Process Algebra

Recursion

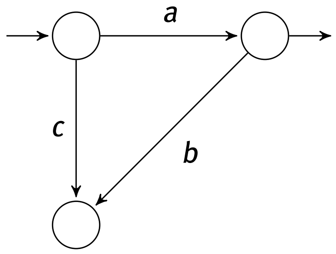
Bas Luttik

s.p.luttik@tue.nl

TU/e

Expressiveness

Can we express the following transition system



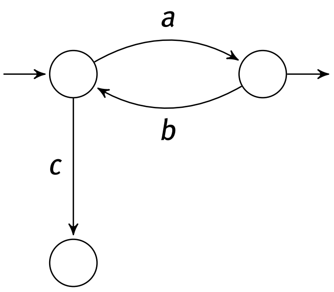
in BSP(A)?

Yes!

The transition system generated by $a.(b.0 + 1) + c.0$ is isomorphic to it.

Expressiveness

Can we express the following transition system



in BSP(A)?

No!

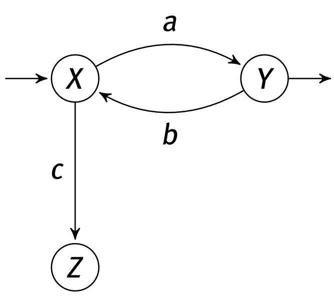
Infinite behaviour cannot be expressed in BSP(A).

How do we know for sure?

Expressiveness

General method for expressing transition systems:

1. label states with *process names* (a.k.a. recursion variables);
2. associate behaviour to every process name; defining equation specifies transition- and termination-behaviour.



$$X = a.Y + c.Z$$
$$Y = b.X + 1$$
$$Z = 0$$

Recursive specifications

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

$$X = t_X \ ,$$

with $X \in V_R$ and t_X a term over Σ and V_R.

We say that the recursive equation $X = t_X$ *defines* X.

A recursive specification over Σ and V_R is a set of recursive equations over Σ and V_R consisting of precisely one recursive equation defining X for every $X \in V_R$.

Operational rules

$$\frac{t_X \xrightarrow{a} t'_X}{X \xrightarrow{a} t'_X} \qquad \frac{t_X\downarrow}{X\downarrow}$$

Example

$$Y = a.b.Y + b.Z + 0$$
$$Z = Y + a.(Z + Y) \ .$$

Exercise: Give transition system for the term $a.Y$.

TU/e

Term model (simplified)

Let E be a recursive specification over BSP(A) and V_R.

Definition

The term algebra for BSP(A)+E is the algebra

$$\mathbb{P}(\mathrm{BSP}(A){+}E) = (\mathcal{C}(\mathrm{BSP}(A){+}E), +, (a.)_{a \in A}, 0, 1, (X)_{X \in V_R}) \ .$$

Proposition

Bisimilarity is a congruence on $\mathbb{P}(\mathrm{BSP}(A){+}E)$.

Theorem

BSP(A)+E is a sound axiomatisation of $\mathbb{P}(\mathrm{BSP}(A){+}E)/\leftrightarrow$.

Is BSP(A)+E also ground-complete for $\mathbb{P}(\mathrm{BSP}(A){+}E)/\leftrightarrow$?

TU/e

Equivalence of recursion variables

Example

Consider the recursive specification

$$\left\{ \begin{array}{l} X = a.X \quad , \\ Y = a.a.Y \end{array} \right\}$$

Then $X \leftrightarrow Y$.

But this equation cannot be derived from BSP$(A) + E$. (Prove!)

Conclusion: we need additional methods to reason about the equivalence of recursion variables.

[Remark: we shall not discuss a full-fledged ground-complete axiomatisation of the algebra $\mathbb{P}(\text{BSP}(A){+}E)/{\leftrightarrow}$, but we'll come close.]

Reasoning (model-independently) about recursive specifcations

Let E be a rec. spec. over signature Σ and set of variables V_R.

Let **A** be a Σ-algebra and ι the associated interpretation.

Definition

A solution of E in **A** is an extension κ of ι with interpretations of the recursion variables in V_R as elements of **A** such that

$$\mathbf{A}, \kappa \models X = t_X$$

for every equation $X = t_X$ in E.

If κ is a solution of E and $X \in V_R$, then we also say that $\kappa(X)$ is a *solution* of X in **A**.

