III

10 Semi-Lattices, Non-Determinism and Basic Process Algebra

The 'non-deterministic choice' construct is understood as the union of a power-set endofunctor, which, categorically, is a monad whose algebras are *semi-lattices*. This leads to a non-deterministic behaviour endofunctor $BX = \check{\mathcal{P}}(1 + \mathsf{Act} \times X)$ whose coalgebras are non-deterministic transition systems. A functorial *denotational* semantics is 'naturally' associated to this behaviour and its induced functorial *operational* semantics turns out to be '*basic process algebra*'.

Let us consider programs with a *non-deterministic*' behaviour. For this, let us introduce the new construct 'or' of **non-deterministic choice**. The intended meaning of a program $u \,\mathsf{or}\, v$ is that it can choose whether to behave either as the subprogram u or as the subprogram v. The following equations should then hold in the operational model $[\![\text{-}]\!]$. For all programs t, u, v,

$$\begin{array}{rcll} [\![(t \,\mathsf{or}\, u) \,\mathsf{or}\, v]\!] & = & [\![t \,\mathsf{or}\, (u \,\mathsf{or}\, v)]\!] & \text{(associativity)} \\ [\![u \,\mathsf{or}\, v]\!] & = & [\![v \,\mathsf{or}\, u]\!] & \text{(commutativity)} \\ [\![t \,\mathsf{or}\, t]\!] & = & [\![t]\!] & \text{(absorption)} \end{array}$$

Algebraically, a set Y with a binary operator $\vee : Y \times Y \to Y$ which is associative, commutative, and absorptive, that is, such that for all x, y, z in Y,

$$\begin{array}{rcl} (x \vee y) \vee z & = & x \vee (y \vee z) \\ x \vee y & = & y \vee x \\ x \vee x & = & x \end{array}$$

forms a **semi-lattice**; the operator $\vee$ is called the **join** of the semi-lattice. The program construct or should then behave as the join of a semi-lattice:

$$[\![u \,\mathsf{or}\, v]\!] = [\![u]\!] \vee [\![v]\!]$$

As an example of a semi-lattice, consider the set $\mathcal{P}X$ of the subsets of a set X: the binary union $\cup : \mathcal{P}X \times \mathcal{P}X \to \mathcal{P}X$ is associative, commutative, and absorptive, hence $\langle \mathcal{P}X, \cup \rangle$ is a semi-lattice. A similar semi-lattice is the one obtained by considering the set

$$\mathcal{P}_{\text{fi}}X \;=\; \{X' \subseteq X \mid X' \text{ finite}\}$$

of finite subsets of a set X, as well as its '*relevant*' part

$$\check{\mathcal{P}}X \;=\; \{X' \subseteq X \mid X' \text{ finite, } X' \neq \emptyset\}$$

obtained by omitting the empty set. This latter semi-lattice (the binary union $\cup : \check{\mathcal{P}}X \times \check{\mathcal{P}}X \to \check{\mathcal{P}}X$ is its join) is of particular importance because it is the free semi-lattice over X; that is, the functor

$$X \mapsto \langle \check{\mathcal{P}}X, \cup \rangle$$

is left adjoint to the forgetful functor

$$\langle Y, \vee \rangle \mapsto Y$$

from the category of semi-lattices and join-preserving functions to sets.

> Write $SL(\mathbf{Set})$ for **the category of semi-lattices** with arrows $f : \langle X, \vee \rangle \to \langle Y, \sqcup \rangle$, the join-preserving functions $f : X \to Y$ between the underlying sets:
>
> $$\begin{tikzcd} X \times X \arrow[r, "f \times f"] \arrow[d, "\vee"'] & Y \times Y \arrow[d, "\sqcup"] \\ X \arrow[r, "f"'] & Y \end{tikzcd}$$
>
> Equationally, for every pair (x, x') in $X \times X$,
>
> $$f(x \vee x') = fx \sqcup fx'$$

Free semi-lattices. Recall that a functor $U : \mathbf{D} \to \mathbf{C}$ has a left adjoint $F : \mathbf{C} \to \mathbf{D}$ if and only if there exists a natural transformation $\eta : I \Rightarrow UF$ such that each η_X is universal from X to U. That is, for every X in $\mathbf{C}$, Z in $\mathbf{D}$, and $f : X \to UZ$ there exists a unique arrow $f^\sharp : FX \to Z$ in $\mathbf{D}$, such that $f = Uf^\sharp \circ \eta_X$:

$$\begin{tikzcd} X \arrow[r, "\eta_X"] \arrow[dr, "f"'] & UFX \arrow[d, "Uf^\sharp"] \\ & UZ \end{tikzcd} \qquad \begin{tikzcd} FX \arrow[d, dashed, "f^\sharp"] \\ Z \end{tikzcd}$$

Let now $U : SL(\mathbf{Set}) \to \mathbf{Set}$ be the above forgetful functor mapping semi-lattices to their carriers and let

$$FX = \langle \check{\mathcal{P}}X, \cup \rangle$$

Then, for every set X, the function

$$\{\text{-}\}_X : X \to \check{\mathcal{P}}X = UFX \qquad\qquad x \mapsto \{x\}$$

mapping every element x of X to the corresponding singleton set $\{x\}$ gives the unit $\eta : I \Rightarrow UF$ of the adjunction:

$$
\begin{tikzcd}
X \arrow[r, "\{\text{-}\}_X"] \arrow[dr, "f"'] & UFX = \check{\mathcal{P}}X \arrow[d, "Uf^\sharp = f^\sharp"] \\
& U\langle Y, \vee\rangle = Y
\end{tikzcd}
\qquad
\begin{tikzcd}
\check{\mathcal{P}}X \arrow[d, dashed, "f^\sharp"'] & \check{\mathcal{P}}X \times \check{\mathcal{P}}X \arrow[l, "\cup"'] \arrow[d, "f^\sharp \times f^\sharp"] \\
Y & Y \times Y \arrow[l, "\vee"]
\end{tikzcd}
$$

Indeed, for every finite subset $\{x_1, \ldots, x_n\} = \{x_1\} \cup \ldots \cup \{x_n\}$ of X

$$f^\sharp\{x_1, \ldots, x_n\} = fx_1 \vee \ldots \vee fx_n$$

is the required unique join-preserving function. (The properties of the join make bracketing irrelevant.) This shows that, for every set X, the pair $\langle \check{\mathcal{P}}X, \cup\rangle$ is the free semi-lattice on X.

As usual, the counit $\varepsilon : FU \Rightarrow I$ of the above adjunction can be obtained by taking for f the identity on $Y = U\langle Y, \vee\rangle$. This gives a 'big join'

$$\bigvee : \check{\mathcal{P}}Y \to Y$$

mapping every finite subset of Y to the join of its elements:

$$\bigvee\{y_1, \ldots, y_n\} = y_1 \vee \ldots \vee y_n$$

In particular, the value of the counit at a free semi-lattice $\langle Y, \vee\rangle = \langle \check{\mathcal{P}}X, \cup\rangle$ is the 'big union'

$$\bigcup : \check{\mathcal{P}}^2 \Rightarrow \check{\mathcal{P}}$$

sending each set of sets into its union. Since every adjunction $F \dashv G$ (with unit η and counit ε) gives rise to a monad $<GF, \eta, G\varepsilon F>$, (cf Section 2), the triple

$$\check{\mathcal{P}} = <\check{\mathcal{P}}, \{\text{-}\}, \bigcup>$$

is a monad. The isomorphism of categories

$$SL(\mathbf{Set}) \cong \mathbf{Set}^{\check{\mathcal{P}}}$$

gives then an alternative description of semi-lattices as algebras of the monad $\check{\mathcal{P}}$. (See "Algebras are T-algebras" in Section 2 or check directly.) Similarly, one can check that the algebras of the the *unrestricted* power-set monad $\mathcal{P} = <\mathcal{P}, \{\text{-}\}, \bigcup>$ are *complete semi-lattices* – semi-lattices with joins of arbitrary cardinality:

$$CSL(\mathbf{Set}) \cong \mathbf{Set}^{\mathcal{P}}$$

Formally, a complete semi-lattice is a partial order $\langle Y, \leq\rangle$ in which *every* subset $Y' \subseteq Y$ has a least upper bound $\bigvee Y'$, while a semi-lattice can be seen as a partial order with least upper bounds only of finite and non empty subsets. (Conversely, every semi-lattice defines a partial order $x \leq y \iff x \vee y = y$.)

In general, 'κ-complete' semi-lattices can be used to define power-set monads of any (*regular*) cardinality κ. Semantically, the cardinality to be used depends on the kind of non-determinism one is interested in. Here only finite determinism is studied, hence (finite) semi-lattices are used.

Even more in general, semi-lattices give an axiomatic description of various 'powerdomains' used in semantics. This holds because semi-lattices can be defined 'internally' in any category $\mathbf{C}$ with binary products:

> A semi-lattice in $\mathbf{C}$ is a pair $\langle Y, \vee \rangle$ with Y an object of $\mathbf{C}$ and $\vee : Y \times Y \to Y$ an arrow of $\mathbf{C}$ which is associative, commutative, and absorptive in a diagrammatic sense. For instance, the commutativity of the join can be described diagrammatically using the canonical 'swap' arrow
>
> $$\begin{tikzcd} & Y \times Y \arrow[dl, "\mathsf{snd}"'] \arrow[d, dashed, "\mathsf{swap}"] \arrow[dr, "\mathsf{fst}"] & \\ Y & Y \times Y \arrow[l, "\mathsf{fst}"] \arrow[r, "\mathsf{snd}"'] & Y \end{tikzcd}$$
>
> as follows.
>
> $$\begin{tikzcd} Y \times Y \arrow[rr, "\mathsf{swap}"] \arrow[dr, "\vee"'] & & Y \times Y \arrow[dl, "\vee"] \\ & Y & \end{tikzcd}$$
>
> Write then $SL(\mathbf{C})$ for the corresponding category with as arrows the join-preserving arrows of $\mathbf{C}$.

For instance, the Plotkin powerdomain monad can be shown to arise from the semi-lattices in a category of complete partial orders and continuous functions. (Notice, the order induced by the semi-lattice structure has nothing to do with the one of the underlying category of complete partial orders.) Similarly, the semi-lattices in a category of complete metric spaces and non-distance-increasing functions give rise to the compact metric powerdomain.

In order to deal with non-deterministic behaviours as introduced by the binary choice construct 'or' consider the new behaviour endofunctor

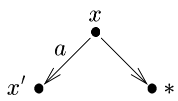
$$B : \mathbf{Set} \to \mathbf{Set} \qquad\qquad X \mapsto \check{\mathcal{P}}(1 + \mathsf{Act} \times X)$$

obtained by composing the (deterministic) behaviour endofunctor $X \mapsto 1 + \mathsf{Act} \times X$ with the semi-lattice monad $\check{\mathcal{P}}$. Its coalgebras are the **finitely branching transition systems**, that is, transition systems which in every state can choose among a *finite* set of transitions. This finite non-determinism reflects the finiteness of the choice construct; this restriction simplifies the presentation, but, in general, one can consider semi-lattices (and corresponding monads) with joins of larger cardinality.

Formally, the correspondence between coalgebras $\langle X, k\rangle$ of the above behaviour and finitely branching transition systems $\langle X, \{\xrightarrow{a}\}_{Act}, \downarrow *\rangle$ is as follows. Omitting, as usual, the injections into the coproduct $1 + \mathsf{Act} \times X$,

$$x \xrightarrow{a} x' \iff k(x) \ni <a, x'> \qquad\qquad x \downarrow * \iff kx \ni *$$

for every $x \in X$. (Cf Section 3.) Notice that a state might both perform an action or become inert; for instance, $k(x) = \{<a, x'>, *\}$ corresponds to the transitions



Notice that above, and whenever convenient, the fact that $x \downarrow *$ holds is treated as a special transition $x \longrightarrow *$:

$$x \downarrow * \iff x \longrightarrow *$$

Next, consider the following 'minimal' language for producing behaviours of type B.

Basic Process Algebra. The basic language for the behaviour $BX = \check{\mathcal{P}}(1+\mathsf{Act}\times X)$ should contain a basic inert program nil, an 'action prefixing' unary operator for every $a \in \mathsf{Act}$, and the binary choice or. Formally, the language is defined by the grammar

$$t ::= x \mid \mathsf{nil} \mid a\,.\,t \mid (t \,\mathsf{or}\, t)$$

and its operational model $[\![\text{-}]\!]$ (a B-coalgebra structure on the above terms) is defined by induction on the structure of the terms as follows.

$$[\![\mathsf{nil}]\!] = \{*\} \qquad [\![a\,.\,t]\!] = \{<a, t>\} \qquad [\![u \,\mathsf{or}\, v]\!] = [\![u]\!] \cup [\![v]\!]$$

(For the treatment of the variables x see the next section.) In terms of transition systems, this corresponds to the following set $\mathcal{R}$ of operational rules.

$$\mathsf{nil} \longrightarrow * \qquad a\,.\,t \xrightarrow{a} t \qquad \frac{u \xrightarrow{a} u'}{u \,\mathsf{or}\, v \xrightarrow{a} u'} \qquad \frac{u \xrightarrow{a} u'}{u \,\mathsf{or}\, v \xrightarrow{a} u'}$$

(In order to simplify the notation, a transition $u \xrightarrow{a} u'$ is here intended possibly to be of the form $u \longrightarrow *$. Thus in particular if $u \longrightarrow *$ then also $u \,\mathsf{or}\, v \longrightarrow *$.)

Basic process algebra is functorial. Let us prove that basic process algebra is functorial by defining a functorial *denotational* semantics Ψ such that its operational dual $\Psi^{\#}$ is equal to the operational semantics induced by the rules of basic process algebra.

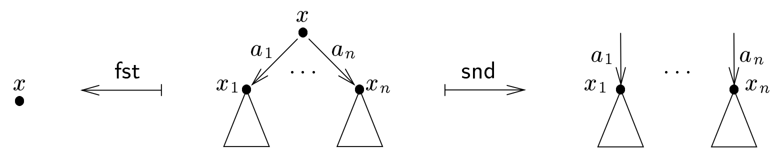
One would like to use the above rules $\mathcal{R}$ for defining directly the functorial operational semantics by induction on a germ $\phi : \Sigma BT \Rightarrow BT$, for Σ and T the signature and the syntactical monad corresponding to basic process algebra, respectively. This is easily done for or and nil using the union $\cup$ and the termination state $*$ available in $B = \check{\mathcal{P}}(1 + \mathsf{Act} \times \text{-})$, but action prefixing causes troubles. Indeed, for any object r of type BT, $a\,.\,r$ should be mapped by ϕ to $\{<a, r>\}$, but this is of type B^2T rather than BT. Instead, the definition of a functorial denotational semantics Ψ lifting the observational comonad $D{=}<D, \varepsilon, \delta>$ to the Σ- (or, equivalently, to the T-) algebras using the rules of basic process algebra causes no problem.

Recall that, for every X, the set DX is the carrier of the final $(X{\times}B)$-coalgebra:

$$X \xleftarrow{\varepsilon_X \,=\, \mathsf{fst}_X} DX \cong X{\times}BDX \xrightarrow{\mathsf{snd}_X} BDX = \check{\mathcal{P}}(1 + \mathsf{Act} \times DX)$$

As shown in Section 13, although the endofunctor $\check{\mathcal{P}} : \mathbf{Set} \to \mathbf{Set}$ is not $\boldsymbol{\omega}^{\mathrm{op}}$-continuous, the final $(X{\times}B)$-coalgebra exists, hence the observational comonad D cofreely generated by B can be defined.

The set DX is the set of global behaviours of states $x \in X$ wrt B. These can be seen as *trees* which are finitely branching, whose nodes are labelled by $x \in X$, and whose arcs are labelled by $a \in \mathsf{Act}$. The counit $\varepsilon_X = \mathsf{fst}_X : DX \to X$ of the comonad gives the label of the root node for each tree in DX and the other projection $\mathsf{snd}_X : DX \to BDX$ gives the remaining part of the tree (and it coinductively extends to give the comultiplication $\delta : D \Rightarrow D^2$ of the comonad D):

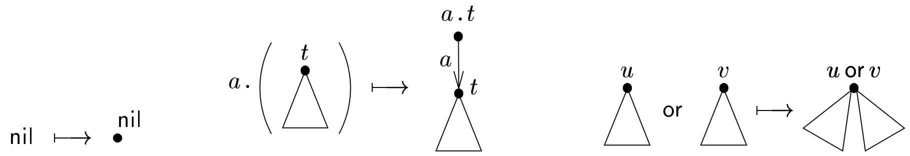


Now, let us first lift the *endofunctor* D to an endofunctor Ψ on the Σ-algebras and then check that also the operations of the comonad D lift. By the equivalence between liftings and actions illustrated in Section 7, the desired endofunctor Ψ is the same as the action

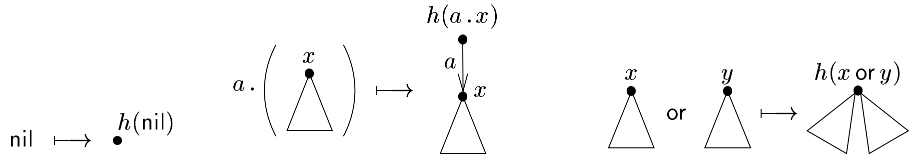
$$\Psi : \Sigma DU^{\Sigma} \Rightarrow DU^{\Sigma}$$

of the constructs Σ on the composite functor $DU^{\Sigma} : \mathbf{Set}^{\Sigma} \to \mathbf{Set}$, where $U^{\Sigma} : \mathbf{Set}^{\Sigma} \to \mathbf{Set}$ is the forgetful functor mapping Σ-algebras to their carriers.

The action Ψ. Let us consider first the case of free Σ-algebras, that is, the action of the program constructs nil, $a\,.$, and or on DT, where, notice, an object of type DT is a tree whose nodes are labelled by terms t of basic process algebra.



