In proceedings of CONCUR '93, LNCS n. 715, 1993

A Classification of Models for Concurrency

Vladimiro Sassone[*] *Mogens Nielsen*[**]
Glynn Winskel[**]

[*]Dipartimento di Informatica, Università di Pisa, Italy
[**]Computer Science Department, Aarhus University, Denmark

Extended Abstract

Models for concurrency can be classified with respect to the three relevant parameters: behaviour/system, interleaving/noninterleaving, linear/branching time. When modelling a process, a choice concerning such parameters corresponds to choosing the level of abstraction of the resulting semantics. The classifications are formalised through the medium of category theory.

Introduction

From its beginning, many efforts in the development of the *theory of concurrency* have been devoted to the study of suitable models for concurrent and distributed processes, and to the formal understanding of their semantics.

As a result, in addition to standard models like languages, automata and transition systems [4, 9], models like *Petri nets* [8], *process algebras* [6, 2], *Hoare traces* [3], *Mazurkiewicz traces* [5], *synchronization trees* [15] and *event structures* [7, 16] have been introduced.

The idea common to the models above is that they are based on atomic units of change—be they called transitions, actions, events or symbols from an alphabet—which are *indivisible* and constitute the steps out of which computations are built.

The difference between the models may be expressed in terms of the parameters according to which models are often classified. For instance, a distinction made explicitly in the theory of Petri nets, but sensible in a wider context, is that between so-called *"system"* models allowing an explicit representation of the (possibly repeating) states in a system, and *"behaviour"* models abstracting away from such information, which focus instead on the behaviour in terms of patterns of occurrences of actions over time. Prime examples of the first type are transition systems and Petri nets, and of the second type, trees, event structures and traces. Thus, we can distinguish among models according to whether they

[*]This author has been supported by a grant of the Danish Research Academy.

Beh./Int./Lin.	Hoare languages	**HL**
Beh./Int./Bran.	synchronization trees	**ST**
Beh./Nonint./Lin.	deterministic labelled event structures	**dLES**
Beh./Nonint./Bran.	labelled event structures	**LES**
Sys./Int./Lin.	deterministic transition systems	**dTS**
Sys./Int./Bran.	transition systems	**TS**
Sys./Nonint./Lin.	deterministic transition systems with independence	**dTSI**
Sys./Nonint./Bran.	transition systems with independence	**TSI**

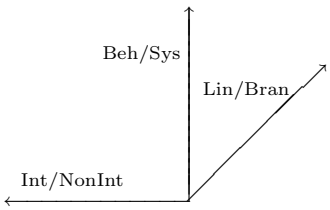
Table 1: The models

are *system* models or *behaviour* models, in this sense; whether they can faithfully take into account the difference between *concurrency* and *nondeterminism*; and, finally, whether they can represent the *branching structure* of processes, i.e., the points in which choices are taken, or not. Relevant parameters when looking at models for concurrency are

Behaviour or System model;
Interleaving or Noninterleaving model;
Linear or Branching Time model.

The parameters cited correspond to choices in the *level of abstraction* at which we examine processes and which are not necessarily fixed for the process once and for all. It is the application one has in mind for the formal semantics which guides the choice of abstraction level.

This work presents a classification of models for concurrency based on the three parameters, which represents a further step towards the identification of systematic connections between transition based models. In particular, we study a *representative* for each of the eight classes of models obtained by varying the parameters *behaviour/system*, *interleaving/noninterleaving* and *linear/branching* in all the possible ways. Intuitively, the situation can be graphically represented, as in the picture below, by a three-dimensional frame of reference whose coordinate axes represent the three parameters.



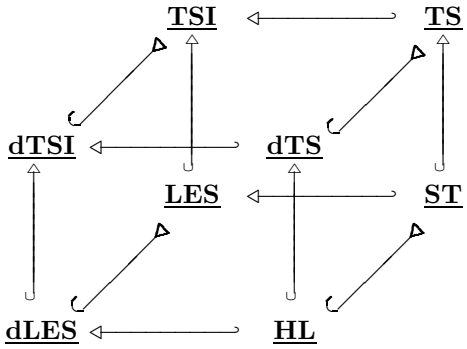
Our choices of models are summarized in Table 1. It is worth noticing that, with the exception of the new model of *transition systems with independence*, each model is well-known.

The formal relationships between models are studied in a *categorical* setting, using the standard categorical tool of *adjunctions*. The "translations" between

models we shall consider are *coreflections* or *reflections*. These are particular kinds of adjunctions between two categories which imply that one category is *embedded*, fully and faithfully, in another.[1]

Here we draw on our experience in recasting models for concurrency as categories, detailed in [17]. Briefly the idea is that each model (transition systems are one such model) will be equipped with a notion of morphism, making it into a category in which the operations of process calculi are universal constructions. The morphisms will preserve behaviour, at the same time respecting a choice of granularity of the atomic changes in the description of processes—the morphisms are forms of *simulations*. One role of the morphisms is to relate the behaviour of a construction on processes to that of its components. The reflections and coreflections provide a way to express that one model is embedded in (is more abstract than) another, even when the two models are expressed in very different mathematical terms. One adjoint will say how to embed the more abstract model in the other, the other will abstract away from some aspect of the representation. The preservation properties of adjoints can be used to show how a semantics in one model *translates* to a semantics in another.

