Technical Report

UCAM-CL-TR-85
ISSN 1476-2986

Number 85

UNIVERSITY OF CAMBRIDGE
Computer Laboratory

Category theory and models for parallel computation

Glynn Winskel

April 1986

15 JJ Thomson Avenue
Cambridge CB3 0FD
United Kingdom
phone +44 1223 763500

http://www.cl.cam.ac.uk/

Technical reports published by the University of Cambridge
Computer Laboratory are freely available via the Internet:

http://www.cl.cam.ac.uk/techreports/

ISSN 1476-2986

Category Theory and Models for Parallel Computation

by
Glynn Winskel
Computer Laboratory,
University of Cambridge.

Introduction.

Here we will illustrate two uses of category theory:

• The use of (elementary) category theory to define semantics in a particular model. How semantic constructions can often be seen as categorical constructions, in particular how parallel compositions are derived from a categorical product and non–deterministic sum is a coproduct. How categorical notions can provide a basis for reasoning about computations. These will be illustrated for the model of Petri nets.

• The use of category theory to relate different semantics. How the relations between various concrete models, like Petri nets, event structures, trees, state machines, are expressed as adjunctions. This will be illustrated by showing the coreflection between safe Petri nets and trees.

No work here relies on any deep result in category theory (*e.g.* knowledge of the first half of [AM] is sufficient). Still, category theory has certainly provided guidelines for definitions. This is another use of category theory which could be advertised, though one which is often suppressed in the final version of a paper. For example, the simple definition of morphism on Petri nets presented here originates with the author and is a great improvement on the standard definition provided in [Br]. It was discovered (see [W3]) in a rather roundabout way in order to achieve a coreflection between a particular subcategory of Petri nets, the safe nets, and a category formed from another model called event structures—a goal which could not even have been formulated without the machinery of category theory.

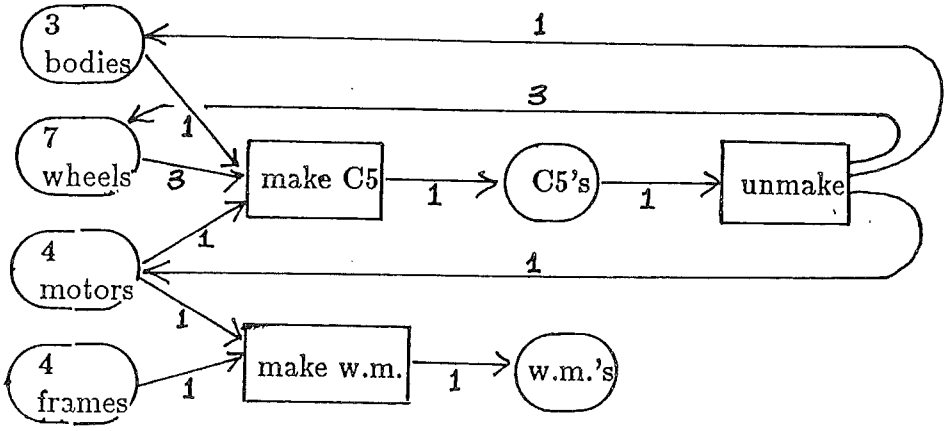
This presentation is essentially a write–up of a talk given at the workshop on Category Theory and Computer Science held at the University of Surrey in the summer 1985. More details can be found in the papers [W1–5] listed at the end.

1. Petri nets.

Petri nets are a very simple and intuitively appealing model of parallel computation. They can be viewed as a generalisation of finite state machines, generalised to express the concurrent structure present in systems. A Petri net has two kinds of elements, *conditions* representing types of resource and *events* representing atomic actions. A condition can hold to a certain multiplicity representing the amount of resource it stands for, and a state of a Petri net, generally called a *marking*, is represented by associating each condition with a nonnegative integer, to show the distribution of resources. The occurrences of events affect the marking because in a Petri net each event is stipulated to consume certain resources and produce others in certain amounts. Events can occur concurrently provided they are not in competition to consume the same resources. These notions will be tightened up in a moment, and note incidentally that although we have talked of events "consuming" and "producing" "resources" these terms should be understood abstractly and fit a wide range of situations.

Example. *The manufacture of C5's and washing machines.*

The idea is most easily seen through an example in which we also introduce the graphical representation of Petri nets. Conditions are drawn as circles, and the multiplicity to which they hold by integer inscriptions or, more often, by numbers of tokens positioned in the circle, and events as squares, and the amount of various resources consumed and produced by events are shown by arcs weighted by nonnegative integers. Later, sometimes we shall use the convention that an arc which carries no weight explicitly is understood to have arc weight 1. The example is more or less self-explanatory. It represents at a very crude (and useless!) level the relationship between two manufacturing processes, that of Hoover washing machines and that of small electric 3-wheeled cars called C5's, produced by Sinclair, which are powered by washing-machine motors. The event of making a C5 uses up 3 wheels, a single body and a motor while the event of making a washing machine requires a frame and a motor which explains the choice of arc weights. If there are sufficient resources (as drawn below there are 3 bodies, 7 wheels, 4 motors and 4 frames) then several C5's and several washing machines can be made concurrently (in this case, for example, 2 C5's and 2 washing machines or, instead, 1 C5 and 3 washing machines, and so on). As various machines are made some resources get consumed and others produced so the marking changes accordingly. Of course once a C5 is made it can be dismantled, the unmake event, and the motors recycled to take be used in the manufacture of washing machines; this explains one of the loops formed by a chain of arcs. Often Petri nets are explained by "playing the token game" on a net; tokens are placed on the conditions to represent an initial marking and then as events occur tokens are removed from certain conditions and placed onto others.