Examples

The recursive specification $E_1 = \{X = a.1\}$has a solution both in $\mathbb{P}(\text{BSP}(A))/{\leftrightarrow}$ and in $\mathbb{P}(\text{BSP}(A){+}E_1)/{\leftrightarrow}$.

The recursive specification $E_2 = \{X = a.X\}$has a solution in $\mathbb{P}(\text{BSP}(A){+}E_2)/{\leftrightarrow}$, but not in $\mathbb{P}(\text{BSP}(A))/{\leftrightarrow}$.

The recursive specification $E_3 = \{X = X\}$has many solutions, both in $\mathbb{P}(\text{BSP}(A))/{\leftrightarrow}$ and in $\mathbb{P}(\text{BSP}(A){+}E_3)/{\leftrightarrow}$.

NB: a solution of E in an algebra of transition systems modulo bisimilarity is an assignment of closed terms to recursion variables such that the recursion equations are true up to bisimilarity.

Equivalence of recursion variables

Consider the recursive specification

$$E = \left\{ \begin{array}{l} X = a.X \;, \\ Y = a.a.Y \end{array} \right\} \;.$$

Every solution of X in **A** is also a solution of Y in **A**:

$$X = a.X = a.a.X \;.$$

Hence, if κ is a solution of E in **A**:

$$\kappa(X) = \kappa(a.a.X) = \mathbf{a}.\mathbf{a}.\kappa(X)$$
$$(\mathbf{a}.\text{ the interpretation of } a.\text{ in } \mathbf{A}) \;,$$

so $\kappa(X)$ is a solution of Y in E.

Is every solution of Y in **A** a solution of X in **A**?

No: it is possible to construct a model of BSP(A)+E in which Y has a solution that is not also a solution of X.

Idea: only consider models in which recursive specifications have unique solutions.

Do there exist (non-trivial) models in which every recursive specification has a unique solution?

Note: $\{X = X\}$ has as many solutions as the cardinality of the model!

Example: equivalence of rec. vars.

Consider the recursive specification

$$\left\{ \begin{array}{l} X = a.X \;, \\ Y = a.a.Y \end{array} \right\} \;.$$

Perhaps if we exclude some models (e.g., the answers to the exercise on the previous slide), then we may be able to say more about the equivalence of X and Y in the above recursive specification.

Note that, for models in which X and Y both have a unique solution, the reasoning on slide 12 would suffice to conclude that X and Y indeed denote the same process!

Guardedness

An occurrence of a recursion variable X in a closed term s is guarded if it occurs in the scope of an action prefix.

A term s is completely guarded if all occurrences of all recursion variables in s are guarded.

Recursive specification E is completely guarded if all right-hand sides of all equations in E are completely guarded.

Recursive specification E is guarded if there exists a completely guarded recursive specification F with $V_R(E) = V_R(F)$ and BSP(A)+$E \vdash X = t$ for all $X = t \in F$.

Example

1. The recursive specification

   $$E_1 = \{X_1 = a.X_1,\ Y_1 = a.X_1\}$$

   is completely guarded.
2. The recursive specification

   $$E_2 = \{X_1 = a.X_1,\ Y_1 = X_1\}$$

   is *not* completely guarded.

But E_1 and E_2 have exactly the same solutions in every model!

Recursive Specification Principle

Let Σ be a signature; we say that Σ-algebra $\mathbf{A}$ satisfies the Recursive Specification Principle (RSP) if every *guarded* recursive specification E over Σ and some set V_R of variables has *at most* one solution.

RSP will be used to prove in a model-independent manner that two recursion variables have the same unique solution in every model.

RSP as a proof principle

Consider rec. spec. E consisting of the following equations:

$$X = a.X + b.X \ ,$$
$$Y = a.Y + b.Z \ , \text{ and}$$
$$Z = a.Z + b.Y \ .$$

We can prove that $X = Y$ in the context of E as follows:

Define sequences of terms $\vec{t} = t_X, t_Y, t_Z$ and $\vec{u} = u_X, u_Y, u_Z$ by $t_X \equiv X$, $t_Y \equiv Y$, $t_Z \equiv Z$, and $u_X \equiv X$, $u_Y \equiv X$, $u_Z \equiv X$. Then both $\vec{t}$ and $\vec{u}$ denote solutions of E:

$$t_X \equiv X = a.X + b.X = a.t_X + b.t_X$$
$$t_Y \equiv Y = a.Y + b.Z = a.t_Y + b.t_Z$$
$$t_Z \equiv Z = a.Z + b.Y = a.t_Z + b.t_Y$$

$$u_X \equiv X = a.X + b.X = a.u_X + b.u_X$$
$$u_Y \equiv X = a.X + b.X = a.u_Y + b.u_Z$$
$$u_Z \equiv X = a.X + b.X = a.u_Z + b.u_Y$$

Since E is guarded, by RSP, $\vec{t} = \vec{u}$, so $X \equiv u_Y = t_Y \equiv Y$.