The picture below, in which arrows represent coreflections and the "backward" arrows reflections, shows the "cube" of relationships.



Although most of the chosen models are well known, among the adjunctions in the cube only **HL** $\hookleftarrow$ **ST**, **ST** $\hookrightarrow$ **TS** and **ST** $\hookrightarrow$ **LES** have already appeared in literature.

Here all proofs are omitted. The reader interested in the details is referred to [10, 11]. Some of the results presented here will appear in [17].

1 Interleaving Models

In this section, we study the interleaving models. We start by briefly recalling some well-known relationships between languages, trees and transition systems [17], and then, we study how they relate to deterministic transition systems.

[1] Here a coreflection is an adjunction in which the unit is natural isomorphism, and a reflection an adjunction where the counit is a natural isomorphism.

DEFINITION 1.1 (*Labelled Transition Systems*)
A labelled transition system is a structure $T = (S, s^I, L, Tran)$ *where* S *is a set of states;* $s^I \in S$ *is the initial state,* L *is a set of labels, and* $Tran \subseteq S \times L \times S$ *is the transition relation.*

Given the labelled transition systems T_0 *and* T_1*, a morphism* $h\colon T_0 \to T_1$ *is a pair* (σ, λ)*, where* $\sigma\colon S_{T_0} \to S_{T_1}$ *is a function and* $\lambda\colon L_{T_0} \rightharpoonup L_{T_1}$ *a partial function, such that*[2]

i) $\sigma(s^I_{T_0}) = s^I_{T_1}$;

ii) $(s, a, s') \in Tran_{T_0}$ *implies if* $\lambda{\downarrow}a$ *then* $\Big(\sigma(s), \lambda(a), \sigma(s')\Big) \in Tran_{T_1}$;
$\sigma(s) = \sigma(s')$ *otherwise.*

This defines the category **TS** *of labelled transition systems.*

DEFINITION 1.2 (*Synchronization Trees*)
A synchronization tree is an acyclic, reachable transition system S *such that*

$$(s', a, s),\ (s'', b, s) \in Tran_S \qquad \text{implies} \qquad s' = s'' \quad \text{and} \quad a = b.$$

We write **ST** *to denote the full subcategory of* **TS** *consisting of synchronization trees.*

DEFINITION 1.3 (*Hoare Languages*)
A Hoare language is a pair (H, L)*, where* $\varnothing \neq H \subseteq L^*$*, and* $sa \in H \Rightarrow s \in H$.
A partial map $\lambda\colon L_0 \rightharpoonup L_1$ *is a morphism of Hoare languages from* (H_0, L_0) *to* (H_1, L_1) *if for each* $s \in H_0$ *it is* $\hat{\lambda}(s) \in H_1$*, where* $\hat{\lambda}\colon L_0^* \to L_1^*$ *is defined by*

$$\hat{\lambda}(\epsilon) = \epsilon \qquad \text{and} \qquad \hat{\lambda}(sa) = \begin{cases} \hat{\lambda}(s)\lambda(a) & \text{if } \lambda{\downarrow}a; \\ \hat{\lambda}(s) & \text{otherwise.} \end{cases}$$

These data give the category **HL** *of Hoare languages.*

Observe that any language (H, L) can be seen as a synchronization tree just by considering the strings of the language as states, the empty string being the initial state, and defining a transition relation where $s \xrightarrow{a} s'$ if and only if $sa = s'$. This extends to a functor from **HL** to **ST**. This functor and the inclusion functor from **ST** to **TS** may be viewed as *embeddings*, as stated in the following theorem, where $\hookrightarrow\!\triangleright$ represents a coreflection and $\hookrightarrow\!\triangleleft$ a reflection, and the functors from right to left abstract away from information about states (from **TS** to **ST**) and branching structure (from **ST** to **HL**).

THEOREM 1.4

$$\mathbf{HL} \hookrightarrow\!\triangleleft \mathbf{ST} \hookrightarrow\!\triangleright \mathbf{TS}$$

[2] We use $f{\downarrow}x$ to mean that a partial function f is defined on argument x.

The right adjoint from **TS** to **ST** is the well-known construction of unfolding, while the left adjoint from **ST** to **HL** is a straightforward "set of paths" construction.

The existence of a (co)reflection from category **A** to **B** tells us that there is a full subcategory of **B** which is *equivalent* (in the formal sense of equivalences of categories) to **A**. Therefore, once we have established a (co)reflection, it is sometimes interesting to identify such subcategories. In the case of **HL** and **ST** such a question is answered below.

PROPOSITION 1.5 (*Languages are deterministic Trees*)
The full subcategory of **ST** *consisting of those synchronization trees which are deterministic, say* **dST**, *is equivalent to the category of Hoare languages.*

Speaking informally, behaviour/system and linear/branching are independent parameters, and we expect to be able to forget the branching structure of a transition system without necessarily losing all the internal structure of the system. This leads us to identify a class of models able to represent the internal structure of processes without keeping track of their branching, i.e., the points at which the choices are actually taken. A suitable model is given by *deterministic transition systems*.

