Structural operational specifications and trace automata

Eric Badouel, Philippe Darondeau

UNITÉ DE RECHERCHE
INRIA-RENNES

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
BP 105
78153 Le Chesnay Cedex
France

Tel. (1) 39 63 55 11

Rapports de Recherche

1992

25ème anniversaire

N° 1631

Programme 2
Calcul Symbolique, Programmation
et Génie logiciel

STRUCTURAL OPERATIONAL SPECIFICATIONS AND TRACE AUTOMATA

Eric BADOUEL
Philippe DARONDEAU

Mars 1992

RR-1631

IRISA

INSTITUT DE RECHERCHE EN INFORMATIQUE
ET SYSTEMES ALEATOIRES

Campus Universitaire de Beaulieu
35042 - RENNES CEDEX FRANCE
Tél. : 99 84 71 00 - Télex : UNIRISA 950 473 F
Télécopie : 99 38 38 32

Structural operational specifications and trace automata

ERIC BADOUEL - PHILIPPE DARONDEAU

Irisa, Campus de Beaulieu , 35042 Rennes Cedex - France
E-mail : ebadouel@irisa.fr - darondeau@irisa.fr

Publication Interne n° 629 - Janvier 1992 - 36 pages

Programme 2

Abstract : *Structural Operational Specifications (SOS) are supplied with concurrent models based on permutations of proved transitions, in the form of trace automata which are deterministic automata equipped with an explicit relation of independence on actions. In order to characterize the finite trace automata which are realized by SOS-algebras, we introduce a new kind of nets which encode exactly the concurrent behaviour of systems specified in SOS and we establish a correspondence between nets and those 'separated' trace automata which are realized in SOS.*

Spécifications opérationnelles structurelles et automates traces

Résumé : *On construit des modèles concurrents fondés sur les permutations de transitions prouvées pour des langages définis par un ensemble de spécifications opérationnelles structurelles (SOS). Ces modèles prennent la forme d'automates traces qui sont des automates déterministes munis d'une relation explicite d'indépendance sur les actions. Afin de caractériser les automates traces finis réalisables en SOS, on introduit un nouveau type de réseaux qui codent exactement le comportement concurrent des systèmes spécifiables en SOS, et on établit une correspondance entre ces réseaux et les automates traces dits 'séparés' qui sont réalisables en SOS.*

Structural operational specifications and trace automata

ERIC BADOUEL - PHILIPPE DARONDEAU

Irisa, Campus de Beaulieu , 35042 Rennes Cedex - France
E-mail : ebadouel@irisa.fr - darondeau@irisa.fr

Abstract

Structural Operational Specifications (SOS) are supplied with concurrent models based on permutations of proved transitions, in the form of trace automata which are deterministic automata equipped with an explicit relation of independence on actions. In order to characterize the finite trace automata which are realized by SOS-algebras, we introduce a new kind of nets which encode exactly the concurrent behaviour of systems specified in SOS and we establish a correspondence between nets and those 'separated' trace automata which are realized in SOS.

1 Introduction

We show that Structural Operational Specifications (SOS) induce models which exhibit the concurrent behaviour of programs, extending Boudol and Castellani's construction for finite CCS [BC88]. Their approach, based on permutations of proved transitions, is to decide, for each pair of co-initial transitions a and b, whether a survives b and symmetrically. Transitions a and b are said to be independent when they both survive the other, in which case they may be amalgamated to a concurrent transition $a.(b/a) = b.(a/b)$, where b/a is the residual of b after a. Since we intend to cover a variety of specifications, the definition of independence should conform to the obligation to deal uniformly with all operators in the signature, forbidding us to distinguish so-called parallel operators. It appears that no satisfactory rule of independence can be stated unless every operator is replaced by exactly one operator in every transition, maintaining the structure of terms. We therefore concentrate on a restriction of De Simone's format [deS84], called *basic SOS*, which ensure that property, and still covers all existing process algebras. Operational specifications given in basic SOS may equivalently be translated to *SOS automata* whose states are term constructors and whose transitions are labelled by proof constructors encoding the inductive rules of the specification. Transition systems induced from process terms are called *process automata*. Each process automaton is a synchronized product of copies of the SOS automaton indexed by the (invariant) tree domain of the process term. In the SOS automaton, pairs of independent moves are exhibited by diamonds; and in the process automaton, two transitions are independent if and only if they project to (locally) independent moves at each occurrence in the tree domain.

Both SOS automata and process automata happen to be instances of trace automata introduced by Stark (see [Sta89b] and [Sta89c]). Those are ordinary deterministic automata equipped with an explicit relation of independence on actions. Trace automata

provide a concrete representation for a subset of Stark's concurrent transition systems [Sta89a] in which transitions are classes for the equivalence by permutations. Boudol and Castellani's model for finite CCS falls in that case, and would lead to infinite trace automata if rational process terms were taken into account. Although most statements expressed in this paper do not depend on the finiteness of terms or automata, the main question we address is the realization of finite trace automata in SOS. The answer is non trivial, and we shall elaborate the example of a finite trace automaton which has no realization in SOS. In order to solve the above question, we introduce a new kind of (SOS) nets which encode exactly the concurrent behaviour of SOS processes, and we construct an adjunction between the so-called saturated nets and separated trace automata, preserving that behaviour. For that purpose, we follow the idea of Nielsen, Rozenberg, and Thiagarajan [NRT90] to form nets places from regions in automata, defined as set of states uniformly entered or exited by all transitions labelled by the same action. The concept of region was introduced originally to relate elementary transition systems and elementary nets. An adapted version was used by Winskel [Win91] to support an adjunction between Bednarczyk's asynchronous transition systems [Bed88] and a variant of elementary nets. In both kind of nets, transitions normally wait for emptiness of their output places. On the contrary, regions associated with process automata correspond to places that may be emptied and refilled by their input and output transitions. We notice that the axiom of uniform commutation for synchrounous transition systems (condition 4.1.1.(ii) in [Bed88]) is not satisfied by process automata, which do not either fit in with elementary transition systems due to self looping states and multiple transitions between pair of states. Now, not every trace automaton corresponds to a net: This holds only for those trace automata which satisfy an axiom of separation ensuring that they have enough regions to separate a state disabling an action from the set of states where it is allowed. Similarly, not every net corresponds to a trace automaton: this holds only for those nets which enjoy a property of saturation dual to the property of separation. We show that every net is equivalent to a saturated net, and the finite trace automata realized by SOS processes are exactly those which satisfy the axiom of separation.

Boudol and Castellani have established in [BC90] the consistency of the model based on proved transitions with two different models for CCS, constructed in the respective settings of flow event structures and flow nets. A similar fact holds in general for arbitrary process algebras defined in basic SOS: every specification in basic SOS may be given three equivalent models in the respective frameworks of trace automata, (SOS) nets, and event structures with a binary conflict. The justification for that claim is twofold. On the one hand, Stark's correspondence between concurrent transition systems and event structures with arbitrary conflict restricts to a correspondence between trace automata and event structure with binary conflict, preserving domain of configurations (this observation will not be elaborated in the present paper). On the other hand, the adjunction between separated trace automata and saturated nets preserves concurrent behaviour, and we obtain by the way compositional net models for structural operational specifications.

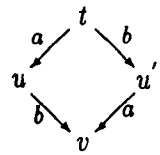
The body of the paper is organized as follows. Section (2) introduces basic SOS, and constructs concurrent models for that format in the setting of trace automata. Section (3) introduces nets and constructs the adjunction between separated trace automata and saturated nets. Relying on that adjunction, we show in section (4) that the finite trace automata realized in SOS are exactly the separated ones.

2 Diamonds in SOS

The objective of this section is to construct inductively from structural operational specifications, see [Plo81], process automata where diamonds reflects concurrency. In a first attempt, let us identify process graphs with pointed transition graphs whose states are the finite or infinite terms over some signature Σ, and whose labelled transitions are proved from some set R of rules which conform to the de Simone format [deS84]:

$$\frac{u_{i_1} \xrightarrow{a_1} v_{i_1}, \ldots, u_{i_k} \xrightarrow{a_k} v_{i_k}}{f(u_1, \ldots, u_n) \xrightarrow{a} C[v_1, \ldots, v_n]}$$

where all variables $\{u_1, \ldots, u_n, v_{i_1}, \ldots, v_{i_k}\}$ are different, and $v_j = u_j$ for $j \notin \{i_1, \ldots, i_k\}$, while $f \in \Sigma_n$ is an n-ary operator symbol, and $C[v_1, \ldots, v_n]$ is a Σ-expression. where each variable v_j occurs at most once. A diamond is then any subgraph of the form:



where opposite transitions bear identical labels and $a \neq b$. Thus, both terms ($a.b.nil + b.a.nil$) and $a.nil \| b.nil$ originate diamonds in CCS, see [Mil80], even though the first diamond does not reflect concurrency. In order to correct that fault, one may consider a different graph whose arcs are labelled by proofs of transitions. However, labelling arcs by plain proofs is too strong: two transitions with the same proof must be identical and the second diamond of the example therefore disappears, even though it reflects concurrency. Arcs of the proved transition graph must therefore be labelled by *schematic proofs* abstracting from the sources of labelled transitions. Let us give a definition before resuming the discussion. We recall that a Σ-tree is a partial mapping $T : (I\!N_+)^* \to \Sigma$ whose domain $Dom(T)$ is a prefix closed set of paths (encoded by sequences of positive integers) including the empty path ϵ, and such that $T(s) \in \Sigma_k \Rightarrow (s.i \in Dom(T)$ **iff** $i \in \{1, \ldots, k\})$.

Definition 2.1 (Schematic proofs) *Given a set R of SOS rules for Σ-terms, we shall also let R denote the signature formed by extracting from each rule a corresponding operator $\rho = <\{i_1, \ldots, i_k\}, <a_1, \ldots, a_k, a>, C(1, \ldots, n)>$ with arity k. The action a is called the action of ρ, denoted as $a = act(\rho)$. A schematic proof is then a finite R-tree A, satisfying for every $s \in Dom(\mathrm{A})$ the following condition of local correctness:*

$$\mathrm{A}(n) = <\{i_1, \ldots, i_k\}, <a_1, \ldots, a_k, a>, C(1, \ldots, n)> \Rightarrow [(\forall j \in \{1, \ldots, k\} \quad act(\mathrm{A}(n.j)) = a_j)]$$

By forgetting the sources of all moves, proofs for transitions $t \xrightarrow{a} u$ are projected to schematic proof trees A where a is the action of the root $\mathrm{A}(\epsilon)$. Now, the proved transition graph $\Gamma(\Sigma, R)$ is the graph formed from the resulting arcs $t \xrightarrow{\mathrm{A}} u$.

Let us examine on examples whether diamonds in the proved transition graph are representative of concurrency. Suppose R is Milner's set of rules for CCS. Then ($a.nil \| b.nil$) is the origin of a diamond in $\Gamma(\Sigma, R)$ whereas ($a.b.nil + b.a.nil$) is not. Unfortunately, no diamond originates from ($(a.nil \| b.nil) + c.nil$), because the schematic proof :

$$\mathrm{A} = <\{1\}, <a, a>, 1> (<\{1\}, <a, a>, 1 \| 2> (<\{\}, <a>, 1>))$$

$$
\begin{array}{rcll}
 & \top & & <\{\},<a>,1> \\
a.nil & \xrightarrow{a} & nil & \\
 & \top & & <\{1\},<a,a>,1\|2> \\
a.nil\|b.nil & \xrightarrow{a} & nil\|b.nil & \\
 & \top & & <\{1\},<a,a>,1> \\
(a.nil\|b.nil)+c.nil & \xrightarrow{a} & nil\|b.nil &
\end{array}
$$

Figure 1: A proof and its schematic encoding

while it applies to $((a.nil\|b.nil)+ c.nil)$ and then produces the proof shown in figure (1), encoded by the proved transition $((a.nil\|b.nil)+ c.nil) \xrightarrow{A} nil\|b.nil$, does not apply to $(a.nil\|nil)$. In that case, the apparent cause of failure is that operator + vanishes when acted upon: CCS-transitions do not strictly maintain the structure of terms. A different cause of failure may be observed for $\Sigma = \{f, nil\}$ when R is the set of rules:

$$\vdash f(u) \xrightarrow{a} f(f(u)) \quad u \xrightarrow{a} v \vdash f(u) \xrightarrow{a} f(v)$$

Let $R = \{\varphi, \psi\}$ with $\varphi =<\{1\},<a,a>,f(f(1))>$ and $\psi =<\{\},<a,a>,f(1)>$, then $f(f(nil)) \xrightarrow{\varphi} f(f(f(nil))) \xrightarrow{\psi(\varphi)} f(f(f(f(nil))))$ and $f(f(nil)) \xrightarrow{\psi(\varphi)} f(f(f(nil))) \xrightarrow{\varphi} f(f(f(f(nil))))$. Unfortunately, this diamond does not represent concurrency: the computation proved by the sequence $\varphi.\psi(\varphi)$ is inherently sequential, while the computations that duplicate the two occurrences of f in parallel are proved by $\psi(\varphi).\varphi$ and $\varphi.\psi(\psi(\varphi))$ which form no diamond! The apparent cause of failure is again that transitions do not strictly maintain the structure of terms, since precisely the operator f may duplicate itself. In order to avoid disagreements between the intuitive notion of concurrency on the one hand and the notion of concurrency based on diamonds on the other hand, we should impose further restrictions on De Simone's format. We call *basic SOS* the restriction defined by forcing rules ρ to the basic format:

$$\frac{u_{i_1} \xrightarrow{a_1} v_{i_1}, \ldots, u_{i_k} \xrightarrow{a_k} v_{i_k}}{f(u_1, \ldots, u_n) \xrightarrow{a} g(v_1, \ldots, v_n)}$$

So, in basic SOS the structure of terms is strictly maintained by transitions: terms t and u have identical domains whenever $t \xrightarrow{a} u$. The usual definition of CCS in De Simone's format does not fit in basic SOS, but CCS may nevertheless be redefined in basic SOS. The price to pay is the introduction of three auxiliary operators (id, π_1 and π_2 such that e.g. $\alpha.nil + nil \xrightarrow{\alpha} \pi_1(id(nil), nil)$). The alternative definition of CCS may be found in table (1).

An operational specification in basic SOS may be equivalently represented by a graph $\gamma(\Sigma, R)$ on Σ, called an SOS-automaton, in which each SOS rule

$$\frac{u_{i_1} \xrightarrow{a_1} v_{i_1}, \ldots, u_{i_k} \xrightarrow{a_k} v_{i_k}}{f(u_1, \ldots, u_n) \xrightarrow{a} g(v_1, \ldots, v_n)}$$

is encoded by an arc $f \xrightarrow{\rho} g$ labelled by $\rho = <\{i_1, \ldots, i_k\}, <a_1, \ldots, a_k, a>, g>$. Thus, in an SOS automaton $\gamma(\Sigma, R)$, states $f \in \Sigma$ are operators on process terms and transition labels

$$nil \quad : \quad \text{no rule}$$

$$a.- \quad : \quad a.t \xrightarrow{a} id(t)$$

$$id \quad : \quad \frac{t \xrightarrow{\mu} t'}{id(t) \xrightarrow{\mu} id(t')}$$

$$+ \quad : \quad \frac{t \xrightarrow{\mu} t'}{t+u \xrightarrow{\mu} \pi_1(t',u)} \qquad \frac{u \xrightarrow{\mu} u'}{t+u \xrightarrow{\mu} \pi_2(t,u')}$$

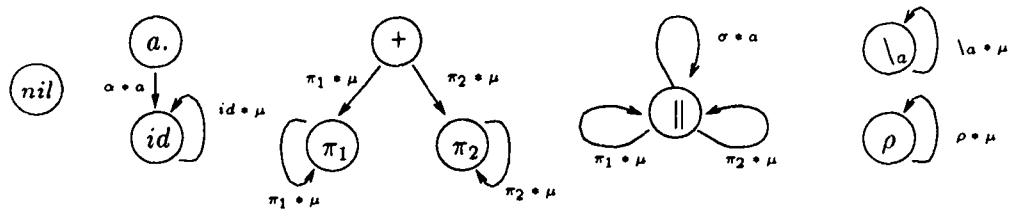
$$\pi_i \quad : \quad \frac{t \xrightarrow{\mu} t'}{\pi_1(t,u) \xrightarrow{\mu} \pi_1(t',u)} \qquad \frac{u \xrightarrow{\mu} u'}{\pi_2(t,u) \xrightarrow{\mu} \pi_2(t,u')}$$

$$\| \quad : \quad \frac{t \xrightarrow{\mu} t'}{t\|u \xrightarrow{\mu} t'\|u} \qquad \frac{u \xrightarrow{\mu} u'}{t\|u \xrightarrow{\mu} t\|u'} \qquad \frac{t \xrightarrow{a} t', u \xrightarrow{\bar{a}} u'}{t\|u \xrightarrow{\tau} t'\|u'}$$

$$\backslash_a \quad : \quad \frac{t \xrightarrow{\mu} t'}{t\backslash_a \xrightarrow{\mu} t'\backslash_a} \quad \mu \notin \{a, \bar{a}\}$$

$$[\rho] \quad : \quad \frac{t \xrightarrow{\mu} t'}{t[\rho] \xrightarrow{\rho(\mu)} t'[\rho]}$$

Table 1: alternative SOS rules for CCS



KEY OF LABELS:

$\alpha * a = <\emptyset, \{a\}, id>$ $\quad$ $id * \mu = <\{1\}, <\mu, \mu>, id>$

$\pi_1 * \mu = <\{1\}, <\mu, \mu>, \pi_1>$ $\quad$ $\pi_2 * \mu = <\{2\}, <\mu, \mu>, \pi_2>$

$\sigma * a = <\{1,2\}, <a, \overline{a}, \tau>, \|>$

$\|_1 * \mu = <\{1\}, <\mu, \mu>, \|>$ $\quad$ $\|_2 * \mu = <\{2\}, <\mu, \mu>, \|>$

$\backslash_a * \mu = <\{1\}, <\mu, \mu\backslash_a>, \backslash_a>$ $\quad$ $\rho * \mu = <\{1\}, <b, \rho(\mu)>, \rho>$

Figure 2: the SOS-automaton for CCS

$\rho \in R$ are operators on proof terms. SOS automaton proceed from the same inspiration as the (more general) context-systems of Larsen and Xinxin [LX90]. It is worth noting that an SOS automaton associated with SOS rules generated from a finite set of schemes of rules, has a finite presentation as for instance the SOS automaton for CCS shown in figure (2), where a and μ are generic actions (respectively visible and arbitrary), $\backslash_a$ is the operator of restriction given by $\mu\backslash_a = \mu$ if $\mu \notin \{a, \overline{a}\}$ and undefined elsewhere, and $\rho : \Lambda \to \Lambda$ is a total renaming function.

