Università degli Studi di Pisa
Corso di Laurea Magistrale in Informatica

Anno Accademico 2021-2022

Insegnamento di

Foundation of Computing

Pagina del corso: `http:/pages.di.unipi.it/montanari/FOC.html`

Note di

CCS

1. I sistemi di riscrittura etichettati (LTS) come coalgebre
2. I sistemi LTS composizionali come bialgebre
3. Il CCS di Milner e i suoi modelli bialgebrici

a cura di
Ugo Montanari
Dipartimento di Informatica
Università degli Studi di Pisa
`ugo@di.unipi.it`

Algebras vs. Coalgebras

We saw an algebraic definition of categories, but we can also give a categorical definition of algebras. Here only Σ-algebras (no axioms), one-sorted.

Take a polinomial functor F on Set. An algebra is any arrow $f: F(A) \to A$. Functor F determines signature Σ.

Set

$$\begin{tikzcd} A \\ F(A) \arrow[u, "f"] \end{tikzcd}$$

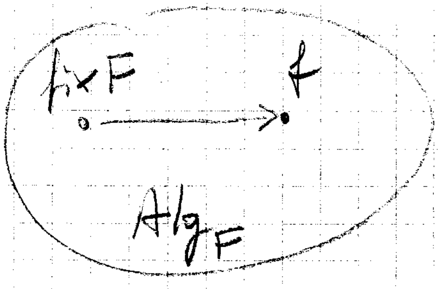
Example. $S = \{number\}$ $\Sigma_0 = (zero)$ $\Sigma_1 = (succ)$ $\Sigma_2 = (mult, sum)$
Then $F(X) = * + X + X \times X + X \times X$, $+$ being disjoint union
Every f is a function assigning a value in A to $* + A + A \times A + A \times A$. This corresponds to compute functions $zero^f$, $succ^f$, $mult^f$ and sum^f.

The category of algebras $\underline{Alg}_F$ has algebras as objects and arrows h in Set as arrows such that

$$\begin{tikzcd} A \arrow[r, "h"] & B \\ F(A) \arrow[u, "f"] \arrow[r, "F(h)"'] & F(B) \arrow[u, "g"'] \end{tikzcd}$$

this diagram commutes.
This is the ordinary notion of homomorphism. In fact, given the argument in $F(A)$ it must be the same to apply an operation in f and then mapping the results via h than to map the argument via $F(h)$ and then to apply the corresponding operation in g.

Category $\underline{Alg_F}$ has an initial algebra $\mathrm{fix}\,F$



which is the minimal solution of the isomorphism equation

$X \sim F(X)$

where $\mathrm{fix}\,F$ is the isomorphism function $F(X) \to X$.

Coalgebras

The dual concept: opposite arrows.

Set

$$A \xrightarrow{f} F(A)$$

Example. Let

$$F(X) = atom + X \times X$$

Here A represents a set of states, which can be understood as binary lists of atoms, finite or infinite, possibly cyclic. For every list a, $f(a)$ gives either the atom on the leaf, or two sublists.

However the most useful case is

$$F(X) = \mathcal{P}_f(L \times X)$$

where function f returns a finite (or countable, in some cases) set of pairs

$$f(a) = \{(l_i, b_i)\}$$

which should be interpreted as labelled transitions outgoing from state a.

Labelled Transition Systems

A LTS is a triple $K = (Q, L, \rightarrow)$

Q : set of states
L : set of labels
$\rightarrow \subseteq Q \times L \times Q$ (written $q \xrightarrow{\ell} q'$) transition relation

Bisimilarity (van Benthem 1978; Milner, Park, 1981)

Direct definition:

A bisimulation R is a relation $R \subseteq Q \times Q$ satisfying:

$p_1 \, R \, p_2$ implies that for every transition
$p_1 \xrightarrow{\ell} q_1$ there is a transition $p_2 \xrightarrow{\ell} q_2$ with $q_1 \, R \, q_2$
and viceverse.

$\simeq = \bigcup R$ bisimilarity. $p_1 \simeq p_2$ iff there is a bisimulation R with $p_1 R p_2$.

There are two players, Alice and Bob.
Given two states p_1 and p_2, Alice wants to prove they are different, Bob that they are equivalent.
If neither p_1 nor p_2 have any transition, Alice has lost.
Otherwise Alice picks up a transition of one of them, and Bob has to find a transition of the other with the same label. Then the game continues from the target states of the two transitions. States p_1 and p_2 are equivalent if Bob has a winning strategy, namely no matter what Alice chooses, he is always able to match it.