DEFINITION 1.6 (*Deterministic Transition Systems*)
A transition system T is deterministic if

$$(s, a, s'),\ (s, a, s'') \in \mathit{Tran}_T \quad \mathit{implies} \quad s' = s''.$$

Let **dTS** *be the full subcategory of* **TS** *consisting of those transition systems which are deterministic.*

Consider the binary relation $\simeq$ on the state of a transition system T defined as the least equivalence which is *forward closed*, i.e.,

$$s \simeq s' \quad \text{and} \quad (s, a, u), (s', a, u') \in \mathit{Tran}_T \quad \Rightarrow \quad u \simeq u';$$

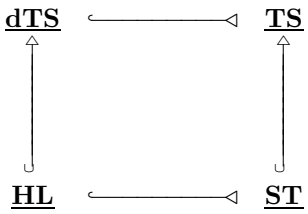
and define $ts.dts(T) = \Big(S/\!\simeq, [s_T^I]_\simeq, L_T, \mathit{Tran}_\simeq\Big)$, where $S/\!\simeq$ are the equivalence classes of $\simeq$, and

$$\Big([s]_\simeq, a, [s']_\simeq\Big) \in \mathit{Tran}_\simeq \quad \Leftrightarrow \quad \exists(\bar{s}, a, \bar{s}') \in \mathit{Tran}_T \text{ with } \bar{s} \simeq s \text{ and } \bar{s}' \simeq s'.$$

This construction defines a functor which is left adjoint to the inclusion **dTS** $\hookrightarrow$ **TS**.

Next, let us present a universal construction from Hoare languages to deterministic transition system. In particular, we exhibit a coreflection **HL** $\hookrightarrow\!\triangleright$ **dTS**. Let (H, L) be a language. The left adjoint is defined on objects as follows: $hl.dts(H, L) = (H, \epsilon, L, \mathit{Tran})$, where $(s, a, sa) \in \mathit{Tran}$ for any $sa \in H$, which is trivially a deterministic transition system. The right adjoint is a standard "set of traces" construction. Thus, we have enriched our diagram and we have a square.

THEOREM 1.7 (*The Interleaving Surface*)



Observe that the construction of the deterministic transition system associated to a language coincides exactly with the construction of the corresponding synchronization tree. However, due to the different objects in the categories, the type of universality of the construction changes. In other words, the same construction shows that **HL** is *reflective* in **ST**—a full subcategory of **TS**—and *coreflective* in **dTS**—another full subcategory of **TS**.

2 Non Interleaving vs. Interleaving Models

Event structures [7, 16] abstract away from the cyclic structure of the process and consider only events (strictly speaking event *occurrences*), assumed to be the *atomic computational steps*, and the cause/effect relationships between them. Thus, we can classify event structures as behavioural, branching and noninterleaving models. Here, we are interested in labelled event structures.

DEFINITION 2.1 (*Labelled Event Structures*)
A labelled event structure is a structure $ES = (E, \#, \leq, \ell, L)$ *consisting of a set of events* E *partially ordered by* $\leq$*; a symmetric, irreflexive relation* $\# \subseteq E \times E$*, the conflict relation, such that*

$$\{\, e' \in E \mid e' \leq e \,\} \text{ is finite for each } e \in E,$$
$$e \# e' \leq e'' \text{ implies } e \# e'' \text{ for each } e, e', e'' \in E;$$

a set of labels L *and a labelling function* $\ell\colon E \to L$*. For an event* $e \in E$*, define* $\lfloor e \rfloor = \{\, e' \in E \mid e' \leq e \,\}$*. Moreover, we write* $\Cup$ *for* $\# \cup \{\, (e,e) \mid e \in E_{ES} \}$*. These data define a relation of concurrency on events:* $co = E_{ES}^2 \setminus (\leq \cup \leq^{-1} \cup \#)$*.*
A labelled event structure morphism from ES_0 *to* ES_1 *is a pair of partial functions* (η, λ)*, where* $\eta\colon E_{ES_0} \rightharpoonup E_{ES_1}$ *and* $\lambda\colon L_{ES_0} \rightharpoonup L_{ES_1}$ *are such that*

i) $\lfloor \eta(e) \rfloor \subseteq \eta(\lfloor e \rfloor)$*;*

ii) $\eta(e) \Cup \eta(e')$ *implies* $e \Cup e'$*;*

iii) $\lambda \circ \ell_{ES_0} = \ell_{ES_1} \circ \eta$*.*

This defines the category **LES** *of labelled event structures.*

The computational intuition behind event structures is simple: an event e can occur when all its *causes*, i.e., $\lfloor e \rfloor \setminus \{e\}$, have occurred and no event which is in conflict with has already occurred. This is formalized by the following notion of *configuration*.

DEFINITION 2.2 (*Configurations*)
Given a labelled event structure ES, define the configurations of ES to be those subsets $c \subseteq E_{ES}$ which are

$$\begin{aligned} \textit{Conflict Free:} &\quad \forall e_1, e_2 \in c,\ \textit{not } e_1 \# e_2 \\ \textit{Left Closed:} &\quad \forall e \in c\ \forall e' \leq e,\ e' \in c \end{aligned}$$

Let $\mathcal{L}(ES)$ denote the set of configurations of ES.
We say that e is enabled at a configuration c, in symbols $c \vdash e$, if

(i) $e \notin c$; *(ii)* $\lfloor e \rfloor \setminus \{e\} \subseteq c$; *(iii)* $e' \in E_{ES}$ *and* $e' \# e$ *implies* $e' \notin c$.

