi_n(s) = \pi_n(t)$.

Proof From Proposition 5.5.27 above, it follows immediately that for any natural number n, there is a closed $(\mathrm{BSP} + \mathrm{PR})(A)$-term p such that $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A) \vdash \pi_n(s) = p = \pi_n(t)$. □

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

Note that the Projection Theorem does not use any essential aspects of the theory $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$, other than the fact that it satisfies the HNF property. It can be generalized to arbitrary process theories extending $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$ that satisfy this property.

Theorem 5.5.29 (Relation between the recursion principles) Let $T(A)$ be some equational theory without recursion extending $(\mathrm{BSP}+\mathrm{PR})(A)$. Let $T_{\mathrm{rec}}(A)$ be the corresponding theory with recursion.

(i) RDP implies RDP^-;
(ii) AIP implies AIP^-;
(iii) if $T_{\mathrm{rec}}(A)$ satisfies the HNF property, then AIP^- implies RSP.

Proof The first two implications follow immediately from the definitions of the principles.
The third one is more involved. Let $\mathbb{M}$ with domain $\mathbf{M}$ be some model of process theory $T_{\mathrm{rec}}(A)$. Suppose that $\mathbb{M} \models \mathrm{AIP}^-$. It has to be proven that $\mathbb{M} \models \mathrm{RSP}$. Let E be an arbitrary guarded recursive specification and X a recursion variable in $V_R(E)$. Suppose that s and t are two $T_{\mathrm{rec}}(A)$-terms satisfying the recursive specification of X. Note that, since E is guarded, this implies that both s and t are guarded. By the generalization of Theorem 5.5.28 (Projection Theorem) to $T_{\mathrm{rec}}(A)$, which is allowed because $T_{\mathrm{rec}}(A)$ satisfies the HNF property, $T_{\mathrm{rec}}(A) \vdash \pi_n(s) = \pi_n(t)$, for all natural numbers $n \in \mathbf{N}$. Applying AIP^- then gives $T_{\mathrm{rec}}(A) \vdash s = t$. Suppose now that ι and κ are two solutions of E. Since s and t can among others be arbitrary variables, without loss of generality, assume that $\iota(X) = \iota_\alpha(s)$ and $\kappa(X) = \kappa_\alpha(t)$ for some variable valuation α. Then, $T_{\mathrm{rec}}(A) \vdash s = t$ implies that $\mathbb{M} \models s = t$ and hence that $\iota_\alpha(s) =_{\mathbf{M}} \kappa_\alpha(t)$. Since E, X, s and t were all chosen arbitrarily, a guarded recursive specification has therefore at most one solution, which means that $\mathbb{M} \models \mathrm{RSP}$. □

Note that Theorem 5.5.11 (Validity of RSP), given earlier in this section, can be derived in a straightforward way from this last result and Theorem 5.5.23 (Validity of AIP^-), as explained in the proof of Theorem 5.5.11.

At the end of this section, it is interesting to consider whether the recursion principles affect the equalities that can be derived between closed terms of the basic theory $(\mathrm{BSP}+\mathrm{PR})(A)$. It turns out that this is not the case.

Theorem 5.5.30 (Conservative ground-extension) Theory $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$ extended with any combination of the five recursion principles considered in this section is a conservative ground-extension of theory $(\mathrm{BSP}+\mathrm{PR})(A)$.

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

Proof Along the lines of the proof of Theorem 5.4.8 (Conservative ground-extension $\mathrm{BSP}_{\mathrm{rec}}(A)$), it can be proven that $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$ is a conservative ground-extension of $(\mathrm{BSP}+\mathrm{PR})(A)$. Since RDP does not affect derivability, the desired conservativity result follows immediately for any extension of $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$ with RDP. It follows from Theorem 4.5.4 (Bounded depth) that any two closed $(\mathrm{BSP}+\mathrm{PR})(A)$-terms that satisfy the condition of AIP are already derivably equal in the theory $(\mathrm{BSP}+\mathrm{PR})(A)$. Hence, also any extension of $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$ with principle AIP is a conservative ground-extension of $(\mathrm{BSP}+\mathrm{PR})(A)$. Theorem 5.5.29 (Relation between the recursion principles) completes the proof. □

Note that the extension of $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$ with any of the principles AIP, AIP^-, or RSP is not a conservative ground-extension of $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$. It is in fact the purpose of these recursion principles to derive equalities between terms with recursion variables that cannot be derived in the basic theory $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$. The extension of $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$ with RDP or RDP^- is identical to $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$, and so these extensions are trivially conservative.

Exercises

5.5.1 Determine whether in the following terms the occurrences of the recursion variable X are guarded, unguarded, or both: $a.X$, $Y+b.X$, $b.(X+Y)$, $a.Y+X$.

5.5.2 Determine whether the following recursive specifications are guarded or unguarded: $\{X=Y, Y=a.X\}$, $\{X=a.Y+Z, Y=b.Z+X, Z=c.X+Y\}$.

5.5.3 Consider $\mathrm{BSP}_{\mathrm{rec}}(A)$ with its model $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrows}$, and assume that $A=\emptyset$. Give an unguarded recursive specification with only one solution.

5.5.4 Consider the recursive specification $\{X=a.X+b.c.X\}$. Calculate $\pi_0(X)$, $\pi_1(X)$, and $\pi_2(X)$.

5.5.5 Consider the recursive specification $E=\{X=a.X\}$. Prove that $(\mathrm{BSP}+\mathrm{PR}+E)(A)\vdash \pi_n(X)=a^n0$ for any natural number $n\in\mathbf{N}$.

5.5.6 Consider the recursive specification $\{X=a.X+b.X\}$. Determine $\pi_n(X)$ for every natural number $n\in\mathbf{N}$.

5.5.7 Consider the recursive specifications $E_1=\{X=a.X+b.X\}$ and $E_2=\{Y=a.Y+b.Z, Z=a.Z+b.Y\}$. Prove that $(\mathrm{BSP}+E_1+E_2+\mathrm{RSP})(A)\vdash X=Y$.

5.5.8 Prove that for any closed $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$-terms p and q and natural number $n\in\mathbf{N}$, $\pi_{n+1}(p)\leftrightarrows\pi_{n+1}(q)$ implies that $\pi_n(p)\leftrightarrows\pi_n(q)$.

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

5.5.9 Consider the term deduction system underlying the standard term model of $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A)$. Given a guarded $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A)$-term t, prove that in one transition only a finite number of terms can be reached from t.
(Hint: use Proposition 5.5.26 (HNF property).)

5.6 Describing a stack

This section illustrates the use of recursion via an example of a specification of a data stack using the process theory $\mathrm{BSP}_{\mathrm{rec}}(A)$. The stack example is used at several points throughout the book to illustrate some of the concepts.

Consider an arbitrary, finite set of data elements $D = \{d_1, d_2, \cdots, d_n\}$, for some natural number $n \in \mathbf{N}$. Using recursion, a data stack with an unlimited capacity can be described in a straightforward way. Such a stack consists of a sequence of data elements from the set D. Elements from D can be added to or removed from the stack at only one end of the sequence. Usually, this end of the stack is called the top of the stack. The first element of the sequence modeling the stack is considered the top. The operation of adding an element d to the stack is modeled as the atomic action $push(d)$. Removing an element d from the stack is denoted $pop(d)$. In the recursive specification of the stack process, the following notations are used:

- The empty sequence of elements from D is denoted as ϵ.
- The set of all finite sequences of elements from D is denoted D^*.
- Concatenation of data elements and sequences into new sequences is denoted by juxtaposition. For example, for an element d from D and a sequence σ in D^*, the concatenation of d and σ is denoted $d\sigma$.
- The stack containing only one datum d, formally denoted by $d\epsilon$, is written as d.

A stack can be described in $\mathrm{BSP}_{\mathrm{rec}}(A)$, with an infinite set of recursive equations.

$$\begin{aligned} Stack1 &= S_\epsilon, \\ S_\epsilon &= 1 + \sum_{d \in D} push(d).S_d, \quad \text{and, for all } d \in D, \sigma \in D^*, \\ S_{d\sigma} &= pop(d).S_\sigma + \sum_{e \in D} push(e).S_{ed\sigma}. \end{aligned}$$

Note that the stack process can terminate if it is empty. This is particularly useful and meaningful if the stack is considered in the context of an environment. If the environment using the stack terminates after emptying the stack, then also the stack can terminate with it. In a theory with parallel composition, developed in Chapter 7, this can be made more precise; see Exercise 7.6.7.

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

In theory $\mathrm{BSP}_{\mathrm{rec}}(A)$, it is not possible to specify a stack with a finite number of recursive equations (see Exercise 5.8.1). In Chapter 6, a theory is introduced that does allow a finite recursive specification of a stack.

Observe that the above recursive specification is not completely guarded. It is guarded though, as it is derivably equal to the specification

$$\begin{aligned} Stack1 &= 1 + \sum_{d\in D} push(d).S_d, \\ S_\epsilon &= 1 + \sum_{d\in D} push(d).S_d, \quad \text{and, for all } d \in D, \sigma \in D^*, \\ S_{d\sigma} &= pop(d).S_\sigma + \sum_{e\in D} push(e).S_{ed\sigma}, \end{aligned}$$

which is completely guarded. Note that each right-hand side in this recursive specification consists of a sum of terms where each term is either an empty-process constant or a prefix operator applied to a recursion variable. Such a form is convenient, because guardedness immediately follows. This specific form of recursive specification is called a *linear* recursive specification, and it occurs often in equational verifications.