Then, in general, for every Σ-algebra $\langle X, h\rangle$, the action of Ψ on $DU^{\Sigma}\langle X, h\rangle = DX$ is



Formally, using the meta-variables p and q to range over objects of type D, ie global behaviours wrt B, the value of the natural transformation $\Psi : \Sigma DU^{\Sigma} \Rightarrow DU^{\Sigma}$ at a Σ-algebra $\langle X, h\rangle$ is defined as follows.

$$
\begin{array}{lll}
\mathsf{nil} & \mapsto & < h(\mathsf{nil}), \{*\} > \\
a\,.\,p & \mapsto & < h(a\,.\,(\mathsf{fst}_X p)), \{< a, p >\} > \\
p\,\mathsf{or}\,q & \mapsto & < h((\mathsf{fst}_X p)\ \mathsf{or}\ (\mathsf{fst}_X q)), (\mathsf{snd}_X p) \cup (\mathsf{snd}_X q) >
\end{array}
$$

Naturality follows from the fact that no assumption is made on the form of the Σ-algebra $\langle X, h\rangle$.

Therefore, for every Σ-algebra structure $h : \Sigma X \to X$, the structure $\Psi h : \Sigma DX \to DX$ is a pair, whose first component is simply the composite function $h \circ \Sigma\mathsf{fst}_X = h \circ \Sigma\varepsilon_X : \Sigma DX \to X$. Writing $\Psi' h : \Sigma DX \to BDX$ for the second component of Ψh, one has the following commuting diagram.

$$
\begin{tikzcd}
\Sigma X \arrow[d, "h"'] & \Sigma DX \arrow[l, "\Sigma\varepsilon_X"'] \arrow[d, "\Psi h"'] \arrow[dr, "\Psi' h = \mathsf{snd}_X \circ \Psi h"] & \\
X & DX \arrow[l, "\varepsilon_X = \mathsf{fst}_X"] \arrow[r, "\mathsf{snd}_X"'] & BDX
\end{tikzcd}
$$

The left square is one of the two diagrams which have to commute in order for $\Psi =< \Psi, \varepsilon, \delta >$ to lift the whole comonad $D =< D, \varepsilon, \delta >$. The other diagram, namely

$$\begin{tikzcd}
\Sigma DX \arrow[r, "\Sigma\delta_X"] \arrow[d, "\Psi h"'] & \Sigma D^2X \arrow[d, "\Psi^2 h"] \\
DX \arrow[r, "\delta_X"'] & D^2X
\end{tikzcd}$$

also commutes, because:

> The composite functions $\delta_X \circ \Psi h$ and $\Psi^2 h \circ \Sigma\delta_X$ both fit as the (unique!) pair $<\Psi h, B\delta_X \circ \Psi' h>: \Sigma DX \to D^2X$
>
> $$\begin{tikzcd}
& \Sigma DX \arrow[dl, equal] \arrow[dd, dashed, "!"] \arrow[dr, "\Psi' h"] & \\
\Sigma DX \arrow[d, "\Psi h"'] & & BDX \arrow[d, "B\delta_X"] \\
DX & D^2X \arrow[l, "\varepsilon_{DX} = \mathsf{fst}_{DX}"] \arrow[r, "\mathsf{snd}_{DX}"'] & BD^2X
\end{tikzcd}$$
>
> Indeed, noticing that Ψ' is natural, everything in sight in the following two diagrams commutes.
>
> $$\begin{tikzcd}
& \Sigma DX \arrow[dl, equal] \arrow[d, "\Psi h"'] \arrow[dr, "\Psi' h"] & \\
\Sigma DX \arrow[d, "\Psi h"'] \arrow[r, phantom, "="] & DX \arrow[r, "\mathsf{snd}_X"] \arrow[d, "\delta_X"] & BDX \arrow[d, "B\delta_X"] \\
DX & D^2X \arrow[l, "\varepsilon_{DX} = \mathsf{fst}_{DX}"] \arrow[r, "\mathsf{snd}_{DX}"'] & BD^2X
\end{tikzcd}
\qquad
\begin{tikzcd}
& \Sigma DX \arrow[dl, equal] \arrow[d, "\Sigma\delta_X"'] \arrow[dr, "\Psi' h"] & \\
\Sigma DX \arrow[d, "\Psi h"'] & \Sigma D^2X \arrow[l, "\Sigma\varepsilon_{DX}"'] \arrow[d, "\Psi^2 h"'] \arrow[dr, "\Psi'^2 h"] & BDX \arrow[d, "B\delta_X"] \\
DX & D^2X \arrow[l, "\varepsilon_{DX} = \mathsf{fst}_{DX}"] \arrow[r, "\mathsf{snd}_{DX}"'] & BD^2X
\end{tikzcd}$$

The above shows thus that $\Psi =< \Psi, \varepsilon, \delta >$ is a functorial denotational semantics lifting $D\!=\!< D, \varepsilon, \delta >$ to the Σ-algebras. It induces a functorial operational semantics $\Psi^\#$ as follows. For every D-coalgebra structure $k : X \to DX$, the structure $\Psi^\# k : TX \to DTX$ is the inductive extension of $\Psi\mathsf{inr}_X$ along the composite $D\eta_X \circ k$:

$$\begin{tikzcd}
X \arrow[r, "\eta_X = \mathsf{inl}_X"] \arrow[d, "k"'] & TX \arrow[d, dashed, "\Psi^\# k"'] & \Sigma TX \arrow[l, "\mathsf{inr}_X"'] \arrow[d, "\Sigma\Psi^\# h"] \\
DX \arrow[r, "D\eta_X"'] & DTX & \Sigma DTX \arrow[l, "\Psi\,\mathsf{inr}_X"]
\end{tikzcd}$$

As shown in Section 7, the triple $\Psi^{\#} =<\Psi^{\#}, \eta, \mu>$ is a lifting of the syntactical monad $T{=}{<}T, \eta, \mu{>}$ to the coalgebras of the comonad D. For comparing it with the operational semantics induced by basic process algebra, one has then to translate it to a lifting to the coalgebras of the endofunctor B. For this, since B cofreely generates D, one can use the isomorphism of categories

$$\vartheta : \mathbf{Set}_D \cong \mathbf{Set}_B \qquad\qquad \langle X, k\rangle \mapsto \langle X, B\mathsf{fst}_X \circ \mathsf{snd}_X \circ k\rangle$$

illustrated in Section 7. Thus the composite

$$\vartheta\Psi^{\#}\vartheta^{-1} : TU_B \Rightarrow BTU_B$$

is of the desired form; let us check that also its 'content' is the right one:

> Consider, without loss of generality, the case $k = 0 : 0 \to B0$, that is, let k be the initial B-coalgebra structure. The isomorphism ϑ^{-1} maps it to the initial D-coalgebra structure $0 : 0 \to D0$. Write, for simplicity,
>
> $$[\![\text{-}]\!]_\Psi = \Psi^{\#}(0) : T0 \to DT0$$
>
> The claim is that, for all terms t,
>
> $$\vartheta[\![t]\!]_\Psi = [\![t]\!]$$
>
> where $[\![t]\!]$ is the operational semantics induced by the rules of basic process algebra. Indeed, omitting the subscript 0,
>
> $$\begin{array}{rcl} \vartheta[\![a\,.t]\!]_\Psi & = & (B\mathsf{fst} \circ \mathsf{snd})[\![a\,.t]\!]_\Psi \\ & = & B\mathsf{fst}(\mathsf{snd} <a\,.t, \{<a, [\![t]\!]_\Psi>\}>) \\ & = & B\mathsf{fst}\{<a, [\![t]\!]_\Psi>\} \\ & = & \{<a, \mathsf{fst}[\![t]\!]_\Psi>\} \\ & = & \{<a, t>\} \\ & = & [\![a\,.t]\!] \end{array}$$
>
> Similarly, one can see that also
>
> $$\vartheta[\![u \,\mathsf{or}\, v]\!]_\Psi = [\![u \,\mathsf{or}\, v]\!] \qquad \text{and} \qquad \vartheta[\![\mathsf{nil}]\!]_\Psi = [\![\mathsf{nil}]\!]$$

This concludes the proof that basic process algebra is functorial.

The syntax as a semi-lattice. Having established that the choice construct or of basic process algebra really behaves as the join of a semi-lattice, let us treat it as a join also in the syntax. That is, let us consider the algebras of the signature $\Sigma = \{\mathsf{nil}, a\,.\,(\text{-}), \mathsf{or}\}$ which validate the equations

$$E = \begin{cases} (x \mathbin{\mathsf{or}} y) \mathbin{\mathsf{or}} z &= x \mathbin{\mathsf{or}} (y \mathbin{\mathsf{or}} z) \\ x \mathbin{\mathsf{or}} y &= y \mathbin{\mathsf{or}} x \\ x \mathbin{\mathsf{or}} x &= x \end{cases}$$

and take for the syntactical monad for basic process algebra the monad

$$T_E = < T_E, \eta, \mu >$$

corresponding to the $\langle\Sigma, E\rangle$-algebras, rather than simply to the Σ-algebras. In other words, the monad T_E is the one arising from the standard adjunction between $\langle\Sigma, E\rangle$-algebras and sets. (See "algebras are T-algebras" in Section 2.)

For every set X, the set $T_E X$ is nothing but the quotient wrt (the congruence relation generated by) E of the free algebra of terms over X; thus one cannot distinguish in this syntax between, for instance, the terms $u \mathbin{\mathsf{or}} v$ and $v \mathbin{\mathsf{or}} u$. Keeping this quotient in mind, one can still regard the elements of $T_E X$ as terms, that is, one can use representatives rather than equivalence classes. The unit $\eta_X : X \to T_E X$ and the multiplication $\mu_X : T_E T_E X \to T_E X$ are the usual operations on variables and terms: the former is the insertion of the variables $x \in X$ into terms; the latter maps every term $t \in T_E T_E X$ containing a sub-term $u \in T_E X$ as a variable to the 'same' term $t \in T_E X$ by removing the distinction between terms and terms as variables. For instance,

$$\mu((a\,.\,t) \mathbin{\mathsf{or}} \eta_{T_E}(u \mathbin{\mathsf{or}} v)) \;=\; (a\,.\,t) \mathbin{\mathsf{or}} u \mathbin{\mathsf{or}} v$$

Now, by definition, the above denotational semantics Ψ for basic process algebra is not only a Σ-action but also a $\langle\Sigma, E\rangle$-action; that is, for every $h : \Sigma X \to X$ which validates the equations E, also $\Psi h : \Sigma D X \to D X$ validates E. In other words, Ψ is a lifting of the observational comonad D to the $\langle\Sigma, E\rangle$-algebras.

$$\begin{tikzcd} \mathbf{Set}^{\langle\Sigma,E\rangle} \arrow[r, "\Psi"] \arrow[d] & \mathbf{Set}^{\langle\Sigma,E\rangle} \arrow[d] \\ \mathbf{Set} \arrow[r, "D"'] & \mathbf{Set} \end{tikzcd}$$

Correspondingly, its operational dual $\Psi^{\#}$ can be seen as a lifting of the monad T_E to the D-coalgebras.

Next, write Φ for the operational monad on the B-coalgebras obtained by applying the isomorphism $\vartheta : \mathbf{Set}_D \cong \mathbf{Set}_B$ between D- and B-coalgebras; that is,

$$\Phi = \vartheta \Psi^{\#} \vartheta^{-1} : T_E U_B \Rightarrow B T_E U_B$$

This coaction, because of the equation $[\![u \,\mathsf{or}\, v]\!]_\Psi = [\![u]\!]_\Psi \cup [\![v]\!]_\Psi$, is join-preserving, that is, the following diagram commutes.

$$\begin{tikzcd}
T_E U_B \arrow[d, Rightarrow, "\Phi"'] & & T_E U_B \times T_E U_B \arrow[ll, Rightarrow, "\mathsf{or}"'] \arrow[d, Rightarrow, "\Phi \times \Phi"] \\
\check{\mathcal{P}}(1 + \mathsf{Act} \times T_E U_B) & & \check{\mathcal{P}}(1 + \mathsf{Act} \times T_E U_B) \times \check{\mathcal{P}}(1 + \mathsf{Act} \times T_E U_B) \arrow[ll, Rightarrow, "\cup"]
\end{tikzcd}$$

In other words, the operational semantics of basic process algebra

$$\begin{tikzcd}
& \mathbf{Set}_B \arrow[dl, bend right, "U_B"'] \arrow[dr, bend left, "U_B"] & \\
\mathbf{Set} \arrow[dr, bend right, "T_E"'] \arrow[rr, phantom, "\overset{\Phi}{\Longrightarrow}"] & & \mathbf{Set} \arrow[dl, bend left, "BT_E"] \\
& \mathbf{Set} &
\end{tikzcd}$$

takes place in the category of semi-lattices:

$$\begin{tikzcd}
& \mathbf{Set}_B \arrow[dl, bend right, "U_B"'] \arrow[dr, bend left, "U_B"] & \\
\mathbf{Set} \arrow[dr, bend right, "\langle T_E(\text{-}),\, \mathsf{or} \rangle"'] \arrow[rr, phantom, "\overset{\Phi}{\Longrightarrow}"] & & \mathbf{Set} \arrow[dl, bend left, "\langle BT_E(\text{-}), \cup \rangle"] \\
& SL(\mathbf{Set}) &
\end{tikzcd}$$

That is,

$$\Phi : \langle T_E U_B,\, \mathsf{or} \rangle \Rightarrow \langle B T_E U_B, \cup \rangle$$

The retraction for basic process algebra. One of the advantages of working with the syntax as a $\langle\Sigma, E\rangle$-algebra is that it gives a simple construction of a retraction for basic process algebra. This retraction is used in the next section to show that a certain class of operational rules (the 'GSOS' rules) is functorial.

Recall, from Section 4, the embedding of the (deterministic) behaviour $X \mapsto 1 + \mathsf{Act} \times X$ into the syntax T of the language with atomic actions and sequential composition:

$$\gamma : 1 + \mathsf{Act} \times (\text{-}) \Rightarrow T \qquad\qquad * \mapsto \mathsf{nil} \qquad\qquad <a, x> \mapsto a \,;\, x$$

The term $a\,;\,x$ behaves like the term $a\,.\,x$ of the above syntax T_E, hence one can write equivalently

$$\gamma : 1 + \mathsf{Act} \times (\text{-}) \Rightarrow T_E \qquad\qquad * \mapsto \mathsf{nil} \qquad\qquad <a, x> \mapsto a\,.\,x$$

Notice that $\check{\mathcal{P}}(1{+}\mathsf{Act}{\times}X)$ is the carrier of the free semi-lattice over the set $1{+}\mathsf{Act}{\times}X$ and that the syntax $\langle T_E X,\, \mathsf{or}\rangle$ is itself a semi-lattice. Then, by taking the left adjunct of γ wrt the standard adjunction from sets to semi-lattices

$$\begin{tikzcd}
1{+}\mathsf{Act}{\times}(\text{-}) \arrow[r, Rightarrow, "\{\text{-}\}"] \arrow[dr, Rightarrow, "\gamma"'] & B = \check{\mathcal{P}}(1{+}\mathsf{Act}{\times}\text{-}) \arrow[d, Rightarrow, "\gamma^\sharp"] \\
& T_E
\end{tikzcd}
\qquad
\begin{tikzcd}
B \arrow[d, Rightarrow, dashed, "\gamma^\sharp"'] & B \times B \arrow[l, Rightarrow, "\cup"'] \arrow[d, Rightarrow, "\gamma^\sharp \times \gamma^\sharp"] \\
T_E & T_E \times T_E \arrow[l, Rightarrow, "\mathsf{or}"]
\end{tikzcd}$$

one obtains a natural transformation

$$\gamma^\sharp : B = \check{\mathcal{P}}(1 + \mathsf{Act} \times \text{ -}) \Rightarrow T_E$$

which embeds the behaviour $BX = \check{\mathcal{P}}(1 + \mathsf{Act} \times X)$ into the above syntax T_E. That is, using the meta-variables r and s to range over objects of type B,

$$\begin{array}{lclcl}
\gamma^\sharp\{*\} & = & \gamma(*) & = & \mathsf{nil} \\
\gamma^\sharp\{<a, x>\} & = & \gamma(<a, x>) & = & a\,.\,x \\
\gamma^\sharp(r \cup s) & = & (\gamma^\sharp r)\,\mathsf{or}\,(\gamma^\sharp s) & &
\end{array}$$

Now the claim is this embedding $\gamma^\sharp$ is a retraction for basic process algebra. That is, for $\Phi =< \Phi, \eta, \mu >$ the above operational monad corresponding to basic process algebra, one has that the composite $\Phi \circ \mu_{U_B} \circ \gamma^\sharp_{T_E U_B}$ is the identity natural transformation on the functor $BT_E U_B$:

$$\Phi \circ \mu_{U_B} \circ \gamma^\sharp_{T_E U_B} = I$$

(Cf Section 4.) In order to prove this, notice that each $\gamma^\sharp_X$ is an arrow in $SL(\mathbf{Set})$, that is,

$$\gamma^\sharp : \langle B, \cup\rangle \Rightarrow \langle T_E,\, \mathsf{or}\,\rangle$$

Therefore, the composite

$$\langle BT_E U_B, \cup\rangle \stackrel{\gamma^\sharp_{T_E U_B}}{\Longrightarrow} \langle T_E^2 U_B, \mathsf{or}\rangle \stackrel{\mu_{U_B}}{\Longrightarrow} \langle T_E U_B, \mathsf{or}\rangle \stackrel{\Phi}{\Longrightarrow} \langle BT_E U_B, \cup\rangle$$

is necessarily the identity on the functor $\langle BT_E U_B, \cup\rangle : \mathbf{Set}_B \to SL(\mathbf{Set})$ because, for every set X, there exists a unique join-preserving arrow from the free semi-lattice $\langle BT_E X, \cup\rangle$ to itself which respects the unit of the monad $\check{\mathcal{P}}$. This proves that the composite $\mu_{U_B} \circ \gamma^\sharp_{T_E U_B}$ is a retraction for the operational semantics Φ induced by basic process algebra.