Not surprisingly the appropriate mathematics to formalise these ideas is that of *multisets* and *multirelations*. A *multiset* over X is a function $f : X \to \mathbf{N}$; it is thought of as a column vector with nonnegative entries $f(x)$, written f_x, in each column $x \in X$. We write μX for the set of multisets over X. The *null multiset* $\underline{0}$ is such that $\underline{0}_x = 0$ for all $x \in X$. For convenience, we shall identify x in X with the *singleton* multiset $\hat{x}$ given by

$$\hat{x}_y = \begin{cases} 1 & \text{if } y = x, \\ 0 & \text{otherwise.} \end{cases}$$

The operations and relations $+, -, \leq$ on multisets are defined pointwise, with multiset difference, $-$, only defined when the result is nonnegative in each component. The natural mappings to take between multisets are *multirelations*. Let X and Y be sets. A *multirelation* from X to Y is a matrix

$$\alpha : Y \times X \to \mathbf{N}.$$

Write $\alpha : X \to_\mu Y$ to mean α is a multirelation from X to Y. We write $\alpha_{y,x}$ for the entry $\alpha(y, x)$ of a multirelation. We write α^{op} for the opposite multirelation to a multirelation α with $(\alpha^{op})_{x,y} = \alpha_{y,x}$. In what follows we shall identify sets, functions and relations with the appropriate multisets

and multirelations in which all entries are at most 1. In defining application and composition of multirelations we must face a little technical discomfort. We shall take *composition* of multirelations to be matrix composition and *application* to a multiset to be matrix application. However because we do not wish to restrict ourselves to just finite Petri nets we must face the problem that these definitions can lead to infinite sums of nonnegative integers so matrix application and composition will not always be well-defined. To accommodate this possibility an extra element ∞, to stand for nonconvergence, is added to $\mathbf{N}$, and multiplication and indexed sums extended accordingly, as done in [AM1] for instance. In this broader framework composition and application of matrices, possibly with entries of ∞, always exist. We shall not go ahead and spell out this obvious extension for the sake of brevity—refer to [W5] for the details—and because, it so happens, all the multisets and multirelations that arise for Petri nets will never have ∞ in any of their components. This is because the extra structure present in Petri nets will define subspaces on which all the operations we consider never yield the value ∞. Still, this is best shown in the broader framework with ∞. We now formalise the definition and behaviour of Petri nets.

The definition of Petri nets

A *Petri net* is a 2-sorted algebra over multisets with *sorts*

μB, where B is a non-null set of *conditions*,

μE, where E is a set of *events*,

with *operations*

$M_0 \in \mu B$, a constant *non-null* multiset of conditions called the *initial marking*,

$^\bullet(\,) : E \to_\mu B$, a multirelation called the *precondition map*, such that $^\bullet e \neq \underline{0}$ for all $e \in E$,

$(\,)^\bullet : E \to_\mu B$, a multirelation called the *postcondition map*, such that $e^\bullet \neq \underline{0}$ for all $e \in E$,

which satisfy:

$$(M_0)_b \neq 0 \quad \text{or} \quad [\exists e \in E.\ (^\bullet e)_b \neq 0] \quad \text{or} \quad [\exists e \in E.\ (e^\bullet)_b \neq 0],$$

for all conditions b, *i.e.* no condition is *isolated*.

The behaviour of Petri nets

Let N be a Petri net.

A *marking* M is a multiset of conditions, *i.e.* $M \in \mu B$.

Let M, M' be markings. Let A be a *finite* multiset of events. Define

$$N : M \xrightarrow{A} M' \text{ iff } {}^\bullet A \leq M \ \ \& \ \ M' = M - {}^\bullet A + A^\bullet.$$

This gives the *transition relation* between markings.

A *reachable marking* of N is a marking M such that

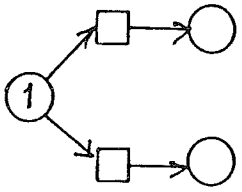
$$M_0 \xrightarrow{A_0} M_1 \xrightarrow{A_1} \cdots \xrightarrow{A_{n-1}} M_n = M$$

for some markings and finite multisets of events.

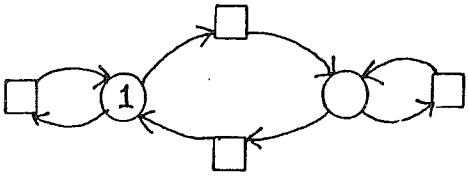
Remark. We insist that A should be a *finite* multiset of events in $M \xrightarrow{A} M'$ to avoid the situation where an event occurs only through the previous occurrence of an infinite set of events. The restriction is reasonable intuitively and has several technical advantages.