Figure 5.6 shows part of the stack's transition system, for the case that $D = \{0, 1\}$. For simplicity, $push(d)$ and $pop(d)$ are written $\overline{d}$ and $\underline{d}$, respectively.

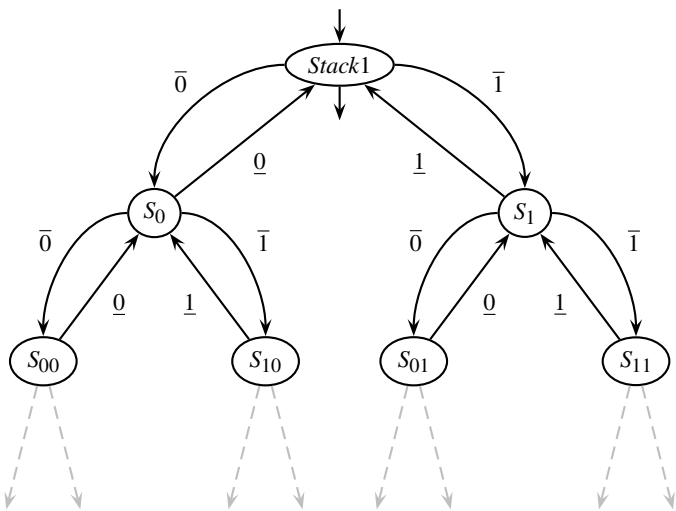

Fig. 5.6. Visualization of (part of) the stack with $D = \{0, 1\}$.

Exercises

5.6.1 Calculate, for the stack process described in this section, $\pi_0(Stack1)$, $\pi_1(Stack1)$, and $\pi_2(Stack1)$.

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

5.6.2 Give a recursive specification describing a stack with capacity C, i.e., a stack that contains at most C elements. You can assume that C is a given constant. Draw the transition system for the stack with capacity $C = 2$ and data elements from $D = \{0, 1\}$.

5.6.3 Consider the recursive specification $\{C_0 = 1 + plus.C_1, C_{n+1} = minus.C_n + plus.C_{n+2} \mid n \in \mathbf{N}\}$ that specifies a counter (that can terminate at the count of zero). Using RSP, prove that the counter (variable C_0) and the stack ($Stack1$) with $D = \{0\}$ are equal when assuming that $plus \equiv push(0)$ and $minus \equiv pop(0)$.

5.6.4 Prove that the counter of the previous exercise and the stack with $D = \{0\}$ are equal when assuming that $plus \equiv push(0)$ and $minus \equiv pop(0)$ but now using AIP^-.

5.7 Expressiveness and definability

At this point, it is interesting to consider the expressiveness of process theories in a bit more detail. In Chapter 4, expressiveness of a process theory was introduced as the class of processes that can be described by the closed terms over the signature of that process theory. It was shown for example that the addition of the 1 constant to the minimal theory of Section 4.2 increases expressiveness, but that the addition of projection operators in general does not increase expressiveness.

The extension of the basic process theory of the previous chapter with recursion in the current chapter was motivated from the expressiveness point of view. With the addition of recursion, however, and in particular given the special role of recursion variables, it is necessary to carefully consider the notion of expressiveness. Given that, in a process theory such as $\text{BSP}_{\text{rec}}(A)$, the recursion variables are simply constants in the signature, one could consider to define expressiveness of such a theory as the class of all processes in any model of the theory that are the interpretation of any closed term over the signature of that process theory. However, Example 5.3.7 (Multiple solutions) shows that this definition does not make much sense, because any process in any model can be the interpretation of constant $\mu X.\{X = X\}$. As an alternative, it is possible to consider the class of processes that can be specified via a closed term in a theory with recursion for a specific model with a specific interpretation. The standard term models and the operational framework of Chapter 3 are of particular interest in this context.

The transition-system framework of Chapter 3 and the standard term models of process theories essentially introduce equivalence classes of transition systems under bisimilarity as processes. Therefore, it is interesting to consider

the universe of all equivalence classes of transition systems as the semantic context for studying the expressiveness of process theories. Given the standard interpretation for term models of process theories, the question then is which of these equivalence classes has a transition system corresponding to a closed term over the signature of some given theory as a representative.

Only one assumption is made, namely that, given a transition system, its set of states can be determined, and that given a state, its set of transitions and the existence of a termination option can be determined. This is an assumption that is not always fulfilled. Consider for example the following definition of transition system p (using the inaction and empty-process constants 0 and 1):

$$p = \begin{cases} 0, \text{ if the decimal expansion of the number } \pi \text{ contains a} \\ \qquad \text{sequence of nine consecutive 9s;} \\ 1, \text{ otherwise.} \end{cases}$$

The definition of p implies that it has exactly one state, but it cannot be determined whether or not this state has a termination option. (It is not possible to define an algorithm that can conclusively decide for any arbitrary sequence of numbers whether or not the decimal expansion of π contains that sequence; the part of the decimal expansion of π known to date does not contain a sequence of nine consecutive 9s.) Transition system p is an example of a *non-computable* transition system. In the remainder, only *computable* transition systems are considered. The formal definition of a computable transition system is omitted. The reader interested in a formal definition of computability is referred to any textbook on automata theory, e.g., (Linz, 2001). Informally, the definition of a computable transition system is that the behavior of the transition system can be determined by an algorithm. This corresponds to the above assumption that the set of states of a transition system can be determined, and that for each state the set of transitions and the existence of a termination option can be determined. Observe that the notion of computability is not limited to transition systems, but carries over to other concepts such as for example recursive specifications and processes, where a recursive specification is computable if and only if its set of equations can be determined by an algorithm, and a process is computable if and only if it is an equivalence class under bisimilarity of computable transition systems.

One final important notion used in the remainder is that of a *countable* set. A set is countable if and only if it has the same number of elements as some subset of the natural numbers, i.e., if it can be indexed using the natural numbers. The definition given here also considers finite sets to be countable. A countable set is also said to have countably many elements.

Definition 5.7.1 ((Countable) computable process) A computable process is an equivalence class under bisimilarity of computable transition systems. A (computable) process is countable if and only if it has a countable transition system as a representative (where a transition system is countable if and only if it has countably many states and countably many transitions).

Note that there are computable processes that are not countable and vice versa. The above example of a non-computable transition system leads to a process that is not computable, but since it has only one state and no transitions, it is countable. Also observe that the definition allows that a countable process may have a representative with an uncountable number of states or transitions. However, any such a transition system is bisimilar to a countable one, which implies that the corresponding process essentially has countably many states and transitions.

Definition 5.7.2 (Expressiveness) A computable process can be specified in a process theory T if and only if it has a transition system corresponding to a closed T-term as defined by the standard term model, as a representative.

The following two results show that in a context with all (computable) recursive specifications, already the basic process theory is sufficient to specify all *countable* computable processes.

Theorem 5.7.3 (Countable computable transition systems) Let the set of recursive specifications contain all computable recursive specifications. A transition system p is an element of the term algebra $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))$ if and only if p is computable and countable.

Proof First, consider the implication from left to right. A general element p of $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))$ corresponds to an arbitrary closed $\mathrm{BSP}_{\mathrm{rec}}(A)$-term. Such a term is by definition finite and may contain a finite number of recursion constants. Since also the right-hand side of the equation defining any recursion constant $\mu X.E$ is a finite term, by following the deduction rules of the term deduction system of Table 5.2, every transition and every state of the transition system defined by p can be determined and counted. As a consequence, transition system p is computable and countable.
Second, consider the implication from right to left. Let p be a countable computable transition system. Since p is countable, it has countably many states and countably many transitions. Enumerate the states of p as s_i with index $i = 0, 1, \ldots$. Enumerate the transitions of each state s_i with index k, $k = 0, 1, \ldots$. Assume that transition k leaving state s_i has label a_{ik}, and that

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

it goes to state $s_{j(i,k)}$. Define a recursive specification E over variables X_{ik} as follows, where variable X_{i0} corresponds to state s_i. For each variable X_{ik}, E contains equation

$$X_{ik} = a_{ik}.X_{j(i,k)0} + X_{i(k+1)}.$$

In case either the set of states or the set of transitions of a state is finite, just put $X_{ik} = 0$ for all remaining variables. If a state s_i has a termination option, which can be determined because p is computable, then a 1 summand has to be added to the equation of X_{i0}.
It is easy to see that the rules of the deduction system of Table 5.2 yield the transition system p for the recursive specification E, which means p is an element of the term algebra $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))$. □

Corollary 5.7.4 (Expressiveness of $\mathrm{BSP}_{\mathrm{rec}}(A)$) Theory $\mathrm{BSP}_{\mathrm{rec}}(A)$ with all computable recursive specifications as the set of recursive specifications allows to specify precisely all countable computable processes, i.e., the term model $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrows}$ contains precisely all countable computable processes.

Note that this corollary implies that the algebra $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrows}$ can be considered as an algebra of all countable computable processes, when the set of recursive specifications is the set of all computable recursive specifications. Given this observation, in the remainder, it is always assumed that the set of recursive specifications of interest is the set of all computable recursive specifications, or, when stated explicitly, a subset of those.