Coalgebra Morphisms

An F-coalg morphism is a function h making the following diagram commute.

$$\begin{tikzcd} A \arrow[r, "h"] \arrow[d, "f"'] & B \arrow[d, "g"] \\ F(A) \arrow[r, "F(h)"'] & F(B) \end{tikzcd} \qquad \underline{\mathrm{Set}}$$

$$F(X) = \mathcal{P}_f(L \times X)$$

$$F(h)(S) = \{(\ell, h(a)) \mid (\ell, a) \in S\}$$

The Kernel of h (i.e $a_1 \sim a_2$ iff $h(a_1) = h(a_2)$) is a bisimulation for the LTS $(A, L, \rightarrow)$ with

$$p \xrightarrow{\ell} q \iff (\ell, q) \in f(p)$$

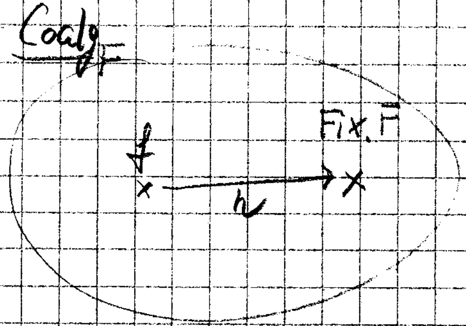
In fact if the diagram commutes we have

$$(F(h))(f(a)) \ \substack{\subseteq \\ \supseteq} \ g(h(a))$$

Thus if $(\ell, b) \in f(a)$, namely if $a \xrightarrow{\ell}_f b$, then $(\ell, h(b)) \in g(h(a))$, namely $h(a) \xrightarrow{\ell}_g h(b)$.
Similarly if $a' \xrightarrow{\ell}_g b'$ then for all a with $f(a) = a'$ there is a transition $a \xrightarrow{\ell}_f b$ with $b' = f(b)$ (zig-zag condition, van Benthem)

Then $a \simeq h(a)$. Since $\simeq$ is transitively closed, then $a_1 \simeq h(a_1)$, $a_2 \simeq h(a_2)$ and $h(a_1) = h(a_2)$ implies $a_1 \simeq a_2$.

The Category $\underline{Coalg}_F$



There is a final coalgebra Fix F which is the maximal solution of the equation

$$X = F(X).$$

Intuitively Fix F is the union of all the states of all the LTS of F and of their transitions, where bisimilar states have been identified.

The image of a LTS f through its unique morphism $!_f : f \to \mathrm{Fix}\,F$ is its minimal realization, i.e. its abstract semantics.

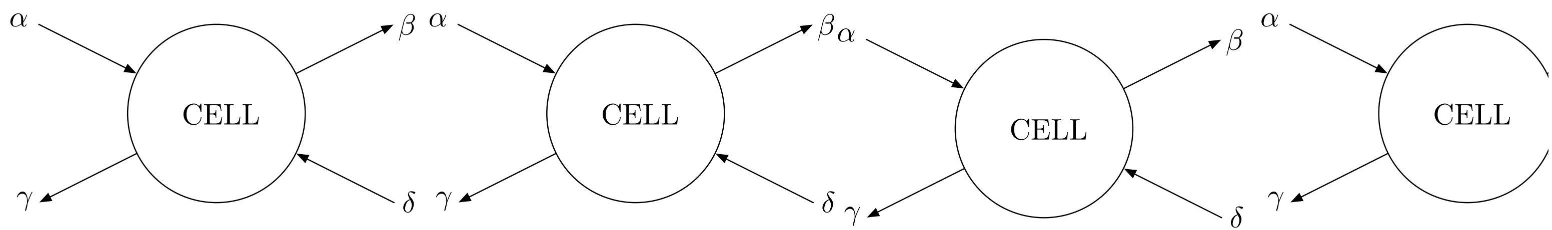
If the states of f are finite, there is a partition refinement algorithm to compute its minimal realization.

$$\underline{Set} \qquad A \xrightarrow{!_A} \{*\}$$

$$\begin{tikzcd} A \arrow[r, "h^n"] \arrow[d, "f"'] & {} \\ F(A) \arrow[r, "F(h^n)"] & {} \end{tikzcd} \qquad h^{n+1} = f \,;\, F(h^n)$$

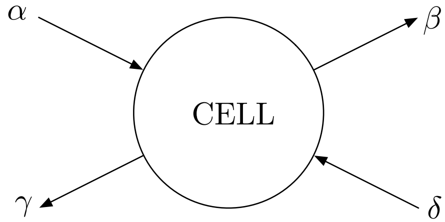
Function h^n is represented by its kernel. The kernel of h^{n+1} is a refinement of that of h^n. Termination occurs if there is no refinement.