Although it is not the only possible one (see [11], where a category of *pomset languages* and a category of *generalized trace languages* are considered), a fair choice for behavioural, linear and noninterleaving models is given by *deterministic labelled event structures*.

DEFINITION 2.3 (*Deterministic Labelled Event Structures*)
A labelled event structure ES is deterministic if for any $c \in \mathcal{L}(ES)$, and for any pair of events $e, e' \in E_{ES}$, whenever $c \vdash e$, $c \vdash e'$ and $\ell(e) = \ell(e')$, then $e = e'$.
This defines the category **dLES** *as a full subcategory of* **LES**.

In [14], it is shown that synchronization trees and labelled event structures are related by a coreflection from **ST** to **LES**. The left adjoint associates with an **ST**-object S, a **LES**-object where the events are the transitions of S, $\leq$ is the tree-ordering of these, and $\#$ is the relation of incompatibility of transitions in this ordering. Hence *co* is empty, and in fact, the analogous of Proposition 1.5 for this coreflection states that synchronization trees are event structures with empty concurrency relations. The coreflection $\mathbf{ST} \hookrightarrow \mathbf{LES}$ restricts to a coreflection $\mathbf{dST} \hookrightarrow \mathbf{dLES}$, and hence, by Proposition 1.5, to a coreflection $\mathbf{HL} \hookrightarrow \mathbf{dLES}$.

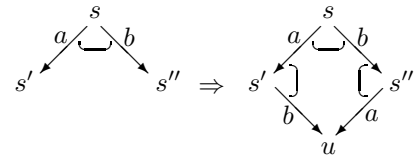
Now, on the system level we look for a way of equipping transition systems with a notion of "concurrency" or "independence", in the same way as **LES** may be seen as adding "concurrency" to **ST**. Moreover, such enriched transition systems should also represent the "system model" version of event structures. Several such models have appeared in the literature [12, 1, 13]. Here we choose a variation of these.

DEFINITION 2.4 (*Transition Systems with Independence*)
A transition system with independence is a structure $(S, s^I, L, Tran, I)$ consisting of a transition system $(S, s^I, L, Tran)$ together with an irreflexive, symmetric relation $I \subseteq Tran^2$ such that

i) $(s,a,s') \sim (s,a,s'') \quad \Rightarrow \quad s' = s''$;

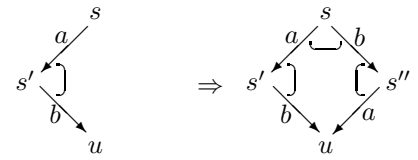
ii) $(s,a,s')\ I\ (s,b,s'') \quad \Rightarrow \quad \exists u.\ (s,a,s')\ I\ (s',b,u)$ *and* $(s,b,s'')\ I\ (s'',a,u)$;

i.e.,



iii) $(s,a,s')\ I\ (s',b,u) \quad \Rightarrow \quad \exists s''.\ (s,a,s')\ I\ (s,b,s'')$ *and* $(s,b,s'')\ I\ (s'',a,u)$;

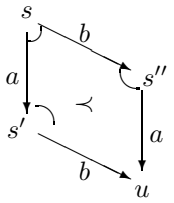
i.e.,



iv) $(s,a,s') \sim (s'',a,u)\ I\ (w,b,w') \quad \Rightarrow \quad (s,a,s')\ I\ (w,b,w')$;

where $\sim$ *is least equivalence on transitions including the relation* $\prec$ *defined by*

$$(s,a,s') \prec (s'',a,u) \quad \Leftrightarrow \quad \begin{array}{l}(s,a,s')\ I\ (s,b,s'') \text{ and} \\ (s,a,s')\ I\ (s',b,u) \text{ and} \\ (s,b,s'')\ I\ (s'',a,u).\end{array}$$



Morphisms of transition systems with independence are morphisms of the underlying transition systems which preserve independence, i.e., such that

$$(s,a,s')\ I\ (\bar{s},b,\bar{s}') \text{ and } \lambda{\downarrow}a,\ \lambda{\downarrow}b \quad \Rightarrow \quad \Big(\sigma(s),\lambda(a),\sigma(s')\Big)\ I\ \Big(\sigma(\bar{s}),\lambda(b),\sigma(\bar{s}')\Big).$$

These data define the category **TSI** *of transition systems with independence. Moreover, let* **dTSI** *denote the full subcategory of* **TSI** *consisting of transition systems with independence whose underlying transition system is deterministic.*

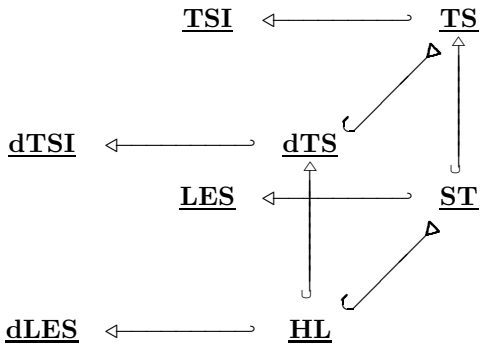
Thus, transition systems with independence are precisely standard transition systems but with an additional relation expressing when one transition is independent of another. The relation $\sim$, defined as the reflexive, symmetric and transitive closure of a relation $\prec$ which simply identifies local "diamonds" of concurrency, expresses when two transitions represent occurrences of the same event. Thus, the equivalence classes $[(s,a,s')]_\sim$ of transitions (s,a,s') are the *events* of the transition system with independence.