The above retraction can be used to give an alternative (more direct) proof of the functoriality of basic process algebra. For this, define the germ

$$\phi^{\mathcal{R}} : \Sigma BT_E \Rightarrow BT_E$$

of the operational semantics corresponding to the rules $\mathcal{R}$ of basic process algebra as follows. For r and s meta-variables ranging over objects of type BT_E,

$$\phi^{\mathcal{R}} = \begin{cases} \mathsf{nil} & \mapsto \ \{*\} \\ a\,.\,r & \mapsto \ \{<a, \gamma^\sharp r>\} \\ r \,\mathsf{or}\, s & \mapsto \ r \cup s \end{cases}$$

Formally the operational monad $\widehat{\phi^{\mathcal{R}}}$ induced by this germ $\phi^{\mathcal{R}}$ is not equal to the above operational monad Φ for basic process algebra. However, the two operational semantics are equivalent in a suitable sense

$$\Phi \sim \widehat{\phi^{\mathcal{R}}}$$

as it is shown in the next section. Here already notice that

$$\mu_{U_B} \circ \gamma^\sharp_{T_E U_B} \text{ is a retraction also for } \widehat{\phi^{\mathcal{R}}}$$

Notes. The interpretation of the non-deterministic choice as a semi-lattics join dates back at least to [HP79], where the Plotkin powerdomain is treated as the semi-lattice monad on a category of complete partial orders.

For a textbook on various non-deterministic languages for concurrency, including basic process algebra, see [BW90].

The above idea of quotienting of the terms (of basic process algebra) by an algebraic congruence for defining the programs of a language is not new: it is used, for instance, in the 'Chemical Abstract Machine' approach to operational semantics [BB92] and in some presentations of the 'π-calculus' [Mil90].

11 GSOS is Functorial

One of the largest classes of 'well-behaved' structural operational rules for transition systems is the class of '*GSOS rules*'. These are rules satisfying suitable syntactic restrictions which ensure the compositionality of the corresponding operational models. Almost all transition systems in the literature are defined by means of GSOS rules. For instance, languages like basic process algebra, *CCS*, and *CSP* have GSOS rules.

It is proved here that the operational semantics induced by a set of GSOS rules is always functorial (under the mild assumption that it embeds basic process algebra). This result shows the generality of the functorial approach to operational semantics motivating the claim that it is a first step towards a mathematical theory of 'well-behaved' operational semantics.

A GSOS rule specifies one possible transition for terms of the form $\sigma(u_1, \ldots u_l)$, for σ a given program construct of arity l:

GSOS Rule

$$\frac{\{u_i \xrightarrow{a_{ij}} v_{ij}\}^{1\leq i\leq l}_{1\leq j\leq m_i} \qquad \{u_i \stackrel{b_{ij}}{\not\longrightarrow}\}^{1\leq i\leq l}_{1\leq j\leq n_i}}{\sigma(u_1, \ldots, u_l) \xrightarrow{a} C[\overrightarrow{u}, \overrightarrow{v}]}$$

The a_{ij}'s and b_{ij}'s are actions in Act; the u_i's and v_{ij}'s are all distinct (meta) variables ranging over terms, the expression $C[\overrightarrow{u}, \overrightarrow{v}]$ is a term formed by the context $C[\overrightarrow{-}]$ and some (meta) variables contained in the set of u_i's and v_{ij}'s. The expression

$$u_i \stackrel{b_{ij}}{\not\longrightarrow}$$

stands for 'u_i cannot perform a transition with action b_{ij}'.

For instance, the rule

$$\frac{u_1 \xrightarrow{a} v_1}{u_1 \ ; u_2 \xrightarrow{a} v_1 \ ; u_2}$$

is in GSOS, as well as the rule

$$\frac{u_1 \longrightarrow * \qquad u_2 \xrightarrow{a} v_2}{u_1 \ ; u_2 \xrightarrow{a} v_2}$$

by considering that a state becomes inert $u \longrightarrow *$ (ie $u \downarrow *$) as a special case of transition $u \xrightarrow{a} v$. In this way, all rules considered so-far are GSOS.

Before setting out to prove the functoriality of GSOS, let us introduce an intermediate notation between transitions and actions of coalgebras $\langle X, k\rangle$ of the behaviour endofunctor

$$BX = \check{\mathcal{P}}(1 + \mathsf{Act} \times X)$$

Write $x \overset{k}{\rightsquigarrow} <a, y>$ for $k(x) \ni <a, y>$ and $x \overset{k}{\not\rightsquigarrow} <a, \text{-}>$ for 'there exists no y such that $<a, y>$ is in $k(x)$'. That is,

$$x \overset{k}{\rightsquigarrow} <a, y> \iff x \xrightarrow{a} y \qquad\qquad x \overset{k}{\not\rightsquigarrow} <a, \text{-}> \iff x \overset{a}{\not\longrightarrow}$$

A GSOS rule is then of the form

$$\frac{\{u_i \rightsquigarrow <a_{ij}, v_{ij}>\}^{1\le i\le l}_{1\le j\le m_i} \qquad \{u_i \not\rightsquigarrow <b_{ij}, \text{-}>\}^{1\le i\le l}_{1\le j\le n_i}}{\sigma(u_1, \ldots, u_l) \rightsquigarrow <a, C[\overrightarrow{u}, \overrightarrow{v}]>}$$

Again, one has that $u \rightsquigarrow *$ is a special case of $u \rightsquigarrow <a, u'>$.

Now, the proof of the functoriality of GSOS given here is based on the assumption that every set $\mathcal{R}$ of GSOS rules embeds the basic process algebra of the previous section. This does not seem to be a serious restriction, because most of the languages defined by means of GSOS rules do have programs behaving like nil, $a\,.\,t$, and $u \,\mathsf{or}\, v$. Therefore, let us assume that the signature Σ of the language contains the basic inert program nil, a unary action-prefixing operator for every action in Act and the binary non-deterministic choice 'or':

$$t ::= x \mid \mathsf{nil} \mid a \mid (t \,\mathsf{or}\, t) \mid \sigma(t, \ldots, t)$$

Moreover, assume that the semi-lattice laws

$$E = \begin{cases} (x \,\mathsf{or}\, y) \,\mathsf{or}\, z &= x \,\mathsf{or}\, (y \,\mathsf{or}\, z) \\ x \,\mathsf{or}\, y &= y \,\mathsf{or}\, x \\ x \,\mathsf{or}\, x &= x \end{cases}$$

for the choice construct hold. Thus, the corresponding syntactical monad

$$T = <T, \eta, \mu>$$

is the free $<\Sigma, E>$-algebra monad. (Cf Sections 2 and 10.) As a consequence, the embedding $\gamma^\sharp : B \Rightarrow T_E$ of the above behaviour into the syntax of basic process algebra extends to an embedding

$$\gamma^\sharp : B \Rightarrow T$$

into this syntax T. Since the rules $\mathcal{R}$ extend the rules of basic process algebra one also has that this embedding is a retraction for (the operational semantics induced by) $\mathcal{R}$. (Cf previous section.)

GSOS is natural. The claim is that every set $\mathcal{R}$ of GSOS rules over T containing basic process algebra can be seen as a natural transformation

$$\lceil\mathcal{R}\rceil : \Sigma B \Rightarrow BT$$

Moreover, the operational models induced by $\mathcal{R}$ and by $\lceil\mathcal{R}\rceil$ are 'observationally equivalent' in the sense that their coinductive extensions are equal.

The definition of the transformation $\lceil\mathcal{R}\rceil : \Sigma B \Rightarrow BT$ is based on the rules $\mathcal{R}$ as follows. Let the meta-variables r and s range over objects of type $B = \check{\mathcal{P}}(1{+}\mathsf{Act}\times\text{-})$. For the rules corresponding to basic process algebra, put

$$\lceil\mathcal{R}\rceil(\mathsf{nil}) = \{*\} \qquad \lceil\mathcal{R}\rceil(a\,.\,r) = \{<a, \gamma^\sharp r>\} \qquad \lceil\mathcal{R}\rceil(r \;\mathsf{or}\; s) = r \cup s$$

and, in general, for every rule

$$\frac{\{u_i \rightsquigarrow <a_{ij}, v_{ij}>\}^{1\le i\le l}_{1\le j\le m_i} \qquad \{u_i \not\rightsquigarrow <b_{ij}, \text{-}>\}^{1\le i\le l}_{1\le j\le n_i}}{\sigma(u_1,\ldots,u_l) \rightsquigarrow <a, C[\overrightarrow{u}, \overrightarrow{v}]>}$$

in $\mathcal{R}$, put

$$<a, C[\overrightarrow{\gamma_X r}, \overrightarrow{x}]> \in \lceil\mathcal{R}\rceil_X(\sigma(r_1,\ldots,r_l))$$

if $<a_{ij}, x_{ij}> \in r_i$ for $1 \le i \le l$ and $1 \le j \le m_i$, and, for every $x \in X$, $<b_{ij}, x> \notin r_i$ for $1 \le i \le l$ and $1 \le j \le n_i$. The only difference between $\mathcal{R}$ and $\lceil\mathcal{R}\rceil$ is in the use in the latter of the embedding $\gamma^\sharp : B \Rightarrow T$, which is necessary in order to plug objects of type B into the context $C[\overrightarrow{\text{-}}, \overrightarrow{x}]$. The fact that this embedding is a retraction wrt the operational semantics will ensure that this difference is observationally irrelevant.

To prove that the above definition of the arrow $\lceil\mathcal{R}\rceil_X : \Sigma BX \to BTX$ is natural in X, let us first use the following more suggestive notation.

$$\frac{\{r_i \ni <a_{ij}, x_{ij}>\}^{1\le i\le l}_{1\le j\le m_i} \qquad \{r_i \not\ni <b_{ij}, \text{-}>\}^{1\le i\le l}_{1\le j\le n_i}}{\sigma(r_1,\ldots,r_l) \overset{\lceil\mathcal{R}\rceil_X}{\rightsquigarrow} <a, C[\overrightarrow{\gamma^\sharp_X r}, \overrightarrow{x}]>}$$

The proof that this definition is natural is a simple generalization of the one given in Section 4 corresponding to the rules for the simple deterministic language used there:

> **Naturality.** The claim is that, for every 'renaming' $f : X \to Y$, the diagram
>
> $$\begin{tikzcd} \Sigma BX \arrow[r, "\Sigma Bf"] \arrow[d, "\lceil\mathcal{R}\rceil_X"'] & \Sigma BY \arrow[d, "\lceil\mathcal{R}\rceil_Y"] \\ BTX \arrow[r, "BTf"'] & BTY \end{tikzcd}$$
>
> commutes. Consider the case of negative premises: if there is no pair $< a, x >$ in $r \in BX$ for a given action a and arbitrary $x \in X$ then

there is also no pair $< a, y >$ in $(Bf)(r) \in BY$ for arbitrary $y \in Y$. Therefore, the problem of proving the naturality of $\lceil\mathcal{R}\rceil$ can be reduced to the problem of proving that the following holds.

$$\begin{array}{ccc}
\sigma(r_1,\ldots,r_l) & \xmapsto{\quad \Sigma Bf \quad} & \sigma((Bf)(r_1),\ldots,(Bf)(r_l)) \\
{\scriptstyle \lceil\mathcal{R}\rceil_X}\downarrow & & \downarrow{\scriptstyle \lceil\mathcal{R}\rceil_Y} \\
<a, C[\overrightarrow{\gamma^\sharp_X r}, \vec{x}]> & \xmapsto[BTf]{} & <a, (Tf)(C[\overrightarrow{\gamma^\sharp_X r}, \vec{x}])>=<a, C[\overrightarrow{\gamma^\sharp_Y (Bf)(r)}, \overrightarrow{fx}]>
\end{array}$$

But, again, like in Section 4, the equation

$$(Tf)(C[\overrightarrow{\gamma^\sharp_X r}, \overrightarrow{x}]) = C[\overrightarrow{\gamma^\sharp_Y (Bf)(r)}, \overrightarrow{fx}]$$

is an immediate consequence of the naturality of the retraction $\gamma^\sharp$ from B to T and of the GSOS condition that all variables in $C[\overrightarrow{u}, \overrightarrow{v}]$ are of the form u_i or v_{ij} (hence $(Tf)C[\ldots] = C[(Tf)\ldots]$).

Notice that it is very easy to violate the naturality of $\lceil\mathcal{R}\rceil$ by relaxing the assumptions on $\mathcal{R}$. For instance, one cannot drop the assumption that all meta-variables v_{ij} on the right hand side of the premises $u_i \xrightarrow{a_{ij}} v_{ij}$ are different. Indeed, one would then permit rules like

$$\frac{u_1 \xrightarrow{a} v \qquad u_2 \xrightarrow{b} v}{\sigma(u_1, u_2) \xrightarrow{c} \mathsf{nil}}$$

which fails to be natural: under the above translation $\mathcal{R} \mapsto \lceil\mathcal{R}\rceil$ and in absence of other rules for the operator σ, one has that $\sigma(<a, x_1>, <b, x_2>)$ cannot perform any transition while, by using the renaming

$$f(x_1) = y = f(x_2)$$

one has that

$$\sigma(<a, f(x_1)>, <b, f(x_2)>) \leadsto <c, \mathsf{nil}>$$

There exists however a useful extension of GSOS which is 'well-behaved' in the sense that it induces operational models which are always compositional. It is the so-called '*ntyft*'-format (see notes below) which is obtained by allowing for whole contexts C_i rather than for simple (meta) variables to appear in the left hand side of the premises of the rules:

$$\frac{\{C_i \xrightarrow{a_{ij}} v_{ij}\}^{1\le i\le l}_{1\le j\le m_i} \qquad \{C_i \stackrel{b_{ij}}{\not\longrightarrow}\}^{1\le i\le l}_{1\le j\le n_i}}{\sigma(u_1,\ldots,u_l) \xrightarrow{a} C}$$

The u_i's and v_{ij}'s are still all distinct meta-variables, but there might now appear some extra meta-variables in the contexts C and C_i. (The induction on these rules is made more problematic by the appearance of contexts also in the premises, hence some restriction (eg, 'stratification') on the use of negative premises is needed.) It is not yet clear whether these rules fit in the present functorial approach.

$\mathcal{R}$ is observationally equivalent to $\lceil\mathcal{R}\rceil$. Like in the example in Section 4, the transformation $\lceil\mathcal{R}\rceil : \Sigma B \Rightarrow BT$ can be made into the germ $\phi^{\mathcal{R}} : \Sigma BT \Rightarrow BT$ of a functorial operational semantics by composing $\lceil\mathcal{R}\rceil$ at the syntax T with the behaviour B applied to the multiplication μ of the syntax:

$$\begin{tikzcd} \Sigma BT \arrow[r, Rightarrow, "\lceil\mathcal{R}\rceil_T"] \arrow[dr, Rightarrow, "\phi^{\mathcal{R}}"'] & BT^2 \arrow[d, Rightarrow, "B\mu"] \\ & BT \end{tikzcd}$$

Spelling out the details, this germ $\phi^{\mathcal{R}} : \Sigma BT \Rightarrow BT$ is defined by 'rules'

$$\frac{\{r_i \ni \,<a_{ij}, t_{ij}>\}^{1\le i\le l}_{1\le j\le m_i} \qquad \{r_i \not\ni \,<b_{ij}, \text{-}>\}^{1\le i\le l}_{1\le j\le n_i}}{\sigma(r_1,\ldots,r_l) \overset{\phi^{\mathcal{R}}}{\rightsquigarrow} \,<a, \mu(C[\overrightarrow{\gamma^\sharp_T r}, \overrightarrow{t}])>}$$

corresponding to the rules in $\mathcal{R}$. The multiplication $\mu : T^2 \Rightarrow T$ is formally needed in order to remove bracketing and make of the term

$$C[\overrightarrow{\gamma^\sharp_T r}, \overrightarrow{t}]$$

(with as *variables* the *terms* $\gamma^\sharp_T r_i$ and t_{ij}) a simpler term with the variables of $\gamma^\sharp_T r_i$ and t_{ij} as variables. In the sequel, for simplicity, μ is omitted.

For every set X, the function $\phi^{\mathcal{R}}_X : \Sigma BTX \to BTX$ is not only a Σ- but also a $\langle\Sigma, E\rangle$-algebra structure for $\langle BTX, \cup\rangle$. That is, $\phi^{\mathcal{R}}$ is join-preserving. Therefore, by the isomorphism between $\langle\Sigma, E\rangle$- and T-algebras (cf Section 2) it can be seen as an action of the monad T on the composite functor BT:

$$\phi^{\mathcal{R}} : TBT \Rightarrow BT$$

Then, for every coalgebra structure $k : X \to BX$, the germ $\phi^{\mathcal{R}}$ induces an operational model

$$[\![\text{-}]\!]^k_{\lceil\mathcal{R}\rceil} : TX \to BTX$$

by taking the left adjunct of the composite arrow $B\eta_X \circ k : X \to BTX = U^T\langle BTX, \phi^{\mathcal{R}}_X\rangle$ wrt the standard adjunction $F^T \dashv U^T$ from the T-algebras to their carriers

$$\begin{tikzcd} X \arrow[r, "\eta_X"] \arrow[d, "k"'] & TX \arrow[d, "{[\![\text{-}]\!]^k_{\lceil\mathcal{R}\rceil}}"'] \\ BX \arrow[r, "B\eta_X"'] & BTX \end{tikzcd} \qquad \begin{tikzcd} TX \arrow[d, dashed, "{[\![\text{-}]\!]^k_{\lceil\mathcal{R}\rceil} = (B\eta_X \circ k)^\sharp}"'] & T^2X \arrow[l, "\mu_X"'] \arrow[d, "{T[\![\text{-}]\!]^k_{\lceil\mathcal{R}\rceil}}"] \\ BTX & TBTX \arrow[l, "\phi^{\mathcal{R}}_X"] \end{tikzcd}$$

(See Section 2.)

Regarding the coalgebra structure $k : X \to BX$ as a set of transitions $x \xrightarrow{a} x'$, with $x, x' \in X$, one can also take the least transition system induced by these transitions and by the rules in $\mathcal{R}$ and obtain another operational model

$$[\![\text{-}]\!]^{k}_{\mathcal{R}} : TX \to BTX$$

> The claim is that these two operational models are *observationally equivalent* in the sense that their coinductive extensions are the same; in other words, they have the same final coalgebra semantics.