A dynamic concurrent stack of equal cells



Four ports for communicating with left and right cells.
Leftmost cell can communicate with external components
Rightmost cell can communicate only on the left

A CELL can be in one of four different states according to the number of data items (natural numbers) it contains

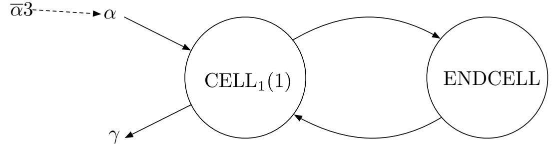


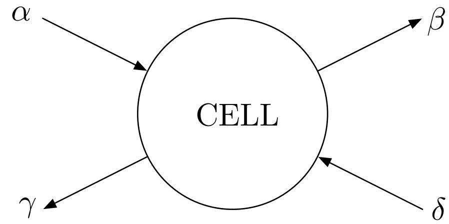
$$\mathrm{CELL}_1(v) \stackrel{\mathrm{def}}{=} \alpha y.\mathrm{CELL}_2(y,v) \quad + \quad \overline{\gamma} v.\mathrm{CELL}_0$$

$$\mathrm{CELL}_2(u,v) \stackrel{\mathrm{def}}{=} \overline{\beta} v.\mathrm{CELL}_1(u)$$

$$\mathrm{CELL}_0 \stackrel{\mathrm{def}}{=} \delta x.\,\textbf{if}\ x = \$\ \textbf{then}\ \mathrm{ENDCELL}\ \textbf{else}\ \mathrm{CELL}_1(x)$$

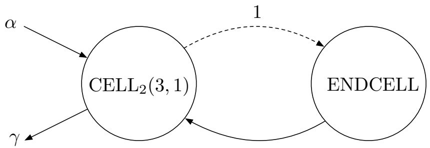
$$\mathrm{ENDCELL} \stackrel{\mathrm{def}}{=} \alpha z.(\mathrm{CELL}_1(z) \underbrace{\frown \mathrm{ENDCELL})}_{\text{a new bottom cell}} \quad + \quad \overline{\gamma}\$.\,\mathbf{nil}$$





$(\mathrm{CELL}_1(1) \frown \mathrm{ENDCELL})\backslash\beta\backslash\delta$ receiving the value 3 on channel α

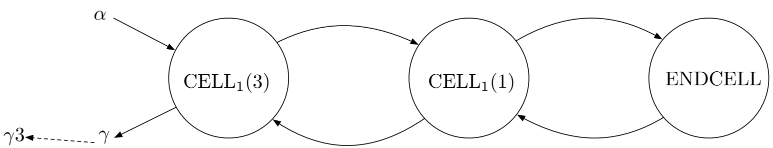
$$\mathrm{CELL}_1(v) \stackrel{\mathrm{def}}{=} \alpha y.\mathrm{CELL}_2(y,v) \quad + \quad \overline{\gamma}v.\mathrm{CELL}_0$$



$(\mathrm{CELL}_2(3,1) \frown \mathrm{ENDCELL})\backslash\beta\backslash\delta$ before right-shifting the value 1

$$\mathrm{CELL}_2(u,v) \stackrel{\mathrm{def}}{=} \overline{\beta}v.\mathrm{CELL}_1(u)$$

$$\mathrm{ENDCELL} \stackrel{\mathrm{def}}{=} \alpha z.(\mathrm{CELL}_1(z) \underbrace{\frown \mathrm{ENDCELL}}_{\text{a new bottom cell}}) \quad + \quad \overline{\gamma}\$.\,\mathbf{nil}$$



$$\mathrm{CELL}_1(3) \frown \mathrm{CELL}_1(1) \frown \mathrm{ENDCELL}\backslash\beta\backslash\delta$$

Calculus for Communicating Systems (CCS)