An important observation is that the notion of expressiveness introduced by Definition 5.7.2 is model-dependent. It assumes the operational framework of Chapter 3. Furthermore, it assumes the standard interpretation of process terms in term models. This implies, among others, that all recursion variables are interpreted as one specific process, which is not really in line with the interpretation of recursion variables as constrained variables. Recall that in Section 5.5, guardedness was introduced as a model-independent way to characterize uniqueness of solutions for recursive specifications, in line with the interpretation of recursion variables as constrained variables. A guarded recursive specification is supposed to *define* a process. Also observe that any closed term in a process theory without recursion can be turned into an equivalent (guarded) recursive specification in the theory extended with recursion by taking that closed term as the right-hand side of an equation in a recursive specification with only that one equation (see, for example, recursive specification E_1 of Example 5.2.2 (Recursive specifications)). These considerations suggest another notion of expressiveness, called *definability*, which is not depending on any particular model, and is consistent with the interpretation of recursion

variables as constrained variables. This latter notion of expressiveness originates from ACP-style process algebra, whereas the notion of expressiveness defined in Definition 5.7.2 finds its origins in CCS-style process theory.

Definition 5.7.5 (Definability) Let T be a process theory without recursion. A process in some model of theory T is *definable* (over T) if and only if it is the unique solution of (some designated recursion variable of) a *guarded* recursive specification over the signature of theory T. A process is *finitely definable* if and only if it is the unique solution of a *finite guarded* recursive specification over the signature of T, i.e., a guarded recursive specification with a finite number of equations.

In the remainder, for simplicity, the recursion variable of the considered recursive specification for which a process is a solution is sometimes left implicit.

Finite definability is introduced because it is of particular interest whether or not a process can be defined via a finite guarded recursive specification. In the previous section, it was for example claimed that the stack process is not finitely definable over $\mathrm{BSP}(A)$. The next section returns to the notion of finite definability in some more detail.

Definition 5.7.5 (Definability) raises the question whether, when returning to the algebra of countable computable processes $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrows}$, all these processes are definable over $\mathrm{BSP}(A)$. It turns out that this is not the case. Observe that the proof of Theorem 5.7.3 (Countable computable transition systems) uses unguarded recursion. It can be shown that this use of unguarded recursion is essential.

Recall Definition 3.1.16 (Finitely branching transition system).

Definition 5.7.6 (Finitely branching process) A finitely branching process is an equivalence class of computable transition systems under bisimilarity containing at least one finitely branching transition system as a representative.

Note that this definition allows that infinitely branching transition systems are elements of the equivalence class defining a finitely branching process. Since any such infinitely branching transition system is bisimilar to a finitely branching one, the corresponding process essentially has finitely many outgoing transitions in each state. Also observe that a finitely branching process is necessarily countable. This follows from the fact that a transition system is defined as a reachable subspace of a transition-system space (see Definition 3.1.5). With finitely many outgoing transitions per state, only a countable number of states can be reached.

Theorem 5.7.7 (Processes definable over BSP(A)) A computable process is definable over BSP(A) if and only if it is finitely branching.

Proof First, consider the implication from left to right. Assume a process p is the unique solution of a guarded recursive specification E over the signature of BSP(A). By Exercise 5.5.9, the state corresponding to any $\mu X.E$ for some recursion variable X of E has only finitely many outgoing transitions. As a consequence, the transition system corresponding to any $\mu X.E$ is finitely branching, which means that process p is finitely branching.

Second, consider the implication from right to left. If a process is finitely branching, then it follows that the process has a finitely branching transition system s as a representative, which has countably many states with finitely many outgoing transitions per state. It is straightforward to define a guarded recursive specification corresponding to transition system s. Enumerate the states of s as s_i with index $i = 0, 1, \ldots$. Enumerate the transitions of each state s_i with index k, $k = 0, 1, \ldots, n_i - 1$, where n_i is the number of outgoing transitions of state s_i. Assume that transition k leaving state s_i has label a_{ik}, and that it goes to state $s_{j(i,k)}$. Define a linear recursive specification E over variables X_i as follows, where variable X_i corresponds to state s_i. For each variable X_i, E contains equation

$$X_i = \sum_{k=0}^{n_i-1} a_{ik}.X_{j(i,k)},$$

where the $\sum$ notation is the generalized choice notation of Notation 5.5.19. If s_i has a termination option, then a 1 summand is added to the equation.

It is straightforward to see that the deduction system of Table 5.2 yields the transition system s for the recursive specification E. Since E is guarded and defined over the signature of theory BSP(A), and since the term model $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrows}$ satisfies RSP, the process represented by s is a unique solution for E and therefore definable over BSP(A). □

As a final note, observe, in the context of the algebra of countable computable processes $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrows}$, the subtle difference between the phrases that 'a process can be expressed' or 'specified' and 'that a process can be defined'. The former means that there is a closed $\mathrm{BSP}_{\mathrm{rec}}(A)$-term with that process as its standard interpretation. The latter means that there is a guarded recursive specification over BSP(A) with that process as a unique solution, where this solution of course is the standard interpretation of this recursive specification.

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

Exercises

5.7.1 Consider the following recursive specifications. Recall the n-fold action-prefix notation of Notation 4.6.6.

(a) $E = \{X_n = X_{n+1} + a^n Y \mid n \in \mathbf{N}\} \cup \{Y = 1\}$;
(b) $F = \{X_n = X_{n+1} + a^n Y \mid n \in \mathbf{N}\} \cup \{Y = a.Y\}$.

These recursive specifications specify processes $\mu X_0.E$ and $\mu X_0.F$ in the standard term model of $\mathrm{BSP}_{\mathrm{rec}}(A)$ (which means that both these processes are expressible in theory $\mathrm{BSP}_{\mathrm{rec}}(A)$). Which of these processes are also finitely definable over theory $\mathrm{BSP}(A)$? Either give a guarded finite recursive specification to show that a process is definable, or give an (informal) argument why such a specification does not exist.

5.8 Regular processes

The previous section has introduced the notion of finite definability. A process is finitely definable over some given theory if and only if it is the unique solution of a finite guarded recursive specification over the signature of that theory. In Chapter 3, the notion of a regular transition system has been introduced, see Definition 3.1.15, which captures precisely all finite transition systems. In light of Theorem 5.7.7 (Processes definable over $\mathrm{BSP}(A)$), note that the regular transition systems form a subset of the finitely branching transition systems, and that finite definability is a restriction of definability. These observations suggest that it is interesting to have a closer look at the relation between regular transition systems and finite definability. It turns out that the regular transition systems can be captured precisely by all finite guarded recursive specifications.

Definition 5.8.1 (Regular process) A regular process is an equivalence class of computable transition systems under bisimilarity containing at least one regular transition system as a representative.

The definition allows non-regular transition systems as representatives of regular processes, which is in line with the earlier definitions of countable and finitely branching processes. Since any such non-regular transition system is bisimilar to a regular one, the corresponding process has finitely many states and transitions, which is the essence of regularity.

Theorem 5.8.2 (Regular processes) A computable process is regular if and only if it is finitely definable over $\mathrm{BSP}(A)$.

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

Proof Consider the implication from left to right. A regular process has a regular transition system s as a representative. Enumerate the states of s as s_i with index $i = 0, 1, \ldots, n-1$, where n is the number of states of s. Enumerate the transitions of each state s_i with index k, $k = 0, 1, \ldots, n_i - 1$, where n_i is the number of outgoing transitions of state s_i. Assume that transition k leaving state s_i has label a_{ik} and that it goes to state $s_{j(i,k)}$. Let recursive specification E over variables X_i, where variable X_i corresponds to state s_i, contain for each variable X_i, equation

$$X_i = \sum_{k=0}^{n_i-1} a_{ik}.X_{j(i,k)},$$

with an additional 1 summand if s_i has a termination option. The deduction system of Table 5.2 yields transition system s for recursive specification E. Since E is guarded, finite, and defined over the signature of BSP(A), and since $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrow}$ satisfies RSP, the process represented by s is a unique solution for E and therefore finitely definable over BSP(A).

Consider the implication from right to left. Assume a process p is the unique solution of a finite guarded recursive specification E over the signature of BSP(A). It follows from Theorem 5.7.7 (Processes definable over BSP(A)) that p is finitely branching. The deduction system of Table 5.2 implies that every state of the transition system defined by E is a subterm of one of the right-hand sides of the equations in E. As E has only finitely many equations, there are finitely many such subterms, and hence the transition system has finitely many states. As a consequence, the transition system is regular, which means that process p is regular. □

Consider again the stack process of Section 5.6. Since the given recursive specification is guarded, the stack is definable over BSP(A). However, it is not finitely definable, as already claimed in Section 5.6 and shown in Exercise 5.8.1. This implies that a stack is not a regular process.

Given the importance of regular processes, let process theory $\mathrm{BSP}_{\mathrm{gfrec}}(A)$, term algebra $\mathbb{P}(\mathrm{BSP}_{\mathrm{gfrec}}(A))$, and process algebra $\mathbb{P}(\mathrm{BSP}_{\mathrm{gfrec}}(A))_{/\leftrightarrow}$ be the theory, term algebra, and term model obtained by extending the basic theory BSP(A) with constants only for guarded finite recursive specifications. Theory $\mathrm{BSP}_{\mathrm{gfrec}}(A)$ and its model $\mathbb{P}(\mathrm{BSP}_{\mathrm{gfrec}}(A))_{/\leftrightarrow}$ can be considered as an equational theory and an algebra of regular processes. Note that the proof of Theorem 5.8.2 implies that all the processes in the algebra of regular processes $\mathbb{P}(\mathrm{BSP}_{\mathrm{gfrec}}(A))_{/\leftrightarrow}$ contain in fact only regular transition systems as representatives, leading to the following corollary.

Corollary 5.8.3 (Regular transition systems) A transition system p is an element of the term algebra $\mathbb{P}(\mathrm{BSP}_{\mathrm{gfrec}}(A))$ if and only if p is regular.