Without loss of generality, let us prove this claim taking for k the 'empty' coalgebra structure $0 : 0 \to B0$ as the base of the induction. Correspondingly, one has the models

$$[\![\text{-}]\!]_{\mathcal{R}} : T0 \to BT0 \qquad \text{and} \qquad [\![\text{-}]\!]_{\lceil\mathcal{R}\rceil} : T0 \to BT0$$

with the set $T0$ of closed terms as carrier. The claim is that, for all closed terms t,

$$[\![t]\!]^{@}_{\lceil\mathcal{R}\rceil} = [\![t]\!]^{@}_{\mathcal{R}}$$

Diagrammatically:

$$\begin{tikzcd}
 & \widehat{B} \arrow[d, "\varphi"', "\text{final coalgebra}"] \arrow[d, phantom, shift right=3ex, "\cong"] \\
T0 \arrow[ur, shift left, "{[\![\text{-}]\!]^{@}_{\mathcal{R}}}"] \arrow[ur, shift right, "{[\![\text{-}]\!]^{@}_{\lceil\mathcal{R}\rceil}}"'] \arrow[ur, phantom, "="] \arrow[d, shift right, "{[\![\text{-}]\!]_{\mathcal{R}}}"'] \arrow[d, shift left, "{[\![\text{-}]\!]_{\lceil\mathcal{R}\rceil}}"] & B\widehat{B} \\
BT0 \arrow[ur, shift left, "{B[\![\text{-}]\!]^{@}_{\mathcal{R}}}"] \arrow[ur, shift right, "{B[\![\text{-}]\!]^{@}_{\lceil\mathcal{R}\rceil}}"'] &
\end{tikzcd}$$

where, recall, $BX = \check{\mathcal{P}}(1 + \mathsf{Act} \times X)$ and the final B-coalgebra $\langle \widehat{B}, \varphi \rangle$ is described in Section 13.

First notice that

$$[\![t]\!]_{\mathcal{R}} \rightsquigarrow * \iff [\![t]\!]_{\lceil\mathcal{R}\rceil} \rightsquigarrow *$$

Thus, consider, without loss of generality, only the case when t might not become inert. Then, the functions $[\![\text{-}]\!]^{@}_{\mathcal{R}}$ and $[\![\text{-}]\!]^{@}_{\lceil\mathcal{R}\rceil}$ are the *unique* functions which, for all t, satisfy the coinductive definitions

$$[\![t]\!]^{@}_{\mathcal{R}} \;=\; \varphi^{-1}\{<a, [\![t']\!]^{@}_{\mathcal{R}}> \mid [\![t]\!]_{\mathcal{R}} \rightsquigarrow <a, t'>\}$$

and

$$[\![t]\!]^{@}_{\lceil\mathcal{R}\rceil} \;=\; \varphi^{-1}\{<a, [\![t']\!]^{@}_{\lceil\mathcal{R}\rceil}> \mid [\![t]\!]_{\lceil\mathcal{R}\rceil} \rightsquigarrow <a, t'>\}$$

respectively, for φ^{-1} the inverse of the final coalgebra isomorphism $\varphi : \widehat{B} \cong B\widehat{B}$. If one can show that, for all terms t, the identity

$$\{<a, [\![t']\!]^{@}_{\lceil\mathcal{R}\rceil}> \mid [\![t]\!]_{\lceil\mathcal{R}\rceil} \rightsquigarrow <a, t'>\} \;=\; \{<a, [\![t']\!]^{@}_{\lceil\mathcal{R}\rceil}> \mid [\![t]\!]_{\mathcal{R}} \rightsquigarrow <a, t'>\} \qquad (3)$$

holds, then one has that, for all t, both

$$[\![t]\!]^{@}_{\lceil\mathcal{R}\rceil} = \varphi^{-1}\{<a, [\![t']\!]^{@}_{\lceil\mathcal{R}\rceil}> \mid [\![t]\!]_{\mathcal{R}} \leadsto <a, t'>\}$$

and

$$[\![t]\!]^{@}_{\mathcal{R}} = \varphi^{-1}\{<a, [\![t']\!]^{@}_{\mathcal{R}}> \mid [\![t]\!]_{\mathcal{R}} \leadsto <a, t'>\}$$

which, by the uniqueness of coinductive extensions, implies that they are the same.

Now, the identity (3) can be proved as follows. Notice that, by definition of $\lceil\mathcal{R}\rceil$,

$$[\![t]\!]_{\lceil\mathcal{R}\rceil} \leadsto <a, t'> \iff [\![t]\!]_{\mathcal{R}} \leadsto <a, C[\overrightarrow{u}, \overrightarrow{v}]> \text{ and } t' = C[\overrightarrow{\gamma^{\sharp}_{T0}[\![u]\!]_{\lceil\mathcal{R}\rceil}}, \overrightarrow{v}]$$

It suffices then to show that

$$[\![C[\overrightarrow{\gamma^{\sharp}_{T0}[\![u]\!]_{\lceil\mathcal{R}\rceil}}, \overrightarrow{v}]]\!]^{@}_{\lceil\mathcal{R}\rceil} = [\![C[\overrightarrow{u}, \overrightarrow{v}]]\!]^{@}_{\lceil\mathcal{R}\rceil}$$

For this, one can use the compositionality of functorial operational semantics (the abstract semantics of a term is invariant under substitution of sub-terms with the same abstract semantics) and reduce it to the problem of proving that

$$[\![\gamma^{\sharp}_{T0}[\![u_i]\!]_{\lceil\mathcal{R}\rceil}]\!]^{@}_{\lceil\mathcal{R}\rceil} = [\![u_i]\!]^{@}_{\lceil\mathcal{R}\rceil}$$

holds. This, in turn, is a consequence of the fact that $\gamma^{\sharp}$ is a retraction for $\lceil\mathcal{R}\rceil$, ie $[\![\gamma^{\sharp}_{T0} r]\!]_{\lceil\mathcal{R}\rceil} = r$ (see previous section):

$$\begin{array}{rcll}
[\![u_i]\!]^{@}_{\lceil\mathcal{R}\rceil} & = & \varphi^{-1} \circ B[\![\text{-}]\!]^{@}_{\lceil\mathcal{R}\rceil} \circ [\![u_i]\!]_{\lceil\mathcal{R}\rceil} & \text{(unfold)} \\
& = & \varphi^{-1} \circ B[\![\text{-}]\!]^{@}_{\lceil\mathcal{R}\rceil} \circ [\![\gamma^{\sharp}[\![u_i]\!]_{\lceil\mathcal{R}\rceil}]\!]_{\lceil\mathcal{R}\rceil} & \text{(retraction)} \\
& = & [\![\gamma^{\sharp}_{T0}[\![u_i]\!]_{\lceil\mathcal{R}\rceil}]\!]^{@}_{\lceil\mathcal{R}\rceil} & \text{(fold)}
\end{array}$$

This concludes the proof.

Structural Coinduction. A more direct way of proving that the set $[\![t]\!]^{@}_{\mathcal{R}}$ is equal to the set $[\![t]\!]^{@}_{\lceil\mathcal{R}\rceil}$ would be to prove that the two sets are equal under the *coinductive* hypothesis that the $[\![t']\!]^{@}_{\mathcal{R}}$'s are equal to the $[\![t']\!]^{@}_{\lceil\mathcal{R}\rceil}$'s. Intuitively, this principle holds by duality wrt the structural induction principle, the algebraic structure of the program constructs being here replaced by the coalgebraic structure of the behaviour observations. However, a formal foundation for this particular 'structural coinduction principle' is still to be investigated.

Guarded Recursion in GSOS. Recall from Section 5 that every set of terms (mutually) recursively defined by means of equations in some variables $x_i \in X$

$$x_1 = t_1[X],\ x_2 = t_2[X], \dots$$

where $t_i[X]$ are elements of TX (hence might contain variables from X), can be seen as a T-coalgebra $k : X \to TX$ by putting $k(x_i) = t_i[X]$. (And vice versa.) Also recall that a system of (mutually) recursive definitions $k : X \to TX$ is *guarded* if it factorizes through a coalgebra

$$g : X \to BTX = \check{\mathcal{P}}(1 + \mathsf{Act} \times TX)$$

of the composite endofunctor BT in the sense that

$$\begin{tikzcd} X \arrow[rr, "k"] \arrow[d, "g"'] & & TX \\ BTX \arrow[r, "\gamma^\sharp_{TX}"'] & T^2X \arrow[ur, "\mu_X"'] & \end{tikzcd}$$

commutes, that is, $k = \mu_X \circ \gamma^\sharp_{TX} \circ g : X \to TX$, where $\mu : T^2 \Rightarrow T$ is the multiplication of the syntactical monad T (cf Section 2) and $\gamma^\sharp : B \Rightarrow T$ is the retraction for basic process algebra. Clearly:

$$g(x_i) = \{<a_{i_1}, t_{i_1}>, \dots, <a_{i_n}, t_{i_n}>\}$$

that is, the equations $x_i = t_i$ are guarded if they are of the form

$$x_i = (a_{i_1} . t_{i_1})\ \mathsf{or}\ \dots\ \mathsf{or}\ (a_{i_n} . t_{i_n})$$

Conversely, every BT-coalgebra can be seen as a set of mutually recursive definitions.

Now, for every set $\mathcal{R}$ of GSOS rules, one can take the left adjunct of every $g : X \to BTX = U^T\langle BTX, \phi^{\mathcal{R}}_X\rangle$ wrt the adjunction $F^T \dashv U^T$ from the T-algebras to their carriers:

$$\begin{tikzcd} X \arrow[r, "\eta_X"] \arrow[dr, "g"'] & TX \arrow[d, "{[\![\text{-}]\!]^g_{\lceil\mathcal{R}\rceil}}"] \\ & BTX \end{tikzcd} \qquad \begin{tikzcd} TX \arrow[d, dashed, "{[\![\text{-}]\!]^g_{\lceil\mathcal{R}\rceil} = g^\sharp}"'] & T^2X \arrow[l, "\mu_X"'] \arrow[d, "{T[\![\text{-}]\!]^g_{\lceil\mathcal{R}\rceil}}"] \\ BTX & TBTX \arrow[l, "\phi^{\mathcal{R}}_X"] \end{tikzcd}$$

Then, the desired interpretation of g as a recursive process is obtained by taking the corresponding final coalgebra semantics $([\![\text{-}]\!]^g_{\lceil\mathcal{R}\rceil})^@ = ([\![\text{-}]\!]^g_{\mathcal{R}})^@ : TX \to \widehat{B}$ precomposed with the insertion-of-variables $\eta_X : X \to TX$. With the usual abuse of notation, write $g^{@} : X \to \widehat{B}$ for this composite arrow:

$$\begin{tikzcd}
X \arrow[r, "\eta_X"] \arrow[dr, "g"'] \arrow[rr, bend left, "g^{@}"] & TX \arrow[r, dashed, "(\llbracket\text{-}\rrbracket^g_{\mathcal{R}})^{@}"] \arrow[d, "\llbracket\text{-}\rrbracket^g_{\lceil\mathcal{R}\rceil}"] & \widehat{B} \arrow[d, "\cong"] \\
& BTX \arrow[r, "B(\llbracket\text{-}\rrbracket^g_{\mathcal{R}})^{@}"'] & B\widehat{B}
\end{tikzcd}$$

Notice that no variable binding operator (like, eg, the operator “fix” in the original definition of GSOS) is needed here to deal with recursion.

As an example, let $\mathcal{R}$ be basic process algebra together with the rules for (simple) *interleaving*

$$\frac{u_1 \xrightarrow{a} v_1}{u_1 \parallel u_2 \xrightarrow{a} v_1 \parallel u_2} \qquad \frac{u_2 \xrightarrow{a} v_2}{u_1 \parallel u_2 \xrightarrow{a} u_1 \parallel v_2}$$

and let g be the BT-coalgebra corresponding to the guarded recursive definition

$$x = a\,.\,x \qquad y = (a\,.\,y) \;\mathsf{or}\; (b\,.\,x) \qquad z = (a\,.\,z) \;\mathsf{or}\; (b\,.\,(x \parallel y))$$

in $X = \{x, y, z\}$. Write, for simplicity,

$$\llbracket\text{-}\rrbracket_g = \llbracket\text{-}\rrbracket^g_{\lceil\mathcal{R}\rceil} : TX \to BTX$$

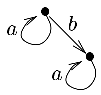
and, correspondingly, let

$$\llbracket\text{-}\rrbracket_g^{@} = (\llbracket\text{-}\rrbracket^g_{\lceil\mathcal{R}\rceil})^{@} = (\llbracket\text{-}\rrbracket^g_{\mathcal{R}})^{@} : TX \to \widehat{B}$$

be its coinductive extension. Then, omitting, as usual, the insertion-of-variables $\eta_X : X \to TX$ and the final coalgebra isomorphism $\widehat{B} \cong B\widehat{B}$,

$$\begin{aligned}
\llbracket a\,.\,t\rrbracket_g^{@} &= \{<a, \llbracket\gamma^{\sharp}_{TX}\llbracket t\rrbracket_g\rrbracket_g^{@}>\} \;=\; \{<a, \llbracket t\rrbracket_g^{@}>\} \\
\llbracket t_1 \;\mathsf{or}\; t_2\rrbracket_g^{@} &= \llbracket t_1\rrbracket_g^{@} \cup \llbracket t_2\rrbracket_g^{@} \\
\llbracket t_1 \parallel t_2\rrbracket_g^{@} &= \{<a, \llbracket t_1' \parallel t_2\rrbracket_g^{@}>\mid t_1 \xrightarrow{a} t_1'\} \cup \{<a, \llbracket t_1 \parallel t_2'\rrbracket_g^{@}>\mid t_2 \xrightarrow{a} t_2'\} \\
g^{@}(x) &= \{<a, g^{@}(x)>\} \;=
\end{aligned}$$

$$g^{@}(y) = \{<a, g^{@}(y)>\} \cup \{<b, g^{@}(x)>\} \;=$$



$$g^{@}(z) = \{<a, g^{@}(z)>\} \cup \{<b, \llbracket x \parallel y\rrbracket_g^{@}>\}$$

GSOS models are Φ-models. The functoriality of GSOS gives a systematic method for deriving an adequate denotational model from any set $\mathcal{R}$ of GSOS rules. Another systematic method proposed in the literature (see notes below) permits to derive a *proof system* from any set $\mathcal{R}$ of GSOS rules. This proof system can be used for proving that the programs of the language of $\mathcal{R}$ satisfy assertions in *Hennessy-Milner* logic.

The main result on this proof system is that it is complete wrt a certain class of '*models*' of $\mathcal{R}$. The problem arises then of finding an independent motivation for the definition of GSOS models. It is here shown that the models of a set $\mathcal{R}$ of GSOS rules are exactly the algebras of the operational monad Φ induced by the rules $\mathcal{R}$. This supports the choice of that class of models as the 'natural' one.

A **model for a set of GSOS rules** $\mathcal{R}$ is a triple $\langle X, h, k\rangle$ with $h : TX \to X$ an algebra of the syntactical monad $T = <T, \eta, \mu>$ corresponding to $\mathcal{R}$ and $k : X \to BX = \check{\mathcal{P}}(1 + \mathsf{Act} \times X)$ a B-coalgebra structure such that

$$\sigma(x_1, \ldots, x_l) \stackrel{h}{\longmapsto} x \stackrel{k}{\rightsquigarrow} <a, x'>$$

holds if and only if there exists a rule

$$\frac{\{u_i \rightsquigarrow <a_{ij}, v_{ij}>\}^{1\le i\le l}_{1\le j\le m_i} \qquad \{u_i \not\rightsquigarrow <b_{ij}, \text{-}>\}^{1\le i\le l}_{1\le j\le n_i}}{\sigma(u_1, \ldots, u_l) \rightsquigarrow <a, C[\overrightarrow{u}, \overrightarrow{v}]>}$$

in $\mathcal{R}$ such that

$$x_i \stackrel{k}{\rightsquigarrow} <a_{ij}, y_{ij}> \qquad x_i \stackrel{k}{\not\rightsquigarrow} <b_{ij}, \text{-}> \qquad C[\overrightarrow{x}, \overrightarrow{y}] \stackrel{h}{\longmapsto} x'$$

(Formally, this definition is obtained from the original definition of GSOS models by using the one-to-one correspondences between $\langle\Sigma, E\rangle$- and T-algebras and (finitely branching) transition systems and B-coalgebras.)

Next, let $\Phi = <\Phi, \eta, \mu>$ be the operational monad induced by a set of GSOS rules $\mathcal{R}$. That is,

$$\begin{tikzcd} X \arrow[r, "\eta_X"] \arrow[d, "k"'] & TX \arrow[d, "\Phi k"] \\ BX \arrow[r, "B\eta_X"'] & BTX \end{tikzcd} \qquad \begin{tikzcd} TX \arrow[d, dashed, "\Phi k"'] & T^2X \arrow[l, "\mu_X"'] \arrow[d, "T\Phi k"] \\ BTX & TBTX \arrow[l, "\phi^{\mathcal{R}}_X"] \end{tikzcd}$$

Recall, from Section 9, that an algebra of the monad Φ is a triple $<X, h, k>$, with

h a T-algebra and k a B-coalgebra structure over the set X such that the diagram

$$\begin{tikzcd} TX \arrow[r, "\Phi k"] \arrow[d, "h"'] & BTX \arrow[d, "Bh"] \\ X \arrow[r, "k"'] & BX \end{tikzcd}$$

commutes. But this means that

$$h(\sigma(x_1, \ldots, x_l)) \overset{k}{\leadsto} <a, x'>$$

holds if and only if

$$\sigma(x_1, \ldots, x_l) \overset{\Phi k}{\leadsto} <a, t> \qquad \text{and} \qquad t \overset{h}{\longmapsto} x'$$

In turn, by definition of Φ, the latter holds if and only if there exists a rule

$$\frac{\{u_i \leadsto <a_{ij}, v_{ij}>\}^{1 \le i \le l}_{1 \le j \le m_i} \qquad \{u_i \not\leadsto <b_{ij}, \text{-}>\}^{1 \le i \le l}_{1 \le j \le n_i}}{\sigma(u_1, \ldots, u_l) \leadsto <a, C[\overrightarrow{u}, \overrightarrow{v}]>}$$

in $\mathcal{R}$ such that

$$x_i \overset{k}{\leadsto} <a_{ij}, y_{ij}> \qquad x_i \overset{k}{\not\leadsto} <b_{ij}, \text{-}> \qquad t = C[\overrightarrow{x}, \overrightarrow{y}]$$

Since $ht = x'$ this proves that every GSOS model is a Φ-algebra, and vice versa. In Section 9, Φ-algebras are also called Φ-models, hence this result can be rephrased formally as *GSOS models are* Φ*-models.*

Notes. The notion of a GSOS model has been introduced in [Sim95]. The GSOS rules have been defined in [BIM88], considerably extending a previous definition of 'well-behaved' rules from [dS85]. More recent proposals are the *tyft* format [GV92] extending GSOS without negative premises and its subsequent *ntyft* format [Gro93] mentioned above. It would be interesting to understand whether the functorial approach can deal also with these latter formats.