An important property of the SOS-automaton is to generate the proved transition graph by means of synchronized products. This fact is made wholly clear in the restating of the definition of proved transition graph given below. First, we notice that the SOS automaton $\gamma(\Sigma, R)$ is deterministic since $f \xrightarrow{\rho} g$ implies that ρ is of the form $<\{i_1, \ldots, i_k\}, <a_1, \ldots, a_k, a>, g>$. This seems an appropriate place for fixing some notations: if $T \subset Q \times \Lambda \times Q$ is a deterministic transition system we use notation $x \xrightarrow{a}$ as an abbreviation for $\exists y \ . \ (x \xrightarrow{a} y)$, and we denote $x.a$ the unique y such that $(x \xrightarrow{a} y)$ when it exists. And we inductively define $q \xrightarrow{u}$ and $q.u$ for $q \in Q$ and $u \in A^*$ by: $q \xrightarrow{\epsilon}$ always holds with $q.\epsilon = q$ and $q \xrightarrow{a.u}$ **iff** $q \xrightarrow{a}$ & $q.a \xrightarrow{u}$ with $q.(a.u) = (q.a).u$.

Definition 2.2 (Proved transition graph) *Given a signature Σ and a set R of rules in basic SOS, the proved transition graph $\Gamma(\Sigma, R)$ is the deterministic transition system on set of (finite or infinite) Σ-terms $t \xrightarrow{\mathrm{A}} u$ labelled by schematic proofs such that, if we let φ_A denote the injective mapping $\varphi_A : Dom(A) \to 2^{(\mathbb{N})_+)^*}$ given by $\varphi_A(\epsilon) = \epsilon$ and $\varphi_A(n.j) = \varphi_A(n).i_j$, then a schematic proof* A *is enabled in t ($t \xrightarrow{\mathrm{A}}$) if, and only if*

1. A *matches the domain of t, i.e. $Im(\varphi_A) \subset Dom(t)$, and*

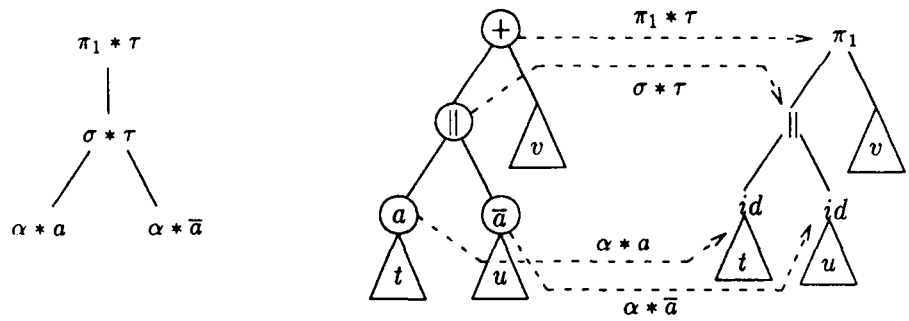


Figure 3: a proved transition and its schematic proof

2. *the proof is locally enabled, i.e.* $\forall s \in Dom(A) \quad t(\varphi_A(s)) \xrightarrow{A(s)}$ *in* $\gamma(\Sigma, R)$.

And then, the unique u *such that* $t \xrightarrow{A} u$ *is given by*

1. $\forall s \in Dom(A) \quad t(\varphi_A(s)) \xrightarrow{A(s)} u(\varphi_A(s))$ *in* $\gamma(\Sigma, R)$, *and*

2. $\forall s \notin Im(\varphi_A) \quad t(s) = u(s)$.

We notice that the part of $\Gamma(\Sigma, R)$ accessible from a Σ-term t corresponds to synchronized product of automata à la Arnold-Nivat (see [Niv79] and [AN82]). More precisely we have as many copies of the SOS-automaton $\gamma(\Sigma, R)$ as there are occurrences $s \in Dom(t)$ in the domain of t, synchronized by the set of vectors $V_A \in (R \cup \{\epsilon\})^{Dom(t)}$ defined from the schematic proofs A enabled in t by: $V_A(s) = A(\varphi_A^{-1}(s))$ if $s \in Im(\varphi_A)$ and $V_A(s) = \epsilon$ otherwise. The specific contribution of SOS is the inductive caracterization of the synchronization vectors via the structure of logical proofs. For instance, the proof term shown on the left of figure (3), encoding the proof

$$\frac{\dfrac{a.t \xrightarrow{a} id(t) \qquad \bar{a}.u \xrightarrow{\bar{a}} id(u)}{a.t \| \bar{a}.u \xrightarrow{\tau} id(t) \| id(u)}}{(a.t \| \bar{a}.u) + v \xrightarrow{\tau} \pi_1(id(t) \| id(u), v)}$$

may be seen as a synchronization vector with four non-empty components. It indicates that the four copies of the SOS-automaton dedicated to the circled nodes should synchronized their respective local moves, indicated by the dashed arrows, while the other copies standing at uncircled nodes stay inactive. The result of the synchronized move is the state vector represented by the rightmost term.

We will now concentrate our attention on concurrency and analyze in terms of synchronized products the circumstances under which diamonds appear in the proved transition graph.

Definition 2.3 (Diamond) *A diamond* $\diamond(q, a, b)$ *in a deterministic transition system* $T \subset Q \times \Lambda \times Q$ *consists of a state* $q \in Q$ *and a pair of distinct actions* $a, b \in \Lambda$ *such that* $q \xrightarrow{a}$, $q \xrightarrow{b}$ *and* $q.ab = q.ba$:

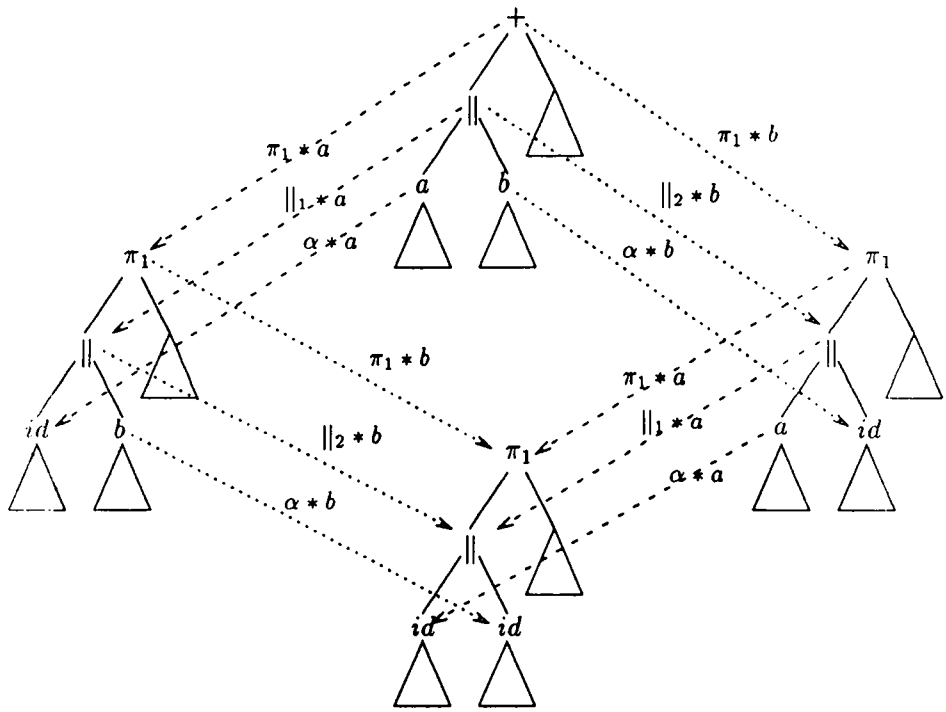
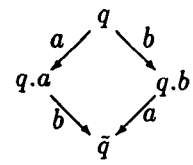
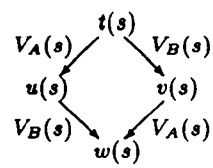


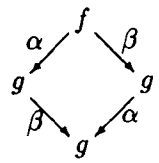
Figure 4: a diamond situation



Such a diamond represents two actions a and b performed in parallel. Since the subgraph of $\Gamma(\Sigma, R)$ accessible from t coincides with an accessible subgraph of the synchronized product $\gamma(\Sigma, R)^{Dom(t)}$, there is a diamond $\diamond(t, A, B)$ in $\Gamma(\Sigma, R)$ if and only if, $A \neq B$ and for every $s \in Dom(t)$, $V_A(s) = \epsilon$ or $V_B(s) = \epsilon$ or there is a *local diamond* $\diamondsuit(t(s), V_A(s), V_B(s))$ in $\gamma(\Sigma, R)$ i.e. a subgraph



with possibly $V_A(s) = V_B(s)$. A typical diamond in $\Gamma(\Sigma, R)$ is shown in figure (4). By the way, since $f \xrightarrow{\alpha} g$ implies $\alpha = <\{i_1, \ldots, i_k\}, <a_1, \ldots, a_k, a>, g>$, it may be observed that a local diamond is any subgraph of $\gamma(\Sigma, R)$ of the particular form:



with possibly $\alpha = \beta$. Now, we can define the independence of schematic proofs.

Definition 2.4 (Independence of schematic proof) *Two schematic proofs* A *and* B *are independent (* A || B *) if and only if* A $\neq$ B *and* $\forall s \in Dom($ A $) \cap Dom($ B $)$. A (s)|| B (s), *where two proof constructors* $\alpha = <\{i_1, \ldots, i_k\}, <a_1, \ldots, a_k, a>, g>$ *and* $\beta = <\{j_1, \ldots, j_l\}, <b_1, \ldots, b_l, b>, h>$ *are independent* $(\alpha \| \beta)$ *if and only if* $g = h$, $g \xrightarrow{\alpha}$ *and* $g \xrightarrow{\beta}$.

Notice that $\alpha \| \beta$ if and only if there exists some local diamond $\diamondsuit(f, \alpha, \beta)$ in the SOS automaton (for the only-if part take $f = g$). And then there is a local diamond $\diamondsuit(f, \alpha, \beta)$ in every state f at which α and β are both enabled. Due to the structure of synchronized product, those observations are also valid for schematic proofs in the proved transition graph: when two distinct proofs A and B apply to some common process term,

$$\text{A} \,\|\, \text{B} \iff \forall t \; [(t \xrightarrow{\text{A}} \; \& \; t \xrightarrow{\text{B}}) \Rightarrow \diamondsuit(t, \text{A}, \text{B})]$$

and without that provision,

$$\text{A} \,\|\, \text{B} \iff \exists t \; \diamondsuit(t, \text{A}, \text{B})$$

It should be clear from the following definition, recalled from [Sta89b], that proved transition graphs $\Gamma(\Sigma, R)$ are instances of Stark's trace automata.

Definition 2.5 (Trace automata) *An automaton* $A = (E, Q, T)$ *(respectively an automaton with initial state* $A = (E, Q, q_0, T)$*) consists of a set of events* E*, a set of states* Q *(resp. a set of states* Q *with initial state* $q_0 \in Q$*), and a transition relation* $T \subset Q \times E \times Q$*. It is a trace automaton when the alphabet* E *comes equipped with a symmetric and irreflexive binary relation* $\| \subset E \times E$*, called the concurrency relation, and when moreover the following two conditions are fullfilled.*

disambiguation : $p \xrightarrow{a} q$ *and* $p \xrightarrow{a} r \Rightarrow q = r$

commutativity : $(a \| b$ *and* $q \xrightarrow{a} r$ *and* $q \xrightarrow{b} s) \Rightarrow (s \xrightarrow{a} p$ *and* $r \xrightarrow{b} p$ *for some* $p)$

We have dropped the unit transitions $p \xrightarrow{\epsilon} p$ appearing in Stark's original definition which do not significantly affect the definition. Notice that SOS automata are not exactly trace automata, because their independence relations are not always irreflexive. Nevertheless we call them *local trace automata* (they satisfy the other conditions), because they generate the proved transition graphs by synchronized products.

Definition 2.6 (Local trace automata) *A local trace automaton* $A = (E, \|, Q, T)$ *(respectively a local trace automaton with initial state* $A = (E, \|, Q, q_0, T)$*) is an automaton (resp. an automaton with initial state) equipped with a symmetric relation* $\| \subset E \times E$ *fulfilling the property of disambiguation and commutativity.*

For synchronized systems we thus exclude auto-concurrency: two occurrences of the same event in a sequence $q \xrightarrow{a} q' \xrightarrow{a} q''$ cannot be amalgamated into a concurrent transition. Neverthess two concurrent events may coincide on some component of the synchronized system. This phenomenon is illustrated by auto-concurrency in the SOS automaton, where events describe only partially the activity of the system. Notice that, due to the property of commutativity, $\alpha \| \alpha$ and $q \xrightarrow{\alpha}$ entail $q.\alpha \xrightarrow{\alpha}$. Therefore, when auto-concurrency appears in a local trace automaton it is necessarily unbounded: an arbitrary number of actions α may be performed in parallel. But in the case of SOS automata, this unbounded auto-concurrency is only potential: the activity of the global system is limited to those synchronized vectors of local activities which represent actual proofs.

Trace automata generated from SOS specifications have the additional property that *independence reflects diamonds* in the sense that two distinct proofs A and B are independent if and only if there exists a diamond $\diamond(t, \mathrm{A}, \mathrm{B})$ for some process term t. They even have the following stronger property

Observation 2.7 *Independence of proof terms is reflected by commutation in traces, which means that for distinct* A *and* B, $t \xrightarrow{\mathrm{A.B}} u$ *and* $t \xrightarrow{\mathrm{B.A}} v$ *in* $\Gamma(\Sigma, R)$ *imply* $u = v$ *and* $\mathrm{A} \| \mathrm{B}$.

which follows from the analogous property of SOS automata:

Observation 2.8 *For any pair of proof constructors* α *and* β, $f \xrightarrow{\alpha.\beta} g$ *and* $f \xrightarrow{\beta.\alpha} h$ *in* $\gamma(\Sigma, R)$ *imply* $g = h$ *and* $\alpha \| \beta$.

Notice that the above statement does not require $\alpha \neq \beta$ and therefore $\alpha \| \alpha$ whenever there exist contiguous transitions $f \xrightarrow{\alpha} g$ and $g \xrightarrow{\alpha} h$ in the SOS automaton.