This corollary confirms the observation made in Example 4.3.2 (Transition systems for closed MPT(A)-terms) that all transition systems corresponding to closed MPT(A)-terms are regular, because all closed MPT(A)-terms are closed $\mathrm{BSP}_{\mathrm{gfrec}}(A)$-terms. Note that the algebra of countable computable processes $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrows}$, obviously, contains all regular processes, but that in $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))_{/\leftrightarrows}$, different from $\mathbb{P}(\mathrm{BSP}_{\mathrm{gfrec}}(A))_{/\leftrightarrows}$, regular processes may contain non-regular transition systems as representatives.

Since regularity and (finite) definability are so closely related, it is interesting to have a look at the validity of the recursion principles of Section 5.5 in the algebra of regular processes $\mathbb{P}(\mathrm{BSP}_{\mathrm{gfrec}}(A))_{/\leftrightarrows}$. Since principles AIP and AIP$^-$ are only meaningful in a context with projection operators, consider theory $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{gfrec}}(A)$, basic process theory with projection and guarded finite recursion, and its term model $\mathbb{P}((\mathrm{BSP}+\mathrm{PR})_{\mathrm{gfrec}}(A))_{/\leftrightarrows}$. The proof of Theorem 5.8.2 (Regular processes) carries over to this setting, showing that $\mathbb{P}((\mathrm{BSP}+\mathrm{PR})_{\mathrm{gfrec}}(A))_{/\leftrightarrows}$ is isomorphic to the algebra of regular processes $\mathbb{P}(\mathrm{BSP}_{\mathrm{gfrec}}(A))_{/\leftrightarrows}$. Obviously, $\mathbb{P}((\mathrm{BSP}+\mathrm{PR})_{\mathrm{gfrec}}(A))_{/\leftrightarrows}$ is a model of BSP(A). It does not satisfy RDP and RDP$^-$, however, because the guarded recursive specification given for the stack in Section 5.6 does not have a solution in $\mathbb{P}((\mathrm{BSP}+\mathrm{PR})_{\mathrm{gfrec}}(A))_{/\leftrightarrows}$ (see Exercise 5.8.1). In fact, the observation that the algebra of regular processes does not satisfy principles RDP$^-$ and RDP corresponds to the observation that regular processes form a strict subclass of the definable and countable computable processes, respectively. The algebra of regular processes does satisfy AIP, AIP$^-$, and RSP. Since any regular process in $\mathbb{P}((\mathrm{BSP}+\mathrm{PR})_{\mathrm{gfrec}}(A))_{/\leftrightarrows}$ can be specified via a (finite) *guarded* recursive specification, Theorem 5.5.23, that shows the validity of recursion principle AIP$^-$ in the term model $\mathbb{P}((\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A))_{/\leftrightarrows}$, shows that the more general principle AIP, formulated for all $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{gfrec}}(A)$-terms, is valid in the algebra $\mathbb{P}((\mathrm{BSP}+\mathrm{PR})_{\mathrm{gfrec}}(A))_{/\leftrightarrows}$. Theorem 5.5.29 (Relation between the recursion principles) then implies that algebra $\mathbb{P}((\mathrm{BSP}+\mathrm{PR})_{\mathrm{gfrec}}(A))_{/\leftrightarrows}$ also satisfies AIP$^-$ and RSP.

Theorem 5.8.4 (Regular processes and the recursion principles) The algebra of regular processes $\mathbb{P}((\mathrm{BSP}+\mathrm{PR})_{\mathrm{gfrec}}(A))_{/\leftrightarrows}$ does not satisfy recursion principles RDP and RDP$^-$ but it does satisfy AIP, AIP$^-$, and RSP.

For readers familiar with automata theory (see e.g. (Linz, 2001)), the results of this section can be rephrased as follows. Finite non-deterministic automata

under bisimulation equivalence correspond to right-linear grammars. This confirms that it is meaningful to speak of regular processes. Note that left-linear grammars cannot be formulated in the present setting. When such a notion can be formulated (in Chapter 6, Section 6.6), it turns out that it can be used to specify also non-regular processes. Also, in Section 6.5, a notion of regular expressions can be formulated. But then it turns out that not all regular processes are equivalent to a regular expression under bisimulation equivalence. The conclusion is that some results of automata theory remain valid in the current algebraic setting, when language equivalence is replaced by bisimilarity, but most do not.

It is also possible to consider equivalence classes of closed $\mathrm{BSP}_{\mathrm{gfrec}}(A)$-terms under language equivalence as a model of equational theory $\mathrm{BSP}_{\mathrm{gfrec}}(A)$. As expected, this yields the usual algebra of finite automata with language equivalence. In Section 6.5, the relation between regular processes as defined in this section and finite automata is considered in a bit more detail.

Exercises

5.8.1 Prove that the stack process of Section 5.6 is not finitely definable over BSP(A), which shows that the stack process is not regular.

5.9 Recursion and BSP$^*(A)$

In the introduction to this chapter, it was already indicated that prefix iteration only allows for the description of a limited class of unbounded processes. To solve this expressiveness problem, recursion has been introduced. In a setting with recursive specifications, prefix-iteration operators are redundant in the sense that every process described by a closed term from the process theory BSP$^*(A)$ can easily be described by recursion in the process theory $\mathrm{BSP}_{\mathrm{rec}}(A)$. This is achieved by replacing all occurrences of a^*p, for some action $a \in A$ and closed BSP$^*(A)$-term p, by a recursion variable X with recursive equation $X = a.X + p$, resulting in a recursive specification.

Example 5.9.1 (Prefix iteration and recursion) The closed BSP$^*(A)$-term $a.(b^*(c.0) + 1)$ defines the same process as the closed $\mathrm{BSP}_{\mathrm{rec}}(A)$-term $a.(X + 1)$ where X is defined by the recursive specification $\{X = b.X + c.0\}$.

The closed BSP$^*(A)$-term $a^*(b.1 + c^*0)$ can be represented by the recursion variable X where $X = a.X + b.1 + c^*0$. In turn, c^*0 can be replaced by Y with $Y = c.Y + 0$. As a result, $a^*(b.1 + c^*0)$ defines the same process as X with $\{X = a.X + b.1 + Y, Y = c.Y\}$.

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

Although the expressiveness of the process theory $\mathrm{BSP}^*(A)$ is less than the expressiveness of the process theory $\mathrm{BSP}_{\mathrm{rec}}(A)$, the latter is not a (conservative) extension of the former. The reason for this is the fact that the signature of $\mathrm{BSP}^*(A)$ is not contained in the signature of $\mathrm{BSP}_{\mathrm{rec}}(A)$. It is interesting to observe that in fact $\mathrm{BSP}^*(A)$ is not more expressive than the theory of regular processes $\mathrm{BSP}_{\mathrm{gfrec}}(A)$ of the previous section. Using Axiom PI1 in Table 4.7, every occurrence of a^*p in some closed $\mathrm{BSP}^*(A)$-term can first be rewritten into $a.(a^*p) + p$, after which the action-prefix occurrence can be replaced by a recursion variable as sketched in the above example. Since any closed $\mathrm{BSP}^*(A)$-term can only contain a finite number of prefix-iteration operators, this conversion results in a finite guarded recursive specification. Thus, theory $\mathrm{BSP}^*(A)$ allows only specifications of regular processes, which confirms the observation made in Example 4.6.1 (Transition systems of $\mathrm{BSP}^*(A)$-terms). It is in fact easy to see that $\mathrm{BSP}^*(A)$ is strictly less expressive than $\mathrm{BSP}_{\mathrm{gfrec}}(A)$; the process in Figure 5.1 is regular but cannot be described in $\mathrm{BSP}^*(A)$.

If for whatever reason it is needed to have both prefix iteration and recursion as means to describing unbounded processes, the addition of recursion to the process theory $\mathrm{BSP}^*(A)$ goes along the same lines as for the process theory $\mathrm{BSP}(A)$, resulting in theory $\mathrm{BSP}^*_{\mathrm{rec}}(A)$. The term model $\mathbb{P}(\mathrm{BSP}^*(A))_{/\leftrightarrows}$ is not a model of theory $\mathrm{BSP}^*_{\mathrm{rec}}(A)$ if the set of recursive specifications is sufficiently rich specifying processes that cannot be described by prefix iteration. The extended term model $\mathbb{P}(\mathrm{BSP}^*_{\mathrm{rec}}(A))_{/\leftrightarrows}$ does result in a model of $\mathrm{BSP}^*_{\mathrm{rec}}(A)$. Furthermore, theory $\mathrm{BSP}^*_{\mathrm{rec}}(A)$ is a conservative ground-extension of theory $\mathrm{BSP}^*(A)$. The notion of guardedness does not change: the occurrence of recursion variable X in a term a^*X cannot be considered guarded because of the possibility that X starts executing immediately. In the extended term model $\mathbb{P}(\mathrm{BSP}^*_{\mathrm{rec}}(A))_{/\leftrightarrows}$, the principles RSP, RDP, and RDP^- are valid. Principle AIP^- is valid if also projection is included.