12 Coalgebraic Bisimulations

There are several notions of *observational equivalence* for a transition system; the most general one corresponds to a relation on its states called *(strong) bisimulation*. The final coalgebra of the behaviour corresponding to transition systems 'classifies' bisimilar states in the sense that two states are bisimilar if and only if they have the same final coalgebra semantics, ie the same abstract global behaviour. In other words, coinduction can be 'pulled back' to bisimulation. As a corollary, the final coalgebra is 'internally fully abstract'.

Categorically, this can be generalized to every behaviour functor B preserving 'weak pullbacks'.

Recall from Section 10 the correspondence between (finitely) non-deterministic transition systems and coalgebras of the behaviour endofunctor

$$BX = \check{\mathcal{P}}(1 + \mathsf{Act} \times X)$$

Recall also the notation $x \overset{k}{\leadsto} < a, x' >$ introduced in Section 11 to express that $< a, x' > \in k(x)$ in a coalgebra structure $k : X \to \check{\mathcal{P}}(1 + \mathsf{Act} \times X)$; in other words, the transition system corresponding to the coalgebra $\langle X, k\rangle$ can perform the transition $x \xrightarrow{a} x'$.

A relation R between the carriers X and Y of two coalgebras $\langle X, k\rangle$ and $\langle Y, \ell\rangle$ **lifts to a (strong) bisimulation** between the two coalgebras when, for all x in X and y in Y such that xRy (ie $< x, y > \in R$), the following three conditions are satisfied.

1. $x \overset{k}{\leadsto} *$ if and only if $y \overset{\ell}{\leadsto} *$
2. if $x \overset{k}{\leadsto} < a, x' >$ then $y \overset{\ell}{\leadsto} < a, y' >$ for some y' such that $x' R\, y'$
3. and, conversely, if $y \overset{\ell}{\leadsto} < a, y' >$ then $x \overset{k}{\leadsto} < a, x' >$ for some x' such that $x' R\, y'$

Notice that bisimulations are themselves coalgebras. Indeed, from the above conditions, one can define a coalgebra structure

$$\widetilde{R} : R \to \check{\mathcal{P}}(1 + \mathsf{Act} \times R)$$

on the relation R by putting

$$xRy \overset{\widetilde{R}}{\leadsto} * \iff x \overset{k}{\leadsto} * \; (\iff y \overset{\ell}{\leadsto} * \,)$$

and

$$xRy \overset{\widetilde{R}}{\rightsquigarrow} \,<a, <x', y'>> \iff x \overset{k}{\rightsquigarrow} \,<a, x'> \qquad y \overset{\ell}{\rightsquigarrow} \,<a, y'> \qquad x'Ry'$$

In the sequel, the above notion of bisimulation is also called **ordinary bisimulations**, in order to distinguish it from the following more general notion of 'coalgebraic bisimulation'.

Bisimulations are coalgebras; now the question is: *Is there a coalgebraic description of bisimulation?* For this, consider the two '**legs**' $r_1 : R \to X$ and $r_2 : R \to Y$ obtained by composing the insertion $R \hookrightarrow X \times Y$ of the relation R into the cartesian product $X \times Y$ with the first and second projection, respectively. Now, if the relation R lifts to an ordinary bisimulation, then its legs r_1 and r_2 lift to coalgebra arrows; that is, the two squares in

$$\begin{tikzcd}
 & R \arrow[dl, "r_1"'] \arrow[dr, "r_2"] \arrow[dd, "\widetilde{R}"] & \\
X \arrow[dd, "k"'] & & Y \arrow[dd, "\ell"] \\
 & BR \arrow[dl, "Br_1"] \arrow[dr, "Br_2"'] & \\
BX & & BY
\end{tikzcd}$$

commute. The converse is also true; namely, if a relation lifts to a coalgebra of the above behaviour endofunctor B in a way that its legs also lift to corresponding coalgebra arrows as in the above diagram, then this relation is a bisimulation. Indeed, the first condition is obvious, while the second and the third follow from the commutativity of the left and the right diagram, respectively. Notice that there might be more structures $\widetilde{R}$ making the above diagram commute, corresponding to the several ways in which, in general, a relation can lift to a bisimulation.

The above diagram can be defined wrt any endofunctor B. Call the corresponding notion **coalgebraic bisimulation**. It applies also to endofunctors on categories other than **Set**, by taking a **relation** between two objects X and Y **in a category C** to be a span

$$\begin{tikzcd}
 & R \arrow[dl, "r_1"'] \arrow[dr, "r_2"] & \\
X & & Y
\end{tikzcd}$$

which is monic in the sense that the two legs are **jointly monic** in **C**; that is, if f and g are two 'parallel' arrows such that

$$r_1 \circ f = r_1 \circ g \quad \text{and} \quad r_2 \circ f = r_2 \circ g$$

then f is equal to g. (To be precise, a monic span is not a relation, but just one representative of an equivalence class (of monic spans) which forms the actual relation – more details below.) Then, a relation R between the carriers X and Y

of two coalgebras $\langle X, k\rangle$ and $\langle Y, \ell\rangle$ of an endofunctor B on $\mathbf{C}$ lifts to a coalgebraic bisimulation if there exists a B-coalgebra structure $\widetilde{R} : R \to BR$ which makes

$$\begin{tikzcd}
 & R \arrow[dl, "r_1"'] \arrow[d, "\widetilde{R}"] \arrow[dr, "r_2"] & \\
X \arrow[d, "k"'] & BR \arrow[dl, "Br_1"] \arrow[dr, "Br_2"'] & Y \arrow[d, "\ell"] \\
BX & & BY
\end{tikzcd}$$

commute. Notice the stress is put on the fact that the legs of the relation R *lift* to coalgebra arrows, rather than on the actual (possibly not unique) coalgebraic structure of R. Therefore, let us *forget* about the coalgebraic structure of R and write

$$\begin{tikzcd}
 & R \arrow[dl, "r_1"'] \arrow[dr, "r_2"] \arrow[d, phantom, "\sim" near start] & \\
\langle X, k\rangle & & \langle Y, \ell\rangle
\end{tikzcd}$$

to express that R is a relation between the carriers X and Y which lifts to a bisimulation between the coalgebras $\langle X, k\rangle$ and $\langle Y, \ell\rangle$.

A canonical way of defining relations is by *pullbacks*: for any diagram

$$\begin{tikzcd}
X \arrow[dr, "f"'] & & Y \arrow[dl, "g"] \\
 & Z &
\end{tikzcd}$$

the two legs of the corresponding pullback

$$\begin{tikzcd}
 & R \arrow[dl, "r_1"'] \arrow[dr, "r_2"] \arrow[dd, phantom, "\lrcorner" rotate=-45, very near start] & \\
X \arrow[dr, "f"'] & & Y \arrow[dl, "g"] \\
 & Z &
\end{tikzcd}$$

are jointly monic (by the universal property of the pullback). For instance, in **Set**, the pullback of two functions f and g is the relation $\{<x, y> \mid fx = gy\}$. Another example is given by the equality relation: the pullback

$$\begin{tikzcd}
 & \mathrm{EQ}(X) \arrow[dl, "e_1"'] \arrow[dr, "e_2"] \arrow[dd, phantom, "\lrcorner" rotate=-45, very near start] & \\
X \arrow[dr, equal] & & X \arrow[dl, equal] \\
 & X &
\end{tikzcd}$$

is the **equality relation on the object** X in a category $\mathbf{C}$ with pullbacks.

> **The equality relation always lifts to a coalgebraic bisimulation.** Firstly, notice that the two legs e_1 and e_2 of the equality are the same. Next, consider the 'diagonal' $\mathsf{d}_X : X \to \mathrm{EQ}(X)$
>
> $$\begin{tikzcd} & X \arrow[d, dashed, "\mathsf{d}_X"] \arrow[ddl, bend right, "\mathsf{id}_X"'] \arrow[ddr, bend left, "\mathsf{id}_X"] & \\ & \mathrm{EQ}(X) \arrow[dl, "e_1"'] \arrow[dr, "e_2"] & \\ X \arrow[dr, equal] & & X \arrow[dl, equal] \\ & X & \end{tikzcd}$$
>
> given by the universal property of $\mathrm{EQ}(X)$. (In **Set**, the value of the diagonal d_X at an element x of X is the pair $< x, x >$.) For any endofunctor B and any B-coalgebra $\langle X, k\rangle$ since the composite $e_i \circ \mathsf{d}_X$ is the identity on X, the diagram
>
> $$\begin{tikzcd} \mathrm{EQ}(X) \arrow[r, "e_i"] \arrow[d, "e_i"'] & X \arrow[r, "k"] & BX \arrow[r, "B\mathsf{d}_X"] & B(\mathrm{EQ}(X)) \arrow[d, "Be_i"] \\ X \arrow[rrr, "k"'] & & & BX \end{tikzcd}$$
>
> commutes; hence, the composite $B\mathsf{d}_X \circ k \circ e_i$ lifts the equality $\mathrm{EQ}(X)$ to a bisimulation on the coalgebra $\langle X, k\rangle$:
>
> $$\begin{tikzcd} & \mathrm{EQ}(X) \arrow[dl, "e_1"'] \arrow[dr, "e_2"] \arrow[d, phantom, "\sim"] & \\ \langle X, k\rangle & & \langle X, k\rangle \end{tikzcd}$$

Next, let B be an endofunctor on a category $\mathbf{C}$ with pullbacks. Recall that pullbacks, like all universals, are determined by two conditions: *uniqueness* and *existence*. When only the existence part is known to hold one speaks of a **weak pullback** (and of a **weak universal** in general). Now, not all pullbacks lift to B-bisimulations, but a sufficient condition is that the functor B **preserves weak pullbacks**. That is, if the image under B of a weak pullback is still a weak pullback, then every pullback in $\mathbf{C}$ of arrows which are coalgebra homomorphisms lifts to a B-bisimulation. Indeed, since pullbacks are also weak pullbacks, for all $f : \langle X, k\rangle \to \langle Z, j\rangle$ and $g : \langle Y, \ell\rangle \to$

$\langle Z, j\rangle$ in $\mathbf{C}_B$, the *existence* of a (possibly not unique) suitable coalgebra structure $\widetilde{R} : R \to BR$ for the pullback R of f and g in $\mathbf{C}$ is ensured by the weakly universal property of the weak pullback BR:

$$\begin{tikzcd}
 & R \arrow[dl, "r_1"'] \arrow[dr, "r_2"] \arrow[rrrr, dotted, "\widetilde{R}"] & & & & BR \arrow[dl, "Br_1"'] \arrow[dr, "Br_2"] & \\
X \arrow[dr, "f"'] \arrow[rrrr, bend left, "k"] & & Y \arrow[dl, "g"] \arrow[rrrr, bend right, "\ell"] & & BX \arrow[dr, "Bf"'] & & BY \arrow[dl, "Bg"] \\
 & Z \arrow[rrrr, "j"'] & & & & BZ &
\end{tikzcd}$$

(The coalgebra structures k and ℓ turn the legs of R into a cone over the diagram for which BR is a weak pullback.)

Pullbacks lift to ordinary bisimulations. Let us check that the behaviour functor $BX = \check{\mathcal{P}}(1 + \mathsf{Act} \times X)$ preserves weak pullbacks and hence, by the above argument, pullbacks lift to (ordinary) bisimulations.

Let us consider the functor $BX = \check{\mathcal{P}}(\mathsf{Act} \times X)$; the proof carries over trivially to the case $BX = \check{\mathcal{P}}(1 + \mathsf{Act} \times X)$. The problem of showing that the functor B preserves weak pullbacks can be reduced to the problem of showing that B maps (ordinary) pullbacks to weak pullbacks. Indeed, the following holds.

In Set, weak pullbacks embed pullbacks. That is, the diagram

$$\begin{tikzcd}
W \arrow[r, "w_2"] \arrow[d, "w_1"'] & Y \arrow[d, "g"] \\
X \arrow[r, "f"'] & Z
\end{tikzcd}$$

is a weak pullback diagram if and only if there exists an injection $m : R \rightarrowtail W$ of the pullback $R = \{<x, y>\mid fx = gy\}$ of f and g into W such that

$$\begin{tikzcd}
R \arrow[dr, tail, "m"] \arrow[drr, bend left, "r_2"] \arrow[ddr, bend right, "r_1"'] & & \\
 & W \arrow[r, "w_2"] \arrow[d, "w_1"] & Y \arrow[d, "g"] \\
 & X \arrow[r, "f"'] & Z
\end{tikzcd}$$

commutes.

Therefore, if

$$BR = \{<a, x, y>\mid a \in \mathsf{Act},\ fx = gy\}$$

is a weak pullback for $Bf : BX \to BZ$ and $Bg : BY \to BZ$, the set BW inherits the weak universality of BR by means of the mediating arrow $Bm : BR \to BW$.

In turn, in order to prove that BR is a weak pullback for Bf and Bg it suffices to prove that the (ordinary) pullback R' of Bf and Bg factorizes through it in the sense that there exists a function $h : R' \to BR$ such that $r'_i = Br_i \circ h$:

$$\begin{tikzcd}
R' \arrow[dr, dotted, "h"] \arrow[drr, bend left, "r'_2"] \arrow[ddr, bend right, "r'_1"'] & & \\
& BR \arrow[r, "Br_2"] \arrow[d, "Br_1"] & BY \arrow[d, "Bg"] \\
& BX \arrow[r, "Bf"'] & BZ
\end{tikzcd}$$

Indeed, then every other cone (f', g') over the co-span $\langle Bf, Bg \rangle$ factorizes through the pullback as follows.

$$\begin{tikzcd}
\cdot \arrow[dr, dashed] \arrow[ddrr, bend left, "g'"] \arrow[dddr, bend right, "f'"'] & & & \\
& R' \arrow[dr, dotted, "h"] \arrow[drr, bend left, "r'_2"] \arrow[ddr, bend right, "r'_1"'] & & \\
& & BR \arrow[r, "Br_2"] \arrow[d, "Br_1"] & BY \arrow[d, "Bg"] \\
& & BX \arrow[r, "Bf"'] & BZ
\end{tikzcd}$$

Let us now try and define such a function $h : R' \to BR$ from the pullback R' of Bf and Bg to the image under B of the pullback R of f and g. By definition of pullbacks in **Set**, the set R' is made of those pairs

$$<\{<a_i, x_i>\}_{i\in I}, \{<a_j, y_j>\}_{j\in J}>$$

such that the index sets I and J are finite and

$$Bf\{<a_i, x_i>\}_{i\in I} = Bg\{<a_j, y_j>\}_{j\in J}$$

The latter holds if and only if for every $i \in I$ there exists a $j \in J$ such that

$$<a_i, fx_i> = <a_j, gy_j> \qquad \text{(ie } a_i = a_j,\ fx_i = gy_j)$$

and, conversely, for every $j \in J$ there exists an $i \in I$ such that $<a_i, fx_i>=<a_j, gy_j>$. But then one can define $h : R' \to BR$ as mapping every pair

$$\{<a_i, x_i>\}_{i\in I}\ R'\ \{<a_j, y_j>\}_{j\in J}$$

to the set

$$\{<a_i, x_i, y_j> \mid a_i = a_j,\ fx_i = gy_j\} \in BR$$

This gives the desired factorization. Notice that the mediating function h is not unique and that this construction also applies to the simpler behaviour $BX = 1 + \mathsf{Act} \times X$.

The semantic import of coalgebraic bisimulation is shown by a list of properties which relate it to final coalgebras. One property is that coinductive extensions identify bisimilar elements; in particular, if two programs are bisimilar, then they have the same final coalgebra semantics. Another way of expressing this fact is to say that the equality on the final coalgebra lifts to the final bisimulation (in a suitable category of relations). As a corollary, final coalgebras are **internally fully-abstract**, in the sense that in a final coalgebra one cannot distinguish between bisimilar elements; this property is also called **strong extensionality**.

Next, if the pullback of two coinductive extensions lifts to a bisimulation, like, eg, when the functor B under consideration preserves weak pullbacks, then this pullback is the greatest relation lifting to a bisimulation. Together with the above property that coinductive extensions identify bisimilar elements, this gives that two programs have the same final coalgebra semantics if and only if they are bisimilar. In other words, coinduction can be 'pulled back' to bisimulation.

Let us look at these properties in detail.

Coinductive extensions identify bisimilar elements. That is, for any relation R lifting to a bisimulation the following diagram commutes.

$$\begin{tikzcd}[column sep=small] & \underset{\sim}{R} \arrow[dl, "r_1"'] \arrow[dr, "r_2"] & \\ \langle X, k\rangle \arrow[dr, "k^{@}"'] & & \langle Y, \ell\rangle \arrow[dl, "\ell^{@}"] \\ & \langle \widehat{B}, \varphi\rangle & \end{tikzcd}$$

This is a trivial consequence of the fact that both composites in the diagram are coalgebra arrows to the final coalgebra, hence they must be the same.