The section ends up with the definition of an interpretation for finite process terms in the setting of labelled trace automata, i.e. trace automata $(E, \|, Q, T)$ whose alphabet E comes equipped with a labelling function $\lambda : E \to \Lambda$. For a given specification, the set of labels Λ is fixed. Rational process terms could be dealt with by least fixed points in an adequate domain of labelled trace automata. In a trace automaton (or local trace automaton) $\mathcal{A} = (E, \|, Q, T)$, each state $q \in Q$ determines an accessible restriction $\mathcal{A} > q = (E', \|, Q', q, T')$, where Q' is the subset of states accessible from q, $T' = T \cap (Q' \times E \times Q')$, and E' is the restriction of E to the events occurring within transitions in T'. The restriction $\gamma(\Sigma, R) > f$ of the SOS automaton is denoted $\mathcal{A}_f$. The restriction $\Gamma(\Sigma, R) > t$ of the proved transition graph (where proof trees are labelled by their actions), called the process automaton of t, is denoted $[\![t]\!]^{\mathcal{A}}_{\Sigma,R}$. We construct from each $\mathcal{A}_f$ an operator $\tilde{f}_{\mathcal{A}}$ on labelled trace automata satisfying the relation:

$$[\![f(t_1, \ldots, t_n)]\!]^{\mathcal{A}}_{\Sigma,R} = \tilde{f}_{\mathcal{A}}([\![t_1]\!]^{\mathcal{A}}_{\Sigma,R}, \ldots, [\![t_n]\!]^{\mathcal{A}}_{\Sigma,R}).$$

Let $\mathcal{A}_f = (E_f, \|_f, Q_f, f, T_f)$ and for $i = 1 \ldots n$ let $\mathcal{A}_i = (E_i, \|_i, Q_i, t_i, T_i, , \lambda_i)$, then we set $\tilde{f}_{\mathcal{A}}(\mathcal{A}_1, \ldots, \mathcal{A}_n) = (E, \|, Q, T, \lambda) > f(t_1, \ldots, t_n)$ where E is the set of events $E = \{\rho(A_{i_1}, \ldots, A_{i_k}) / \rho = < \{i_1, \ldots, i_k\}, < a_1, \ldots, a_k, a >, g > \in E_f \ \& \ \forall p \ . \ A_{i_p} \in E_{i_p} \ \& \ \lambda_{i_p}(A_{i_p}) = a_p\}$, equipped with the labelling function $\lambda(\rho(A_{i_1}, \ldots, A_{i_k})) = act(\rho)$. The set of states Q is $\{g(u_1, \ldots, u_n) / g \in E_f \ \& \ \forall i \ . \ u_i \in Q_i\}$. The relation of independence is given by $A \| A'$ if and only if $A = \rho(A_{i_1}, \ldots, A_{i_k})$, $A' = \rho'(A'_{j_1}, \ldots, A'_{j_l})$ with

$$(\rho \|_f \rho') \ \& \ \forall p, q, m \ . \ (i_p = j_q = m \Rightarrow A_m \|_m A'_m)).$$

And the transition system is the synchronized product

$$g(u_1,\ldots u_n) \xrightarrow{\rho(A_{i_1},\ldots,A_{i_k})} h(v_1,\ldots v_n) \Longleftrightarrow \begin{cases} g \xrightarrow{\rho} h \text{ in } \mathcal{A}_f \\ \forall j \in \{1,\ldots,k\} \quad u_{i_j} \xrightarrow{A_{i_j}} v_{i_j} \text{ in } \mathcal{A}_i \\ u_i = v_i \text{ elsewhere} \end{cases}$$

A question which arises naturally is to find an axiomatic characterization for the class of (finite) process automata. In order to solve that question, we establish in the next section a connection between trace automata and a new type of nets introduced for that purpose. By the way, that connection induces an alternative interpretation for process terms in the setting of nets, exhibiting their concurrent behaviour.

3 Trace automata and nets

For any relation $\nabla \subset A \times B$ we shall denote $()^{\nabla} : A \to 2^B$ and ${}^{\nabla}() : B \to 2^A$ the mappings given by:

$$a \nabla b \quad \textbf{iff} \quad b \in a^{\nabla} \quad \textbf{iff} \quad a \in {}^{\nabla}b$$

We say that a family of subsets $X_i \subset A$ is a *partitioning* of A whenever they are pairwise disjoint: $X_i \cap X_j = \emptyset$ for $i \neq j$, and cover the whole of A: $\bigcup_i X_i = A$. It is like a partition except that some of the components of a partitioning may be empty. Notice that a family $\nabla_i \subset A \times B$ of binary relations is a partitioning of $A \times B$ **iff** for all $a \in A$, the family a^{∇_i} is a partitioning of B **iff** for all $b \in B$ the family ${}^{\nabla_i}b$ is a partitioning of A.

Definition 3.1 (Nets) *A net $\mathcal{N} = (P, A, \overset{0}{\leftarrow}, \overset{1}{\leftarrow}, \overset{0}{\rightarrow}, \overset{1}{\rightarrow}, \perp, M_0)$ consists of a set of places P, a set of actions (or events) A, flow relations $\overset{0}{\leftarrow}, \overset{1}{\leftarrow}, \overset{0}{\rightarrow}, \overset{1}{\rightarrow}, \perp$ which give a partitioning of $A \times P$, and an initial marking $M_0 \subset P$.*

If $M \subset P$ is a marking with $x \in M$ (respectively $x \notin M$) we say that the place x is *full* (resp. *empty*) for the marking M. x is said to be an *input place* for a if $a \leftarrow x$ where $\leftarrow = \overset{0}{\leftarrow} \cup \overset{1}{\leftarrow}$ and x is said to be an *output place* for a if $a \rightarrow x$ where $\rightarrow = \overset{0}{\rightarrow} \cup \overset{1}{\rightarrow}$. The behaviour of the net is the following: an action a is *enabled* at a marking M (in notation $M \xrightarrow{a}$) if every input place of a is full for the marking M, i.e. the input places are the *pre-conditions* for an action to be *fired*. When an action a is fired, the places x such that $a\mathbf{0}x$ (where $\mathbf{0} = \overset{0}{\leftarrow} \cup \overset{0}{\rightarrow}$) are emptied if they were not already empty, and the places x such that $a\mathbf{1}x$ (where $\mathbf{1} \overset{1}{\leftarrow} \cup \overset{1}{\rightarrow}$) are filled if they were not already full. Therefore, if $\mathcal{M} = 2^P$ is the set of *markings*, the set of transitions $\mathcal{T} \subset \mathcal{M} \times A \times \mathcal{M}$ for the net $\mathcal{N}$ is given by:

$$M \xrightarrow{a} M' \quad \textbf{iff} \quad a^{\leftarrow} \subset M \quad \text{and} \quad M' = (M \setminus a^{\mathbf{0}}) \cup a^{\mathbf{1}}$$

To sum up, there are two types of input places for an action a: the first type corresponds to pre-conditions that must hold for a to execute but which are unaffected by this action (we say that the action a *tests* the condition x when $a \overset{1}{\leftarrow} x$), whereas pre-conditions of the second type do no longer hold after a has executed (we say that the action a *consumes* the resource x when $a \overset{0}{\leftarrow} x$). As for the ouput places of an action a, they do not condition the firing of a but on the contrary, they are set unconditionally to a value at each execution of a. The remaining case when $a \perp x$ (read a and x are *orthogonal*) corresponds to conditions which are neither tested nor affected by the action a. A net may be depicted as shown in figure (5) where an arc is drawn from a place x (depicted by a circle) to an action a

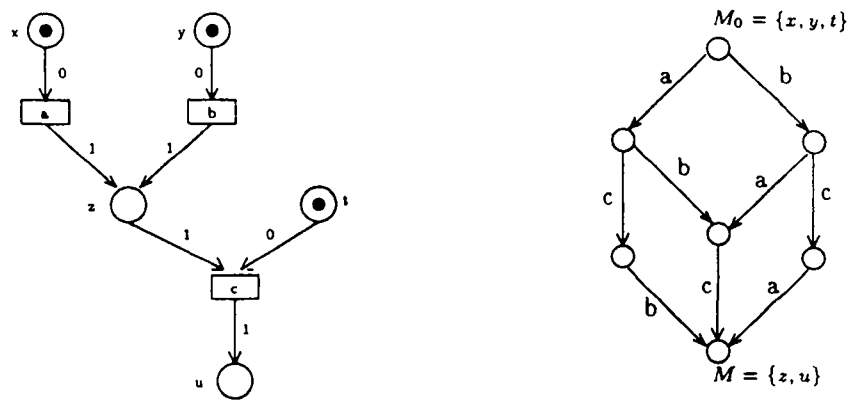


Figure 5: a net and its transition system

(depicted by a box) when x is an input place for a, and conversely an arc is drawn from an action a to every of its ouput places. An arc between a place and an action is labelled according to the value of the place after the execution of the action (empty: 0 or full: 1). A marking M may then be represented by setting one token in each of its places. In figure (5) we have also represented the transition system associated to the net with initial marking as indicated. In this example, the actions a and b consume their respective resources x and y and each of them turns on the condition z, the action c tests the condition z and thus awaits for a or b (or both) to be performed, and then consumes its resource t and turns on the condition u. Intuitively the actions a and b may be performed in parallel, and when for instance a has been fired, the actions b and c may also be performed in parallel. This is reflected by the presence of *diamond situations* $\diamond(M, a, b)$ in the transition system, which are defined by a marking M and two actions a and b such that $M \xrightarrow{ab}$, $M \xrightarrow{ba}$ and $M.ab = M.ba$. A diamond situation occurs when two independent actions are both enabled at a given marking where:

Definition 3.2 (Independence in nets) *Two* distinct *actions a and b in a net $\mathcal{N}$ are said to be* independent *(in notation $a\|b$) when* (i) *there exists an accessible marking M at which they are both enabled,* (ii) $a^{\overset{0}{\leftarrow}} \subset b^{\perp}$ & $a^{\mathbf{0}} \cap b^{\mathbf{1}} = \emptyset$, *and symmetrically* (iii) $b^{\overset{0}{\leftarrow}} \subset a^{\perp}$ & $b^{\mathbf{0}} \cap a^{\mathbf{1}} = \emptyset$.

Two actions are structurally independent (conditions *(ii)* and *(iii)*) when no resource consumed by one action is either tested or possibly modified by the other one, and they do not deliver incompatible values to shared places. The condition *(i)* is not structural but it ensures that all the information about independence is *visible* on the transition system in the sense that two actions a and b are independent if and only if they occur in a diamond situation. The following definition and proposition state this property more precisely.

Definition 3.3 (Canonical trace automata) *A trace automaton $(A, \|, Q, q_0, T)$ is said to be canonical when* (i) *every state is accessible from the initial state,* (ii) *each pair of independent actions is enabled in at least one state (giving rise to at least one diamond*

situation), and (iii) *independence reflects diamond situations: two* distinct *actions a and b are independent whenever $q \xrightarrow{ab}$, $q \xrightarrow{ba}$ and $q.ab = q.ba$ for some state q.*

Proposition 3.4 *Let $\mathcal{T}_a \subset \mathcal{M}_a \times A \times \mathcal{M}_a$ be the restriction of the transition system associated with a net $\mathcal{N}$ to its accessible markings $\mathcal{M}_a \subset 2^P$, then $(A, \|, \mathcal{M}_a, M_0, \mathcal{T}_a)$ is a canonical trace automaton.*

Proof: we prove first that independence *creates* diamond situations :

$$[a\|b \;\;\&\;\; (\exists M \in \mathcal{M}_a).(M \xrightarrow{a} \text{ and } M \xrightarrow{b})] \;\Rightarrow\; [M.a \xrightarrow{b} \text{ and } M.b \xrightarrow{a} \text{ and } M.ab = M.ba]$$

and second that independence reflects diamond situations, i.e. for every pair of distinct actions a and b:

$$(\exists M \in \mathcal{M}_a).(M \xrightarrow{ab} \text{ and } M \xrightarrow{ba} \text{ and } M.ab = M.ba) \;\Rightarrow\; a\|b$$

First, let us assume that $a\|b$, $\exists M \in \mathcal{M}_a$ such that $M \xrightarrow{a}$ and $M \xrightarrow{b}$. The action b is enabled at M i.e. $b^- \subset M$, now since $a\|b$ we have $a^0 \cap b^- = \emptyset$ and thus $b^- \subset M.a = (M \setminus a^0) \cup a^1$, i.e. $M.a \xrightarrow{b}$. Symmetrically, $M.b \xrightarrow{a}$. Now

$$\begin{array}{rcll} (M.a).b & = & (((M \setminus a^0) \cup a^1) \setminus b^0) \cup b^1) & \\ & = & (M \setminus (a^0 \cup b^0)) \cup (a^1 \cup b^1) & \text{because } a^1 \cap b^0 = \emptyset \\ & = & (M.b).a & \text{by symmetry} \end{array}$$

The proof that independence reflects diamond situations is sketched in figure (6) where each slot in an array represents the set of places satisfying the flow relations indicated by its two coordinates, e.g. the slot with coordinates $a^{\overset{0}{\leftarrow}}$ and $b^{\overset{1}{\rightarrow}}$ represents the set $a^{\overset{0}{\leftarrow}} \cap b^{\overset{1}{\rightarrow}}$. As indicated, from $M.a \xrightarrow{b}$ we deduce (1) $a^0 \cap b^- = \emptyset$, and symmetrically from $M.b \xrightarrow{a}$ we deduce (2) $b^0 \cap a^- = \emptyset$, and finally, the equality (3) $M.ab = M.ba$ gives us $a^1 \cap b^0 = b^1 \cap a^0 = \emptyset$.

□

Definition 3.5 (Behaviour of nets) *Let $\mathcal{A}^{(1)}$ and $\mathcal{A}^{(2)}$ be canonical trace automata, and let $\eta : A^{(1)} \to A^{(2)}$ be a surjective mapping between their respective sets of actions, such that $a\|b \Leftrightarrow \eta(a)\|\eta(b)$. An η-reduction of $\mathcal{A}^{(1)}$ into $\mathcal{A}^{(2)}$ is a mapping $\sigma : Q^{(1)} \to Q^{(2)}$ between their sets of states which is a rooted bisimulation modulo η, thus satisfying* (i) $\sigma(q_0^{(1)}) = q_0^{(2)}$, (ii) $q \xrightarrow{a} \Leftrightarrow \sigma(q) \xrightarrow{\eta(a)}$ *and,* (iii) $\sigma(q.a) = \sigma(q).\eta(a)$ *whenever a is enabled in q. If $\mathcal{A}^{(1)}$ and $\mathcal{A}^{(2)}$ have the same set of actions A and η is the identity on A, then the existence of an η-reduction between $\mathcal{A}^{(1)}$ and $\mathcal{A}^{(2)}$ (called a strict reduction in that case) ensures that both automata have an identical concurrent behaviour. We call* behaviour *of a canonical trace automaton its class for the equivalence generated by strict reductions, i.e. by the equivalence of rooted bisimulation. The behaviour of a net is the behaviour of its associated (canonical) trace automaton.*

The use of the term *reduction* is justified par the following proposition.