Example 5.9.2 (Prefix iteration and recursion) Consider again the closed $\mathrm{BSP}^*(A)$-term $a.(b^*(c.0) + 1)$ and the recursive specification $E_1 = \{X = b.X + c.0\}$. It can be shown that

$$(\mathrm{BSP}^* + E_1 + \mathrm{RSP})(A) \vdash a.(b^*(c.0) + 1) = a.(X + 1),$$

which confirms the claim made in Example 5.9.1 that the two terms in the equality define the same process. To show this equality, it suffices to show that $(\mathrm{BSP}^* + E_1 + \mathrm{RSP})(A) \vdash X = b^*(c.0)$. It follows from Axiom PI1 in Table 4.7 that $\mathrm{BSP}^*(A) \vdash b^*(c.0) = b.(b^*(c.0)) + c.0$. This implies that $b^*(c.0)$ is a solution of X. Assuming RSP implies that (the process specified by) $b^*(c.0)$ is the only solution, showing that $(\mathrm{BSP}^* + E_1 + \mathrm{RSP})(A) \vdash X = b^*(c.0)$.

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

Consider now closed $\mathrm{BSP}^*(A)$-term $a^*(b.1 + c^*0)$ and recursive specification $E_2 = \{X = a.X + b.1 + Y, Y = c.Y\}$. It is not difficult to show that

$$(\mathrm{BSP}^* + E_2 + \mathrm{RSP})(A) \vdash a^*(b.1 + c^*0) = X,$$

which confirms also the second claim in Example 5.9.1.

Exercises

5.9.1 Show that $\mathbb{P}(\mathrm{BSP}^*_{\mathrm{rec}}(A))_{/\leftrightarrows} \models \mu X.\{X = a.X\} = a^*0$.

5.9.2 Show that $(\mathrm{BSP}^* + E_2 + \mathrm{RSP})(A) \vdash a^*(b.1 + c^*0) = X$, with E_2 as specified in Example 5.9.2.

5.9.3 Find two different solutions of the recursive equation $X = a^*X$ in the term model $\mathbb{P}(\mathrm{BSP}^*_{\mathrm{rec}}(A))_{/\leftrightarrows}$.

5.9.4 Consider the proper-iteration operators of Exercise 4.6.6. Give an argument that recursion variable X in term $a^{\oplus}X$ can be considered guarded. Show that recursive equation $X = a^{\oplus}X$ has a unique solution in the term model of theory $\mathrm{BSP}(A)$ with proper iteration and recursion.

5.10 The projective limit model

An important motivation for equational reasoning is that it is model-independent. That is, any derivation in an equational theory yields an equality that is valid in any model of that theory. Recursion principles have been introduced for similar reasons. They allow us to reason about certain meaningful classes of models in the context of recursion, without limiting ourselves to specific models.

So far, term models have been the most important models that were considered, but also the initial algebra (see Definition 2.3.19) has been mentioned. Considering process theory $(\mathrm{BSP} + \mathrm{PR})(A)$ as the basic theory, an interesting observation is that none of the term models $\mathbb{P}((\mathrm{BSP} + \mathrm{PR})(A))_{/\leftrightarrows}$, $\mathbb{P}((\mathrm{BSP} + \mathrm{PR})_{\mathrm{gfrec}}(A))_{/\leftrightarrows}$ and $\mathbb{P}((\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A))_{/\leftrightarrows}$, and none of the initial algebras of the various considered equational theories $\mathbb{I}((\mathrm{BSP} + \mathrm{PR})(A))$, $\mathbb{I}((\mathrm{BSP} + \mathrm{PR})_{\mathrm{gfrec}}(A))$, and $\mathbb{I}((\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A))$ validates both RDP and AIP, the two strongest recursion principles (see Theorem 5.5.29).

Table 5.3 gives an overview of the validity of the recursion principles in the various models. The results for the three term models have been proven in Sections 5.5 and 5.8. Note that the inclusion of the projection operators in the theories does not affect the results derived for principles RDP and RDP^-. The fact that the initial algebras $\mathbb{I}((\mathrm{BSP}+\mathrm{PR})(A))$ and $\mathbb{I}((\mathrm{BSP}+\mathrm{PR})_{\mathrm{gfrec}}(A))$

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

do not satisfy RDP$^-$ and RDP follows from the same examples used to show the invalidity of these principles in the corresponding term models. The fact that $\mathbb{I}((\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A))$ satisfies RDP and RDP$^-$ follows from the simple observation that theory $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$ contains a constant for each recursion variable in each recursive specification. Initial algebra $\mathbb{I}((\mathrm{BSP}+\mathrm{PR})(A))$ satisfies AIP (and therefore AIP$^-$ and RSP) because the model only contains bounded-depth processes, whereas $\mathbb{I}((\mathrm{BSP}+\mathrm{PR})_{\mathrm{gfrec}}(A))$ and $\mathbb{I}((\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A))$ do not satisfy RSP, AIP$^-$, or AIP simply because without recursion principles the equational theories $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{gfrec}}(A)$ and $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$ do not have any generally applicable means to derive equalities between recursion variables specified by different recursive specifications. As already explained on Page 132, this implies that it is easy to construct guarded recursive specifications with multiple solutions. Note that AIP and AIP$^-$ are valid in the models $\mathbb{I}((\mathrm{BSP}+\mathrm{PR})(A))$ and $\mathbb{P}((\mathrm{BSP}+\mathrm{PR})(A))_{/\leftrightarrows}$ for arbitrary $(\mathrm{BSP}+\mathrm{PR})(A)$-terms, i.e., terms of the basic theory without recursion. The results in Table 5.3 for the term models and initial algebras of $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{gfrec}}(A)$ and $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$ refer to AIP and AIP$^-$ formulated for arbitrary closed $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{gfrec}}(A)$- and $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$-terms, respectively, i.e., for terms including recursion variables. An interesting observation is the fact that the initial algebra $\mathbb{I}((\mathrm{BSP}+\mathrm{PR})_{\mathrm{gfrec}}(A))$ does not satisfy any of the five recursion principles.

	RDP$^-$	RDP	RSP	AIP$^-$	AIP
$\mathbb{I}((\mathrm{BSP}+\mathrm{PR})(A))$	no	no	yes	yes	yes
$\mathbb{P}((\mathrm{BSP}+\mathrm{PR})(A))_{/\leftrightarrows}$	no	no	yes	yes	yes
$\mathbb{I}((\mathrm{BSP}+\mathrm{PR})_{\mathrm{gfrec}}(A))$	no	no	no	no	no
$\mathbb{P}((\mathrm{BSP}+\mathrm{PR})_{\mathrm{gfrec}}(A))_{/\leftrightarrows}$	no	no	yes	yes	yes
$\mathbb{I}((\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A))$	yes	yes	no	no	no
$\mathbb{P}((\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A))_{/\leftrightarrows}$	yes	yes	yes	yes	no
$\mathbb{I}(((\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}+\mathrm{AIP})(A))$	yes	yes	yes	yes	yes
$\mathbb{I}^{\infty}((\mathrm{BSP}+\mathrm{PR})(A))$	yes	yes	yes	yes	yes

Table 5.3. Validity of recursion principles in models for $(\mathrm{BSP}+\mathrm{PR})(A)$.

So, is it possible to create a model that satisfies all five recursion principles introduced in Section 5.5? It is in fact relatively straightforward to do so. The

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

initial algebra $\mathbb{I}(((\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}+\mathrm{AIP})(A))$ satisfies RDP and AIP by definition, and hence by Theorem 5.5.29 (Relation between the recursion principles) also the other principles. However, it could be that this model is a one-point model, which contains only one process and in which everything is equal. It has already been mentioned before that these models are not very interesting.

This section introduces another model that satisfies all five recursion principles, namely the so-called projective limit model $\mathbb{I}^\infty((\mathrm{BSP}+\mathrm{PR})(A))$. This model is not trivial in the sense that it contains many processes, and it provides a meaningful and intuitive interpretation of process specifications. Since in the initial algebra $\mathbb{I}(((\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}+\mathrm{AIP})(A))$ the valid equalities are precisely those that are derivably equal, any equality valid in this initial algebra must be valid in any other model of $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$ that satisfies AIP as well. In particular, if $\mathbb{I}(((\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}+\mathrm{AIP})(A))$ is a one-point model, any other model of $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$ that satisfies AIP must be a one-point model. The existence of the projective limit model therefore implies that also the initial algebra $\mathbb{I}(((\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}+\mathrm{AIP})(A))$ is not a trivial model. Note that for both $\mathbb{I}(((\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}+\mathrm{AIP})(A))$ and $\mathbb{I}^\infty((\mathrm{BSP}+\mathrm{PR})(A))$, AIP and AIP^- are valid for arbitrary $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$-terms, i.e., for terms including recursion variables of arbitrary recursive specifications.

Besides the fact that the projective limit model is interesting from the point of view of recursion, it is also an example of a model for process theories that is constructed in an entirely different way than the term models and the initial algebras seen so far. The basic idea behind the projective limit model is that behavior is something that evolves over time. As there is no explicit notion of time in the process theories and algebras considered so far, the abstract notion of counting actions that have been executed is used to represent evolution. For this purpose, the projection operators are very useful. In the projective limit model, a process is an infinite sequence of finite projections. Two processes are considered equal if and only if their sequences of projections are the same.

Definition 5.10.1 (Projective limit model) Recall Definition 2.3.19 (Initial algebra). Consider the initial algebra $\mathbb{I}((\mathrm{BSP}+\mathrm{PR})(A))$ of the process theory $(\mathrm{BSP}+\mathrm{PR})(A)$. Let $\mathbf{I}$ denote the universe of the equivalence classes of closed $(\mathrm{BSP}+\mathrm{PR})(A)$-terms under derivability, and let $+_\vdash$, $(\pi_{n\vdash})_{n\in\mathbf{N}}$, $(a._\vdash)_{a\in A}$, $0_\vdash$, and $1_\vdash$ denote the operators and constants of the algebra. An infinite sequence $(p_0, p_1, p_2, \ldots)$ of elements of $\mathbf{I}$ is called a *projective sequence* if and only if it satisfies the following: $\pi_{n\vdash}(p_{n+1}) =_{\mathbf{I}} p_n$, for all natural numbers $n \in \mathbf{N}$.