$p ::= nil \mid p+p \mid p|p \mid p/\alpha \mid \mu.p \mid x \mid rec\, x.p$ agents

$\mu ::= \tau \mid \lambda \qquad \lambda ::= \alpha \mid \bar{\alpha} \quad$ with $\bar{\bar{\alpha}} = \alpha$ prefixes

An infinite LTS $(Q, L, \rightarrow)$ can be defined, where Q is the set of all agents, $L = \{\tau\} \cup \Lambda$ the labels are actions λ and the silent action τ, and $\rightarrow$ is defined by the following SOS (structural operational semantics) rules in De Simone format (except for rec).

$$\mu.p \xrightarrow{\mu} p$$

$$\frac{p \xrightarrow{\mu} q}{p+r \xrightarrow{\mu} q} \qquad \frac{p \xrightarrow{\mu} q}{r+p \xrightarrow{\mu} q}$$

$$\frac{p \xrightarrow{\mu} q}{p|r \xrightarrow{\mu} q|r} \qquad \frac{p \xrightarrow{\mu} q}{r|p \xrightarrow{\mu} r|q} \qquad \frac{p_1 \xrightarrow{\lambda} q_1 \quad p_2 \xrightarrow{\bar{\lambda}} q_2}{p_1|p_2 \xrightarrow{\tau} q_1|q_2}$$

$$\frac{p \xrightarrow{\tau} q \quad \lambda \neq \alpha, \bar{\alpha}}{p/\alpha \xrightarrow{\tau} q/\alpha}$$

$$\frac{p[rec\, x.p/x] \rightarrow q}{rec\, x.p \rightarrow q} \qquad \text{or equivalently} \quad rec\, x.p = p[rec\, x.p/x]$$

Example 11.2 (Derivation). We show an example of the use of the derivation rules we have introduced. Let us take the (guarded) CCS process: $((p \mid q) \mid r)\backslash\alpha$, where:

$$p \stackrel{\text{def}}{=} \mathbf{rec}\ x.\ (\alpha.x + \beta.x) \qquad q \stackrel{\text{def}}{=} \mathbf{rec}\ x.\ (\alpha.x + \gamma.x) \qquad r \stackrel{\text{def}}{=} \mathbf{rec}\ x.\ \overline{\alpha}.x.$$

First, let us focus on the behaviour of the simpler, deterministic agent r. We have:

$$\begin{array}{rl} \mathbf{rec}\ x.\ \overline{\alpha}.x \xrightarrow{\lambda} r' \qquad\qquad \nwarrow_{\text{Rec}} & \overline{\alpha}.(\mathbf{rec}\ x.\ \overline{\alpha}.x) \xrightarrow{\lambda} r' \\ \nwarrow_{\text{Act},\ \lambda=\overline{\alpha},\ r'=\mathbf{rec}\ x.\ \overline{\alpha}.x} & \square \end{array}$$

where we have annotated each derivation step with the name of the applied rule. Thus, $r \xrightarrow{\overline{\alpha}} r$ and since there are no other rules applicable during the above derivation, the LTS associated with r consists of a single state and one looping arrow with label $\overline{\alpha}$. Correspondingly, the agent is able to perform the action $\overline{\alpha}$ indefinitely. However, when embedded in the larger system above, then the action $\overline{\alpha}$ is blocked by the topmost restriction $\cdot\backslash\alpha$. Therefore, the only opportunity for r to execute a transition is by synchronising on channel α with either one or the other of the two (nondeterministic) agents p and q. In fact the synchronisation on α produces an action τ which is not blocked by $\cdot\backslash\alpha$. Note that p and q are also available to interact with some external agent on other non-restricted channels (β or γ).

By using the rules of the operational semantics of CCS we have, e.g.:

$$\begin{array}{rl} ((p \mid q) \mid r)\backslash\alpha \xrightarrow{\mu} s \qquad\qquad \nwarrow_{\text{Res},\ s=s'\backslash\alpha} & (p \mid q) \mid r \xrightarrow{\mu} s', \quad \mu \neq \alpha, \overline{\alpha} \\ \nwarrow_{\text{Com},\ \mu=\tau,\ s'=s'' \mid r_1} & p \mid q \xrightarrow{\lambda} s'', \quad r \xrightarrow{\overline{\lambda}} r_1 \\ \nwarrow_{\text{Par},\ s''=p \mid q_1} & q \xrightarrow{\lambda} q_1, \quad r \xrightarrow{\overline{\lambda}} r_1 \\ \nwarrow_{\text{Rec}} & \alpha.q + \gamma.q \xrightarrow{\lambda} q_1, \quad r \xrightarrow{\overline{\lambda}} r_1 \\ \nwarrow_{\text{Sum}} & \alpha.q \xrightarrow{\lambda} q_1, \quad r \xrightarrow{\overline{\lambda}} r_1 \\ \nwarrow_{\text{Act},\ \lambda=\alpha,\ q_1=q} & r \xrightarrow{\overline{\alpha}} r_1 \\ \nwarrow_{\text{Rec}} & \overline{\alpha}.r \xrightarrow{\overline{\alpha}} r_1 \\ \nwarrow_{\text{Act},\ r_1=r} & \square \end{array}$$

From which we derive:

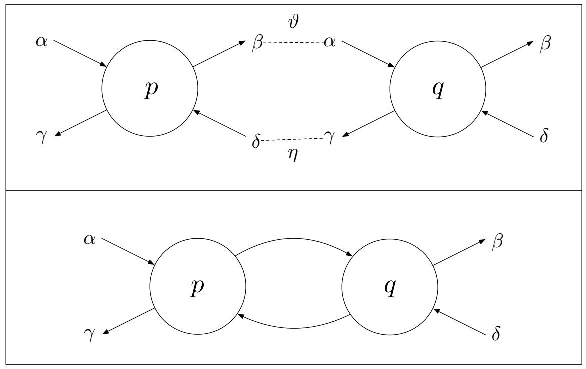


Fig. 11.8: Graphically illustration of the concatenation operator $p \smile q$

$$
\begin{aligned}
r_1 = r &= \textbf{rec}\ x.\ \overline{\alpha}.x \\
q_1 = q &= \textbf{rec}\ x.\ \alpha.x + \gamma.x \\
s'' = p \mid q_1 &= (\textbf{rec}\ x.\ \alpha.x + \beta.x) \mid \textbf{rec}\ x.\ \alpha.x + \gamma.x \\
s' = s'' \mid r_1 &= ((\textbf{rec}\ x.\ \alpha.x + \beta.x) \mid (\textbf{rec}\ x.\ \alpha.x + \gamma.x)) \mid \textbf{rec}\ x.\ \overline{\alpha}.x \\
s = s' \backslash \alpha &= (((\textbf{rec}\ x.\ \alpha.x + \beta.x) \mid (\textbf{rec}\ x.\ \alpha.x + \gamma.x)) \mid \textbf{rec}\ x.\ \overline{\alpha}.x) \backslash \alpha \\
\mu &= \tau
\end{aligned}
$$

and thus:

$$((p \mid q) \mid r) \backslash \alpha \xrightarrow{\tau} ((p \mid q) \mid r) \backslash \alpha$$

Note that during the derivation we had to choose several times between different rules which could have been applied; while in general it may happen that wrong choices can lead to dead ends, our choices have been made so to complete the derivation satisfactorily, avoiding any backtracking. Of course other transitions are possible for the agent $((p \mid q) \mid r) \backslash \alpha$: we leave it as an exercise to identify all of them and draw the complete LTS (see Problem 11.1).

Example 11.3 (Dynamic stack: linking operator). Let us consider again the extensible stack from Example 11.1. We show how to formalise in CCS the linking operator $\smile$. We need two new channels ϑ and η, which will be private to the concatenated cells. Then, we let:

$$p \smile q = (p[\phi_{\beta,\delta}] \mid q[\phi_{\alpha,\gamma}]) \backslash \vartheta \backslash \eta$$

where $\phi_{\beta,\delta}$ is the relabelling that switches β with ϑ, δ with η and is the identity otherwise, while $\phi_{\alpha,\gamma}$ switches α with ϑ, γ with η and is the identity otherwise. Notably, ϑ and η are restricted, so that their scope is kept local to p and q, avoiding any conflict on channel names from the outside. For example, messages sent on β by p are redirected to ϑ and must be received by q that views ϑ as α. Instead, messages sent on β by q are not redirected to ϑ and will appear as messages sent on β by the whole process $p \smile q$ (see Figure 11.8).