Proposition 3.6 *Let σ be an η reduction of $\mathcal{A}^{(1)}$ into $\mathcal{A}^{(2)}$, then σ is a surjective mapping.*

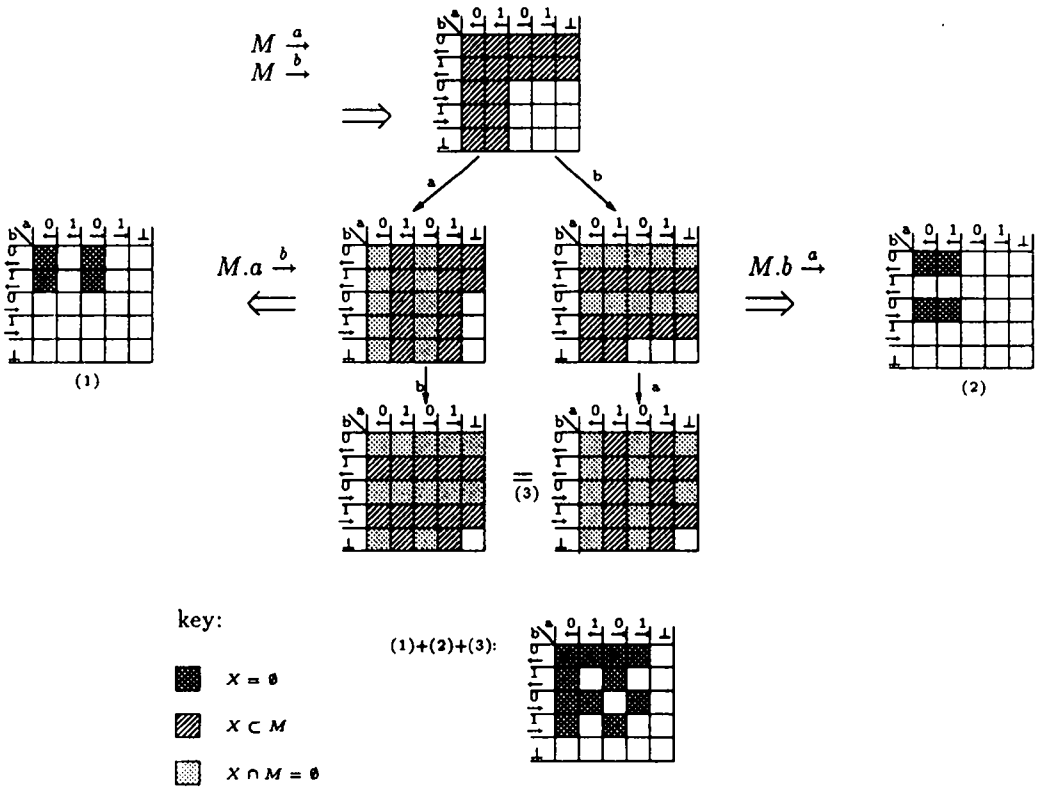


Figure 6: creation of diamond situations

Proof: using the fact that in a canonical trace automaton every state is accessible from the initial state, we show that every state $q \in Q^{(2)}$ is reached by σ by induction on the minimal length of a path from the initial state $q_0^{(2)}$ to q. The base case corresponding to $q = q_0^{(2)}$ follows from $q_0^{(2)} = \sigma(q_0^{(1)})$. For the induction step, we assume $q^{(2)} \xrightarrow{b} q$ and $q^{(2)} = \sigma(q^{(1)})$, since η is surjective, there exists an action $a \in A^{(1)}$ such that $\eta(a) = b$ and thus $q = \sigma(q^{(1)}.a)$. □

Our purpose is to establish a connection between nets and their behaviours. Technically we build an adjunction between subcategories of nets and canonical trace automata such that two objects put in correspondence exhibit the same behaviour. Although some restrictions on nets and trace automata are necessary (we shall restrict to the so-called saturated nets and separated trace automata), all behaviours of nets will be represented (for every net there is a saturated net with the same behaviour). But the analog result for trace automata does not hold (there exist canonical trace automata whose behaviour does not coincide with the behaviour of any separated trace automata). This asymmetry is typical of the duality between structure and semantics: every structure (here net) can be given a semantics but there usually exist semantic objects (here canonical trace automata) which do not represent any structure.

Let $\mathcal{N}$ be a net with set of actions A and set of places P. The language of the net is the set of words $\mathcal{L}(\mathcal{N}) = \{u \in A^* / \ M_0 \xrightarrow{u}\}$, where M_0 is the initial marking. The restriction $\mathcal{N}/P'$ of the net $\mathcal{N}$ to a subset of places $P' \subset P$ is the net with initial state $M_0/P' = M_0 \cap P'$ and with flow relations $\nabla/P' = \nabla \cap A \times P'$ deduced from the corresponding flow relations of $\mathcal{N}$. Notice $M \xrightarrow{a}$ in $\mathcal{N}$ entails $M/P' \xrightarrow{a}$ in $\mathcal{N}/P'$ hence *(i)* $a||b$ in $\mathcal{N}$ implies $a||b$ in $\mathcal{N}/P'$ and *(ii)* $\mathcal{L}(\mathcal{N}) \subset \mathcal{L}(\mathcal{N}/P')$. Then P' is said to be *dense* in $\mathcal{N}$ if the restriction to P' doesn't modify the sequential behaviour of $\mathcal{N}$ i.e. if $\mathcal{L}(\mathcal{N}) = \mathcal{L}(\mathcal{N}/P')$. Clearly P' is dense in $\mathcal{N}$ if and only if the following relation holds for every accessible marking M and for every action a:

$$a^{\leftarrow} \cap P' \subset M \quad \Rightarrow \quad a^{\leftarrow} \subset M$$

In some sense, those places -viewed as conditions- which are ouside a dense subset $P' \subset P$ are *consequences* of the conditions inside P', but this notion of logical consequence depends in a crucial way on the initial marking. Now morphisms between nets may be introduced as *reduction* which allow to identify two actions if their respective flow relations coincide on a dense subset of places and which allow at the same time to suppress the places outside this dense subset. More precisely,

Definition 3.7 (Category of nets) *A morphism $(\eta, \beta) : \mathcal{N}^{(1)} \to \mathcal{N}^{(2)}$ between two nets consist of a mapping $\eta : A^{(1)} \to A^{(2)}$ between the respective sets of actions, and a mapping $\beta : P^{(2)} \to P^{(1)}$ in the opposite direction between the sets of places such that:*

1. *η is a surjective mapping s.t. $a||b \iff \eta(a)||\eta(b)$,*
2. *the image of β is a dense subset of places in $\mathcal{N}^{(1)}$,*
3. *$\beta^{-1}(M_0^{(1)}) = M_0^{(2)}$, and*
4. *compatibility with the flow relations: $a \in {}^{\nabla}\beta(x) \Leftrightarrow x \in \eta(a)^{\nabla}$*

for all $a, b \in A^{(1)}$, $x \in P^{(2)}$, and $\nabla \in \{ \xleftarrow{0}, \xleftarrow{1}, \xrightarrow{0}, \xrightarrow{1}, \perp \}$.

Trace automata may be derived from nets as the induced transitions systems. Conversely we can associate to a trace automaton a net whose places are *regions* in the following sense.

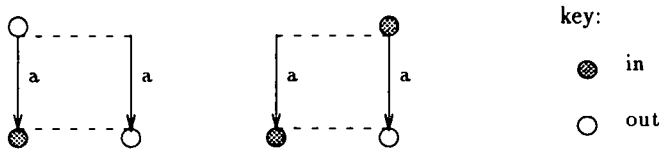


Figure 7: forbidden cases for a region

Definition 3.8 (Regions) *Let $(A, \|, Q, q_0, T)$ be a trace automaton, we define the following (disjoint) flow relations on $A \times 2^Q$:*

$a \overset{0}{\leftarrow} x$ **iff** $q \xrightarrow{a} q' \Rightarrow q \in x \ \ \& \ \ q' \notin x$

$a \overset{0}{\rightarrow} x$ **iff** $(q \xrightarrow{a} q' \Rightarrow q' \notin x) \ \& \ (\exists q \xrightarrow{a} q' \ \textit{with} \ q \notin x)$

$a \overset{1}{\leftarrow} x$ **iff** $q \xrightarrow{a} q' \Rightarrow (q \in x \ \ \& \ \ q' \in x)$

$a \overset{1}{\rightarrow} x$ **iff** $(q \xrightarrow{a} q' \Rightarrow q' \in x) \ \& \ (\exists q \xrightarrow{a} q' \ \textit{with} \ q \notin x)$

$a \perp x$ **iff** $[q \xrightarrow{a} q' \Rightarrow (q \in x \ \textit{iff} \ q' \in x)] \ \& \ (\exists q \xrightarrow{a} q' \ \textit{with} \ q \notin x) \ \& \ (\exists q \xrightarrow{a} q' \ \textit{with} \ q \in x)$

A set of states $x \subset Q$ is said to be a region *(or place) if $\{\overset{0}{\leftarrow}x, \overset{1}{\leftarrow}x, \overset{0}{\rightarrow}x, \overset{1}{\rightarrow}x, \perp x\}$ is a partitioning of the set A of actions. The above defined relations then restrict to homonymic flow relations on $A \times P$ where P is the set of regions. The net derived from the trace automaton is then defined as $(P, A, \overset{0}{\leftarrow}, \overset{1}{\leftarrow}, \overset{0}{\rightarrow}, \overset{1}{\rightarrow}, \perp, M_0)$ with $M_0 = \{ x \in P / \ q_0 \in x\}$ as initial state.*

Like we did for nets we also define on $A \times 2^Q$ the derived flow relations $\rightarrow = \overset{0}{\rightarrow} \cup \overset{1}{\rightarrow}$, $\leftarrow = \overset{0}{\leftarrow} \cup \overset{1}{\leftarrow}$, $\mathbf{0} = \overset{0}{\leftarrow} \cup \overset{0}{\rightarrow}$, and $\mathbf{1} = \overset{1}{\leftarrow} \cup \overset{1}{\rightarrow}$. The last two have a clear characterization:

$$a\mathbf{0}x \ \ \textbf{iff} \ \ q \xrightarrow{a} q' \Rightarrow q' \notin x$$
$$a\mathbf{1}x \ \ \textbf{iff} \ \ q \xrightarrow{a} q' \Rightarrow q' \in x$$

In order to obtain a simple characterization of regions, let us finally state:

$$a \dagger x \ \ \textbf{iff} \ \ q \xrightarrow{a} q' \Rightarrow (q \in x \ \text{iff} \ q' \in x)$$

then x is a region if and only if $A = {}^0x \cup {}^1x \cup {}^\dagger x$. Thus $x \subset Q$ is a region if an only if none of two *forbidden cases* depicted in figure (7) occurs for x, more precisely x is not a region if and only if

$$\exists a \in A \ \ \exists q_1 \xrightarrow{a} q_1' \in T \ \ \exists q_2 \xrightarrow{a} q_2' \in T \ \ \text{with} \ \ q_1' \in x \ \& \ q_2' \notin x \ \& \ (q_1 \notin x \ \text{or} \ q_2 \in x)$$

Therefore, if $x \subset Q$ is a region, its complement $\overline{x} = Q \setminus x$ is also a region, with ${}^0\overline{x} = {}^1x$, ${}^1\overline{x} = {}^0x$, and ${}^\perp\overline{x} = {}^\perp x$. In figure (8) we have represented 8 out of the 14 regions for the transition system associated with the net in figure (5). The missing items are the trivial regions (the whole set Q and the empty set) plus four regions obtained by exchanging a and b in the upper half. By adding the flow relations and initial marking indicated in definition (3.8) and shown in figure (8) we recover the original net of figure (5) enriched with additional places but with unchanged behaviour (see figure 9). As we shall see this is a general situation when we start from a net. However, if we consider an arbitrary trace automaton, the behaviour of the derived net may be totally different from the behaviour of the trace automaton. For instance if we consider the transition system depicted in

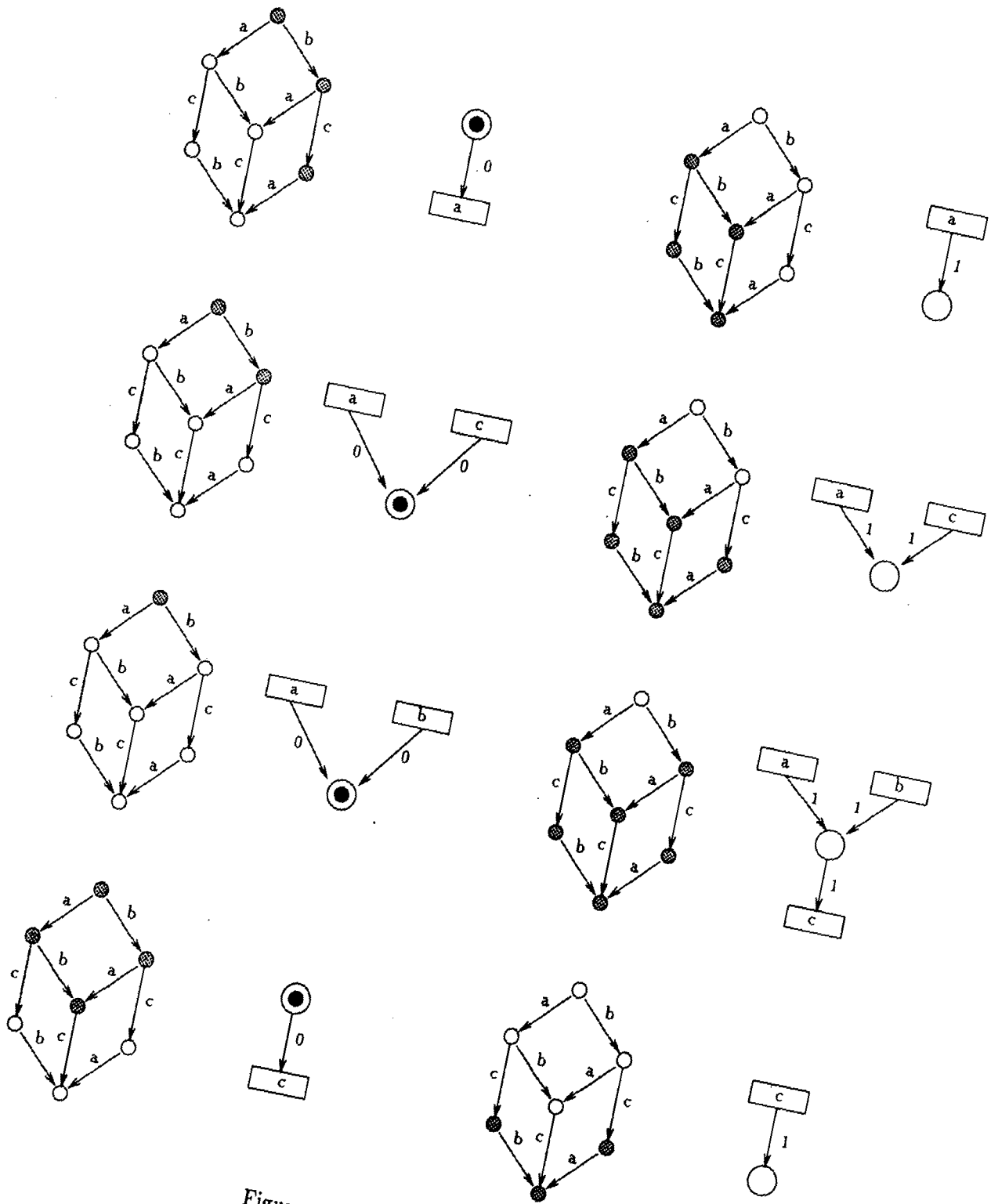


Figure 8: examples of regions

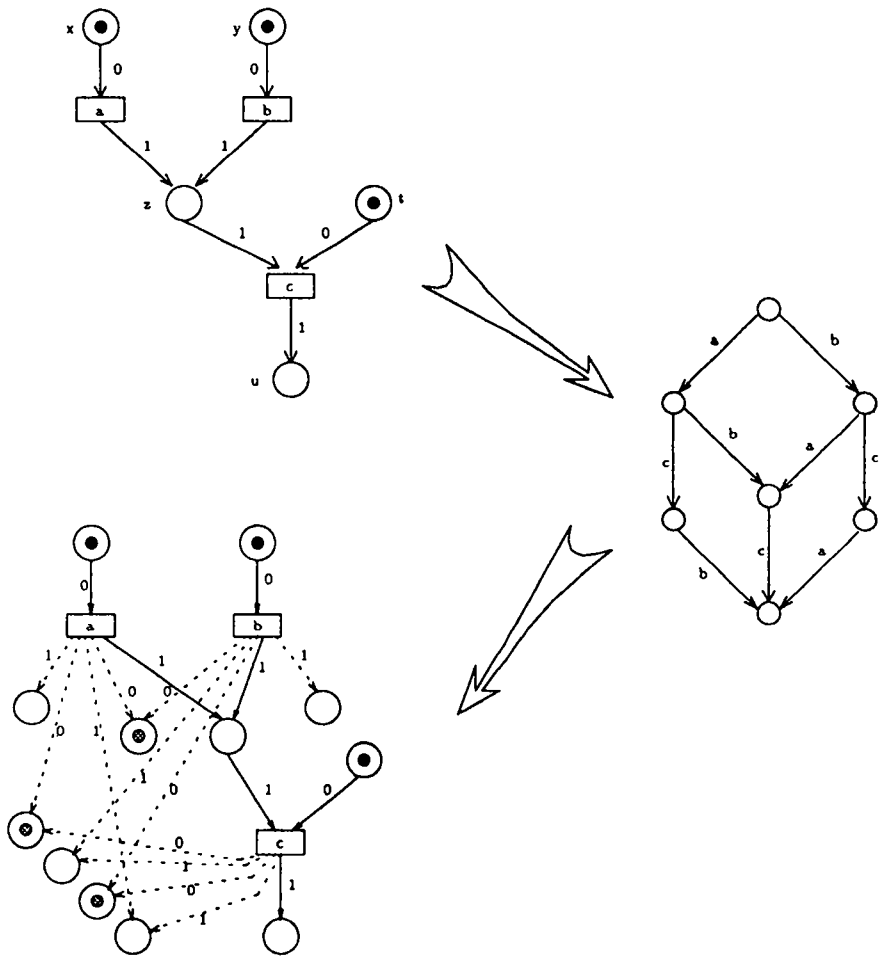


Figure 9: saturation of nets

figure (5) and identify the states M_0 and M we obtain a cyclic trace automaton with only the trivial regions. That case of collapsing is due to *contact situations* (there exist a state q, an action a and transitions $q_1 \xrightarrow{a} q$ and $q \xrightarrow{a} q_2$) which can easily be eliminated by a finite unfolding operation which produces an equivalent trace automaton. But there exists more involved cases, as we shall see, which cannot be dealt with so easily. We are therefore looking for additional conditions to impose on nets and trace automata, such that translations in both directions between nets and automata preserve behaviours. On the one hand, since the complement of a region is a region, we know that a net which is produced from a trace automaton comes up equipped with a duality operation $\overline{()} : P \to P$ on places, which is an involutive mapping such that $(\forall x \in P)\ x \in M_0 \Leftrightarrow \overline{x} \notin M_0$, and ${}^0\overline{x} = {}^1x$, ${}^1\overline{x} = {}^0x$, and ${}^\perp\overline{x} = {}^\perp x$. Notice that by the very definition of the transition system associated with a net, the above property of the initial marking is shared by all accessible markings: $(\forall M \in \mathcal{M}_a)\ x \in M \Leftrightarrow \overline{x} \notin M$. On the other hand, there follows from the definition of a region x associated with a transition system, that the two conditions *(i)*: $a \leftarrow x$ and, *(ii)*: $\forall q \in Q\ q \xrightarrow{a} \Rightarrow q \in x$ are equivalent (see lemma (3.11) below). Therefore a net produced from a trace automaton should satisfy the following property of completeness:

$$(\forall x \in P)\ [(\forall M \in \mathcal{M}_a)(M \xrightarrow{a} \Rightarrow x \in M) \Rightarrow a \leftarrow x]$$

The nets produced from trace automata are *saturated* nets according to the following definition.