Define $\mathbb{I}^\infty((\mathrm{BSP}+\mathrm{PR})(A))$ as the algebra with as its universe $\mathbf{I}^\infty$ the set of projective sequences of elements of $\mathbf{I}$. By definition, $(p_0, p_1, p_2, \ldots) =_{\mathbf{I}^\infty} (q_0, q_1, q_2, \ldots)$ if and only if $p_n =_{\mathbf{I}} q_n$ for all $n \in \mathbf{N}$.

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

The operators $+^\infty$, π_n^∞ (for each $n \in \mathbf{N}$), and $a.^\infty$ (for each $a \in A$) are defined as follows:

$$(p_0, p_1, \ldots) +^\infty (q_0, q_1, \ldots) = (p_0 +_\vdash q_0, p_1 +_\vdash q_1, \ldots),$$
$$\pi_n^\infty(p_0, p_1, \ldots) = (\pi_{n\vdash}(p_0), \pi_{n\vdash}(p_1), \ldots), \text{ and}$$
$$a.^\infty(p_0, p_1, \ldots) = (0_\vdash, a._\vdash p_0, a._\vdash p_1, \ldots).$$

The proof that these defining equations are sound, i.e., that the right-hand sides are projective sequences, is left as Exercise 5.10.1.

Finally, the constants 0^∞ and 1^∞ are defined as the projective sequences $(0_\vdash, 0_\vdash, 0_\vdash, \ldots)$ and $(1_\vdash, 1_\vdash, 1_\vdash, \ldots)$, respectively.

Algebra $\mathbb{I}^\infty((\mathrm{BSP}+\mathrm{PR})(A)) = (\mathbf{I}^\infty, +^\infty, (\pi_n^\infty)_{n\in\mathbf{N}}, (a.^\infty)_{a\in A}, 0^\infty, 1^\infty)$ is called the *projective limit model* of theory $(\mathrm{BSP}+\mathrm{PR})(A)$.

The above definition defines the projective limit model for the basic theory $(\mathrm{BSP}+\mathrm{PR})(A)$, but it can be defined in a similar way for any other theory with projection operators. It turns out that the projective limit model of $(\mathrm{BSP}+\mathrm{PR})(A)$ is in fact a model of $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$ that satisfies all five recursion principles.

As a first step, it is shown that $\mathbb{I}^\infty((\mathrm{BSP}+\mathrm{PR})(A))$ is a model of $(\mathrm{BSP}+\mathrm{PR})(A)$. The expected interpretation of the signature of $(\mathrm{BSP}+\mathrm{PR})(A)$ into the functions and constants of the algebra $\mathbb{I}^\infty((\mathrm{BSP}+\mathrm{PR})(A))$ is used, namely the interpretation that maps every operator or constant symbol f in the signature of $(\mathrm{BSP}+\mathrm{PR})(A)$ to the function or constant f^∞ of $\mathbb{I}^\infty((\mathrm{BSP}+\mathrm{PR})(A))$. Let ι denote this interpretation.

The following proposition shows that the projective sequence of a closed $(\mathrm{BSP}+\mathrm{PR})(A)$-term can be calculated by deriving all its projections. Each such a projection is the representative of one of the equivalence classes in the projective sequence.

Proposition 5.10.2 (Projective sequences) Let p be a closed $(\mathrm{BSP}+\mathrm{PR})(A)$-term. Projective sequence $\iota(p)$ is equal to $([\pi_0(p)]_\vdash, [\pi_1(p)]_\vdash, [\pi_2(p)]_\vdash, \ldots)$.

Proof The proof is straightforward via induction on the structure of closed $(\mathrm{BSP}+\mathrm{PR})(A)$-term p.

- $p \equiv 1$. It follows from Axiom PR1 in Table 4.5 that $(\mathrm{BSP}+\mathrm{PR})(A) \vdash \pi_n(1) = 1$ for all $n \in \mathbf{N}$. Hence, it is straightforward to show the desired result: $\iota(1) = 1^\infty = (1_\vdash, 1_\vdash, 1_\vdash, \ldots) =_{\mathbf{I}^\infty} ([\pi_0(1)]_\vdash, [\pi_1(1)]_\vdash, [\pi_2(1)]_\vdash, \ldots)$.
- $p \equiv 0$. It follows from Axiom PR2 that $(\mathrm{BSP}+\mathrm{PR})(A) \vdash \pi_n(0) = 0$ for all $n \in \mathbf{N}$, which makes the proof straightforward also in this case.

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

- $p \equiv a.q$ for some $a \in A$ and closed $(\mathrm{BSP} + \mathrm{PR})(A)$-term q. It follows that $(\mathrm{BSP} + \mathrm{PR})(A) \vdash \pi_0(p) = 0$ and $\pi_{n+1}(p) = a.\pi_n(q)$, for all $n \in \mathbf{N}$. It also follows that $\iota(p) = a.^\infty\iota(q)$. By induction, $\iota(q) = ([\pi_0(q)]_\vdash, [\pi_1(q)]_\vdash, [\pi_2(q)]_\vdash, \ldots)$. Hence, $a.^\infty\iota(q) =_{\mathbf{I}^\infty} ([0]_\vdash, [a.\pi_0(q)]_\vdash, [a.\pi_1(q)]_\vdash, \ldots)$, which proves this case.
- $p \equiv q + r$ for some closed $(\mathrm{BSP} + \mathrm{PR})(A)$-terms q and r. $(\mathrm{BSP} + \mathrm{PR})(A) \vdash \pi_n(p) = \pi_n(q) + \pi_n(r)$, for all $n \in \mathbf{N}$. By induction, $\iota(q) = ([\pi_0(q)]_\vdash, [\pi_1(q)]_\vdash, [\pi_2(q)]_\vdash, \ldots)$ and $\iota(r) = ([\pi_0(r)]_\vdash, [\pi_1(r)]_\vdash, [\pi_2(r)]_\vdash, \ldots)$. It follows that $\iota(p) = \iota(q) +^\infty \iota(r) =_{\mathbf{I}^\infty} ([\pi_0(q) + \pi_0(r)]_\vdash, [\pi_1(q) + \pi_1(r)]_\vdash, [\pi_2(q) + \pi_2(r)]_\vdash, \ldots) =_{\mathbf{I}^\infty} ([\pi_0(q+r)]_\vdash, [\pi_1(q + r)]_\vdash, [\pi_2(q + r)]_\vdash, \ldots)$.
- $p \equiv \pi_n(q)$ for natural number $n \in \mathbf{N}$ and closed $(\mathrm{BSP} + \mathrm{PR})(A)$-term q. It follows from Exercise 4.5.6 that $(\mathrm{BSP} + \mathrm{PR})(A) \vdash \pi_k(p) = \pi_k(q)$, for all natural numbers $k < n$, and $(\mathrm{BSP} + \mathrm{PR})(A) \vdash \pi_l(p) = \pi_n(q)$, for all natural numbers $l \geq n$. By induction, $\iota(q) = ([\pi_0(q)]_\vdash, [\pi_1(q)]_\vdash, [\pi_2(q)]_\vdash, \ldots)$. Thus, again using the result of Exercise 4.5.6, it follows that $\iota(p) =_{\mathbf{I}^\infty} ([\pi_n(\pi_0(q))]_\vdash, [\pi_n(\pi_1(q))]_\vdash, \ldots, [\pi_n(\pi_n(q))]_\vdash, [\pi_n(\pi_{n+1}(q))]_\vdash, \ldots) =_{\mathbf{I}^\infty} ([\pi_0(q)]_\vdash, [\pi_1(q)]_\vdash, \ldots, [\pi_n(q)]_\vdash, [\pi_n(q)]_\vdash, \ldots)$, which completes also this last case.

□

Example 5.10.3 (Projective sequences) The projective sequence associated to the $(\mathrm{BSP} + \mathrm{PR})(A)$-term $p \equiv a.(b.0 + c.a.1)$ can be obtained in two ways, namely directly via the definition of the projective limit model and the accompanying interpretation ι or via Proposition 5.10.2.