Example

$$\left((\alpha.p_1 + \beta.q_1) \mid (\overline{\alpha}.p_2 + \overline{\beta}.q_2)\right) \backslash \alpha \backslash \beta \quad \begin{array}{l} \xrightarrow{\tau} p_1 \mid p_2 \\ \xrightarrow{\tau} q_1 \mid q_2 \end{array}$$

$$\dfrac{\dfrac{\dfrac{\alpha.p_1 \xrightarrow{\alpha} p_1}{\alpha.p_1 + \beta.q_1 \xrightarrow{\alpha} p_1} \qquad \dfrac{\overline{\alpha}.p_2 \xrightarrow{\overline{\alpha}} p_2}{\overline{\alpha}.p_2 + \overline{\beta}.q_2 \xrightarrow{\overline{\alpha}} p_2}}{(\alpha.p_1 + \beta q_1) \mid (\overline{\alpha}.p_2 + \overline{\beta}.q_2) \xrightarrow{\tau} p_1 \mid p_2}}{\dfrac{\vdots}{((\alpha.p_1 + \beta.q_1) \mid (\overline{\alpha}.p_2 + \overline{\beta}.q_2)) \backslash \alpha \backslash \beta \xrightarrow{\tau} p_1 \mid p_2}}$$

Bisimilarity

Two agents p_1 and p_2 are bisimilar $p_1 \simeq p_2$ when they are bisimilar when considered as states of the LTS defined by the inference rules.

Congruence

For CCS (strong) bisimilarity is a congruence, namely for every CCS context $C[-]$ without recursion we have

$$p \simeq q \implies C(p) \simeq C(q)$$

This means that abstract semantics is compositional, i.e. replacing any agent for an equivalent agent the semantics (behaviour) does not change.

De Simone Format

Given a signature Σ and a set of labels L an inference rule is in De Simone format if it has the form

$$\frac{\{ x_i \xrightarrow{l_i} y_i \mid i \in I \}}{op(x_1, \dots, x_n) \xrightarrow{l} p}$$

where $op \in \Sigma$, $I \subseteq \{1, \dots, n\}$ p is linear in $\{y_i\}_{i=1,\dots,n}$ and $x_i, y_i \neq x_j, y_j$ except for $y_i = x_i$ if $i \notin I$.

Bisimilarity is a congruence for all the calculi defined using De Simone inference rules.

The result has been extended to more general formats: tyft, GSOS, etc.

For calculi in these formats, compositionality descends from the coexistence of algebraic and coalgebraic aspects within the same structure. These structures are called structured coalgebras or bialgebras.

Structured Coalgebras

$\underline{Alg}_\Sigma$

$$\begin{tikzcd} A \arrow[r, "h"] \arrow[d, "f"'] & B \arrow[d, "g"] \\ F(A) \arrow[r, "F(h)"'] & F(B) \end{tikzcd}$$

Structured coalgebras are coalgebras in a category of algebras $\underline{Alg}_\Sigma$ rather than in $\underline{Set}$. Thus now h is at the same time an arrow in $\underline{Alg}_\Sigma$ (i.e. a Σ-homomorphism) and an arrow in $\underline{Coalg}_F$, (i.e. its kernel is a bisimulation).

Often $A = T_\Sigma$, i.e. A is the initial object in $\underline{Alg}_\Sigma$, which is the case when the states in A are syntactic terms (as for CCS).

For labelled transition systems in De Simone format we have:

$|F(X)| = \mathcal{P}_f(L \times |X| + |X|)$ on carriers and whenever

$$\frac{\{x_i \xrightarrow{l_i} y_i \mid i \in I\}}{op(x_1, \ldots, x_n) \xrightarrow{l} p}$$

then

$$\frac{p_i \in X \quad i = 1, \ldots, n}{op^X(p_1 \ldots p_n) \in op^{F(X)}(p_1 \ldots p_n)}$$

De Simone rules specify the operations of $F(A)$

$$\frac{\langle l_i, p_i \rangle \in S_i,\ i \in I \qquad q_j \in S_j,\ j \notin I}{\langle l, p[p_i/y_i,\ i \in I,\ q_j/y_j,\ j \notin I] \rangle \in op^{F(X)}(S_1, \ldots, S_n)}$$

$$(F(h))(S) = \{\langle l, h(p) \rangle \mid \langle l, p \rangle \in S \cap (L \times |X|)\} \cup \{h(p) \mid p \in S \cap |X|\}$$

The Structured Coalgebra of CCS

$\mathbf{Alg}_{\Sigma_{CCS}}$

$$\begin{tikzcd} T_{\Sigma_{CCS}} \arrow[d, "f"'] \\ F(T_{\Sigma_{CCS}}) \end{tikzcd}$$

$\leftarrow$ the unique arrow from the initial object.