Corollary (Strong Extensionality): Final coalgebras are internally fully-abstract. That is, every relation which lifts to a bisimulation on the final coalgebra has equal legs:

$$\begin{tikzcd}[column sep=small] & \underset{\sim}{R} \arrow[dl, "r_1"'] \arrow[dr, "r_2"] & \\ \langle \widehat{B}, \varphi\rangle \arrow[dr, equal] & & \langle \widehat{B}, \varphi\rangle \arrow[dl, equal] \\ & \langle \widehat{B}, \varphi\rangle & \end{tikzcd}$$

The equality on the final coalgebra lifts to the final bisimulation. Consider the category having as objects relations lifting to bisimulations of an endofunctor B and as arrows triples of arrows $<r, f, g>$ making everything in sight in

$$\begin{tikzcd}
& R \arrow[dl, "r_1"'] \arrow[dr, "r_2"] \arrow[rrr, "r"] \arrow[d, phantom, "\sim"] & & & R' \arrow[dl, "r_1'"'] \arrow[dr, "r_2'"] \arrow[d, phantom, "\sim"] & \\
\langle X, k\rangle \arrow[rrr, "f"'] & {} & \langle Y, \ell\rangle \arrow[rrr, "g"] & \langle X', k'\rangle & {} & \langle Y', \ell'\rangle
\end{tikzcd}$$

commute – where f and g are arrows in $\mathbf{C}_B$, while r is an arrow in $\mathbf{C}$. Then the equality $\mathrm{EQ}(\widehat{B})$ on (the carrier of) the final coalgebra is the final object of this category. This is an immediate consequence of the fact that $\mathrm{EQ}(\widehat{B})$ is a pullback (in $\mathbf{C}$):

$$\begin{tikzcd}
& R \arrow[dl, "r_1"'] \arrow[dr, "r_2"] \arrow[rrr, dashed] \arrow[d, phantom, "\sim"] & & & \mathrm{EQ}(\widehat{B}) \arrow[dl, "e_1"'] \arrow[dr, "e_2"] \arrow[d, phantom, "\sim"] & \\
\langle X, k\rangle \arrow[rrr, "k^{@}"'] & {} & \langle Y, \ell\rangle \arrow[rrr, "\ell^{@}"] & \langle \widehat{B}, \varphi\rangle & {} & \langle \widehat{B}, \varphi\rangle
\end{tikzcd}$$

That is, from any relation R lifting to a bisimulation there is a mediating arrow to the equality $\mathrm{EQ}(\widehat{B})$ on the final coalgebra because the two legs of R can be coinductively prolonged to form a suitable cone on (the carrier of) the final coalgebra.

Greatest bisimulations. So far, we have made no distinction between relations and monic spans (like pullbacks). To be precise, one should first define an equivalence relation among monic spans with a common codomain and then take the corresponding equivalence classes as the actual relations; this equivalence relation is defined as follows.

For any two monic spans with a common codomain

$$\begin{tikzcd}
& M \arrow[dl, "m_1"'] \arrow[dr, "m_2"] & \\
X & & Y
\end{tikzcd}
\qquad
\begin{tikzcd}
& M' \arrow[dl, "m_1'"'] \arrow[dr, "m_2'"] & \\
X & & Y
\end{tikzcd}$$

write

$$\begin{tikzcd}
& M \arrow[dl, "m_1"'] \arrow[dr, "m_2"] & \\
X & & Y
\end{tikzcd}
\leq
\begin{tikzcd}
& M' \arrow[dl, "m_1'"'] \arrow[dr, "m_2'"] & \\
X & & Y
\end{tikzcd}$$

if there is an arrow $f : M \to M'$ such that M_i factorizes as $M_i' \circ f$, for both $i = 1$

and $i = 2$:

$$\begin{tikzcd} & M \arrow[r, "f"] \arrow[dl, "m_1"'] \arrow[drr, "m_2"' near end] & M' \arrow[dll, "m'_1" near end] \arrow[dr, "m'_2"] & \\ X & & & Y \end{tikzcd}$$

The two monic spans are then equivalent (hence represent the same relation) if the converse also holds, that is, if also

$$\begin{tikzcd} & M \arrow[dl, "m_1"'] \arrow[dr, "m_2"] & & & & M' \arrow[dl, "m'_1"'] \arrow[dr, "m'_2"] & \\ X & & Y & \geq & X & & Y \end{tikzcd}$$

The above defines a partial order '$\leq$' of relations (and also of relations which lift to bisimulations). If the cartesian product $X \times Y$ of two objects X and Y in a category exists, then its equivalence class is the *greatest relation* between X and Y wrt this partial order. If the category has finite limits, then products are pullbacks wrt the final object; in particular,

$$\begin{tikzcd} & X \times Y \arrow[dl, "\mathsf{fst}"'] \arrow[dr, "\mathsf{snd}"] \arrow[dd, phantom, "\lrcorner" rotate=-45, very near start] & \\ X \arrow[dr, dashed] & & Y \arrow[dl, dashed] \\ & 1 & \end{tikzcd}$$

In semantics, the 'base' category $\mathbf{C}$ should, like $\mathbf{Set}$, have all finite limits. The same cannot be said in general of the category $\mathbf{C}_B$ of coalgebras of the behaviour endofunctor B. What certainly is true is that the behaviour should have a final coalgebra, that is, the category $\mathbf{C}_B$ should have a final object. Now, recall that the coinductive extension $k^@ : X \to \widehat{B}$ of a coalgebra structure $k : X \to BX$ is the unique coalgebra arrow from the coalgebra $\langle X, k \rangle$ to the final coalgebra $\langle \widehat{B}, \varphi \rangle$; then one can take the pullback (in $\mathbf{C}$) of two coinductive extensions and, *if it lifts to a bisimulation* between the corresponding coalgebras

$$\begin{tikzcd} & \cdot \arrow[dl] \arrow[dr] \arrow[dd, phantom, "\sim", very near start] & \\ \langle X, k \rangle \arrow[dr, "k^@"'] & & \langle Y, \ell \rangle \arrow[dl, "\ell^@"] \\ & \langle \widehat{B}, \varphi \rangle & \end{tikzcd}$$

then this is the **greatest** (relation lifting to a) **bisimulation** between the coalgebras $\langle X, k \rangle$ and $\langle Y, \ell \rangle$.

Write $\overset{k,\ell}{\sim}$ for the relation obtained above by 'pulling back' the coinductive extensions of the coalgebra structures k and ℓ. Then, in **Set**, if the relation $\overset{k,\ell}{\sim}$ lifts to a bisimulation,

$$x \overset{k,\ell}{\sim} y \iff k^{@}(x) = \ell^{@}(y)$$

for any two elements $x \in X$ and $y \in Y$. (The implication from left to right follows the property that coinductive extensions always identify bisimilar elements.) Semantically, for an operational model $[\![\text{-}]\!] : TX \to BTX$ with syntax T and behaviour $BX = \check{\mathcal{P}}(1 + \mathsf{Act} \times X)$, two programs $t, t' \in TX$ are bisimilar if and only if they have the same final coalgebra semantics:

$$t \overset{[\![\text{-}]\!]}{\sim} t' \iff [\![t]\!]^{@} = [\![t']\!]^{@}$$

Notice the underlying assumption that the pullback $\overset{[\![\text{-}]\!]}{\sim}$ lifts to a bisimulation on the operational model $[\![\text{-}]\!]$:

$$\begin{tikzcd}
 & \dot{\sim} \arrow[dl] \arrow[dr] \arrow[dd, phantom, "\Diamond"] & \\
\langle TX, [\![\text{-}]\!] \rangle \arrow[dr, "{[\![\text{-}]\!]^{@}}"'] & & \langle TX, [\![\text{-}]\!] \rangle \arrow[dl, "{[\![\text{-}]\!]^{@}}"] \\
 & \langle \widehat{B}, \varphi \rangle &
\end{tikzcd}$$

As shown above, pullbacks lift to ordinary bisimulations, ie to the bisimulations of the behaviour functor $BX = \check{\mathcal{P}}(1 + \mathsf{Act} \times X)$. As a consequence, one can thus obtain the familiar result that the union of all bisimulations on a transition system is itself a bisimulation.

Bisimulations along arrows. The fact that coinductive extensions can be pulled back to bisimulations can be generalized to coinductive extensions *along arrows*. This leads to a new, more general notion of ordinary bisimulation in which not only the actions but also some (properties of the) states can be observed.

Recall that final coalgebras $\widehat{B} \cong B\widehat{B}$ are a special case of cofree coalgebras $DX \cong X \times BDX$ (namely $\widehat{B} = D1$) and that, correspondingly, the coinduction principle of final coalgebras generalizes to the arbitrary cofree coalgebras: for every coalgebra structure $k : X \to BX$ and arrow $f : X \to Z$ one has a unique coalgebra arrow $f^\flat : \langle X, k\rangle \to \langle DZ, \mathsf{snd}_Z\rangle$, namely the coinductive extension of k along f:

$$\begin{tikzcd} & X \arrow[r, "k"] \arrow[dl, "f"'] \arrow[d, dashed, "f^\flat"] & BX \arrow[d, "Bf^\flat"] \\ Z & DZ \arrow[l, "\varepsilon_Z = \mathsf{fst}_Z"] \arrow[r, "\mathsf{snd}_Z"'] & BDZ \end{tikzcd}$$

(Cf Section 7.)

Next, consider a **relation** R **between two arrows** $f : X \to Z$ **and** $g : Y \to Z$ **over the object** Z, that is, a relation between X and Y such that the diagram

$$\begin{tikzcd} & R \arrow[dl, "r_1"'] \arrow[dr, "r_2"] & \\ X \arrow[dr, "f"'] & & Y \arrow[dl, "g"] \\ & Z & \end{tikzcd}$$

commutes. Then, if X and Y carry coalgebra structures $k : X \to BX$ and $\ell : Y \to BY$ respectively and the relation R lifts to a bisimulation between them

$$\begin{tikzcd} & R \arrow[dl, "r_1"'] \arrow[dr, "r_2"] \arrow[d, phantom, "\sim"] & \\ \langle X, k\rangle & {} & \langle Y, \ell\rangle \end{tikzcd}$$

then also the diagram

$$\begin{tikzcd} & R \arrow[dl, "r_1"'] \arrow[dr, "r_2"] \arrow[d, phantom, "\sim"] & \\ \langle X, k\rangle \arrow[dr, "f^\flat"'] & {} & \langle Y, \ell\rangle \arrow[dl, "g^\flat"] \\ & \langle DZ, \mathsf{snd}_Z\rangle & \end{tikzcd}$$

commutes, because both composites $f^\flat \circ r_1$ and $g^\flat \circ r_2$ fit as the unique coinductive extension of the (no matter which!) coalgebra structure on R along the composite $f \circ r_1 = g \circ r_2 : R \to Z$.

If pullbacks lift to B-bisimulation, then the pullback (in the base category) of the coinductive extensions $f^\flat$ and $g^\flat$ of k and ℓ along f and g is the greatest relation between f and g which lifts to a bisimulation between $\langle X, k\rangle$ and $\langle Y, \ell\rangle$.

As an example, consider the simple behaviour $BX = 1 + \mathsf{Act} \times X$ and, correspondingly, ordinary bisimulation for deterministic transition systems. Let the set Act of actions be trivial, that is, let Act be made of only one action a. Let $\langle X, k\rangle$ and $\langle Y, \ell\rangle$ be the same coalgebra having as carrier the set $\mathbb{Z}$ of integers and as structure $\ell : \mathbb{Z} \to B(\mathbb{Z})$ the one corresponding to the following (deterministic) transition system: 0 is inert, a positive integer n performs a transition to its predecessor $n-1$, and a negative integer $-n$ performs a transition to its successor $-n+1$:

$$0 \downarrow * \qquad n \xrightarrow{a} n-1 \qquad -n \xrightarrow{a} -n+1$$

(Cf Example in Section 7.) Finally, let Z be the three-elements set $\{0, \clubsuit, \diamondsuit\}$. Thus:

$$X = \mathbb{Z} = Y \qquad Z = \{0, \clubsuit, \diamondsuit\} \qquad \mathsf{Act} = \{a\}$$

Now, different bisimulations are possible according to the choice of the functions $f, g : \mathbb{Z} \to \{0, \clubsuit, \diamondsuit\}$. Let us fix the function $g : \mathbb{Z} \to \{0, \clubsuit, \diamondsuit\}$ to be the one mapping odd numbers to $\clubsuit$ and even numbers to $\diamondsuit$. If f is equal to g, then every number is bisimilar to itself and to its opposite. For instance,

$$f^\flat(-3) \;=\; \clubsuit \xrightarrow{a} \diamondsuit \xrightarrow{a} \clubsuit \xrightarrow{a} 0 \;=\; g^\flat(3)$$

and thus -3 is bisimilar to 3 (wrt g).

The above amounts to assume that one can observe in both transition systems whether a number is odd or even. If, instead, in the first transition system one can observe this only for positive numbers, thus, eg, $f(-n) = 0$ and $f(n) = g(n)$, then one has that a positive number n is bisimilar to both $-n$ and n (wrt f and g) but its opposite $-n$ is not bisimilar to any number in the second transition system.

Finally, if one cannot observe at all in the first transition system whether a number is odd or even (ie $f(z) = 0$ for all $z \in \mathbb{Z}$) then only the two 0's are bisimilar.

(Notice that the arrows f and g can be regarded as *abstract interpretations* of the states.)

Another example is when one has a distinguished subset $Obs(X) \subset X$ of states which are 'observable'. This can be expressed by taking $Z = Obs(X) \cup \{\perp\}$ and $f : X \to Obs(X) \cup \{\perp\}$ to be

$$f(x) \;=\; \begin{cases} x & \text{if } x \in Obs(X) \\ \perp & \text{otherwise} \end{cases}$$

Bisimulations vs Congruences

Consider the case in which, like for the above behaviour functor, pullbacks lift to coalgebraic bisimulations. Then, in any situation like in functorial operational semantics

$$\begin{tikzcd}
\Sigma T0 \arrow[rr, "\Sigma\llparenthesis\text{-}\rrparenthesis^{\#}"] \arrow[d, "\cong", "\text{initial algebra}"'] & & \Sigma\widehat{B} \arrow[d, "\llparenthesis\text{-}\rrparenthesis"] \\
T0 \arrow[rr, shift left, "\text{initial algebra semantics } \llparenthesis\text{-}\rrparenthesis^{\#}"] \arrow[rr, shift right, "\text{final coalgebra semantics } \llbracket\text{-}\rrbracket^{@}"'] \arrow[rr, phantom, "="] \arrow[d, "\llbracket\text{-}\rrbracket"'] & & \widehat{B} \arrow[d, "\cong"', "\text{final coalgebra}"] \\
BT0 \arrow[rr, "B\llbracket\text{-}\rrbracket^{@}"'] & & B\widehat{B}
\end{tikzcd}$$

in which both an operational and a denotational model are given and the denotational model is adequate wrt the operational one in the sense that initial algebra and final coalgebra semantics coincide, one has that 'bisimulation is a congruence'. That is, if

$$u_1 \stackrel{\llbracket\text{-}\rrbracket}{\sim} v_1, \ldots, u_n \stackrel{\llbracket\text{-}\rrbracket}{\sim} v_n$$

for terms u_i and v_i, then, for every n-ary construct σ in Σ,

$$\sigma(u_1, \ldots, u_n) \stackrel{\llbracket\text{-}\rrbracket}{\sim} \sigma(v_1, \ldots, v_n)$$

Indeed, using the hypothesis that pullbacks lift to coalgebraic bisimulations, one has that, for all terms t and t',

$$t \stackrel{\llbracket\text{-}\rrbracket}{\sim} t' \iff \llbracket t \rrbracket^{@} = \llbracket t' \rrbracket^{@}$$

hence, for $i = 1, \ldots, n$,

$$\llbracket u_i \rrbracket^{@} = \llbracket v_i \rrbracket^{@}$$

and thus

$$\begin{array}{rcl}
\llbracket \sigma(u_1, \ldots, u_n) \rrbracket^{@} & = & \llparenthesis\, \sigma(u_1, \ldots, u_n) \,\rrparenthesis^{\#} \\
& = & \llparenthesis\, \sigma \,\rrparenthesis(\llparenthesis\, u_1 \,\rrparenthesis^{\#}, \ldots, \llparenthesis\, u_n \,\rrparenthesis^{\#}) \\
& = & \llparenthesis\, \sigma \,\rrparenthesis(\llbracket u_1 \rrbracket^{@}, \ldots, \llbracket u_n \rrbracket^{@}) \\
& = & \llparenthesis\, \sigma \,\rrparenthesis(\llbracket v_1 \rrbracket^{@}, \ldots, \llbracket v_n \rrbracket^{@}) \\
& = & \llbracket \sigma(v_1, \ldots, v_n) \rrbracket^{@}
\end{array}$$

Therefore,

$$\sigma(u_1, \ldots, u_n) \stackrel{\llbracket\text{-}\rrbracket}{\sim} \sigma(v_1, \ldots, v_n)$$

which means that the (bisimulation) relation $\stackrel{\llbracket\text{-}\rrbracket}{\sim}$ is a *congruence*. In general, a relation R between the carriers X and Y of two Σ-algebras $\langle X, h \rangle$ and $\langle Y, l \rangle$ is a

congruence when, for all $x_1, \ldots, x_n$ in X and $y_1, \ldots, y_n$ in Y and n-ary construct σ in Σ,

$$\text{if } x_1\, R\, y_1, \ldots, x_n\, R\, y_n \text{ then } h(\sigma(x_1, \ldots, x_n))\; R\; l(\sigma(y_1, \ldots, y_n))$$

Diagrammatically, this is equivalent to saying that the relation R lifts to the Σ-algebras in the sense that there exists a Σ-algebra structure $\widetilde{R} : \Sigma R \to R$ making the following diagram commute.

$$\begin{tikzcd}
& \Sigma R \arrow[dl, "\Sigma r_1"'] \arrow[dd, "\widetilde{R}"] \arrow[dr, "\Sigma r_2"] & \\
\Sigma X \arrow[dd, "h"'] & & \Sigma Y \arrow[dd, "l"] \\
& R \arrow[dl, "r_1"'] \arrow[dr, "r_2"] & \\
X & & Y
\end{tikzcd}$$

This definition generalizes to algebras of (arbitrary) monads T:

$$\begin{tikzcd}
& TR \arrow[dl, "Tr_1"'] \arrow[dd, "\widetilde{R}"] \arrow[dr, "Tr_2"] & \\
TX \arrow[dd, "h"'] & & TY \arrow[dd, "l"] \\
& R \arrow[dl, "r_1"'] \arrow[dr, "r_2"] & \\
X & & Y
\end{tikzcd}$$

In particular, if R is a Σ-congruence, then its inductive extension is a congruence of the monad T freely generated by Σ. This amounts to the well-known fact that if R is a (Σ-) congruence then, for every context $C[\text{-}]$, if $x\ R\ y$ then $C[x]\ R\ C[y]$.