Transition systems with independence admit **TS** as a coreflective subcategory. In this case, the adjunction is easy. The left adjoint associates to any

transition system T the transition system with independence whose underlying transition system is T itself and whose independence relation is empty. The right adjoint simply forgets about the independence, mapping any transition system with independence to its underlying transition system. From the definition of morphisms of transition systems with independence, it follows easily that these mappings extend to functors which form a coreflection $\underline{\mathbf{TS}} \hookrightarrow \underline{\mathbf{TSI}}$. Moreover, such a coreflection trivially restricts to a coreflection $\underline{\mathbf{dTS}} \hookrightarrow \underline{\mathbf{dTSI}}$.

So, we are led to the following diagram.

THEOREM 2.5 (*Moving along the "interleaving/noninterleaving" axis*)



3 Non Interleaving Models

Now, the question is whether the relationships between the interleaving models generalize to our chosen noninterleaving counterparts. This is indeed so, and in the following we sketch the generalizations and their proofs.

First of all, it is quite easy to see that the inclusion functor from $\underline{\mathbf{ST}}$ to $\underline{\mathbf{TS}}$ generalizes naturally to a functor *les.tsi* from $\underline{\mathbf{LES}}$ to $\underline{\mathbf{TSI}}$. Consider a labelled event structure $ES = (E, \leq, \#, \ell, L)$. Define $les.tsi(ES)$ to be the transition system with independence of the *finite configurations* of ES, i.e.,

$$les.tsi(ES) = \Big(\mathcal{L}_F(ES), \varnothing, L, \mathit{Tran}, I\Big),$$

where

- $\mathcal{L}_F(ES)$ is the set of finite configuration of ES;
- $(c, a, c') \in \mathit{Tran}$ if and only if $c = c' \setminus \{e\}$ and $\ell(e) = a$;
- $(c, a, c')\ I\ (\bar{c}, b, \bar{c}')$ if and only if $(c' \setminus c)\ co\ (\bar{c}' \setminus \bar{c})$.

This is basically nothing but formalizing the computational intuition of event structures (Definition 2.2) in terms of $\underline{\mathbf{TSI}}$. Our proof that this is part of a coreflection, involves (yet another) characterization of $\underline{\mathbf{LES}}$—here in terms of $\underline{\mathbf{TSI}}$—which also supports our claim of transition systems with independence being a "generalization" of event structures.

THEOREM 3.1 (*Event Structures are Transition Systems with Independence*) *Let* **oTSI** *(occurrence transition systems with independence) denote the full subcategory of* **TSI** *where the objects* $OTI = (S, s^I, L, Tran, I)$ *are reachable, acyclic and such that*

$$(1) \qquad \begin{array}{l} (s', a, u) \neq (s'', b, u) \in Tran \quad implies \\ \quad \exists s.\ (s, b, s')\ I\ (s, a, s'')\ and\ (s, b, s')\ I\ (s', a, u) \\ \qquad\qquad\qquad\qquad\qquad\qquad and\ (s, a, s'')\ I\ (s'', b, u), \end{array}$$

or, in other words, (s', a, u) *and* (s'', b, u) *form the bottom of a concurrency diamond, and*

$$(2) \qquad t\ I\ t' \quad \Rightarrow \quad \exists s.\ (s, a, s') \sim t \quad and \quad (s, b, s'') \sim t'.$$

Then **oTSI** *and* **LES** *are equivalent categories. Moreover, the equivalence cuts down to an equivalence of* **doTSI** *and* **dLES**.

It is immediate to realize that $les.tsi(ES)$ is an occurrence transition system with independence; thus, *les.tsi* factorizes through $les.otsi$: **LES** $\rightarrow$ **oTSI** and the inclusion **oTSI** $\hookrightarrow$ **TSI**. In order to give a hint about the equivalences of Theorem 3.1, we sketch the transformation from **oTSI** to **LES** opposite to *les.otsi*. The labelled event structure associated to OTI is

$$otsi.les(OTI) = \Big(Tran_{OTI}/\!\sim, \leq, \#, \ell, L_{OTI}\Big),$$

where

- $[(s, a, s')]_\sim \leq [(\bar{s}, b, \bar{s}')]_\sim$ if and only if

  for any path π of OTI, $\quad [(\bar{s}, b, \bar{s}')]_\sim \in \pi \quad \Rightarrow \quad [(s, a, s')]_\sim \in \pi$;

- $[(s, a, s')]_\sim \ \# \ [(\bar{s}, b, \bar{s}')]_\sim$ if and only if

  for any path π of OTI, $\quad [(\bar{s}, b, \bar{s}')]_\sim \in \pi \quad \Rightarrow \quad [(s, a, s')]_\sim \notin \pi$;

- $\ell\Big([(s, a, s')]_\sim\Big) = a$;

and where we write $[(s, a, s')]_\sim \in \pi$ to mean that a representative of $[(s, a, s')]_\sim$ occurs in the path π. An example of this construction is given in Figure 1.