Example

Consider $E = \left\{ \begin{array}{lcl} X & = & a.a.X \\ Y & = & (a.Y) \cdot b.1 \end{array} \right\}$. We prove that $X = Y$ in every model satisfying RSP.

To this end, we define an auxiliary guarded recursive specification $E' = \{Z_n = a.Z_{n+1} \mid n \in \mathbb{N}\}$ and define two sequences of terms denoting a solution for E:

Define $t_0, t_1, \ldots, t_i, \ldots$ ($i \in \mathbb{N}$) by $t_i \equiv X$ if i is even, and $t_i \equiv a.X$ if i is odd.

If i is even, then (using that $i + 1$ is odd and hence $t_{i+1} = a.X$) we have $t_i \equiv X = a.a.X = a.t_{i+1}$.

If i is odd, then (using that $i + 1$ is even and hence $t_{i+1} = X$) we have $t_i \equiv a.X = a.t_{i+1}$.

In both cases we have argued that $t_i = a.t_{i+1}$, so $t_0, t_1, t_2, \ldots$ denotes a solution of E'.

Define $u_0, u_1, \ldots, u_i, \ldots$ ($i \in \mathbb{N}$) with induction on i by $u_0 \equiv Y$ and $u_{i+1} \equiv u_i \cdot (b.1)$.

We prove that $u_i = a.u_{i+1}$ for all $i \in \mathbb{N}$ with induction on i:
If $i = 0$, then
$u_i \equiv u_0 = Y = (a.Y) \cdot b.1 \stackrel{A10}{=} a.(Y \cdot b.1) = a.u_1$.
Let $i \in \mathbb{N}$ and suppose that $u_i = a.u_{i+1}$ (IH). Then

$$\begin{aligned} u_{i+1} &\equiv u_i \cdot b.1 \\ &\stackrel{IH}{=} (a.u_{i+1}) \cdot b.1 \\ &\stackrel{A10}{=} a.(u_{i+1} \cdot (b.1)) \equiv a.u_{i+2} \ . \end{aligned}$$

Hence, both $t_0, t_1, \ldots, t_i, \ldots$ and $u_0, u_1, \ldots, u_i, \ldots$ denote solutions of E', and therefore, in every model satisfying RSP we have that $X \equiv t_0 = u_0 \equiv Y$.

Extending a process theory with recursion

[This material is not (completely) covered in the book!]

Let A be a set of actions;
let $T(A)$ be a process theory over A.

Let *Rec* be the set of all recursive specifications over $T(A)$.

We define $T(A)_{\mathrm{rec}}$ as the extension of $T(A)$ with for every $E \in \mathit{Rec}$ and for every $(X = t_X) \in E$:

1. a constant symbol $\mu X.E$, and
2. an axiom $\mu X.E = \mu t_X.E$ (Rec).

$\mu t.E$ is the term obtained from t by replacing all occurrences of a recursion variable X by $\mu X.E$ (see book for an inductive definition).

Term model

The term algebra $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))$ for $\mathrm{BSP}_{\mathrm{rec}}(A)$ is the algebra

$$(\mathcal{C}(\mathrm{BSP}_{\mathrm{rec}}(A)), +, (a.)_{a\in A}, 0, 1, (\mu X.E)_{E\in \mathit{Rec}, X\in V_R(E)})$$

Operational semantics

$$\frac{}{a.x \xrightarrow{a} x} \qquad \frac{x \xrightarrow{a} x'}{x+y \xrightarrow{a} x'} \qquad \frac{y \xrightarrow{a} y'}{x+y \xrightarrow{a} y'} \qquad \frac{\mu t_X.E \xrightarrow{a} t'_X}{\mu X.E \xrightarrow{a} t'_X}$$

$$\frac{}{1\downarrow} \qquad \frac{x\downarrow}{(x+y)\downarrow} \qquad \frac{y\downarrow}{(x+y)\downarrow} \qquad \frac{\mu t_X.E\downarrow}{\mu X.E\downarrow}$$

1. Bisimilarity is a congruence on $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))$.
2. $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))/\underline{\leftrightarrow}$ is a model for $\mathrm{BSP}(A)$, the so called term model for $\mathrm{BSP}(A)$.