Definition 3.9 (Category of saturated nets) *A saturated net is a net that satisfies the following property of completeness:*

$$(\forall x \in P)\ [(\forall M \in \mathcal{M}_a)(M \xrightarrow{a} \Rightarrow x \in M) \Rightarrow a \leftarrow x]$$

and that is moreover equipped with an involutive mapping $\overline{()} : P \to P$ on places such that $(\forall x \in P)$ $(x \in M_0 \Leftrightarrow \overline{x} \notin M_0)$ and ${}^0\overline{x} = {}^1x$, ${}^1\overline{x} = {}^0x$ and ${}^\perp\overline{x} = {}^\perp x$. A morphism of saturated nets $(\eta, \beta) : \mathcal{N}^{(1)} \to \mathcal{N}^{(2)}$ is a morphism of nets which commutes with the complementation operation, in the sense that $\beta(\overline{x}) = \overline{\beta(x)}$. In particular the image of β is not only dense but stable under complementation.

We shall prove latter on that for every net $\mathcal{N}$ there exists a saturated net $\overline{\mathcal{N}}$ with the same behaviour.

We already saw (proposition 3.4) that the trace automaton derived from a net is *canonical*. Now, whenever an action a cannot be fired at a given (accessible) marking in the net, a token should be missing in some input place of a, and this dynamic property of nets entails the following property of separation for the trace automaton derived from the net.

Definition 3.10 (Separated trace automata) *The category* **STA** *of separated trace automata is the subcategory of trace automata formed of those trace automata which are canonical (definition 3.3) and which moreover satisfy the following property of separation:*

$$q \overset{a}{\not\to} \ \Rightarrow\ \exists x \in P \text{ s.t. } q \notin x \ \&\ x \in a^{\leftarrow}$$

A morphism between two separated trace automata consists of two mappings $\eta : A^{(1)} \to A^{(2)}$ and $\sigma : Q^{(1)} \to Q^{(2)}$ between their respective sets of actions and states such that σ is an η-reduction (definition 3.5).

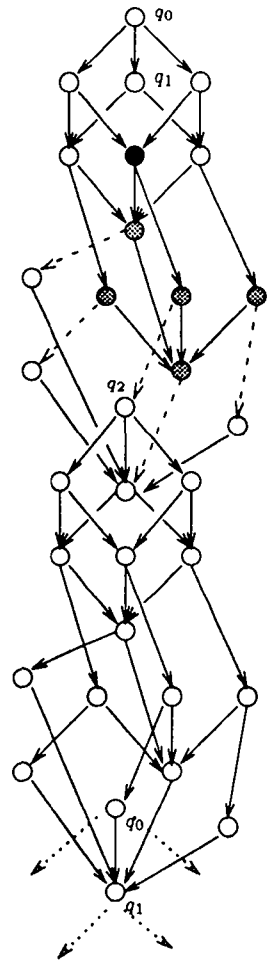


Figure 10: a non separated trace automaton

Lemma 3.11 *Let x be a region, then conditions* (i) $a \leftarrow x$ *and* (ii) $\forall q \in Q \quad q \xrightarrow{a} \Rightarrow q \in x$ *are equivalent.*

Proof: By the definition of regions, $(a \leftarrow x \ \& \ q \xrightarrow{a}) \Rightarrow q \in x$. Conversely, assume $\forall q \in Q$ $q \xrightarrow{a} \Rightarrow q \in x$ but $a \not\leftarrow x$. In that case, there should exist two transitions $q_1 \xrightarrow{a} q_1'$ and $q_2 \xrightarrow{a} q_2'$ such that $q_1' \in x$ and $q_2' \notin x$, but this is impossible since we have neither $a \xrightarrow{0} x$ because $q_1' \in x$, nor $a \xrightarrow{1} x$ because $q_2' \notin x$, nor $a \perp x$ because $q_2 \xrightarrow{a} q_2'$ with $q_2 \in x$ and $q_2' \notin x$.
□

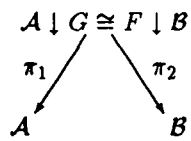
In view of the above lemma, the separation property means that for every action a and state q at which a is not enabled, we can find a region which separate q from a in the sense that it contains every state at which a is enabled but not q. Figure (10) displays the simplest example we were able to produce of a contact-free ($q \xrightarrow{a} \Rightarrow not(\xrightarrow{a} q)$) but not separated trace automaton. The vertical arrows represent one action, say a, and the dashed arrows represent another action, say b. Action b is enabled in the grey states but not in the black one. Let us prove by contradiction, that an input region of b must contain

the black state. Assume $b \overset{0}{\leftarrow} x$ and x does not contain the black state. Then all grey states belong to x and necessarily $a1x$, but this is impossible since there exists one state which is reached by both a and b. Assume $b \overset{1}{\leftarrow} x$ and x does not contain the black state. Then all sources and targets of dashed arrows and all targets of vertical arrows are in x, Using the second excluded pattern shown in figure (7), it is easy to propagate downwards the property $q \in x$ from state q_2 to all the other states q such that $q \xrightarrow{a}$, including the state q_0 and the black state. Due to the existence of one action (namely a) which may never be taken but remains always live, that case of non-separation cannot be escaped by any sort of limited unfolding.

We are going to show that the previously defined correspondences between nets and trace automata restrict to an adjunction $F \dashv G : \mathbf{SNets} \to \mathbf{STA}$ such that the objects set in correspondence have the same behaviour.

Theorem 3.12 *There exists an adjunction between the categories* **STA** *and* **SNets** *of separated trace automata and saturated nets, such that both left and right adjoint functors preserve behaviours.*

In order to facilitate the construction, we introduce an intermediate category and split the desired adjunction into two basic parts, a reflection and a coreflection, following the general principle recalled below. A functor $F : \mathcal{A} \to \mathcal{B}$ induces an *image* of $\mathcal{A}$ into $\mathcal{B}$, and the comma category $F \downarrow \mathcal{B}$, whose objects (A, f, B) consist of an object in each category together with an arrow $f : FA \to B$ relating them within $\mathcal{B}$, describe how the image of $\mathcal{A}$ approximates $\mathcal{B}$. For instance, if $\mathcal{B}$ is an ordered set viewed as a category and $\mathcal{A}$ a subset of $\mathcal{B}$, the comma category $F \downarrow \mathcal{B}$ (where F is the inclusion) is the union of all those sets $F \downarrow B = \{A \in \mathcal{A} / \ A \leq B\}$ which approximate the element $B \in \mathcal{B}$ by elements in $\mathcal{A}$. An adjunction situation $F \dashv G : \mathcal{B} \to \mathcal{A}$ is a natural isomorphism $\mathcal{B}(Fa, b) \cong \mathcal{A}(a, Gb)$ expressing that the approximation of $\mathcal{B}$ by $\mathcal{A}$ via F coincides with the approximation of $\mathcal{A}$ by $\mathcal{B}$ via G. Actually, such an adjunction amounts to an isomorphism between the comma categories $\mathcal{A} \downarrow G \cong F \downarrow \mathcal{B}$ which commutes with the projections.



The objects of the intermediate category are *compound objects* (A, f, B) consisting of an object in each category together with a connection $f = (f_\#, f^\#)$ where $f_\# : A \to GB$ in $\mathcal{A}$ and $f^\# : FA \to B$ in $\mathcal{B}$ are adjoint arrows. A morphism between two compound objects $(A^{(1)}, f^{(1)}, B^{(1)})$ and $(A^{(2)}, f^{(2)}, B^{(2)})$ is a pair of arrows between their respective components: $a : A^{(1)} \to A^{(2)}$ in $\mathcal{A}$ and $b : B^{(1)} \to B^{(2)}$ in $\mathcal{B}$ which respects the connections in each compound object, i.e. $Gb \circ f^{(1)}_\# = f^{(2)}_\# \circ a$ or equivalently $b \circ f^{(1)\#} = f^{(2)\#} \circ Fa$. Now the projection $\pi_1 : \mathcal{A} \downarrow G \to \mathcal{A}$ has a right-inverse left-adjoint $\rho_1 : \mathcal{A} \to \mathcal{A} \downarrow G$ which takes an object A of $\mathcal{A}$ to the unit of the adjunction $\eta_A : A \to GFA$ and takes the arrow $f : A \to B$ to the pair (f, Gf). In the same way, the second projection $\pi_2 : F \downarrow \mathcal{B} \to \mathcal{B}$ has a right-inverse right-adjoint $\rho_2 : \mathcal{B} \to F \downarrow \mathcal{B}$ which takes an object B to the co-unit $\epsilon_B : FGB \to B$. And we recover the initial adjunction as the composite of the two basic adjunctions $\rho_1 \dashv \pi_1$ (a

co-reflection) and $\pi_2 \dashv \rho_2$ (a reflection). Some basics facts on reflections and co-reflections are recalled in the appendix. Here we are going to use a category of compound objects called *trace-nets* which is a little bit simpler than the comma categories $\mathcal{A} \downarrow G \cong F \downarrow \mathcal{B}$ but playing the same role. Actually we have a family of adjunctions indexed by the alphabets so we can restrict ourselves to compound objects whose both components have the same alphabet.

Definition 3.13 (Trace nets) *A trace net is a structure with the following fields: a set of states Q including an initial state $q_0 \in Q$, a set of places P equipped with an involutive mapping $\overline{()} : P \to P$, an enabling relation $\models \subset Q \times P$ ($q \models x$ reads as "the state q satisfies the condition x"), a set of actions A equipped with an independence relation $\| \subset A \times A$, a set of labelled transitions $T \subset Q \times A \times Q$, and five flow relations $\overset{0}{\leftarrow}, \overset{1}{\leftarrow}, \overset{0}{\rightarrow}, \overset{1}{\rightarrow}, \perp \subset A \times P$ satisfying as a whole the following conditions:*

1. $(A, \|, Q, q_0, T)$ **is a canonical trace automaton** *i.e.,*
   - *(a)* $\| \subset A \times A$ *is irreflexive and symmetric,*
   - *(b)* $T \subset Q \times A \times Q$ *is a deterministic transition system,*
   - *(c) every state is accessible from the initial state,*
   - *(d) if a and b are independent, there exists at least one state at which both are enabled,*
   - *(e) commutativity: whenever $q \xrightarrow{a} q_a$ and $q \xrightarrow{b} q_b$ with $a \| b$, $q_b \xrightarrow{a} q_{a,b}$ and $q_a \xrightarrow{b} q_{a,b}$ for some state $q_{a,b}$,*
   - *(f) independence reflects diamond situations: $a \| b$ whenever $q \xrightarrow{ab}$, $q \xrightarrow{ba}$ and $q.ab = q.ba$ for some state q.*
2. $(P, A, \overset{0}{\leftarrow}, \overset{1}{\leftarrow}, \overset{0}{\rightarrow}, \overset{1}{\rightarrow}, \perp, M_0)$ **is a net** *i.e. $\{\overset{0}{\leftarrow}, \overset{1}{\leftarrow}, \overset{0}{\rightarrow}, \overset{1}{\rightarrow}, \perp\}$ is a partitioning of $A \times P$. As for the initial marking, we let $M_0 = q_0{}^{\models}$, i.e. $M_0 = \{x \in P / \; q_0 \models x\}$.*
3. **Properties of duality:**
   - *(a)* $\forall q \in Q \; \forall x \in P \; (q \models x \;\; \textbf{iff} \;\; q \not\models \overline{x})$
   - *(b)* ${}^{0}\overline{x} = {}^{1}x \;\; \& \;\; {}^{1}\overline{x} = {}^{0}x$ and ${}^{\perp}\overline{x} = {}^{\perp}x$.
4. **Connection between independence relations:** $a \| b \Rightarrow a^{\overset{0}{\leftarrow}} \subset b^{\perp}$ *and* $a^{0} \cap b^{1} = \emptyset$.
5. **Connection between the transition and flow relations:**
   - *(a)* $q \xrightarrow{a} \Leftrightarrow a^{\leftarrow} \subset q^{\models}$ *and in that case* $(q.a)^{\models} = (q^{\models} \setminus a^{0}) \cup a^{1}$.
   - *(b)* $\forall a \in A$ *and* $\forall x \in P$ *we have*

$$
\begin{array}{lll}
(i) & a \overset{0}{\leftarrow} x \;\; \textbf{iff} & q \xrightarrow{a} q' \Rightarrow (q \models x \;\; \& \;\; q' \not\models x) \\
(ii) & a \overset{0}{\rightarrow} x \;\; \textbf{iff} & (q \xrightarrow{a} q' \Rightarrow q' \not\models x) \;\; \& \;\; (\exists q \xrightarrow{a} q' \;\; \textit{with} \;\; q \not\models x) \\
(iii) & a \overset{1}{\leftarrow} x \;\; \textbf{iff} & q \xrightarrow{a} q' \Rightarrow (q \models x \;\; \& \;\; q' \models x) \\
(iv) & a \overset{1}{\rightarrow} x \;\; \textbf{iff} & (q \xrightarrow{a} q' \Rightarrow q' \models x) \;\; \& \;\; (\exists q \xrightarrow{a} q' \;\; \textit{with} \;\; q \not\models x) \\
(v) & a \perp x \;\; \textbf{iff} & [q \xrightarrow{a} q' \Rightarrow (q \models x \;\; \textit{iff} \;\; q' \models x)] \;\; \& \\
 & & (\exists q \xrightarrow{a} q' \;\; \textit{with} \;\; q \not\models x) \;\; \& \;\; (\exists q \xrightarrow{a} q' \;\; \textit{with} \;\; q \models x)
\end{array}
$$

A morphism between two trace nets is a pair of mappings $\eta : A^{(1)} \to A^{(2)}$ *and* $\sigma : Q^{(1)} \to Q^{(2)}$ *between their respective sets of actions and states together with a mapping* $\beta : P^{(2)} \to P^{(1)}$ *in the opposite direction between their sets of places, satisfying the following conditions, where* $a, b \in A^{(1)}$, $x \in P^{(2)}$, *and* $\nabla \in \{ \overset{0}{\leftarrow}, \overset{1}{\leftarrow}, \overset{0}{\rightarrow}, \overset{1}{\rightarrow}, \perp \}$:

1. η **is a surjective mapping s.t.** $a \| b \iff \eta(a) \| \eta(b)$,
2. (η, σ) **is a reduction of trace automata**, *i.e.*
   (a) $\sigma(q_0^{(1)}) = q_0^{(2)}$,
   (b) (η, σ) *is a bisimulation, i.e.* (i) $q \xrightarrow{a} \Leftrightarrow \sigma(q) \xrightarrow{\eta(a)}$ *and,* (ii) $\sigma(q.a) = \sigma(q).\eta(a)$.
3. (η, β) **is a reduction of saturated nets**, *i.e.*
   (a) *the image of* β *is a dense subset of places in* $\mathcal{N}^{(1)}$ *such that* $\overline{\beta(x)} = \beta(\overline{x})$,
   (b) $\beta^{-1}(M_0^{(1)}) = M_0^{(2)}$, *and*
   (c) *compatibility with the flow relations:* $a \in {}^{\nabla}\beta(x) \Leftrightarrow x \in \eta(a)^{\nabla}$.
4. σ **and** β **are adjoint mappings:** $q \models \beta(x) \Leftrightarrow \sigma(q) \models x$.