The proof of Theorem 3.1 consists of showing that *otsi.les* and *les.otsi* form an equivalence of categories.

Now, the right adjoint to *les.tsi* can be defined in terms of a notion of unfolding *tsi.otsi* of **TSI**'s into **oTSI**'s which is the right adjoint of the inclusion functor. The unfolding construction takes into account that computations which only differ for the order in which (representatives of) independent events ($\sim$-classes) occur must be considered equivalent. Thus, differently from the interleaving

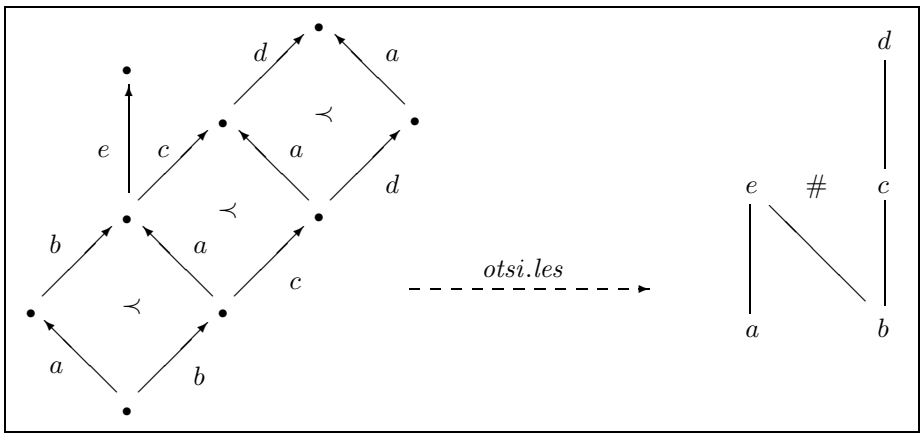
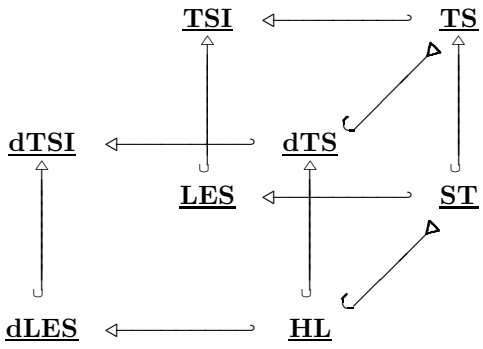


Figure 1:

case, the unfolding is not a tree: the states reached by equivalent computations coincide in the unfolded transition system. The proof that this form a coreflection is rather involved, but once established it is quite easy to see that it restricts to a coreflection **doTSI** $\hookrightarrow\!\triangleright$ **dTSI**. Thus, we have the coreflections **LES** $\hookrightarrow\!\triangleright$ **TSI** and **dLES** $\hookrightarrow\!\triangleright$ **dTSI**. The result is summarized in the following theorem.

THEOREM 3.2 (*Moving along the "behaviour/system" axis*)



Finally, we consider the relationship between **dTSI** and **TSI**, looking for a generalization of the reflection **dTS** $\hookrightarrow\!\triangleleft$ **TS**. Of course, the question to be answered is whether a left adjoint for the inclusion functor **dTSI** $\hookrightarrow$ **TSI** exists or not. This is actually a complicated issue and for a long while we could not see the answer. However, we can now answer positively! The definition of the functor and the proof of the reflection are rather involved, since the presence of an independence relation, together with the axioms of transition systems with independence, raises considerably the complexity of the problem we faced easily

in the interleaving case. In particular, the object component of the functor is an inductive construction which builds on *ts.dts* taking independence into account. It is quite easy to realize that applying the quotient construction which defines *ts.dts* to transition systems with independence does not yield transition systems with independence, since it does not preserve the irreflexivity of the independence relation and properties *(ii)* and *(iii)* of Definition 2.4. However, an inductive construction which *"completes"* the outcome of *ts.dts*, building a transition system with independence upon it, can be defined. Technically, this is achieved by constructing an ω-chain of transition systems which fail to be transition systems with independence at most because of the irreflexivity of the independence relation and axioms *(ii)* and *(iii)*, say pre-transition systems with independence, and by showing that the (co)limit of such a chain exists and is a deterministic transition system with independence. Such a (co)limit construction is the object component of the left adjoint to the inclusion $\underline{\mathbf{dTSI}} \hookrightarrow \underline{\mathbf{TSI}}$.

Of course, the non-irreflexivity of independence can be dealt only by eliminating the transitions involved, and since it arises from *autoconcurrency*, i.e., diamonds of concurrency whose edges carry the same label, this means that we eliminate the nondeterminism arising from autoconcurrency by eliminating all the events involved. However, it is not difficult to understand that, in our context, this is the only possibile choice. (The reader can convince himself by studying the problem on the very simple transition system with independence consisting of a single diamond of autoconcurrency.)