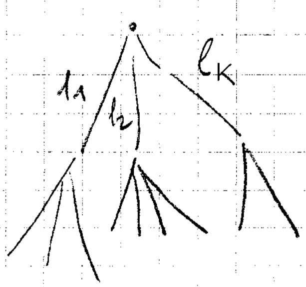
$$\begin{aligned} S_1 | S_2 = \; & \{ \langle \mu, p_1|p_2 \rangle \mid \langle \mu, p_1 \rangle \in S_1,\ p_2 \in S_2 \} \cup \\ & \{ \langle \mu, p_1|p_2 \rangle \mid p_1 \in S_1,\ \langle \mu, p_2 \rangle \in S_2 \} \cup \\ & \{ \langle \tau, p_1|p_2 \rangle \mid \langle \lambda, p_1 \rangle \in S_1,\ \langle \bar{\lambda}, p_2 \rangle \in S_2 \}. \end{aligned}$$

similarly for the other operations.

The final coalgebra Fix F contains the solutions of the domain equation

$$|F(X)| \sim \mathcal{P}_f(L \times |X| + |X|)$$

which are synchronization trees of the form:



synch$(p|q) =$ synch$($ [illegible]

The image of f through the unique arrow in Coalg_F from f to Fix F gives the minimal realization of CCS.

[illegible]

Exercise 2

A Petri net is a semi c-monoidal graph N with the monoid of nodes freely generated by the set of places:

$N^{node} = S^{\otimes}$ $\qquad \varepsilon$ is the identity of $S^{\otimes}$

$N^{arc} = T$

$s(t) = {}^{\bullet}t$

$d(t) = t^{\bullet}$

The ~~cat~~ graph $C[N]$ is the reflexive c-monoidal graph freely generated by N:

$$\frac{u \in N^{node}}{u \in C[N]^{node}} \qquad \frac{t : u \to v \in N^{arc}}{t : u \to v \in C[N]^{arc}} \qquad \frac{u \in C[N]^{node}}{id_u : u \to u \in C[N]^{arc}}$$

$$(\alpha)\ \frac{t_1 : u_1 \to v_1 \ ,\ t_2 : u_2 \to v_2 \in C[N]^{arc}}{t_1 \otimes t_2 : u_1 \otimes u_2 \to v_1 \otimes v_2 \ \in C[N]^{arc}}$$

with all the axioms.

The ~~cat~~ graph $C[N]$ can be seen as a coalgebra, where function f is defined by the inference rule:

Set

$$\begin{array}{c} S^{\otimes} \\ \downarrow f \\ \mathcal{P}_{fin}(T^{\otimes} \times S^{\otimes}) = F(S^{\otimes}) \end{array}$$

$$\frac{t = id_u \otimes \bigotimes_i t_i \in C[N]^{arc},\ t_i \in N^{arc},\ i = 1, \dots, n}{(\otimes_i t_i, t^{\bullet}) \in f({}^{\bullet}t)}$$

i.e. $f(u) = \{(t, v) \mid u\ [t\rangle\ v\} \cup \{(1, u)\}$

Notice that the label of a transition $t \in C[N]$, when included in the LTS f, is not t, but only the part $\otimes_i t_i$ of t corresponding to transitions of the net, excluding idle tokens.

In the example $S=\{a,b,c\}$ and $T=\{t\}$ with $^\circ t = a\otimes b$ and $t^\circ = c$, bisimilarity is the equivalence relation generated on $S^\otimes$ by:

$$u \cong u\otimes c \qquad na\otimes mb \cong Ka\otimes Kb \quad \text{where } K=\min(n,m)$$

thus in particular $a\cong\varepsilon$ but $a\otimes b \neq b$. Notice that if the idle tokens of a transitions were observable, then only identical markings would be bisimilar. Even if the fact that bisimilarity is not a congruence makes sure that coalgebra f in <u>Set</u> cannot be lifted to an algebra g in $\underline{Alg}_{\Gamma_{CMon}} = \underline{CMon}$, <u>CMon</u> let us try to do it.