(i) Interpretation of the operators and constants:

$$\begin{aligned}
& \iota(a.(b.0 + c.a.1)) \\
= \;& a.^\infty(b.^\infty 0^\infty +^\infty c.^\infty a.^\infty 1^\infty) \\
=_{\mathbf{I}^\infty} \;& a.^\infty(b.^\infty(0_\vdash, 0_\vdash, \ldots) +^\infty c.^\infty a.^\infty(1_\vdash, 1_\vdash, \ldots)) \\
=_{\mathbf{I}^\infty} \;& a.^\infty((0_\vdash, b.\vdash 0_\vdash, b.\vdash 0_\vdash, \ldots) +^\infty \\
& \qquad c.^\infty(0_\vdash, a.\vdash 1_\vdash, a.\vdash 1_\vdash, \ldots)) \\
=_{\mathbf{I}^\infty} \;& a.^\infty((0_\vdash, b.\vdash 0_\vdash, b.\vdash 0_\vdash, \ldots) +^\infty \\
& \qquad (0_\vdash, c.\vdash 0_\vdash, c.\vdash a.\vdash 1_\vdash, c.\vdash a.\vdash 1_\vdash, \ldots)) \\
=_{\mathbf{I}^\infty} \;& a.^\infty(0_\vdash +_\vdash 0_\vdash, b.\vdash 0_\vdash +_\vdash c.\vdash 0_\vdash, b.\vdash 0_\vdash +_\vdash c.\vdash a.\vdash 1_\vdash, \\
& \qquad b.\vdash 0_\vdash +_\vdash c.\vdash a.\vdash 1_\vdash, \ldots) \\
=_{\mathbf{I}^\infty} \;& a.^\infty([0]_\vdash, [b.0 + c.0]_\vdash, [b.0 + c.a.1]_\vdash, [b.0 + c.a.1]_\vdash, \ldots) \\
=_{\mathbf{I}^\infty} \;& ([0]_\vdash, a.\vdash [0]_\vdash, a.\vdash [b.0 + c.0]_\vdash, a.\vdash [b.0 + c.a.1]_\vdash, \\
& \qquad a.\vdash [b.0 + c.a.1]_\vdash, \ldots) \\
=_{\mathbf{I}^\infty} \;& ([0]_\vdash, [a.0]_\vdash, [a.(b.0 + c.0)]_\vdash, [a.(b.0 + c.a.1)]_\vdash, \\
& \qquad [a.(b.0 + c.a.1)]_\vdash, \ldots).
\end{aligned}$$

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

(ii) Via projections: it is straightforward to see that
$(\mathrm{BSP}+\mathrm{PR})(A)\vdash$
$\pi_0(p)=0,$
$\pi_1(p)=a.0,$
$\pi_2(p)=a.\pi_1(b.0+c.a.1)=a.(\pi_1(b.0)+\pi_1(c.a.1))$
$=a.(b.0+c.0),$
$\pi_n(p)=a.(b.0+c.a.1)=p$ (for $n\geq 3$).
Hence, the projective sequence $\iota(p)$ equals
$([0]_{\vdash},[a.0]_{\vdash},[a.(b.0+c.0)]_{\vdash},[a.(b.0+c.a.1)]_{\vdash},$
$[a.(b.0+c.a.1)]_{\vdash},\ldots),$
which is the same result as derived in the previous case.

The example shows that the projective sequence corresponding to a bounded-depth closed term (i.e., any closed $(\mathrm{BSP}+\mathrm{PR})(A)$-term, see Theorem 4.5.4 (Bounded depth)) 'converges' in the sense that the projections at some point in the sequence do not change anymore.

Theorem 5.10.4 (Soundness of $(\mathrm{BSP}+\mathrm{PR})(A)$ **for** $\mathbb{I}^{\infty}((\mathrm{BSP}+\mathrm{PR})(A))$**)** Theory $(\mathrm{BSP}+\mathrm{PR})(A)$ is a sound axiomatization of the projective limit model $\mathbb{I}^{\infty}((\mathrm{BSP}+\mathrm{PR})(A))$, i.e., $\mathbb{I}^{\infty}((\mathrm{BSP}+\mathrm{PR})(A))\models(\mathrm{BSP}+\mathrm{PR})(A)$.

Proof It must be shown that $\mathbb{I}^{\infty}((\mathrm{BSP}+\mathrm{PR})(A))$ validates all axioms of $(\mathrm{BSP}+\mathrm{PR})(A)$.
Consider Axiom A1, $x+y=y+x$. Let p_n,q_n, for all $n\in\mathbf{N}$, be closed $(\mathrm{BSP}+\mathrm{PR})(A)$-terms such that $s_1=([p_0]_{\vdash},[p_1]_{\vdash},[p_2]_{\vdash},\ldots)$ and $s_2=([q_0]_{\vdash},[q_1]_{\vdash},[q_2]_{\vdash},\ldots)$ are projective sequences. It needs to be shown that $s_1+^{\infty}s_2=_{\mathbf{I}^{\infty}}s_2+^{\infty}s_1$. From the definitions of a projective sequence and an initial algebra, it follows that

$$\begin{aligned}
& s_1+^{\infty}s_2\\
= \;& ([p_0]_{\vdash},[p_1]_{\vdash},[p_2]_{\vdash},\ldots)+^{\infty}([q_0]_{\vdash},[q_1]_{\vdash},[q_2]_{\vdash},\ldots)\\
=_{\mathbf{I}^{\infty}} \;& ([p_0]_{\vdash}+_{\vdash}[q_0]_{\vdash},[p_1]_{\vdash}+_{\vdash}[q_1]_{\vdash},[p_2]_{\vdash}+_{\vdash}[q_2]_{\vdash},\ldots)\\
=_{\mathbf{I}^{\infty}} \;& ([p_0+q_0]_{\vdash},[p_1+q_1]_{\vdash},[p_2+q_2]_{\vdash},\ldots).
\end{aligned}$$

Similarly, $s_2+^{\infty}s_1=_{\mathbf{I}^{\infty}}([q_0+p_0]_{\vdash},[q_1+p_1]_{\vdash},[q_2+p_2]_{\vdash},\ldots)$. Obviously, $(\mathrm{BSP}+\mathrm{PR})(A)\vdash p+q=q+p$ for any closed $(\mathrm{BSP}+\mathrm{PR})(A)$-terms p and q. This implies that for all $n\in\mathbf{N}$, $(\mathrm{BSP}+\mathrm{PR})(A)\vdash p_n+q_n=q_n+p_n$, and thus that $[p_n+q_n]_{\vdash}=_{\mathbf{I}}[q_n+p_n]_{\vdash}$. Hence, $s_1+^{\infty}s_2=_{\mathbf{I}^{\infty}}s_1+^{\infty}s_2$, completing the proof for Axiom A1.
The other axioms are left as an exercise. □

The next step is to show that the projective limit model of $(\mathrm{BSP}+\mathrm{PR})(A)$ satisfies recursion principle RDP. For this purpose, it needs to be shown that every recursive specification has a solution in this model. Proposition 5.5.27

(Projections of guarded recursive specifications) implies that the projections of recursion variables of guarded recursive specifications correspond to closed $(\mathrm{BSP} + \mathrm{PR})(A)$ terms. Thus, a recursion variable of such a specification can be interpreted as the projective sequence of these projections.

Example 5.10.5 (Interpretation of recursion constants) Consider the recursion constant X defined by the guarded recursive specification $E = \{X = a.X\}$. Then, $(\mathrm{BSP} + \mathrm{PR} + E)(A) \vdash \pi_0(X) = 0$, $\pi_1(X) = a.0$, and $\pi_2(X) = a.\pi_1(X) = a.(a.0)$. In general, $(\mathrm{BSP} + \mathrm{PR} + E)(A) \vdash \pi_n(X) = a^n 0$, where a^n is the n-fold action prefix of Notation 4.6.6.

Let $s = ([0]_{\vdash}, [a.0]_{\vdash}, [a.a.0]_{\vdash}, \ldots, [a^n 0]_{\vdash}, [a^{n+1}0]_{\vdash}, \ldots)$ be the projective sequence associated with X. To show that s is indeed a solution of X, it needs to be shown that $s =_{\mathbf{I}^\infty} a.^\infty s$. This follows immediately from the definition of $a.^\infty$, see Definition 5.10.1 (Projective limit model).

The construction of solutions for *unguarded* recursive specifications is more involved. The details are beyond the scope of this book. The interested reader is referred to (Bergstra & Klop, 1982). Given solutions for arbitrary recursive specifications, the following theorem follows.

Theorem 5.10.6 (Validity of RDP**)** Assuming theory $(\mathrm{BSP} + \mathrm{PR})(A)$ as the basic process theory, recursion principle RDP is valid in $\mathbb{I}^\infty((\mathrm{BSP} + \mathrm{PR})(A))$, i.e., $\mathbb{I}^\infty((\mathrm{BSP} + \mathrm{PR})(A)) \models \mathrm{RDP}$.

Proof See (Bergstra & Klop, 1982), where the result is proven for a slightly different theory. □

Theorem 5.5.4 ($\mathrm{BSP}_{\mathrm{rec}}(A)$ and RDP), which generalizes to process theory $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A)$, yields immediately the following corollary.

Corollary 5.10.7 (Soundness $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A)$ **for** $\mathbb{I}^\infty((\mathrm{BSP} + \mathrm{PR})(A))$**)** Theory $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A)$ is a sound axiomatization of the projective limit model $\mathbb{I}^\infty((\mathrm{BSP} + \mathrm{PR})(A))$, i.e., $\mathbb{I}^\infty((\mathrm{BSP} + \mathrm{PR})(A)) \models (\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A)$.

The final step is to prove the validity of AIP.

Theorem 5.10.8 (Validity of AIP**)** Recursion principle AIP, as formulated in Definition 5.5.17 for all $(\mathrm{BSP}+\mathrm{PR})_{\mathrm{rec}}(A)$-terms, is valid in the projective limit model of $(\mathrm{BSP} + \mathrm{PR})(A)$, i.e., $\mathbb{I}^\infty((\mathrm{BSP} + \mathrm{PR})(A)) \models \mathrm{AIP}$.