Given sets S and L, consider triples of the kind $(X, \equiv, I)$, where $X \subseteq S \cdot L^* = \{s\alpha \mid s \in S \text{ and } \alpha \in L^*\}$, and $\equiv$ and I are binary relations on X. On such triples, the following closure properties can be considered.

(Cl1) $x \equiv z$ and $za \in X$ implies $xa \in X$ and $xa \equiv za$;

(Cl2) $x \equiv z$ and $za\ I\ yc$ implies $xa\ I\ yc$;

(Cl3) $xab \equiv xba$ and $xa\ I\ xb$ or $xa\ I\ xab$ implies $xa\ I\ yc \Leftrightarrow xba\ I\ yc$.

We say that $(X, \equiv, I)$ is *suitable* if $\equiv$ is an equivalence relation, I is a symmetric relation and it enjoys properties *(Cl1)*, *(Cl2)* and *(Cl3)*. Suitable triples are meant to represent deterministic pre-transition systems with independence, the elements in X representing both states and transitions. Namely, xa represents the state reached from (the state corresponding to) x with an a-labelled transition, and that transition itself. Thus, equivalence $\equiv$ relate paths which lead to the same state and relation I expresses independence of transitions. With this understanding, *(Cl1)* means that from any state there is at most one a-transition, while *(Cl2)* says that I acts on transitions rather than on their representation. Finally, *(Cl3)*—the analogous of axiom *(iv)* of transition systems with independence—tells that transitions on the opposite edges of a diamond behave the same with respect to I.

For $x \in S \cdot L^*$ and $a \in L$, let $x\restriction a$ denote the pruning of x with respect to a.

Formally,

$$s{\restriction}a = s \qquad \text{and} \qquad (xb){\restriction}a = \begin{cases} x{\restriction}a & \text{if } a = b \\ (x{\restriction}a)b & \text{otherwise} \end{cases}$$

Of course it is possibile to use unambiguously $x{\restriction}A$ for $A \subseteq L$. Moreover, we use $X{\restriction}A$ to denote the set $\{x{\restriction}A \mid x \in X\}$ and for R a binary relation on X, $R{\restriction}A$ will be $\{(x{\restriction}A, y{\restriction}A) \mid (x,y) \in R\}$.

For a transition system with independence $TI = (S, s^I, L, \mathit{Tran}, I)$, we define the sequence a triples $(S_i, \equiv_i, I_i)$, for $i \in \omega$, inductively as follows. For $i = 0$, $(S_0, \equiv_0, I_0)$ is the least (with respect to componentwise set inclusion) *suitable* triple such that

$$S \cup \Big\{ sa \;\Big|\; (s,a,u) \in \mathit{Tran} \Big\} \subseteq S_0; \quad \Big\{ (sa,u) \;\Big|\; (s,a,u) \in \mathit{Tran} \Big\} \subseteq \equiv_0;$$

and

$$\Big\{ (sa, s'b) \;\Big|\; (s,a,u)\ I\ (s',b,u') \Big\} \subseteq I_0;$$

and, for $i > 0$, $(S_i, \equiv_i, I_i)$ is the least *suitable* triple such that

(ℑ) $S_{i-1}{\restriction}A_{i-1} \subseteq S_i$; $\equiv_{i-1}{\restriction}A_{i-1} \subseteq \equiv_i$; $(I_{i-1} \setminus TA_{i-1}){\restriction}A_{i-1} \subseteq I_i$;

(D1) $xa,\ xb \in S_{i-1}{\restriction}A_{i-1}$ and $xa\ (I_{i-1} \setminus TA_{i-1}){\restriction}A_{i-1}\ xb$ implies $xab,\ xba \in S_i$ and $xab \equiv_i xba$;

(D2) $xa,\ xab \in S_{i-1}{\restriction}A_{i-1}$ and $xa\ (I_{i-1} \setminus TA_{i-1}){\restriction}A_{i-1}\ xab$ implies $xb,\ xba \in S_i$ and $xab \equiv_i xba$;

where $A_i = \{a \in L \mid xa\ I_i\ xa\}$ and $TA_i = \{(xa, yb) \in I_i \mid a \in A_i \text{ or } b \in A_i\}$.

The inductive step extends a triple towards a transition system with independence by means of the rules *(D1)* and *(D2)*, whose intuitive meaning is clearly that of closing possibly incomplete diamonds. The process could create *autoindependent* transitions which must be eliminated. This is done by *(ℑ)* which drops them and adjusts $\equiv_i$ and I_i. We define the *limit* of the sequence as

$$\Big(S_\omega = \bigcup_{i\in\omega} \bigcap_{j\geq i} S_j, \quad \equiv_\omega = \bigcup_{i\in\omega} \bigcap_{j\geq i} \equiv_j, \quad I_\omega = \bigcup_{i\in\omega} \bigcap_{j\geq i} I_j \Big),$$

and consider $\mathit{TSys}_\omega = \Big(S_\omega/{\equiv_\omega}, [s^I]_{\equiv_\omega}, L_\omega, \mathit{Tran}_{\equiv_\omega}, I_{\equiv_\omega} \Big)$, where

- $([x]_{\equiv_\omega}, a, [x']_{\equiv_\omega}) \in \mathit{Tran}_{\equiv_\omega}$ if and only if $x' \equiv_\omega xa$;
- $([x]_{\equiv_\omega}, a, [x']_{\equiv_\omega})\ I_{\equiv_\omega}\ ([\bar{x}]_{\equiv_\omega}, b, [\bar{x}']_{\equiv_\omega})$ if and only if $xa\ I_\omega\ \bar{x}b$;
- $L_\omega = L \setminus \bigcup_{j\in\omega} A_j$.