SOS meta-theory: the *path* format

Definition

A collection of *operational rules* is in path format if for every rule it holds that

1. target of every premise is variable;
2. source of conclusion is either a variable, or of the form $f(x_1, \ldots, x_n)$ with f an n-ary function symbol and x_i $(1 \leq i \leq n)$ variables;
3. variables in targets of premises and source of conclusion all distinct.

Theorem

If the operational rules of a process calculus are all in the path format, then bisimilarity is a congruence for that process calculus.

Corollary

Rules of $\mathrm{BSP}_{\mathrm{rec}}(A)$ *are all in path format, so bisimilarity is a congruence for it.*

Term model (properties)

Definition

Let Σ be a signature; we say that Σ-algebra **A** satisfies the Recursive Definition Principle (RDP) if every recursive specification E over Σ and some set V_R of variables has **at least** one solution.

Theorem

$\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))/\underline{\leftrightarrow} \models \mathrm{RDP}$.

Proof.

Let E be a recursive specification.
Define κ as the extension of ι such that, for every recursion variable X in E,

$$\kappa(X) = [\mu X.E]_{\underline{\leftrightarrow}} \quad .$$

Then $\kappa(X) = \kappa(t_X)$ for every recursion variable X in E (verify!).
So κ is indeed a solution of E in $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))/\underline{\leftrightarrow}$. □

Definition

Let Σ be a signature; we say that Σ-algebra **A** satisfies the Recursive Specification Principle (RSP) if every *guarded* recursive specification E over Σ and some set V_R of variables has **at most** one solution.

Theorem

$\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))/\underline{\leftrightarrow} \models \mathrm{RSP}$.

Proof.

[Postponed.] □

Regularity

Recall: a transition system is *regular* iff both its set of states and its set of transitions are finite.

Definition

A regular behaviour is an equivalence class of transition systems modulo $\leftrightarrow$ containing at least one regular (i.e., finite-state) transition system.

Irregularity Lemma

Let T be a transition system, and suppose that the set of (reachable) states of T has an infinite subset of which the elements are all pairwise not bisimilar. Then T is not bisimilar to a regular transition system.

Proof.

See next slide. □

Irregularity Lemma (proof)

Lemma

Let T be a transition system, and suppose that the set of (reachable) states of T has an infinite subset of which the elements are all pairwise not bisimilar. Then T is not bisimilar to a regular transition system.

Proof.

Let U be an infinite subset of the set of states of T of which all elements are pairwise not bisimilar.

Suppose, towards a contradiction, that T is bisimilar to a regular transition system T'.

Then there exists a bisimulation relation R relating the initial states of T and T', and hence, since all states in a transition system are reachable, for every state s in T there is a state s' in T' such that $s \mathrel{R} s'$.

Now T', since it is regular, has only finitely many states, so there must exist two distinct elements u and v that are related to the same state s' of T' according to R.

It then follows that $u \leftrightarrow s' \leftrightarrow v$, and hence $u \leftrightarrow v$ which contradicts our assumption that u and v are pairwise not bisimilar. □

Irregularity Lemma (example application)

Consider the recursive specification

$$E = \{X_n = X_{n+1} + a^n.Y \mid n \in \mathbb{N}\} \cup \{Y = 1\} \ ;$$

We prove that the behaviour denoted by $\mu X_0.E$ (in $\mathbb{P}(\mathrm{BSP}_{\mathrm{rec}}(A))/\leftrightarrow$) is not regular, with an application of the Irregularity Lemma. It suffices to argue that the transition system associated with $\mu X_0.E$ has an infinite subset of (reachable) pairwise not bisimilar states.

To this end, define the infinite sequence of terms $p_0, p_1, \dots, p_i, \dots$ ($i \in \mathbb{N}$) inductively by $p_0 \equiv \mu Y.E$ and $p_{i+1} \equiv a.p_i$.