Notice that for coalgebras one speaks of relations *lifting* to bisimulations while for algebras one speaks of relations *being* congruences. The point is that, while there are many ways of lifting a relation to a bisimulation, it is often the case that there exists a unique way of lifting a relation to a congruence. This is certainly true with pullback relations:

> **Pullbacks uniquely lift to T-congruences.** The lifting $\widetilde{R} : TR \to R$ in
>
> $$\begin{tikzcd}
& TR \arrow[dl, "Tr_1"'] \arrow[dr, "Tr_2"] \arrow[rrr, dashed, "\widetilde{R}"] & & & R \arrow[dl, "r_1"'] \arrow[dr, "r_2"] \arrow[d, phantom, "\lrcorner" very near start] & \\
TX \arrow[rrr, bend left, "h"] \arrow[dr, "Tf"'] & & TY \arrow[rrr, bend right, "l"'] \arrow[dl, "Tg"] & X \arrow[dr, "f"'] & & Y \arrow[dl, "g"] \\
& TZ \arrow[rrr, "j"'] & & & Z &
\end{tikzcd}$$

is given by the unique mediating arrow from the cone $\langle h \circ TR_1, k \circ TR_2 \rangle$ to the pullback R of f and g. The universality of R can be used to prove that the function $\widetilde{R} : R \to BR$ is a T-algebra structure.

One can check that the above implies that $\langle R, \widetilde{R} \rangle$ is the pullback of f and g in the T-algebras:

$$\begin{tikzcd}
& \langle R, \widetilde{R} \rangle \arrow[dl, "r_1"'] \arrow[dr, "r_2"] & \\
\langle X, h \rangle \arrow[dr, "f"'] & & \langle Y, l \rangle \arrow[dl, "g"] \\
& \langle Z, j \rangle &
\end{tikzcd}$$

The fact that pullbacks of functions between carriers of algebras lift uniquely to pullbacks (of the same functions but) in the category of algebras amounts to say that the forgetful functor $U^T : \mathbf{C}^T \to \mathbf{C}$ **creates pullbacks**. In turn, this is a consequence of the more general fact (see, eg, §VI.2 of [Mac71]) that

The forgetful functor $U^T : \mathbf{C}^T \to \mathbf{C}$ creates limits.

In other words, a category of algebras has the same limits as its base category. Colimits are more difficult. Dually, a category of coalgebras has the same colimits as its base category, ie:

The forgetful functor $U_B : \mathbf{C}_B \to \mathbf{C}$ creates colimits.

Instead, in general, the limits (eg, products and pullbacks) of coalgebras are difficult. This explains why there is no systematic way of lifting a pullback relation to a bisimulation, and extra assumptions are needed like the preservation of weak pullbacks.

Notes. Preliminary material presented in this section has appeared in [RT93, RT94]. Bisimulations along arrows appear here for the first time.

The notion of an ordinary bisimulation stems from the work of Park [Par81] and Milner [Mil80] on concurrency. Coalgebraic bisimulations were introduced in [AM89], while their dual algebraic congruences already appear in [Man76, page 167]. An order-enriched form of coalgebraic bisimulations is studied by Marcelo Fiore in [Fio93] (improving a previous definition from [RT93]); Fiore's notion cuts down, for a particular functor, to the notion of an *applicative bisimulation* from [Abr90].

For a categorical definition of relations see, eg, [FS90]. When dealing with categories other than **Set** like, eg, the category **pCpo** as in [Fio93], one might want to use a more subtle definition of relations, considering only a class of *admissible* monic spans, closed under pullbacks.

A drawback of the present definition of coalgebraic bisimulations is that it requires that the relations live in the same category as the coalgebras. In **Set** this is not a problem, but when one is working with more structured objects it might be too strong a requirement. For instance, in categories of complete partial orders one has to consider *chain-closed* relations.

Andy Pitts [Pit94a, Pit93, Pit94c] has introduced a different notion of generalized bisimulations for the functor types most commonly used in semantics which overcomes this problem and, moreover, it is 'compositional': if two composable relations are bisimulations (in the sense of Pitts) wrt two different functors F and G, then their composition is a bisimulation wrt the composite functor FG, which is not the case for coalgebraic bisimulations. These two properties really make the 'pulling back' of coinduction to (generalized) bisimulation a useful method for reasoning about coinductively defined objects. (Notice, however, that the actual construction of bisimulation relations can be quite involved, hence it would be important to generalize to functorial operational semantics the existing methods for constructing ordinary bisimulation like those treated in [San95].)

Pitts' notion is implicitly based on lifting the functors to a category of relations. This idea is formalized by Claudio Hermida and Bart Jacobs [Her93, HJ95a, HJ95b, Jac95] by means of the categorical notion of a 'fibration': a category $\mathbf{R}$ of relations over a given category $\mathbf{C}$ is a certain fibration on $\mathbf{C}$; functors F on $\mathbf{C}$ defined by universal properties lift to functors $\widetilde{F}$ on $\mathbf{R}$; a bisimulation wrt to F is then a $\widetilde{F}$-coalgebra in $\mathbf{R}$. Notice that an object of $\mathbf{R}$ does *not* need to be an object of $\mathbf{C}$ as well.

An alternative categorical approach to generalized bisimulations is pursued in [JNW93]; its relationship with the above approaches is still to be investigated.

13 The Observational Comonad for Bisimulation

The behaviour $BX = \check{\mathcal{P}}(1+\mathsf{Act}\times X)$ is not an $\boldsymbol{\omega}^{\mathsf{op}}$-continuous endofunctor, because the power-set functor $\check{\mathcal{P}}$ is not, hence its final coalgebra cannot be obtained as the limit of the usual $\boldsymbol{\omega}^{\mathsf{op}}$-chain. This section illustrates two alternative methods for establishing the existence of the final coalgebra of the finite power-set functor. The first method, due to Peter Aczel, amounts to quotienting a *weakly final* coalgebra by its greatest bisimulation.

The second method is due to Michael Barr. It amounts to finding a 'generating set' for the coalgebras of the finite power-set functor, that is, a (small) set $\{\langle X_i, k_i\rangle\}_I$ of coalgebras such that every coalgebra is a quotient of a coproduct of $\langle X_i, k_i\rangle$'s. By the *Special Adjoint Functor Theorem (SAFT)*, the final coalgebra is then the greatest quotient of the coproduct of all the $\langle X_i, k_i\rangle$'s.

More generally, SAFT ensures the existence of a right adjoint for the forgetful functor mapping coalgebras to their carriers. This right adjoint maps a set to its cofree coalgebra and the whole adjunction defines the cofree comonad for the finite power-set functor. The same can be done with the composite behaviour functor $BX = \check{\mathcal{P}}(1+\mathsf{Act}\times X)$ thus obtaining the observational comonad D for bisimulation.

For simplicity, let us consider the finite power-set functor

$$\mathcal{P}_{\mathrm{fi}} : \mathbf{Set} \to \mathbf{Set} \qquad\qquad X \mapsto \{X' \subseteq X \mid X' \text{ finite}\}$$

instead of its 'relevant' part only, the functor $\check{\mathcal{P}}$ which does not produce the empty set. The coalgebras of the finite power-set functor $\mathcal{P}_{\mathrm{fi}}$ are in a one-to-one correspondence with the finitely branching, directed graphs. Indeed, a coalgebra structure $k : X \to \mathcal{P}_{\mathrm{fi}}X$ defines a graph with $x \in X$ as nodes and with arcs $x \to x'$ for every $x' \in k(x)$. That is, the children of x in the graph are the elements of the image of x under k. Conversely, every finitely branching and directed graph defines a $\mathcal{P}_{\mathrm{fi}}$-coalgebra:

$$x \to x' \iff x' \in k(x)$$

Next, if the final coalgebra $\widehat{\mathcal{P}_{\mathrm{fi}}} \cong \mathcal{P}_{\mathrm{fi}}\widehat{\mathcal{P}_{\mathrm{fi}}}$ of the finite power-set functor exists, then every coalgebra structure $k : X \to \mathcal{P}_{\mathrm{fi}}(X)$ can be coinductively extended to a function $k^{@} : X \to \widehat{\mathcal{P}_{\mathrm{fi}}}$ such that

$$k^{@}(x) = \{k^{@}(x_i) \mid x_i \in k(x)\}$$

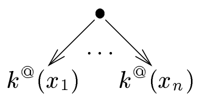
Moreover, this coinductive extension is unique:

$$\begin{tikzcd} X \arrow[r, dashed, "k^{@}"] \arrow[d, "k"'] & \widehat{\mathcal{P}_{\mathrm{fi}}} \arrow[d, "\cong"] \\ \mathcal{P}_{\mathrm{fi}}X \arrow[r, "\mathcal{P}_{\mathrm{fi}}(k^{@})"'] & \mathcal{P}_{\mathrm{fi}}\widehat{\mathcal{P}_{\mathrm{fi}}} \end{tikzcd}$$

The equation

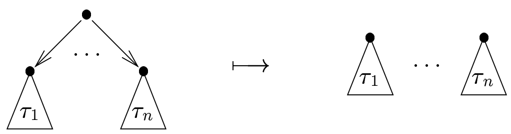
$$k^{@}(x) = \{k^{@}(x_i) \mid x_i \in k(x)\}$$

can be seen as the recursive definition of a tree:

$k^{@}(x) =$ 

(for $k(x) = \{x_1, \ldots, x_n\}$)

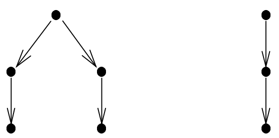
This is a rooted tree, finitely branching, and possibly of infinite depth. Neither nodes nor arcs are labelled. The set $\mathcal{T}$ of these rooted finitely branching trees can be seen as (the carrier of) a coalgebra of the finite power-set functor: every tree $\tau \in \mathcal{T}$ is mapped to the (finite) set $\{\tau_1, \ldots, \tau_n\}$ of children of its root:



This coalgebra is not a final but a *weakly final* coalgebra, that is, it is a coalgebra which ensures the existence but not the uniqueness of coinductive extensions. For example, the coalgebra structure $k : X = \{x, x_1, x_2, x'_1, x'_2\} \to \mathcal{P}_{\mathrm{fi}}(X)$

$$k(x) = \{x_1, x_2\} \quad k(x_1) = \{x'_1\} \quad k(x_2) = \{x'_2\} \quad k(x'_1) = 0 = k(x'_2)$$

can be extended to both the following trees.



Proposition. The final coalgebra of the finite power-set functor is the set of rooted finitely branching trees quotiented by the corresponding (greatest) coalgebraic bisimulation.

More generally, the quotient modulo bisimulation of any weakly final $\mathcal{P}_{\mathrm{fi}}$-coalgebra yields the final $\mathcal{P}_{\mathrm{fi}}$-coalgebra.

Recall, from the previous section, that a relation R between the carriers X and Y of two coalgebras $\langle X, k\rangle$ and $\langle Y, \ell\rangle$ lifts to a coalgebraic bisimulation if there exists a coalgebra structure $\widetilde{R}$ on the relation making its legs coalgebra arrows:

$$\begin{tikzcd}
& R \arrow[dl, "r_1"'] \arrow[d, "\widetilde{R}"] \arrow[dr, "r_2"] & \\
X \arrow[dd, "k"'] & \mathcal{P}_{\mathrm{fi}}R \arrow[ddl, "\mathcal{P}_{\mathrm{fi}}(r_1)"'] \arrow[ddr, "\mathcal{P}_{\mathrm{fi}}(r_2)"] & Y \arrow[dd, "\ell"] \\
& & \\
\mathcal{P}_{\mathrm{fi}}X & & \mathcal{P}_{\mathrm{fi}}Y
\end{tikzcd}$$

That is, for all x in X and y in Y such that xRy,

- if $x \to_k x'$ then $y \to_\ell y'$ for some y' such that $x' R\, y'$
- and, conversely, if $y \to_\ell y'$ then $x \to_k x'$ for some x' such that $x' R\, y'$.

(Here the notation $x \to_k x'$ stands for 'there is an arc from x to x' in the graph corresponding to the coalgebra $\langle X, k\rangle$'.)

As shown in the previous section, the finite power-set functor preserves weak pullbacks, hence pullbacks lift to $\mathcal{P}_{\mathrm{fi}}$-bisimulations. As a consequence, for every $\mathcal{P}_{\mathrm{fi}}$-coalgebra $\langle X, k\rangle$, the greatest relation on X lifting to a $\mathcal{P}_{\mathrm{fi}}$-bisimulation exists if the final $\mathcal{P}_{\mathrm{fi}}$-coalgebra exists. The argument is *not* circular because, later in this section, the existence of the final $\mathcal{P}_{\mathrm{fi}}$-coalgebra is proved by means of SAFT and *without* using bisimulations.

Next, consider the quotient of a $\mathcal{P}_{\mathrm{fi}}$-coalgebra $\langle X, k\rangle$ modulo its greatest bisimulation R_k. Categorically, this amounts to taking the *coequalizer* $q : X \to X/R_k$ of the two legs $r_1, r_2 : R_k \to X$ of the relation R_k:

$$\begin{tikzcd}
R_k \arrow[r, shift left, "r_1"] \arrow[r, shift right, "r_2"'] \arrow[d, dotted] & X \arrow[r, "q"] \arrow[d, "k"] & X/R_k \arrow[d, dashed] \\
\mathcal{P}_{\mathrm{fi}}(R_k) \arrow[r, shift left, "\mathcal{P}_{\mathrm{fi}}(r_1)"] \arrow[r, shift right, "\mathcal{P}_{\mathrm{fi}}(r_2)"'] & \mathcal{P}_{\mathrm{fi}}X \arrow[r, "\mathcal{P}_{\mathrm{fi}}(q)"'] & \mathcal{P}_{\mathrm{fi}}(X/R_k)
\end{tikzcd}$$

Notice this lifts to a coequalizer in the category of coalgebras. The coalgebra structure for X/R_k is given by the universal property of the coequalizer. Indeed, since the legs of the relation R_k lift to coalgebra arrows, the composite function $\mathcal{P}_{\mathrm{fi}}(q) \circ k : X \to \mathcal{P}_{\mathrm{fi}}(X/R_k)$ equates the two legs of the relation R_k. The corresponding unique mediating function from X/R_k to $\mathcal{P}_{\mathrm{fi}}(X/R_k)$ is the desired structure. Write $\langle X, k\rangle/R_k$ for this quotient coalgebra.

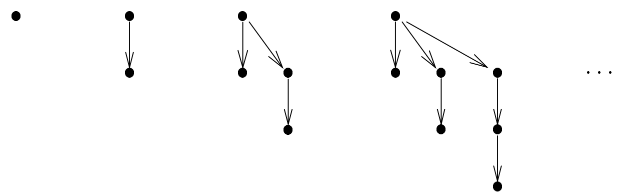
> **Lemma.** From every coalgebra there is *at most one arrow* to the quotient coalgebra $\langle X,k\rangle/R_k$.
>
> Indeed, consider two coalgebra arrows $f,g:\langle Y,\ell\rangle\to\langle X,k\rangle/R_k$. Since, as shown in the previous section, the equality relation always lifts to a coalgebraic bisimulation
>
> $$\begin{tikzcd} & \mathrm{EQ}(Y) \arrow[dl, "e_1"'] \arrow[dr, "e_2"] \arrow[d, phantom, "\sim"] & \\ \langle Y,\ell\rangle & & \langle Y,\ell\rangle \end{tikzcd}$$
>
> one has that the equality on Y with as legs the composites $f\circ e_1, g\circ e_2:\mathrm{EQ}(Y)\to X/R_k$ lifts to a bisimulation on the quotient coalgebra $\langle X,k\rangle/R_k$:
>
> $$\begin{tikzcd} & & \mathrm{EQ}(Y) \arrow[dl, "e_1"'] \arrow[dr, "e_2"] \arrow[d, phantom, "\sim"] & & \\ & \langle Y,\ell\rangle \arrow[dl, "f"'] & & \langle Y,\ell\rangle \arrow[dr, "g"] & \\ \langle X,k\rangle/R_k & & & & \langle X,k\rangle/R_k \end{tikzcd}$$
>
> Therefore, for every $y\in Y$, $f(y)$ is bisimilar to $g(y)$. Since, by construction, the quotient $\langle X,k\rangle/R_k$ is *strongly extensional*, that is, bisimulation is the equality, one has that $f(y)$ is equal to $g(y)$ for every $y\in Y$, hence $f=g$ and the lemma is proved. (Cf [Acz88, Theorem 2.19].)

Therefore, the quotient modulo bisimulation of a weakly final $\mathcal{P}_{\mathrm{fi}}$-coalgebra is necessarily final: the existence of an arrow from every coalgebra is guaranteed by being the quotient of a weakly final coalgebra, the uniqueness is guaranteed by the above property of quotients modulo bisimulation. In particular, the weakly final coalgebra of rooted finitely branching trees can be thus quotiented by bisimulation to yield the final coalgebra of the finite power-set functor. This concludes the proof of the above proposition.

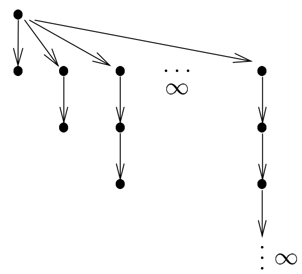
Notice that the finite power-set functor is *not* $\boldsymbol{\omega}^{\mathsf{op}}$-continuous, that is, the limit of the following chain is not a fixed point for the finite power-set functor $\mathcal{P}_{\mathrm{fi}}$. (Cf Section 5.)

$$1 \xleftarrow{\;1\;} \mathcal{P}_{\mathrm{fi}}1 \xleftarrow{\;\mathcal{P}_{\mathrm{fi}}1\;} {\mathcal{P}_{\mathrm{fi}}}^2 1 \xleftarrow{\;{\mathcal{P}_{\mathrm{fi}}}^2 1\;} \cdots$$

Indeed: Each object ${\mathcal{P}_{\mathrm{fi}}}^n 1$ of the chain is the set of finitely branching trees with depth at most n, quotiented by bisimulation. Correspondingly, the following sequence of trees belong to the above chain.



The problem is then that the limit has to contain the following tree with *infinitely* many branches,



while the final coalgebra, as shown above, contains only *finitely* branching trees.

* * *

The coequalizer $q : \langle X, k\rangle \to \langle X, k\rangle/R_k$ of the two legs of the greatest bisimulation on a coalgebra $\langle X, k\rangle$ is the 'greatest quotient' of $\langle X, k\rangle$. Formally, a **quotient** is an equivalence class of epis, just like a relation is an equivalence class of monic spans (see previous section). Coequalizers are always epi, ie they are 'right-cancellable': given a coequalizer $q : X \to Y$ and two parallel arrows $f, g : Y \to Z$, if $f \circ q = g \circ q$ then $f = g$. (This is immediate because of the universal property of coequalizers.)