$$\begin{array}{c} S^\otimes \\ \Big\downarrow g \\ \mathcal{P}_{fin}(T^\otimes \times S^\otimes) = G(S^\otimes) \end{array}$$

The problem is that $G(X)$ must be a Γ_{CMon}-algebra. The only reasonable definition is

(this rule is different if there are transitions with empty precondition)

$$\frac{(t_1,v_1)\in S_1 \quad (t_2,v_2)\in S_2}{(t_1\otimes t_2,\ v_1\overset{X}{\otimes} v_2)\in S_1\otimes^{G(X)} S_2} \qquad 1^{G(X)}=\{(id_\varepsilon,\varepsilon)\}$$

it is easy to check that these operations satisfy the axioms

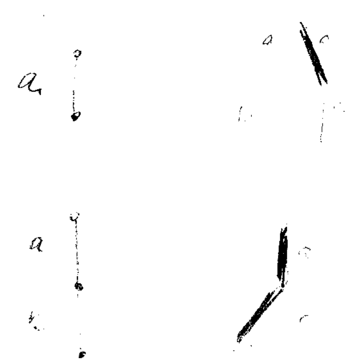
But f defined previously is not a homomorphism in our example.

$f(a)=\{(id_\varepsilon,a)\}$ $\quad f(b)=\{(id_\varepsilon,b)\}$ $\quad f(a)\otimes f(b)=\{(id_\varepsilon,a\otimes b)\}$

while $f(a\otimes b)=\{(id_\varepsilon,a\otimes b),(t,c)\}$

Exercise

Given a finite Petri net N with set of places S and set of transitions T, let $\mathcal{C}[N]$ be its case graph. Interpret $\mathcal{C}[N]$ as a coalgebra $f: S^{\otimes} \to \mathcal{P}_{fin}(T^{\otimes} \times S^{\otimes})$ in **Set**, where $\mathcal{P}_{fin}$ is the finite powerset functor and $X^{\otimes}$ is the commutative monoid freely generated by X. As an example, show that if $S = \{a, b, c\}$ and $T = \{t : a \otimes b \to c\}$, than a and the empty marking 1 are bisimilar. Notice however that markings $a \otimes b$ and b are not bisimilar, i.e. bisimilarity is not a congruence w.r.t the monoidal operation $\otimes$. Conclude that the above coalgebra cannot be seen as a coalgebra $g: S^{\otimes} \to \mathcal{Q}_{fin}(T^{\otimes} \times S^{\otimes})$ in $\mathbf{Alg}_{\Gamma^{CMon}}$, with Γ^{CMon} the theory of commutative monoids and $\mathcal{Q}_{fin}$ some poweralgebra functor. Show instead (optional) that this is possible for the class of Petri nets where the precondition ${}^{\bullet}t$ of each transition t is a singleton, and thus that, for this class of nets, bisimilarity is a congruence.



In the special case where the precondition $^{\bullet}t$ of each $t \in H^{arc}$ is a singleton and no transition has an empty precondition, f is always a homomorphism, i.e.

$$f(u_1 \otimes^{S^\otimes} u_2) = f(u_1) \otimes^{G(S^\otimes)} f(u_2).$$

$$f(\varepsilon) = 1^{G(S^\otimes)}$$

The second equation is obvious under the above conditions. The first equation in the direction $\supseteq$ is also obvious, since if $(t, v) \in f(u_1) \otimes^{G(S^\otimes)} f(u_2)$ then it is generated by the rule defining $\otimes^{G(S^\otimes)}$, i.e. there are (t_1, v_1) in $f(u_1)$ and (t_2, v_2) in $f(u_2)$ with $t = t_1 \otimes t_2$, $u = u_1 \otimes u_2$ and $v = v_1 \otimes v_2$. But the same transition is also in $f(u_1 \otimes^{S^\otimes} u_2)$, since $t_1' \otimes t_2' : u_1 \otimes u_2 \to v_1 \otimes v_2$ (is in $C[N]^{arc}$), where t_1' and t_2' correspond to (t_1, v_1) and (t_2, v_2) including idles.

The other direction descends from the fact that if a transition from u belongs to $C[N]^{arc}$, then there is a proof for it in the inference system for $C[N]$ where its last step uses $\otimes$ for every u_1, u_2 with $u = u_1 \otimes u_2$. This is not true if some precondition is not a singleton.