The next three statements show that a trace net is a schizophrenic object which is at the same time a separated trace automaton and a saturated net both bridged by a structural link $\models$, and exhibiting the same behaviour.

Proposition 3.14 (Properties of separation) *Trace nets enjoy the following two properties of separation:*

1. $(\forall q \in Q)\ [((\forall x \in P)\ (a \leftarrow x \Rightarrow q \models x) \Rightarrow q \xrightarrow{a})]$,
2. $(\forall x \in P)\ [((\forall q \in Q)\ (q \xrightarrow{a} \Rightarrow q \models x) \Rightarrow a \leftarrow x)]$.

Proof: the implication $a^{-} \subset q^{\models} \Rightarrow q \xrightarrow{a}$ which is one half of axiom (5 a) is equivalent to the first separation property. The second separation property follows from an analog of lemma (3.11) which states that for every place x in a trace net, the conditions *(i)* $a \leftarrow x$ and *(ii)* $\forall q \in Q \quad q \xrightarrow{a} \Rightarrow q \models x$ are equivalent (one needs just to replace $\in$ by $\models$ in the proof of lemma (3.11)).
□.

Proposition 3.15 *There exist a pair of functors from the category* **TrNets** *of trace nets to the respective categories of separated trace automata and saturated nets*

$$\mathbf{STA} \xleftarrow{\pi_1} \mathbf{TrNets} \xrightarrow{\pi_2} \mathbf{SNets}$$

which project a trace net to its underlying trace automaton and net, and send a morphism (η, σ, β) *respectively to* (η, σ) *and*(η, β).

Proof: Let us verify that the underlying trace automaton of a trace net is separated. In view of the axioms (5 b) and (2) we see that every place $x \in P$ has an associated region ${}^{\models}x = \{q \in Q / \ q \models x\} \subset 2^{Q}$. By the first separation property, stating that $q \overset{a}{\not\rightarrow} \Rightarrow \exists x \in P$ s.t. $a \leftarrow x$ & $q \not\models x$,there exists a *region* ${}^{\models}x$ of the underlying trace automaton such that *(i)* $q \notin {}^{\models}x$ and *(ii)* $q' \xrightarrow{a} \Rightarrow q' \in {}^{\models}x$.
Next, let us verify that the underlying net of a trace net is saturated. Since $M_0 = q_0{}^{\models}$, the axiom (5 a) shows that the set of accessible markings is $\mathcal{M}_a = \{q^{\models} / \ q \in Q\}$, and the second separation property gives the result.
□

Corollary 3.16 *Trace nets are exactly those triples* $(\mathcal{A}, \models, \mathcal{N})$ *where* $\mathcal{A} = (A, \|, Q, q_0, T)$ *is a separated trace automaton,* $\mathcal{N} = (P, \overline{()}, A, \overset{0}{\leftarrow}, \overset{1}{\leftarrow}, \overset{0}{\rightarrow}, \overset{1}{\rightarrow}, \perp, M_0)$ *is a saturated net on the same alphabet, and the relation* $\models \subset Q \times P$ *is a link between the two structures satisfying as a whole* $M_0 = q_0{}^{\models}$ *and the conditions (4) and (5) from definition (3.13). Thus* $()^{\models}$ *is a strict reduction (see definition 3.5) of* $\mathcal{A}$ *into the separated trace automaton associated to the net* $\mathcal{N}$*. Hence* $\mathcal{A}$ *and* $\mathcal{N}$ *have the same behaviour.*

In order to establish theorem (3.12), it suffices to prove that both projection functors π_1 and π_2 have right-inverse (extending respectively separated trace automata and saturated nets into trace nets) providing adjunction pairs $\rho_1 \vdash \pi_1$ and $\rho_2 \vdash \pi_2$.

Proposition 3.17 π_1 *has a right-inverse left-adjoint.*

Proof: We can extend a separated trace automaton $\mathcal{A} = (A, \|, Q, q_0, T)$ into a trace net $\rho_1(\mathcal{A}) = (\mathcal{A}, \models, (P, \overline{()}, A, \overset{0}{\leftarrow}, \overset{1}{\leftarrow}, \overset{0}{\rightarrow}, \overset{1}{\rightarrow}, \perp, M_0))$ where P is the set of regions of $\mathcal{A}$, $\overline{()} : P \to P$ is the involution that maps x to its complement $\overline{x} = Q \setminus x$ in Q, $\models$ is the membership relation, the flow relations are the restriction to $A \times P$ of the corresponding relations on $A \times 2^Q$ (see definition 3.8), and $M_0 = \{x \in P / \; q_0 \models x\}$. In order to show that $\rho_1(\mathcal{A})$ is a trace net, we have just to verify conditions (4) and (5a) from the definition of trace nets.

Let us establish condition (4), namely: $a \| b \;\Rightarrow\; a^{\overset{0}{\leftarrow}} \subset b^{\perp} \;\&\; a^0 \cap b^1 = \emptyset$
Since a and b are independent, there exists at least one state q in which both actions are enabled, giving rise to a diamond situation

$$q \xrightarrow{a} q.a \xrightarrow{b} q' \;\&\; q \xrightarrow{b} q.b \xrightarrow{a} q'$$

Let $x \in a^{\overset{0}{\leftarrow}}$, then by the definition of regions $q \in x$, $q.b \in x$, $q.a \notin x$, and $q' \notin x$. And thus $b \perp x$ since neither $b0x$ (as $q.b \in x$), nor $b1x$ (as $q' \notin x$). Hence $a^{\overset{0}{\leftarrow}} \subset b^{\perp}$. Let $x \in a^0$, then by the definition of regions $q' \notin x$, hence we cannot have $b1x$. Thus $a^0 \cap b^1 = \emptyset$

Let us establish condition (5a), namely: $q \xrightarrow{a} \Leftrightarrow a^{\leftarrow} \subset q^{\epsilon}$ and, when a is enabled in q, $q.a^{\epsilon} = (q^{\epsilon} \setminus a^0) \cup a^1$. The implication $q \xrightarrow{a} \Rightarrow a^{\leftarrow} \subset q^{\epsilon}$ follows immediately from the definition of $a^{\leftarrow}$, and the converse implication, which is equivalent to the first separation property, follows from the assumption that $\mathcal{A}$ is a *separated* trace automaton. Assuming $q \xrightarrow{a} q'$, let us prove that

$$q' \in x \quad \textbf{iff} \quad [(q \in x \;\&\; a \notin {}^0x) \text{ or } a \in {}^1x]$$

1. $([(q \in x \;\&\; a \notin {}^0x) \text{ or } a \in {}^1x] \;\&\; q \xrightarrow{a} q') \Rightarrow q' \in x$:
   (a) $(a \in {}^1x \;\&\; q \xrightarrow{a} q') \Rightarrow q' \in x$ by definition of ${}^1x$.
   (b) $(q \in x \;\&\; a \notin {}^0x \;\&\; a \notin {}^1x \;\&\; q \xrightarrow{a} q') \Rightarrow q' \in x$ by definition of a region x.
2. $(q' \in x \;\&\; q \xrightarrow{a} q') \Rightarrow [(q \in x \;\&\; a \notin {}^0x) \text{ or } a \in {}^1x]$
   $(q' \in x \;\&\; q \xrightarrow{a} q') \;\Rightarrow\; a \notin {}^0x$, and $(q' \in x \;\&\; q \xrightarrow{a} q') \;\Rightarrow\; (q \in x \text{ or } a \in {}^1x)$, because $(q \notin x \;\&\; q \xrightarrow{a} q' \;\&\; q' \in x) \Rightarrow a \in {}^1x$ for any region x.

We are now going to prove that ρ_1 is left-adjoint to π_1: every morphism $(\eta, \sigma) : \mathcal{A} \to \pi_1(\mathcal{X})$ in **STA** can be extended in a unique way into a morphism $(\eta, \sigma, \beta) : \rho_1(\mathcal{A}) \to \mathcal{X}$ in **TrNets**. We

recall that the functoriality of ρ_1 follows from that universal property. Let $\rho_1(\mathcal{A}) = (\mathcal{A}^{(1)}, \models^{(1)}, \mathcal{N}^{(1)})$ and $\mathcal{X} = (\mathcal{A}^{(2)}, \models^{(2)}, \mathcal{N}^{(2)})$. By condition (4) for morphisms in definition (3.13), $\beta : P^{(2)} \rightarrow P^{(1)}$ must be defined as $\beta(x) = \{q \in Q^{(1)} / \ \sigma(q) \models^{(2)} x\}$. The extended morphism (η, σ, β), if it exists, is therefore unique. It remains to verify that (η, β) is a reduction between nets, i.e. satisfies conditions (3) for morphisms in definition (3.13). Let $\beta : P^{(2)} \rightarrow 2^{Q^{(1)}}$ be defined as $\beta(x) = \{q \in Q^{(1)} / \ \sigma(q) \models^{(2)} x\}$. In order to show that $\beta : P^{(2)} \rightarrow P^{(1)}$, i.e. that $\beta(x)$ is always a region of $\mathcal{A}^{(1)}$, there suffices to establish the following relations which also entail condition (3c) from definition (3.13)

$$\forall a \in A^{(1)} \ \forall x \in P^{(2)} \ \forall \nabla \in \{\overset{0}{\leftarrow}, \overset{1}{\leftarrow}, \overset{0}{\rightarrow}, \overset{1}{\rightarrow}, \perp\} \quad a \nabla \beta(x) \Leftrightarrow \eta(a) \nabla x$$

Because $\{\overset{0}{\leftarrow}x, \overset{1}{\leftarrow}x, \overset{0}{\rightarrow}x, \overset{1}{\rightarrow}x, \perp x\}$ is a partitioning of $A^{(2)}$, $\{\overset{0}{\leftarrow}\beta(x), \overset{1}{\leftarrow}\beta(x), \overset{0}{\rightarrow}\beta(x), \overset{1}{\rightarrow}\beta(x), \perp\beta(x)\}$ is then a partitioning of $A^{(1)}$, i.e. $\beta(x)$ is a region.
Notice that ∇ in the left-hand side stands for the flow relations $\nabla \subset A^{(1)} \times 2^{Q^{(1)}}$ associated to the trace automaton $\mathcal{A}^{(1)}$ according to definition (3.8), whereas ∇ in the right-hand side stands for the flow relations $\nabla \subset A^{(2)} \times P^{(2)}$ of the trace net $\mathcal{N}^{(2)}$. The equivalences to be proven follow from the surjectivity of σ (proposition 3.6) plus the fact that (η, σ) is a bisimulation. We give hereafter the detailed proof for $\nabla = \overset{0}{\rightarrow}$, the others cases are similar.

1. $\eta(a) \overset{0}{\rightarrow} x \Rightarrow a \overset{0}{\rightarrow} \beta(x)$
   First, $q \xrightarrow{a} q' \Rightarrow \sigma(q) \xrightarrow{\eta a} \sigma(q') \Rightarrow \sigma(q') \not\models x \Rightarrow q' \notin \beta(x)$. Second, we must find a transition $q \xrightarrow{a} q'$ with $q \notin \beta(x)$. Since $\eta(a) \overset{0}{\rightarrow} x$, there exists a transition $q_1 \xrightarrow{\eta a} q_1'$ with $q_1 \not\models x$, and since (η, σ) is a bisimulation and σ is onto (by proposition 3.6), there must exist a transition $q \xrightarrow{a} q'$ with $\sigma(q) = q_1$ and $\sigma(q') = q_1'$. Now $\sigma(q) = q_1 \not\models x$ entails $q \notin \beta(x)$, as required.

2. $a \overset{0}{\rightarrow} \beta(x) \Rightarrow \eta(a) \overset{0}{\rightarrow} x$
   Assuming $q_1 \xrightarrow{\eta a} q_1'$, we know there exists $q \xrightarrow{a} q'$ with $\sigma(q) = q_1$ and $\sigma(q') = q_1'$, but $q' \in \beta(x)$ (since $a \overset{0}{\rightarrow} \beta(x)$) hence $\sigma(q') = q_1' \not\models x$. Moreover there exists a transition $q \xrightarrow{a} q'$ with $q \notin \beta(x)$ hence a transition $q_1 \xrightarrow{\eta a} q_1'$ with $q_1 \not\models x$ by taking $q_1 = \sigma(q)$ and $q_1' = \sigma(q')$.

Let us check condition (3b) in definition (3.13), namely: $\beta^{-1}(M_0^{(1)}) = M_0^{(2)}$. This statement is in fact redundant in definition (3.13), because relations $M_0^{(i)} = q_0^{i\models} = \{x \in P^{(i)} / \ q_0^i \models x\}$, $\sigma(q_0^{(1)}) = q_0^{(2)}$, and $q \models \beta(x) \Leftrightarrow \sigma(q) \models x$ entail together:

$$\beta^{-1}(M_0^{(1)}) = \beta^{-1}(q_0^{1\models}) = \{x \in P^{(2)} / q_0^1 \models \beta(x)\} = \{x \in P^{(2)} / q_0^2 = \sigma(q_0^1) \models x\} = M_0^{(2)}$$

There remains to check condition (3a) in definition (3.13), i.e. to show that the image of β is a dense subset of places in $\mathcal{N}^{(1)}$ such that $\overline{\beta(x)} = \beta(\overline{x})$. Since (η, σ) is a bisimulation, $\sigma(q) \xrightarrow{\eta(a)}$ in $T^{(2)}$ is equivalent to $q \xrightarrow{a}$ in $T^{(1)}$. Now the former relation is equivalent to $\forall x \in P^{(2)} \ \eta(a) \leftarrow x \Rightarrow \sigma(q) \models x$ which is in turn (by conditions (3 c) and (4) for morphisms) equivalent to $\forall x \in P^{(2)} \ a \leftarrow \beta(x) \Rightarrow q \models \beta(x)$, whereas the latter relation is equivalent to $\forall y \in P^{(1)} \ a \leftarrow y \Rightarrow q \models y$. Hence we deduce that $a^{\leftarrow} \cap Im(\beta) \subset q^{\models}$ entails $a^{\leftarrow} \subset q^{\models}$, i.e. the image of β is a dense subset of places in $\mathcal{N}^{(1)}$. And finally

$$\beta(\overline{x}) = \{q \in Q^{(1)} / \sigma(q) \models \overline{x}\} = \{q \in Q^{(1)} / \sigma(q) \not\models x\} = Q \setminus \beta(x) = \overline{\beta(x)}$$

□

Proposition 3.18 *π_2 has a right-inverse right-adjoint.*

Proof: Let $\mathcal{N} = (P, \overline{()}, A, \overset{0}{\leftarrow}, \overset{1}{\leftarrow}, \overset{0}{\rightarrow}, \overset{1}{\rightarrow}, \perp, M_0)$ be a saturated net. We let $\mathcal{M} = 2^P$ be the set of markings, we recall that the transitions of the net $\mathcal{N}$ are given by:

$$M \xrightarrow{a} M' \quad \textbf{iff} \quad a^{\leftarrow} \subset M \quad \text{and} \quad M' = (M \setminus a^{\mathbf{0}}) \cup a^{\mathbf{1}}$$

and that two distinct actions a and b are said to be independent (written $a \| b$) when *(i)* there exists an accessible marking $M \in \mathcal{M}_a$ at which they are both enabled, *(ii)* $a \overset{0}{\leftarrow} \subset b^{\perp}$ & $a^{\mathbf{0}} \cap b^{\mathbf{1}} = \emptyset$, and symmetrically *(iii)* $b \overset{0}{\leftarrow} \subset a^{\perp}$ & $b^{\mathbf{0}} \cap a^{\mathbf{1}} = \emptyset$. We extend the saturated net $\mathcal{N}$ into a trace net $\rho_2(\mathcal{N}) = ((A, \|, \mathcal{M}_a, M_0, \mathcal{T}_a), \models, \mathcal{N})$ where $\mathcal{T}_a$ is the set of transitions between accessible markings and $\models \subset \mathcal{M} \times P$ is the inverse of the membership relation: $M \models x$ **iff** $x \in M$. By proposition 3.3 $(A, \|, \mathcal{M}_a, M_0, \mathcal{T}_a)$ is a canonical trace automaton. In order to prove that $\rho_2(\mathcal{N})$ is a trace net, there remains to check condition (5b) from definition 3.13.
For the one hand, since $M \xrightarrow{a} M' \Leftrightarrow a^{\leftarrow} \subset M$ and $M' = (M \setminus a^{\mathbf{0}}) \cup a^{\mathbf{1}}$ we have:

$$\begin{array}{rcl}
x \in a \overset{0}{\leftarrow} \ \& \ M \xrightarrow{a} M' & \Rightarrow & x \in M \ \& \ x \notin M' \\
x \in a \overset{0}{\rightarrow} \ \& \ M \xrightarrow{a} M' & \Rightarrow & x \notin M' \\
x \in a \overset{1}{\leftarrow} \ \& \ M \xrightarrow{a} M' & \Rightarrow & x \in M \ \& \ x \in M' \\
x \in a \overset{1}{\rightarrow} \ \& \ M \xrightarrow{a} M' & \Rightarrow & x \in M' \\
x \in a^{\perp} \ \& \ M \xrightarrow{a} M' & \Rightarrow & x \in M \text{ iff } x \in M'
\end{array}$$

On the other hand, by virtue of the second separation property $a \not\leftarrow x \Rightarrow \exists M$ s.t. $a^{\leftarrow} \subset M$ & $x \notin M$, thus :

$$(a \overset{0}{\rightarrow} x \text{ or } a \overset{1}{\rightarrow} x \text{ or } a \perp x) \Rightarrow \exists M \xrightarrow{a} M' \text{ with } x \notin M$$

Whence, finally, by exchanging x and $\overline{x}$ (recall that $a \perp x$ iff $a \perp \overline{x}$ and $x \in M$ iff $\overline{x} \notin M$) we obtain $a \perp x \Rightarrow \exists M \xrightarrow{a} M'$ with $x \in M$. To sum up, we have proved the implications (5b) from left to right. The converse implications may be established by contraposition, using the fact that the relations $(\overset{0}{\leftarrow}, \overset{1}{\leftarrow}, \overset{0}{\rightarrow}, \overset{1}{\rightarrow}, \perp)$ give a partitioning of $A \times P$. For instance the implication

$$a \overset{0}{\not\leftarrow} \quad \Rightarrow \quad (\exists M \xrightarrow{a} M') \ \ (x \notin M \text{ or } x \in M')$$

follows from equivalence $a \overset{0}{\not\leftarrow} \Leftrightarrow (a \overset{1}{\leftarrow} x \text{ or } a \overset{0}{\rightarrow} x \text{ or } a \overset{1}{\rightarrow} x \text{ or } a \perp x)$, by inspection of each possible case. The detailed verification is left to the reader.