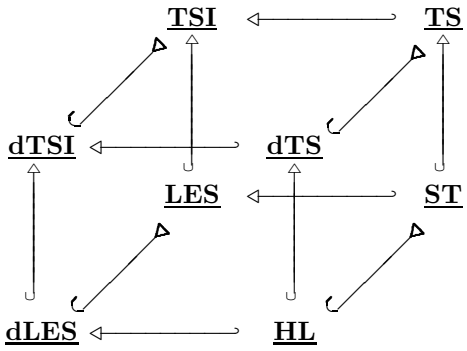
THEOREM 3.3
$TSys_\omega$ is a deterministic transition system with independence.

Thus, $TSys_\omega$, which can be seen as the colimit of an appropriate ω-diagram in the category of pre-transition system with independence, is the deterministic transition system with independence associates to the transition system with independence *TI* by the left adjoint to the inclusion **dTSI** $\hookrightarrow$ **TSI**.

Once the reflection is established, it is then easy to see that it restricts to a reflection **doTSI** $\hookleftarrow$ **oTSI**, and hence from Theorem 3.1, to a reflection **dLES** $\hookleftarrow$ **LES**.

So, our results may be summed up in the following "cube" of relationships among models.

THEOREM 3.4 (*The Cube*)



Conclusion

We have established a "cube" of formal relationships between well-known and (a few) new models for concurrency, obtaining a picture of how to translate between these models via adjunctions along the axes of "interleaving/noninterleaving", "linear/branching" and "behaviuor/system". Notice the pleasant conformity in the picture, with *coreflections* along the "interleaving/noninterleaving" and "behaviour/system" axes, and *reflections* along "linear/branching". All squares (surfaces) of the "cube" commute, with directions along those of the embeddings.

References

[1] M.A. Bednarczyk. *Categories of Asynchronous Transition Systems*. PhD Thesis, University of Sussex, 1988.

[2] M. Hennessy. *Algebraic Theory of Processes*. Cambridge, Massachusetts, 1988.

[3] C.A.R. Hoare. *Communicating Sequential Processes*. Englewood Cliffs, 1985.

[4] R.M. Keller. Formal Verification of Parallel Programs. *Communications of the ACM*, n. 19, vol. 7, pp. 371–384, 1976.

[5] A. Mazurkiewicz. Basic Notions of Trace Theory. In *lecture notes for the REX summerschool in temporal logic*, LNCS n. 354, pp. 285–363, Springer-Verlag, 1988.

[6] R. Milner. *Communication and Concurrency*. Englewood Cliffs, 1989.

[7] M. Nielsen, G. Plotkin, and G. Winskel. *Petri nets, Event Structures and Domains, part 1.* Theoretical Computer Science, n. 13, pp. 85–108, 1981.

[8] C.A. Petri. *Kommunikation mit Automaten*. PhD thesis, Institut für Instrumentelle Mathematik, Bonn, FRG, 1962.

[9] G. Plotkin. *A Structural Approach to Operational Semantics*. Technical report DAIMI FN-19, Computer Science Department, Aarhus University, 1981.

[10] V. Sassone, M. Nielsen, and G. Winskel. *A Classification of Models for Concurrency*. To appear as Technical Report DAIMI, Computer Science Department, Aarhus University, 1993.

[11] V. Sassone, M. Nielsen, and G. Winskel. *Deterministic Behavioural Models for Concurrency*. To appear as Technical Report DAIMI, Computer Science Department, Aarhus University, 1993. Extended abstract to appear in Proceedings of *MFCS '93*.

[12] M.W. Shields. Concurrent Machines. *Computer Journal*, n. 28, pp. 449–465, 1985.

[13] A. Stark. Concurrent Transition Systems. *Theoretical Computer Science*, n. 64, pp. 221–269, 1989.

[14] G. Winskel. Event Structure Semantics of CCS and Related Languages. In proceedings of *ICALP '82*, LNCS n. 140, pp. 561–567, Springer-Verlag, 1982. Expanded version available as technical report DAIMI PB-159, Computer Science Department, Aarhus University.

[15] G. Winskel. Synchronization Trees. *Theoretical Computer Science*, n. 34, pp. 33–82, 1985.

[16] G. Winskel. Event Structures. In *Advances in Petri nets*, LNCS n. 255, pp. 325–392, Springer-Verlag, 1987.

[17] G. Winskel, and M. Nielsen. Models for Concurrency. To appear in the *Handbook of Logic in Computer Science*. A draft appears as technical report DAIMI PB-429, Computer Science Department, Aarhus University, 1992.