Proof Let s_1 and s_2 be projective sequences such that, for all $n \in \mathbf{N}$, $\pi_n^\infty(s_1) =_{\mathbf{I}^\infty} \pi_n^\infty(s_2)$. It needs to be shown that $s_1 =_{\mathbf{I}^\infty} s_2$.

https://doi.org/10.1017/CBO9781139195003.009 Published online by Cambridge University Press

Let p_n, q_n, for all $n \in \mathbf{N}$, be equivalence classes of closed $(\mathrm{BSP} + \mathrm{PR})(A)$-terms such that $s_1 = (p_0, p_1, p_2, \ldots)$ and $s_2 = (q_0, q_1, q_2, \ldots)$. It follows from the definition of the π_n^∞ operators that $\pi_n^\infty(s_1) =_{\mathbf{I}^\infty} (\pi_{n\vdash}(p_0), \pi_{n\vdash}(p_1), \pi_{n\vdash}(p_2), \ldots)$ and $\pi_n^\infty(s_2) =_{\mathbf{I}^\infty} (\pi_{n\vdash}(q_0), \pi_{n\vdash}(q_1), \pi_{n\vdash}(q_2), \ldots)$. It follows from the definition of projective sequences that $\pi_{n\vdash}(p_{n+1}) =_{\mathbf{I}} p_n$ and that $\pi_{n\vdash}(q_{n+1}) =_{\mathbf{I}} q_n$. The assumption that $\pi_n^\infty(s_1) =_{\mathbf{I}^\infty} \pi_n^\infty(s_2)$ implies that $p_n =_{\mathbf{I}} \pi_{n\vdash}(p_{n+1}) =_{\mathbf{I}} \pi_{n\vdash}(q_{n+1}) =_{\mathbf{I}} q_n$. Hence, for all $n \in \mathbf{N}$, $p_n =_{\mathbf{I}} q_n$, which means that $s_1 =_{\mathbf{I}^\infty} s_2$. □

Since $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A)$ satisfies the HNF property (Proposition 5.5.26), Theorem 5.5.29 (Relation between the recursion principles) yields the following corollary.

Corollary 5.10.9 (Recursion principles and the projective limit model) Projective limit model $\mathbb{I}^\infty((\mathrm{BSP}+\mathrm{PR})(A))$ satisfies recursion principles RDP, RDP$^-$, AIP, AIP$^-$, and RSP.

At the end of this section, it is interesting to once more consider the nature of the projective limit model. Many facts about processes in this book (for instance the notion of a regular process in Section 5.8) are stated in terms of transition systems, and so do not directly apply to other models such as the projective limit model considered here. However, any model can be turned into a transition-system space. The following definition illustrates this transformation for the projective limit model. The transition-system space can be turned into a model for theory $(\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}}(A)$, that is isomorphic to the projective limit model, and therefore satisfies all five recursion principles. Note however that the notion of equivalence on this transition-system space induced by the identity in the projective limit model is not bisimilarity, but simply identity, because there is a one-to-one correspondence between projective sequences and transition systems.

Definition 5.10.10 (Transition-system space of the projective limit model) The projective limit model $\mathbb{I}^\infty((\mathrm{BSP} + \mathrm{PR})(A))$ with set of projective sequences $\mathbf{I}^\infty$ induces the transition-system space $(\mathbf{I}^\infty, A, \rightarrow, \downarrow)$ with, for any projective sequences $s, t \in \mathbf{I}^\infty$, $s \xrightarrow{a} t$ if and only if $s =_{\mathbf{I}^\infty} a.^\infty t +^\infty s$ and $s\downarrow$ if and only if $s =_{\mathbf{I}^\infty} 1^\infty +^\infty s$.

This definition can be used to define a notion of regular processes in the context of the projective limit model. Recall Definitions 3.1.15 (Regular transition system) and 5.8.1 (Regular process). The first definition is still valid in the current setting. The second definition defines a regular process in the

context of the standard operational framework with bisimulation equivalence as an equivalence class of transition systems under bisimilarity that contains at least one regular transition system as a representative. Adapting this definition to the current context with identity as the equivalence yields the following straightforward definition of a regular projective sequence.

Definition 5.10.11 (Regular projective sequence) A projective sequence is regular if and only if its corresponding transition system is regular.

Exercise 5.10.6 investigates the possibility to give a direct definition of regularity of projective sequences, not using a transformation to a transition-system space.

Exercises

5.10.1 Prove that for an arbitrary projective sequence s from the projective limit model $\mathbb{I}^\infty((\mathrm{BSP}+\mathrm{PR})(A))$, the sequences $a.^\infty s$ (for any $a \in A$) and $\pi_n^\infty(s)$ (for any natural number $n \in \mathbf{N}$) are again projective sequences. Also, prove that for any two projective sequences s_1 and s_2, the sequence $s_1 +^\infty s_2$ is a projective sequence.

5.10.2 Complete the proof of Theorem 5.10.4 (Soundness).

5.10.3 Consider the recursive specification $\{X_n = a^n 0 + X_{n+1} \mid n \in \mathbf{N}\}$, used in the proof of Theorem 5.5.20 ((In-)validity of AIP). Show that the solution for variable X_0 in the projective limit model $\mathbb{I}^\infty((\mathrm{BSP}+\mathrm{PR})(A))$ is the process $([0]_\vdash, [a.0]_\vdash, [a.0+a^2 0]_\vdash, \ldots)$.

5.10.4 The algebra $\mathbb{I}^n(\mathrm{BSP}(A))$, for some $n \in \mathbf{N}$, has as its domain $\mathbf{I}_n = \{\pi_{n\vdash}(p) | p \in \mathbf{I}\}$, where $\mathbf{I}$ is the domain of the initial algebra $\mathbb{I}((\mathrm{BSP}+\mathrm{PR})(A))$, operators $+_n$, $(a._n)_{a\in A}$ defined by $p +_n q = p +_\vdash q$ and $a._n p = \pi_{n\vdash}(a._\vdash p)$, and constants $0_n = 0_\vdash$ and $1_n = 1_\vdash$.

(a) Show that $\mathbb{I}^n(\mathrm{BSP}(A)) \models \mathrm{BSP}(A)$.

(b) Define projection operators on $\mathbf{I}_n$, with algebra $\mathbb{I}^n((\mathrm{BSP}+\mathrm{PR})(A))$ as a result, such that $\mathbb{I}^n((\mathrm{BSP}+\mathrm{PR})(A)) \models (\mathrm{BSP}+\mathrm{PR})(A)$.

(c) What process in $\mathbb{I}^n((\mathrm{BSP}+\mathrm{PR})(A))$ is a solution of $X = a.X$?

(d) Show that recursion principles AIP, RDP^-, and RSP are valid in $\mathbb{I}^n((\mathrm{BSP}+\mathrm{PR})(A))$.

(e) In what sense is $\mathbb{I}^\infty((\mathrm{BSP}+\mathrm{PR})(A))$ the limit of the algebras $\mathbb{I}^n((\mathrm{BSP}+\mathrm{PR})(A))$, for all $n \in \mathbf{N}$?

5.10.5 The so-called *Limit Rule* is the statement that all equations (containing variables) that are derivable from some given process theory when closed terms are substituted for the variables, are derivable from that

theory in general. Formulate this Limit Rule in a formal way and show, by using AIP and Proposition 5.5.26 (HNF property), that it holds for theory $((\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}} + \mathrm{AIP})(A)$. (Note that this implies that theory $((\mathrm{BSP} + \mathrm{PR})_{\mathrm{rec}} + \mathrm{AIP})(A)$ is ω-complete, as defined in Section 2.3; observe that the projective limit model introduced in this section is a model of that theory, but that, due to the assumption of AIP, the standard term model is not a model.)

5.10.6 Definition 5.10.11 (Regular projective sequence) shows how to define regularity of projective sequences via a transformation to a transition-system space. Define regularity directly in terms of projective sequences and argue that your definition gives the same notion of regularity as the one defined in Definition 5.10.11.
(Hint: the number of outgoing transitions of every state, the branching degree, stabilizes.)

5.11 Bibliographical remarks

Recursion is an essential ingredient of every concurrency theory. There are algebraic treatments in CCS (Milner, 1980), CSP (Hoare, 1985), ACP (Baeten & Weijland, 1990), topological process theory (De Bakker & Zucker, 1982a) and in other places. The present treatment owes most to (Bergstra & Klop, 1984b).

The notation $\mu X.E$ is inspired by the notation for least fixed points, and similar notations used in CCS and CSP. The CCS notation $\mu X.t$, where t is a term (potentially) containing recursion variable X, is generalized in the current notation to $\mu X.\{X = t\}$. Notation $\mu X.E$ corresponds to the notation $\langle X \mid E \rangle$ in ACP-style process algebra, that originates from (Bergstra & Klop, 1988).

A closer look at recursion in process algebra is found in (Kranakis, 1987; Usenko, 2002). The deduction rules in Table 5.2 are from (Van Glabbeek, 1987), and based on (Milner, 1980). The results in Section 5.4 are based on (Van Glabbeek, 1987; Baeten & Verhoef, 1995). For Section 5.8, see (Bergstra & Klop, 1984b) and (Mauw & Mulder, 1994). The dual role of recursion variables, as both constants and constrained variables, is also discussed in (Baeten & Bravetti, 2006); see also (Baeten & Bravetti, 2005).

The notion of guardedness is found in (Milner, 1980; Milne, 1982), but is derived from older ideas. In the present setting, guardedness comes down to having a right-linear grammar, see e.g. (Linz, 2001).

The principle RSP was formulated in (Bergstra & Klop, 1986c). Principles RDP, RDP^-, AIP, and AIP^-, as well as the proof of Theorem 5.5.23, are from (Baeten *et al.*, 1987b). A slightly less restrictive form of AIP^-, using the

notion of bounded non-determinism, can be found in (Van Glabbeek, 1987). Proposition 5.5.26 (HNF property) is from (Baeten & Van Glabbeek, 1987). The Projection Theorem is from (Baeten *et al.*, 1987a).

For the material on the projective limit model, see (Bergstra & Klop, 1982; De Bakker & Zucker, 1982a). For the Limit Rule, see (Baeten & Bergstra, 1988).