We are now going to prove that ρ_2 is right-adjoint to π_2: every morphism $(\eta, \beta) : \pi_2(\mathcal{X}) \rightarrow \mathcal{N}$ in **SNets** can be extended in a unique way into a morphism $(\eta, \sigma, \beta) : \mathcal{X} \rightarrow \rho_2(\mathcal{N})$ in **Tr-Nets**. Again functoriality of ρ_2 follows from that universal property. Let $\mathcal{X} = (\mathcal{A}^{(1)}, \models^{(1)}, \mathcal{N}^{(1)})$ and $\rho_2(\mathcal{N}) = (\mathcal{A}^{(2)}, \models^{(2)}, \mathcal{N}^{(2)})$. By condition (4) for morphisms in definition (3.13), $\sigma : Q^{(1)} \rightarrow Q^{(2)}$ must be defined as $\sigma(q) = \{x \in P^{(2)} / \ q \models^{(1)} \beta(x)\}$. The extended morphism (η, σ, β), if it exists, is therefore unique. It remains to verify that (η, σ) is a reduction between trace automata, i.e. satisfies conditions (2) for morphisms in definition (3.13). Let $\sigma : Q^{(1)} \rightarrow Q^{(2)}$ be defined as $\sigma(q) = \{x \in P^{(2)} / \ q \models^{(1)} \beta(x)\}$. In order to show that $\sigma : Q^{(1)} \rightarrow Q^{(2)}$, i.e. that $\sigma(q)$ is always an accessible marking of $\mathcal{N}^{(2)}$, there suffices to verify that σ is a rooted bisimulation. Since every state in $Q^{(1)}$ is accessible from the initial state

$q_0^{(1)}$, the fact that every marking $\sigma(q)$ is an accessible marking (i.e. an element of $Q^{(2)}$) will then immediately follow. First, $M_0^{(1)} = \{x \in P^{(1)} / \ q_0^{(1)} \models x\}$ because $\mathcal{X}$ is a trace net, and $q_0^{(2)} = M_0^{(2)} \subset 2^{P^{(2)}}$ by construction of $\rho_2(\mathcal{N})$, whence $\beta^{-1}(M_0^{(1)}) = \{y \in P^{(2)} / \ q_0^{(1)} \models \beta(x)\} = \sigma(q_0^{(1)})$ and $\sigma(q_0^{(1)}) = q_0^{(2)}$. Second, we have the equivalences

$$\begin{array}{lll} q \xrightarrow{a} & \textbf{iff} & a^- \subset q^\models \\ & \textbf{iff} & \forall x \in P^{(1)} \ \ a \leftarrow x \Rightarrow q \models x \end{array}$$

and similarly

$$\begin{array}{lll} \sigma(q) \xrightarrow{\eta(a)} & \textbf{iff} & \forall y \in P^{(2)} \ \ \eta(a) \leftarrow y \Rightarrow \sigma(q) \models y \\ & \textbf{iff} & \forall y \in P^{(2)} \ \ a \leftarrow \beta(y) \Rightarrow q \models \beta(y) \end{array}$$

hence the implication $q \xrightarrow{a} \Rightarrow \sigma(q) \xrightarrow{\eta(a)}$ automatically holds; the converse implication holds as well because the image of β is a dense subset of places in $\mathcal{N}_1$. Third, when the action a is enabled in q one has

$$\begin{array}{lll} x \in \sigma(q.a) & \textbf{iff} & q.a \models \beta(x) \qquad \text{by definition of } \sigma \\ & \textbf{iff} & (q \models \beta(x) \ \ \& \ \ a \notin {}^0\beta(x)) \text{ or } a \in {}^1\beta(x) \qquad \text{because } (q.a)^\models = (q^\models \setminus a^0) \cup a^1 \\ & \textbf{iff} & (\sigma(q) \models x \ \ \& \ \ \eta(a) \notin {}^0x) \text{ or } \eta(a) \in {}^1x \qquad \text{because (3 c) and (4)} \\ & \textbf{iff} & \sigma(q).\eta(a) \models x \qquad \text{i.e. } x \in \sigma(q).\eta(a) \end{array}$$

hence $\sigma(q.a) = \sigma(q).\eta(a)$.

□

The desired adjunction $F \dashv G$: **SNets** $\rightarrow$ **STA** between separated trace automata and saturated nets is the composition of the elementary adjunctions $\rho_1 \dashv \pi_1$ and $\pi_2 \dashv \rho_2$. The fact that both F and G preserves behaviours follows from corollary (3.16) which states that the underlying net and trace automaton of a trace net have the same behaviour. Actually among the trace nets we can find the elements $(\mathcal{A}, \in, F\mathcal{A})$ for $\mathcal{A} \in |\textbf{STA}|$ and the elements $(G\mathcal{N}, \ni, \mathcal{N})$ for $\mathcal{N} \in |\textbf{SNets}|$.

The following corollary emphasizes the fact (already mentioned) that all classes of behaviourally equivalent nets are represented in that correspondance.

Corollary 3.19 (Saturation of nets) *For every net $\mathcal{N}$ there exists a saturated net $\overline{\mathcal{N}}$ with the same behaviour.*

Proof: let $\mathcal{A}$ be the separated automaton induced by $\mathcal{N}$ (see proposition 3.4). The behaviour of $\mathcal{N}$ is the class of $\mathcal{A}$ in the equivalence generated by all strict reductions between canonical trace automata (definition 3.5). Hence the saturated net $\overline{\mathcal{N}} = F\mathcal{A}$ has the same behaviour as $\mathcal{N}$.

□

So we obtain saturated nets from arbitrary nets using the translations between nets and trace automata in both directions (see figure 9). The asymmetry between automata and nets reflects the definition of behaviours in term of transition systems.

A similar connection between nets and *local* trace automata may be established with strictly unchanged proofs applied to slightly modified statements. The modifications are the following. A local trace automaton is canonical if it satisfies the requirements stated in definition (3.3) except that in condition *(iii)* we no longer require the actions a and

b to be distinct. A local trace automaton is separated if it furthermore meets the property of separation stated in definition (3.10). The only modification to the translations between nets and automata is in the interpretation of independence in nets: we drop the requirement that actions a and b be distinct in definition (3.2). With that interpretation, an action such that $a^{\overset{0}{\frown}} = \emptyset$ is auto-concurrent. The result is the following.

Theorem 3.20 *There exists an adjunction $LF \vdash LG$: **SNets** → **SLTA** between the categories of separated local trace automata and saturated nets, such that both left and right adjoint functors preserve behaviours.*

4 Representation results

We will henceforth identify trace automata which differ only on their independence relation by pairs of events with no common enabling state. The purpose of this section is to establish that finite trace automata realized in basic SOS are exactly the separated ones. That assertion is equivalent to the following.

Proposition 4.1 (main representation result) *The set of trace automata realized by finite terms in process algebras defined in basic SOS is the set of all finite and separated trace automata.*

That statement relies heavily on the property to preserve concurrent behaviours satisfied by the adjunction $F \dashv G$: **Snets** → **STA** between separated trace automata and saturated nets. Suppose we have proved that every finite process automaton is separated, then the main representation result reduces to the following.

Proposition 4.2 (auxiliary representation result) *The set of nets realized by finite terms in process algebras defined in basic SOS is the set of all finite nets.*

We recall from section (3) that every net $\mathcal{N}$ has an associated trace automaton $G\mathcal{N}$ which is separated (the notation $G\mathcal{N}$ used till now for saturated nets is extended here to all nets). We also recall that the concurrent behaviour of a net $\mathcal{N}$ is the equivalence class of $G\mathcal{N}$ for bisimulation equivalence. Proposition (4.2) splits to the following pair of statements.

Lemma 4.3 *Given a finite net $\mathcal{N}$, one can construct an operational specification (Σ, R) in basic SOS and a finite Σ-term t such that $\|t\|_{\Sigma,R} = G\mathcal{N}$.*

Lemma 4.4 *For every operational specification (Σ, R) in basic SOS and for every finite Σ-term t, $\|t\|_{\Sigma,R} = G\mathcal{N}$ for some finite net $\mathcal{N}$.*

Lemma (4.4) tells us that every finite process automaton is separated since equal to $G\mathcal{N}$ for some $\mathcal{N}$. In order to establish simultaneously proposition (4.1) and proposition (4.2), there suffices therefore to prove lemma (4.3) and lemma (4.4).

Proof of lemma (4.3):
Given a finite net $\mathcal{N}$, we fill the associated signature Σ with an operator of synchronization ϕ with as many arguments as there are places in the net, plus two constant operators *empty* and *full*, allowing to encode bi-univocally markings $M \subset 2^P$ by terms $t_M = \phi(q_M^{(1)}, \ldots, q_M^{(n)})$,

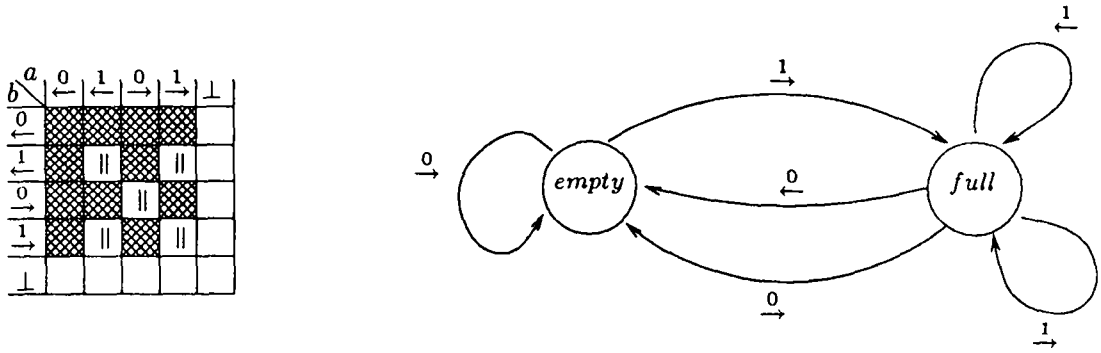


Figure 11: coincidence of the independence relations

where $q_M^{(i)} = full$ if $i \in M$ and $q_M^{(i)} = empty$ otherwise. We fill the operational specification R of Σ with six axioms ($empty \xrightarrow{(\overset{0}{\rightarrow})} empty$, $empty \xrightarrow{(\overset{1}{\rightarrow})} full$, $full \xrightarrow{(\overset{0}{\rightarrow})} empty$, $full \xrightarrow{(\overset{1}{\rightarrow})} full$, $full \xrightarrow{(\overset{0}{\leftarrow})} empty$, and $full \xrightarrow{(\overset{1}{\leftarrow})} full$) plus one inference rule per action a in the net, namely

$$\frac{x_i \xrightarrow{a_i} y_i \quad i \in I}{\phi(x_1, \ldots, x_n) \xrightarrow{a} \phi(y_1, \ldots, y_n)}$$

where $I = a^- \cup a^\rightarrow$ and $a_i = \nabla$ if and only if $a \nabla i$ (where $\nabla \in \{\overset{0}{\leftarrow}, \overset{1}{\leftarrow}, \overset{0}{\rightarrow}, \overset{1}{\rightarrow}\}$). Clearly, $M \xrightarrow{a}$ if and only if $t_M \xrightarrow{a}$ and then $t_{M.a} = (t_M).a$, whence the underlying transition systems of $G\mathcal{N}$ and $[\![t]\!]_{\Sigma,R}$ are isomorphic. Now, two actions a and b are independent according to the net $(a||b)$ if and only if all the hachured slots in the array on figure (11) are empty. And according to SOS, $a||b$ if and only if $a_i||b_i$ for every index i outside $a^\perp \cup b^\perp$ where the symmetric relation $||$ on $\{\overset{0}{\leftarrow}, \overset{1}{\leftarrow}, \overset{0}{\rightarrow}, \overset{1}{\rightarrow}\}$ is generated by $\overset{1}{\leftarrow}||\overset{1}{\leftarrow}$, $\overset{1}{\leftarrow}||\overset{1}{\rightarrow}$, $\overset{0}{\rightarrow}||\overset{0}{\rightarrow}$, and $\overset{1}{\rightarrow}||\overset{1}{\rightarrow}$ (see definition 2.4 and the SOS automaton in figure 11). Hence both notions of independence coincide, as shown by the array in figure (11).
□

Proof of lemma (4.4):
We recall from section (2) that $[\![t]\!]_{\Sigma,R} = \Gamma(\Sigma, R) > t$, where $\Gamma(\Sigma, R)$ is the Arnold-Nivat product $\prod_{s \in Dom(t)} \gamma(\Sigma, R)$ synchronized by the set $\mathcal{V}$ of vectors $V_A \in (R \cup \{\epsilon\})^{Dom(t)}$ defined from the schematic proofs A enabled in t by: $V_A(s) =$ A $(\varphi_A^{-1}(s))$ if $s \in Im(\varphi_A)$ and $V_A(s) = \epsilon$ otherwise. For an arbitrary set $\mathcal{V} \subset (R \cup \{\epsilon\})^I$ of vectors of synchronization, we let $\prod_{\mathcal{V}}$ denote the operator of synchronized product on local trace automata:

$$\prod_{\mathcal{V}} : \mathbf{LTA}^I \hookrightarrow \mathbf{TA}$$

where $\prod_{\mathcal{V}}(\mathcal{A}^{(i)}, i \in I)$ is the trace automaton with set of actions $\mathcal{V}$, where two distinct synchronization vectors $V, V' \in \mathcal{V}$ are independent ($V||V'$) if and only if for all index i, $V(i) = \epsilon$ or $V'(i) = \epsilon$ or $V(i)||V'(i)$ in $\mathcal{A}_i$. The proof that $\Gamma(\Sigma, R) > t$ is separated relies on two lemmas, stated below.