Then it is clear from the structural operational rules that $\mu X_{i+1}.E \xrightarrow{a} p_i$ for all $i \in \mathbb{N}$. Furthermore, since $\mu X_{i+1} \xrightarrow{a} p'$ implies $\mu X_i \xrightarrow{a} p'$ for all p', it follows that $\mu X_0.E \xrightarrow{a} p_i$ for all $i \in \mathbb{N}$. Hence, each p_i is reachable. Moreover, from the definition of p_i and the structural operational semantics, it immediately follows that if $p_i \xrightarrow{a} p'$ for some p' then $i > 0$ and $p' \equiv p_{i-1}$.

To show that the p_i ($i \in \mathbb{N}$) are pairwise not bisimilar, suppose, towards a contradiction, that there exist $i \neq j$ such that $p_i \leftrightarrow p_j$. Without loss of generality we can then consider the least natural number i for which there exists $j > i$ with $p_i \leftrightarrow p_j$. Then, since $j > 0$ and hence $p_j \equiv a.p_{j-1}$, by the structural operational semantics it follows that $p_j \xrightarrow{a} p_{j-1}$. Since $p_i \leftrightarrow p_j$, it follows that $p_i \xrightarrow{a} p_{i-1}$ such that $p_{i-1} \leftrightarrow p_{j-1}$. From $j > i$ it follows that $j - 1 > i - 1$. So we find that $j - 1 > i - 1$ and $p_{i-1} \leftrightarrow p_{j-1}$, which contradicts our choice of i as the least natural number for which there exists $j > i$ with $p_i \leftrightarrow p_j$.

Stack

Let $D = \{d_1, \dots, d_n\}$ ($n \in \mathbb{N}$) be a finite set of data.

Let D^* be the set of all finite sequences of elements of D.

The recursive specification consisting of the following equations defines the behaviour of a *stack* over D:

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

Stack

Theorem

The behaviour of a stack is not regular.

Proof.

Every recursion variable S_σ denotes a *reachable* (!) state in the transition system associated with *Stack1*; consider S_σ and $S_{\sigma'}$.

Then σ and σ' characterise the unique sequence of $pop(d)$-transitions to the state denoted by S_ϵ, the only state with the termination option. It follows that the states denoted by S_σ and $S_{\sigma'}$ are bisimilar only if $\sigma = \sigma'$.

So *Stack1* has infinitely many reachable states S_σ, all pairwise non-bisimilar. Therefore, *Stack1* cannot be bisimilar to a regular transition system. □

Definability

Definition

Let T be a process theory.

A process in some model of 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 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.

Theorem

A behaviour is finitely definable over BSP(A) *iff it is regular.*

Finite definability over BSP(A)

Theorem

A behaviour is finitely definable over BSP(A) *iff it is regular.*

Proof.

We have already observed that a regular transition system can be specified with a finite guarded recursive specification BSP(A).

Observe (by inspecting the operational rules for BSP(A)+E) that if E is a finite completely guarded recursive specification E and X is some recursion variable defined in E, then all closed terms reachable in one or more steps from X are subterms of some right-hand side of an equation in E. (Exercise: Prove this formally and generalise to completely guarded $\mathrm{BSP_{rec}}(A)$-terms!)

There are finitely many such subterms, so the transition system associated with a recursion variable has finitely many states.

The number of summands of a completely guarded $\mathrm{BSP_{rec}}(A)$-term is an upper bound on the outdegree of the associated state. □

Stack revisited

Recall the (infinite) guarded recursive specification over BSP(A) defining the behaviour of a *stack* over D:

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

We have established that the behaviour of a stack is not regular, and hence is not finitely definable over BSP(A).

Interestingly, the behaviour of a stack *is* finitely definable over TSP(A), by the following guarded recursive specification:

$$\mathit{Stack2} = 1 + \sum_{d\in D} \mathit{push}(d).\mathit{Stack2} \cdot \mathit{pop}(d).\mathit{Stack2} \ .$$

Definability and expressiveness in TSP(A)

TSP(A) is as expressive as BSP(A) when it comes to specifying transition systems: in both theories precisely all countable transition systems can be specified.

In TSP(A) it is possible to finitely define infinite-state behaviours, which are not regular and hence not finitely definable in BSP(A) (e.g., the behaviour of a stack).