Given two epis $f : X \twoheadrightarrow Z$ and $g : X \twoheadrightarrow Y$ with a common domain X put

$$f \leq g \iff f = f' \circ g$$

for some (necessarily unique and epi) arrow $f' : Y \twoheadrightarrow Z$. The two epis are equivalent (hence represent the same quotient) if the converse also holds, that is, if also

$$g \leq f$$

It is wrt this partial order on quotients that one can prove that $q : \langle X, k\rangle \twoheadrightarrow \langle X, k\rangle/R_k$ is (a representative of) the *greatest quotient* of the coalgebra $\langle X, k\rangle$. Indeed, since the pullback of $\check{\mathcal{P}}$-coalgebra arrows lifts to coalgebraic bisimulations (see previous section), the pullback $K(f)$ of (two copies of) every other quotient f lifts to a bisimulation and hence it is smaller than the relation R_k:

$$\begin{tikzcd}
& & \langle X,k\rangle \arrow[dr, two heads, "q"'] \arrow[drr, "f"] & & \\
K(f) \arrow[urr] \arrow[drr] \arrow[r, phantom, "\leq"] & R_k \arrow[ur] \arrow[dr] & & \langle X,k\rangle/R_k \arrow[r, dashed] & \langle Y,\ell\rangle \\
& & \langle X,k\rangle \arrow[ur, two heads, "q"] \arrow[urr, "f"'] & &
\end{tikzcd}$$

Therefore $f \leq q$, for every quotient $f : \langle X, k\rangle \twoheadrightarrow \langle Y, \ell\rangle$ of $\langle X, k\rangle$.

The greatest quotient of an object can be seen as the least upper bound of all quotients of that object. Dually, and more generally, also the greatest lower bound, ie the **intersection**, of all quotients of a suitable object can be used for finding the final object of a category.

In general, the intersection of a set of quotients of an object is their *pushout*, if this exists, because "pushouts of epis are epi" and "epis are closed under composition". (See, eg, [Mac71, §V.7].) For instance, the following diagram shows that the pushout of two epis f_1 and f_2 is their least upper bound.

$$\begin{tikzcd}
 & \cdot \arrow[dr, two heads, "f_1'"] \arrow[drr, two heads, bend left] & & \\
\cdot \arrow[ur, two heads, "f_1"] \arrow[dr, two heads, "f_2"'] & & \cdot \arrow[r, dashed, two heads] & \cdot \\
 & \cdot \arrow[ur, two heads, "f_2'"'] \arrow[urr, two heads, bend right] & &
\end{tikzcd}$$

Indeed, by definition of pushout, the composite $f_1' \circ f_1$ is equal to the composite $f_2' \circ f_2$, it is an epi, and it is greater that both f_1 and f_2. Moreover it is smaller than every other upper bound for f_1 and f_2, because of the universal property of pushouts.

Even if pushouts exist, the intersection of *all* quotients of an object in a category might fail to exist: one needs that the category be '**co-well-powered**', that is, the collection of all quotients of a given object should be a (small) set, so that its pushout can be taken. Now, by a standard cardinality argument, for every coalgebra $\langle X, k\rangle$ of an *arbitrary* endofunctor B on **Set**, one can form the set of its quotients, hence

$\mathbf{Set}_B$ is co-well-powered.

for all $B : \mathbf{Set} \to \mathbf{Set}$ (and thus the finite power-set functor in particular).

As for pushouts, these are colimits and coalgebras inherit all colimits from their underlying category, since, as mentioned in the previous section,

The forgetful functor $U_B : \mathbf{C}_B \to \mathbf{C}$ creates colimits.

Therefore, since **Set** is cocomplete (ie it has all colimits), the category of coalgebras of an endofunctor B on **Set** is also cocomplete:

$\mathbf{Set}_B$ is cocomplete.

Now, a more general way of finding a final object in a cocomplete and co-well-powered category is by finding a (small) set of objects $\{X_i\}_I$ such that every object in the category is the quotient of a coproduct of X_i's. (A set $\{X_i\}_I$ with this property is called a **generating set** for the (cocomplete) category.)

From a generating set to the final object. In a cocomplete category with a generating set $\{X_i\}_I$, if the intersection of all quotients $q : \coprod_I X_i \twoheadrightarrow Q$ of the coproduct $\coprod_I X_i$ exists, then Q is the final object of the category.

Let us first check uniqueness, that is, that from every object Y there exists at most one arrow to Q. Indeed, if there were two distinct arrows one can coequalize them. Let

$$q' : Q \twoheadrightarrow E$$

be this coequalizer. Since coequalizers are epi, the composite

$$\coprod_I X_i \overset{q}{\twoheadrightarrow} Q \overset{q'}{\twoheadrightarrow} E$$

would then be greater than q, which is a contradiction.

For the uniqueness part one only uses the fact that Q is the greatest quotient of an object. It is for the existence part that the generating set $\{X_i\}_I$ is used. Indeed, by definition of generating sets, every object Y is (the codomain of) a quotient

$$q' : \coprod_J X_j \twoheadrightarrow Y$$

of a coproduct $\coprod_J X_j$ of elements of $\{X_i\}_I$. Since every X_j is an element of $\{X_i\}_I$, there is, by the universal property of the coproduct, a function from $\coprod_J X_j$ to $\coprod_I X_i$, mapping each X_j to the corresponding X_i. (Notice that this function is not an embedding, because there might be more copies in $\coprod_J X_j$ of the same X_i.) One can then take the pushout

$$\begin{tikzcd}
\coprod_J X_j \arrow[r, two heads, "q'"] \arrow[d] & Y \arrow[d] \arrow[dl, phantom, "\ulcorner", very near start] \\
\coprod_I X_i \arrow[r, two heads, "q''"'] & Q'
\end{tikzcd}$$

of this function and the quotient $q' : \coprod_J X_j \twoheadrightarrow Y$. Since pushouts of epis are epi, the arrow $q'' : \coprod_I X_i \twoheadrightarrow Q'$ is epi, hence there exists an arrow from Q' to the codomain Q of the intersection of all quotients of $\coprod_I X_i$. One can then form a composite $Y \to Q' \twoheadrightarrow Q$, which proves the existence of an arrow from an arbitrary Y to Q. This concludes the proof. (Cf, eg, [Mac71, Theorem V.8.1].)

A generating set for the $\mathcal{P}_{\mathrm{fi}}$-coalgebras.

The set
$$\mathcal{G} = \{\langle U, k\rangle \mid k : U \to \mathcal{P}_{\mathrm{fi}} U \text{ and } U \subseteq \omega\}$$
of $\mathcal{P}_{\mathrm{fi}}$-coalgebras with ordinals less than or equal to ω as carriers is a (small) generating set for the category of $\mathcal{P}_{\mathrm{fi}}$-coalgebras.

Firstly notice that, if a set U has cardinality $\leq \omega$, then also its set of finite subsets $\mathcal{P}_{\mathrm{fi}} U$ has cardinality $\leq \omega$. Therefore the above collection $\mathcal{G}$ really is a set (and not a proper class).

Next, let us show that the set $\mathcal{G}$ is a generating set for the $\mathcal{P}_{\mathrm{fi}}$-coalgebras. For this, notice that, in a category which (like the one of $\mathcal{P}_{\mathrm{fi}}$-coalgebras) is cocomplete, a set $\mathcal{G}$ of objects is generating if and only if for every two parallel arrows f_1 and f_2 such that $f_1 \neq f_2$ there exists an arrow g from an object in $\mathcal{G}$ such that
$$f_1 \circ g \neq f_2 \circ g$$
This definition is easier to check (and it makes sense also in categories which are not cocomplete). For instance, in **Set**, two functions $f_1, f_2 : X \to Y$ are distinct if and only if there exists an $x \in X$ such that $f_1(x) \neq f_2(x)$, therefore, the singleton set 1 is a generator.

Two coalgebra arrows
$$f_1, f_2 : \langle X, k\rangle \to \langle Y, \ell\rangle$$
are functions, thus also they have a distinct value at some $x \in X$. However, the coalgebras with carrier 1 do not suffice to form a generating set for the $\mathcal{P}_{\mathrm{fi}}$-coalgebras, because the discriminating x will be mapped by $k : X \to \mathcal{P}_{\mathrm{fi}} X$ to a set $\{x_1, \ldots, x_m\}$ in which the x_i's are, in general, different from x.

The idea is that, since every $\mathcal{P}_{\mathrm{fi}}$-coalgebra structure $k : X \to \mathcal{P}_{\mathrm{fi}} X$ maps elements $x \in X$ to *finite* subsets of X, one can start from x and recursively apply ($\mathcal{P}_{\mathrm{fi}}$ of) k to it. Thus at the first step one has $\{x\}$ only, at the second $\{x\} \cup \{x_1, \ldots, x_m\}$, and so on, until a subset $U \subseteq X$ is found such that $x \in U$ and k restricted to U is a $\mathcal{P}_{\mathrm{fi}}$-coalgebra structure on U itself. Because at each step only finitely many x_i's are added, the set U cannot be larger than ω. Therefore, the coalgebra $\langle U, k\rangle$ is isomorphic to a coalgebra in $\mathcal{G}$.

Formally, given a $\mathcal{P}_{\mathrm{fi}}$-coalgebra $\langle X, k\rangle$ and an $x \in X$, let
$$U = \bigcup_{n \in \omega} U_n$$
where
$$U_0 = \{x\} \qquad U_{n+1} = U_n \cup \mathcal{P}_{\mathrm{fi}}(k)(U_n)$$
By definition U is a subset of X of cardinality at most ω. It remains thus only to show that, if $x_i \in U$, then $k(x_i) \subseteq \mathcal{P}_{\mathrm{fi}} U$. But this follows from the fact that $x_i \in U$ implies there exists an n such that $x_i \in U_n$, hence $k(x_i) \subseteq U_{n+1} \subseteq U$.

This concludes the proof that the above $\mathcal{G}$ is a generating set for the $\mathcal{P}_{\text{fi}}$-coalgebras. (Cf [Bar93, Proposition 1.3].)

> **Corollary 1.** The final coalgebra of the finite power-set functor is the intersection of all quotients of the coproduct $\coprod_{\mathcal{G}}\langle U, k\rangle$.

Notice that, by essentially the same argument, one can show that the set

$$\mathcal{G}_B = \{\langle U, k\rangle \mid k : U \to BU \text{ and } U \subseteq \omega\}$$

is a generating set for the behaviour $BX = \check{\mathcal{P}}(1 + \mathsf{Act} \times X)$. Thus:

> **Corollary 2.** The final coalgebra of the behaviour $BX = \check{\mathcal{P}}(1+\mathsf{Act}\times X)$ is the intersection of all quotients of the coproduct $\coprod_{\mathcal{G}_B}\langle U, k\rangle$.

Next, a category is **locally small** if the collection of arrows between every two objects forms a (small) set. For instance, **Set** is locally small. Now, the above proof of the existence of a final coalgebra by means of a generating set is an application of the following general theorem.

> **The Special Adjoint Functor Theorem (SAFT).** If **D** is cocomplete, co-well-powered, and with a (small) generating set, and if **C** is locally small, then every cocontinuous functor $F : \mathbf{D} \to \mathbf{C}$ has a right adjoint.
>
> (For a proof see, eg, [FS90] or [Mac71].)

Indeed, by instantiating the above theorem to the unique functor

$$\mathbf{Set}_B \to \mathbf{1}$$

from the coalgebras of the endofunctor $BX = \check{\mathcal{P}}(1 + \mathsf{Act} \times X)$ (or $BX = \mathcal{P}_{\text{fi}}X$) to the (final) category $\mathbf{1}$ with one object and one arrow (the identity), one obtains the existence of the final B-coalgebra.

A functor which creates colimits also preserves them, that is, it is cocontinuous, hence, for every endofunctor B on **Set**,

The forgetful functor $U_B : \mathbf{Set}_B \to \mathbf{Set}$ is cocontinuous.

Therefore, the Special Adjoint Functor Theorem also shows that, for every endofunctor (like the behaviour $BX = \check{\mathcal{P}}(1 + \mathsf{Act} \times X)$ or the finite power-set functor) whose coalgebras have a generating set,

The forgetful functor $U_B : \mathbf{Set}_B \to \mathbf{Set}$ has a right adjoint.

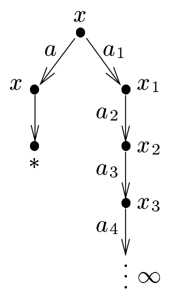
This adjunction gives rise to a cofree comonad $D{=}{<}D, \varepsilon, \delta{>}$:

> Let $G_B : \mathbf{Set} \to \mathbf{Set}_B$ be the right adjoint of the above forgetful functor $U_B : \mathbf{Set}_B \to \mathbf{Set}$. By definition of right adjoint, given a set X, a coalgebra $\langle Y, \ell \rangle$, and a function $f : Y \to X$ there exists a unique coalgebra arrow $f^\flat : \langle Y, \ell \rangle \to G_B X$ such that $f = \varepsilon_X \circ U_B f^\flat$, where $\varepsilon : U_B G_B \Rightarrow I$ is the counit of the adjunction. (Cf Section 8.)
>
> Write DX for the carrier of the coalgebra $G_B X$ and $\lambda_X : DX \to BDX$ for its structure. Then $U_B f^\flat : Y \to DX$ is the unique $(X \times B)$-coalgebra arrow from the coalgebra (structure) $<f, \ell> : Y \to X \times BY$ to the coalgebra (structure) $<\varepsilon_X, \lambda_X> : DX \to X \times BDX$, which means that the latter is the (structure of the) final $(X \times B)$-coalgebra.
>
> Because final coalgebras are isomorphisms, this implies that $DX \cong X \times BDX$, the counit at X is the first projection $\mathsf{fst}_X : DX \to X$ and the structure $\lambda_X : DX \to BDX$ is the second projection.
>
> As shown in Section 7, the operation $X \mapsto DX$ extends to an endofunctor $D : \mathbf{Set} \to \mathbf{Set}$, and the counit $\varepsilon : D \Rightarrow I$ and the coinductive extension $\delta : D \Rightarrow D^2$ of the second projection along the identity are comonad operations for it.

In particular, the cofree comonad corresponding to the behaviour $BX = \check{\mathcal{P}}(1 + \mathsf{Act} \times X)$ is the **observational comonad for bisimulation**.

Concretely, the value of the observational comonad for bisimulation at a set X can be obtained by means of a quotient construction in terms of trees and bisimulations as follows.

Let $\mathcal{T}_X$ be the set of trees which are coinductively generated by finitely branching transition systems. That is, the set $\mathcal{T}_X$ is the set of trees which are rooted, finitely branching, with nodes labelled by $x \in X$, arcs labelled by $a \in \mathsf{Act}$, and whose leaves are labelled by $*$ (and the arcs to leaves are then unlabelled). These trees are possibly of infinite depth. For instance, the following is a tree in $\mathcal{T}_X$ if $x, x_1, x_2, \ldots$ are in X.



This tree has root labelled by x, one leaf, and one infinite branch.

Just like the set $\mathcal{T}$ given at the beginning of this section can be seen as a coalgebra of the finite power-set functor, this set $\mathcal{T}_X$ can be seen as an $(X \times B)$-coalgebra. The function $\mathcal{T}_X \to X$ is the operation which, given a tree, returns the label $x \in X$ of its root.

One can check that, with this structure, the set $\mathcal{T}_X$ is a weakly final $(X \times B)$-coalgebra. The final $(X \times B)$-coalgebra (with carrier DX!) can be then obtained by taking the quotient modulo the greatest $(X \times B)$-bisimulation. By instantiating the coalgebraic notion of bisimulation to the functor $(X \times B)$, one obtains relations R on $\mathcal{T}_X$ such that two trees $\tau_1, \tau_2 \in \mathcal{T}_X$ by R (ie $\tau_1 R \tau_2$) iff the following four conditions are satisfied.

1. The label $x \in X$ of the root of τ_1 is the same as the label of the root of τ_2;
2. $\tau_1 \longrightarrow *$ if and only if $\tau_2 \longrightarrow *$;
3. if $\tau_1 \xrightarrow{a} \tau_1'$ then $\tau_2 \xrightarrow{a} \tau_2'$ for some τ_2' such that $\tau_1' R \tau_2'$,
4. and, conversely, if $\tau_2 \xrightarrow{a} \tau_2'$ then $\tau_1 \xrightarrow{a} \tau_1'$ for some τ_1' such that $\tau_1' R \tau_2'$.

The first clause is the one corresponding to the extra information given by the states $x \in X$. By putting $X = 1$ one recovers the ordinary notion of bisimulation between trees (with unlabelled nodes).

Notes. The idea of defining semantics by taking quotients of transition systems (ie coalgebras) by greatest (ordinary) bisimulations dates back at least to [Mil80]. The Final Coalgebra Theorem in [AM89] (based on a previous result in [Acz88]) generalizes that idea: it shows that final coalgebras of endofunctors can be obtained by quotienting weakly final coalgebras by the greatest (coalgebraic) *congruence*. This is stated for 'set-based' endofunctors on the category **SET** of classes (ie large sets – cf Part V): an endofunctor is *set-based* if its value at a class X is determined by its value at the (small) subsets of X [Acz88, Definition 6.1]. An example is the endofunctor $\mathcal{P}_S : \mathbf{SET} \to \mathbf{SET}$ mapping a class to the class of its (small) subsets, which is used in Part V. If an endofunctor preserves weak pullbacks then the notion of a (coalgebraic) conguence cuts down to that of a (coalgebraic) bisimulation [AM89, Proposition 6.2].

In [Bar93], the final coalgebra theorem of [Acz88] is reformulated in **Set** (thus without use of classes) by replacing the set-based condition by that of 'accessibility', modelling with inaccessible cardinals the size distinction between sets and classes. In particular, the endofunctors $\mathcal{P}_{\mathrm{fi}}$ and $BX = \check{\mathcal{P}}(1 + \mathsf{Act} \times X)$ are accessible and the above 'construction' of the corresponding generating sets is a special case of that in [Bar93, Proposition 1.3].