Lemma 4.5 *For every operational specifications (Σ, R) and for every operator symbol $f \in \Sigma$*

$$\gamma(\Sigma, R) > f = LG \circ LF\ (\gamma(\Sigma, R) > f)$$

where $LF \vdash LG$: **SNets** $\rightarrow$ **SLTA** *is the adjunction between local trace automata and saturated nets.*

Proof of lemma (4.5):
Let $\gamma(\Sigma, R) > f = \mathcal{A}_f = (R_f, \Sigma_f, f, \rightarrow_f)$ thus $R_f \subset R$ and $\Sigma_f \subset \Sigma$. We recall from section (2) that $\rho \| \rho'$ if and only if $g \xrightarrow{\rho} g$ and $g \xrightarrow{\rho'} g$ for some $g \in \Sigma_f$. By abuse of notation, let us rename Σ_f to Σ, R_f to R, and $\rightarrow_f$ to $\rightarrow$, whence $\mathcal{A}_f = (R, \Sigma, f, \rightarrow)$. In the remaining of the proof, we extend notations $LF(\mathcal{A})$ and $LG(\mathcal{N})$ to arbitrary local trace automata and nets respectively, keeping in mind that the correspondence (LF, LG) yields an adjunction only when restricted to separated trace automata and saturated nets. Let us construct the net $LF(\mathcal{A}_f)$. Since each proof constructor $\rho \in R$ codes the unique operator $g = \mathbf{target}(\rho)$ such that $h \xrightarrow{\rho} g$ for some h, every subset of Σ is a region of $\mathcal{A}_f$ (the forbidden cases indicated in figure (7) are certainly not possible). Thus, in $F\mathcal{A}_f = (P, R, \overset{0}{\leftarrow}, \overset{1}{\leftarrow}, \overset{0}{\rightarrow}, \overset{1}{\rightarrow}, \perp, M_0)$, $P = 2^{\Sigma}$. For $\rho \in R$, let $\mathbf{sources}(\rho) = \{g \in \Sigma / \ g \xrightarrow{\rho}\}$, then every place $x \in P$ is affected by every action $\rho \in R$, as shown by the following table:

	$\mathbf{sources}(\rho) \subset x$	$\mathbf{sources}(\rho) \not\subset x$
$\mathbf{target}(\rho) \in x$	$\rho \overset{1}{\leftarrow} x$	$\rho \overset{1}{\rightarrow} x$
$\mathbf{target}(\rho) \notin x$	$\rho \overset{0}{\leftarrow} x$	$\rho \overset{0}{\rightarrow} x$

In particular, $\perp = \emptyset$!

We will now show that $\mathcal{A}_f$ and $LG \circ LF\ (\mathcal{A}_f)$ are identical, which entails $\mathcal{A}_f \in$ **LSTA** and establishes the lemma.
By definition, the initial marking M_0 of $LF\ \mathcal{A}_f$ is $\{x \subset \Sigma\ /\ f \in x\}$. In order to prove that the underlying transition systems of local trace automata $\mathcal{A}_f$ and $LG \circ LF\ (\mathcal{A}_f)$ coincide, there suffices to show that every accessible markings of $LF\ (\mathcal{A}_f)$ has the form $\{x \subset \Sigma\ /\ h \in x\}$ for some h, and that $h \xrightarrow{\rho} g$ is a transition of $\mathcal{A}_f$ if and only if

$$\{x \subset \Sigma\ /h \in x\} \xrightarrow{\rho} \{x \subset \Sigma\ /g \in x\}$$

is a transition of $LG \circ LF\ (\mathcal{A}_f)$. By definition of M_0 and by induction on computations, those facts are immediate from the above table. There remains to show that the respective relations of independence in $\mathcal{A}_f$ and in $LG \circ LF\ (\mathcal{A}_f)$ coincide, i.e. that for every $\rho, \rho' \in R$ (where possibly $\rho = \rho'$):

$$\rho \| \rho' \quad \textbf{iff} \quad \rho^{\overset{0}{\leftarrow}} \subset \rho'^{\perp}\ , \quad \rho^{\mathbf{0}} \cap \rho'^{\mathbf{1}} = \emptyset, \quad \text{and symmetrically.}$$

Now, $\rho'^{\perp}$ is always empty, and the following equivalences hold:

$$\rho^{\mathbf{0}} = \emptyset \quad \textbf{iff} \quad \forall x\ .\ (\mathbf{sources}(\rho) \subset x \Rightarrow \mathbf{target}(\rho) \in x) \quad \textbf{iff} \quad \mathbf{target}(\rho) \xrightarrow{\rho} \mathbf{target}(\rho)$$

and

$$\rho^{\mathbf{0}} \cap \rho'^{\mathbf{1}} = \emptyset \quad \textbf{iff} \quad \mathbf{target}(\rho) = \mathbf{target}(\rho').$$

Thus, the relation to be established is equivalent to the following

$$\rho \| \rho' \quad \textbf{iff} \quad \exists g\ .\ g \xrightarrow{\rho} g\ \&\ g \xrightarrow{\rho'} g$$

which is true by definition.
□

Lemma 4.6 *One may construct a synchronized product $\prod'_{\mathcal{V}}$ of saturated nets making the following diagram of operators commute.*

$$\begin{array}{ccc} \mathbf{SLTA}^I & \xrightarrow{\prod_{\mathcal{V}}} & \mathbf{TA} \\ {\scriptstyle LG^I}\uparrow & & \uparrow{\scriptstyle G} \\ \mathbf{SNets}^I & \xrightarrow[\prod'_{\mathcal{V}}]{} & \mathbf{Nets} \end{array}$$

Proof of lemma (4.6):
For $i \in I$ let $\mathcal{N}_i$ $(P_i, A_i, \overset{0}{\leftarrow}_i, \overset{1}{\leftarrow}_i, \overset{0}{\rightarrow}_i, \overset{1}{\rightarrow}_i, \perp_i, (M_0)_i)$ then $\prod'_{\mathcal{V}} = (\mathcal{N}_i, i \in I) = (P, A, \overset{0}{\leftarrow}, \overset{1}{\leftarrow}, \overset{0}{\rightarrow}, \overset{1}{\rightarrow}, \perp, M_0)$ is the net with set of places $P = \coprod_{i \in I} P_i$, initial marking $M_0 = \bigcup_{i \in I} in_i((M_0)_i)$ where $in_i : P_i \rightarrow P$ is the canonical injections, and flow relations $p \nabla V$ are defined as follows for $p = in_i(p_i) \in P$, $\nabla \in \{\overset{0}{\leftarrow}, \overset{1}{\leftarrow}, \overset{0}{\rightarrow}, \overset{1}{\rightarrow}, \perp\}$, $V \in \mathcal{V}$:

$$p \nabla V \iff p_i \nabla_i V(i)$$

The transition systems underlying the respective trace automata $G(\prod'_{\mathcal{V}}(\mathcal{N}_i, i \in I))$ and $\prod_{\mathcal{V}}(LG\,(\mathcal{N}_i), i \in I))$ coincide because they are constructed from the same set of synchronization vectors. Now, two distinct vectors $V, V' \in \mathcal{V}$ are independent in the former trace automaton if and only if $V \overset{0}{\leftarrow} \subset V'^{\perp}$ & $V^0 \cap V'^1 = \emptyset$ and symmetrically, i.e. independence is the largest symmetric relation on synchronization vectors such that $\forall i \in I$

$$V(i) = \epsilon \quad \text{or} \quad V'(i) = \epsilon \quad \text{or} \quad (V(i) \overset{0}{\leftarrow} \subset V'(i)^{\perp} \;\&\; V(i)^0 \cap V'(i)^1 = \emptyset)$$

whence $V \| V'$ if and only if $\forall i \in I$ $(V(i) = \epsilon$ or $V'(i) = \epsilon$ or $V(i) \| V'(i)$ in $LG(\mathcal{N}_i))$, which corresponds to independence in the latter trace automaton.
□
Proof of lemma (4.4) continued:
By putting lemmas (4.5) and (4.6) together, we deduce $[\![t]\!]_{\Sigma,R}$ is separated since it is the image of a net (via G):

$$[\![t]\!]_{\Sigma,R} = G \circ \prod'_{\mathcal{V}} \circ [\prod_{s \in Dom(t)} LF(\gamma(\Sigma, R) > t(s))]$$

□

As a by-product, we obtain an alternative interpretation for basic SOS specifications in the setting of labelled nets, i.e. nets whose set of events is equipped with a labelling function $\lambda : E \rightarrow \Lambda$. Given (Σ, R), we define for each operator symbol $f \in \Sigma_n$ an alternative operator $\tilde{f}_{\mathcal{N}}$ on labelled nets, constructed as follows from $\mathcal{N}_f = LG\,(\gamma(\Sigma, R) > f)$. Let $E_f \subset R$ be the set of events of $\mathcal{N}_f$, and for $i = 1, \ldots, n$ let $\mathcal{N}_i$ be a net with set of events E_i and labelling functions $\lambda_i : E_i \rightarrow \Lambda$. Then $\tilde{f}_{\mathcal{N}}(\mathcal{N}_1, \ldots, \mathcal{N}_n) = \prod'_{\mathcal{V}}(\mathcal{N}_f, \mathcal{N}_1, \ldots, \mathcal{N}_n)$ where $\mathcal{V}$ is the set of vectors $< e_1, \ldots, e_n >$ satisfying the following conditions and equipped with the labelling function $\lambda < e_1, \ldots, e_n > = act(e_0)$:

- $e_0 = < \{i_1, \ldots, i_k\}, < a_1, \ldots, a_k, a >, g > \in E_f$,

- $e_{i_p} \in E_{i_p}$ and $\lambda_{i_p}(e_{i_p}) = a_p$ for $p \in \{1, \ldots, k\}$,
- $e_j = \epsilon$ for $j \notin \{i_1, \ldots, i_k\}$.

Let $[\![t]\!]^{\mathcal{N}}_{\Sigma,R}$ denote the induced net interpretation for Σ-terms t. By lemmas (4.5) and (4.6), $[\![t]\!]^{\mathcal{N}}_{\Sigma,R} = G\,[\![t]\!]^{\mathcal{A}}_{\Sigma,R}$, showing the adequacy of that alternative interpretation.

Acknowledgement: *The idea according to which true concurrency could be extracted from SOS was suggested to us by Frits Vaandragger.*

References

[AN82] ARNOLD, A., and NIVAT, M., *Comportements de processus.* Colloque AFCET « Les Mathématiques de l'Informatique », (1982) 35-68.

[Bed88] BEDNARCZYK, M.A., *Categories of asynchronous systems.* PhD thesis, University of Sussex, report no.1/88 (1988).

[BC88] BOUDOL, G., and CASTELLANI, I., *A non-interleaving semantics for CCS based on proved transitions.* Fundamenta Informaticae XI (1988) 433-452.

[BC90] BOUDOL, G., and CASTELLANI, I., *Three Equivalent Semantics for CCS.* Semantics of Systems of Concurrent Processes, I. Guessarian (Ed.) LNCS 469 (1990) 96-141.

[deS84] DE SIMONE, R., *Calculabilité et expressivité dans l'algèbre de processus MEIJE.* Thèse de 3eme Cycle, Université de Paris VII (1984).

[LX90] LARSEN, K., and XINXIN, *Compositionality through an operational semantics of contexts.* Proc. 17^{th} ICALP (Warwick), LNCS 443 (1990) 526-539.

[Mil80] MILNER, R., *A calculus of communicating systems.* Springer-Verlag LNCS 92 (1980).

[NRT90] NIELSEN, M., ROZENBERG, G., AND THIAGARAJAN, P.S., *Elementary Transition Systems.* DAIMI PB-310 Aarhus (1990).

[Niv79] NIVAT, M., *Sur la synchronisation des processus.* Revue technique Thomson-CSF, 11, (1979) 899-919.

[Plo81] PLOTKIN, G., *A structural approach to operational semantics.* DAIMI-FN-19 Aarhus (1981).

[Sta89a] STARK, E.W., *Concurrent transition systems.* Theoretical Computer Science 64 (1989) 221-269.

[Sta89b] STARK, E.W., *Connections between a concrete and an abstract model of concurrent systems.* 5^{th} Mathematical Foundations of programming semantics (1989) 53-79.

[Sta89c] STARK, E.W., *Compositional Relational Semantics for Indeterminate Dataflow Networks.* Summer Conference on Category Theory and Computer Science, LNCS 389 (1989) 52-74.

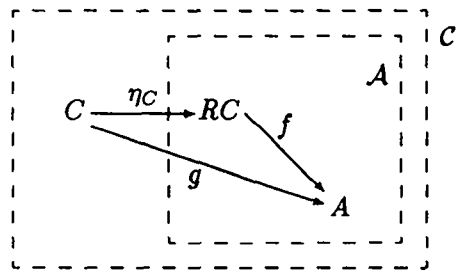
[Win91] WINSKEL, G., *Categories of Models for Concurrency.* Advanced school on the algebraic, logical, and categorical foundations of concurrency, Gargnano del Garda (1991).

Appendix

A subcategory $\mathcal{A}$ of $\mathcal{C}$ is called *reflective* when the inclusion functor $K : \mathcal{A} \to \mathcal{C}$ has a left adjoint called *reflector*. I.e. we have a bijective correspondence

$$\mathcal{A}(RC, A) \cong \mathcal{C}(C, A)$$

natural in C and A. R can be compared to a geometrical projection which takes an object C in $\mathcal{C}$ to its *best approximation* in $\mathcal{A}$. Actually, RA satisfies the following universal property : there exists an arrow $\eta_C : C \to RC$ and, for any object of $\mathcal{A}$ and arrow $g : C \to A$ there exists a unique arrow $f : RC \to A$ such that $g = f \circ \eta_C$.



More generally a functor $S : \mathcal{C} \to \mathcal{B}$ is said to be a *reflector* for $T : \mathcal{B} \to \mathcal{C}$ when S is a *left adjoint left inverse* of T, i.e. we have an adjunction $S \dashv T : \mathcal{B} \to \mathcal{C}$ with the identity $ST = 1_{\mathcal{B}}$ as co-unit. As T is split-mono the image of T is a full-subcategory of $\mathcal{C}$: T is clearly injective on objects and

$$\mathcal{C}(TB, TB') \overset{\phi}{\cong} \mathcal{B}(STB, B') = \mathcal{B}(B, B')$$

And actually the image of T is a reflective sub-category of $\mathcal{C}$.

Dually $\mathcal{A} \subset \mathcal{C}$ is *coreflective* when the inclusion functor has a right adjoint and more generally a *coreflector* is a right adjoint left inverse. In the ordered case we obtain :

$$UF = 1_{\mathcal{A}} \quad \textit{and} \quad FU \sqsubseteq 1_{\mathcal{B}}$$

(where $\sqsubseteq$ stands for the extensional order). So adjunctions with unit identity generalize the injection-projection pairs of Scott.

LISTE DES PUBLICATIONS INTERNES IRISA 1992

PI 624 SIGNAL AS A MODEL FOR REAL-TIME AND HYBRID SYSTEMS
Albert BENVENISTE, Michel LE BORGNE, Paul LE GUERNIC
Janvier 1992, 22 pages.

PI 625 ON THE CENTRAL-LIMIT THEOREM FOR TRACKING ESTIMATORS WITH SMALL GAIN - INFINITE HORIZON CASE
Bernard DELYON, Anatoli JUDITSKY
Janvier 1992, 16 pages.

PI 626 A MONTE CARLO METHOD BASED ON ANTITHETIC VARIATES FOR NETWORK RELIABILITY COMPUTATIONS
Mohamed EL KHADIRI, Gerardo RUBINO
Janvier 1992, 28 pages.

PI 627 CONSTRAINED MULTISCALE MARKOV RANDOM FIELDS AND THE ANALYSIS OF VISUAL MOTION
Fabrice HEITZ, Patrick PEREZ, Patrick BOUTHEMY
Janvier 1992, 40 pages.

PI 628 ON ITERATIVE REFINEMENT FOR THE SPECTRAL DECOMPOSITION OF SYMMETRIC MATRICES
Alexander N. MALYSHEV
Janvier 1992, 26 pages.

PI 629 STRUCTURAL OPERATIONAL SPECIFICATIONS AND TRACE AUTOMATA
Eric BADOUEL, Philippe DARONDEAU
Janvier 1992, 36 pages.

PI 630 EREBUS, A DEBUGGER FOR ASYNCHRONOUS DISTRIBUTED COMPUTING SYSTEM
Michel HURFIN, Noël PLOUZEAU, Michel RAYNAL
Janvier 1992, 14 pages.

PI 631 PROTOCOLES SIMPLES POUR L'IMPLEMENTATION REPARTIE DES SEMAPHORES
Michel RAYNAL
Janvier 1992, 14 pages.

PI 632 L-STABLE PARALLEL ONE-BLOCK METHODS FOR ORDINARY DIFFERENTIAL EQUATIONS
Philippe CHARTIER, Bernard PHILIPPE
Janvier 1992, 28 pages.

PI 633 ON EFFICIENT CHARACTERIZING SOLUTIONS OF LINEAR DIOPHANTINE EQUATIONS AND ITS APPLICATION TO DATA DEPENDENCE ANALYSIS
Christine EISENBEIS, Olivier TEMAM, Harry WIJSHOFF
Janvier 1992, 22 pages.

PI 634 UN NOYAU DE SYSTEME REPARTI POUR LES APPLICATIONS GEREES PAR UN TEMPS VIRTUEL
Janvier 1992, 20 pages.

ISSN 0249-6399