Examples

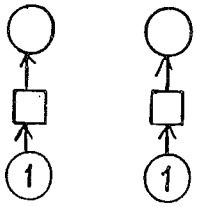
The purpose of these examples is to indicate the expressive power of Petri nets. They can represent computution trees where branching stands for nondeterministic choice as in for example:



They can represent any finite–state machine—take its states as conditions and represent its transitions as single events, *e.g.* as in:



But they also have the ability to express concurrent or parallel activity as in the simple net:



2. Morphisms on Petri nets.

As Petri nets are now viewed as algebras it is natural to take morphisms as some kind of homomorphism.

Let N_0 and N_1 be nets. A *homomorphism* from N_0 to N_1 is a pair of multirelations (η, β) with $\eta : E_0 \to_\mu E_1$ and $\beta : B_0 \to_\mu B_1$ such that

$$\beta M_0 = M_1 \text{ and } {}^\bullet(\eta A) = \beta({}^\bullet A) \text{ and } (\eta A)^\bullet = \beta(A^\bullet),$$

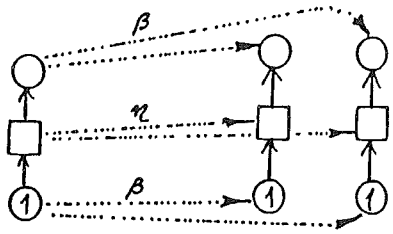
for all $A \in \mu E_0$.

A homomorphism is *finitary* when ηe is a finite multiset for all events e.

Finitary homomorphisms preserve the behaviour of nets in the following sense.

Theorem. *Let $(\eta, \beta) : N_0 \to N_1$ be a finitary homomorphism of Petri nets. Then β preserves the initial marking and*

$$N_0 : M \xrightarrow{A} M' \;\Rightarrow\; N_1 : \beta M \xrightarrow{\eta A} \beta M'.$$

Example. A finitary homomorphism: 

The morphisms on nets which arise in practice are homomorphisms which preserve the nature of events—the homomorphism above does not, in the sense that one event is sent to two. It may be possible to argue on the basis of our understanding of the notion of event in Petri nets that finitary homomorphisms are too general. Certainly in terms of the categorical constructions and relations with other models it pays to restrict to morphisms on nets defined as follows.

A *morphism* on Petri nets $N \to N'$ is a homomorphism

$$(\eta, \beta) : N \to N',$$

on the nets viewed as algebras, in which η is a partial function.
(We identify partial and total functions with their linear extensions to multirelations.)

Say a morphism (η, β) of nets is *synchronous* when η is a total function on events.

We have the corresponding categories: **Net** that of nets with morphisms, and $\mathbf{Net}_{syn}$ the subcategory with synchronous morphisms.

3. Categorical constructions.

We investigate the categorical constructions in the categories **Net** and $\mathbf{Net}_{syn}$.

Product

Let N_0 and N_1 be nets. Their *product* has *events*

$$E = \{(e_0, \underline{0}) \mid e_0 \in E_0\} \cup \{(\underline{0}, e_1) \mid e_1 \in E_1\} \cup \{(e_0, e_1) \mid e_0 \in E_0 \ \ \& \ \ e_1 \in E_1\},$$

with projections $\pi_i : E \to E_i$ where $\pi_i(e_0, e_1) = e_i$, for $i = 0, 1$, and *conditions*, the disjoint union,

$$B = B_0 \uplus B_1,$$

with projections $\rho_i : B \to B_i$, where ρ_i^{op} are the obvious injections $B_i \to B$, and *initial marking*

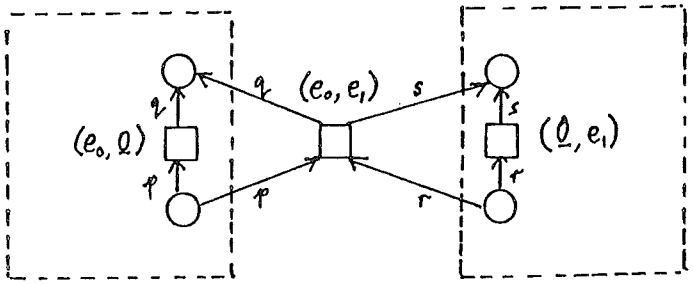
$$M = \rho_0^{op} M_0 + \rho_1^{op} M_1,$$

and *pre and post condition maps* given by

$$\begin{aligned} {}^\bullet e &= \rho_0^{op}[{}^\bullet(\pi_0 e)] + \rho_1^{op}[{}^\bullet(\pi_1 e)] \\ e^\bullet &= \rho_0^{op}[(\pi_0 e)^\bullet] + \rho_1^{op}[(\pi_1 e)^\bullet]. \end{aligned}$$

The product is associated with a simple construction on the graphical representation of nets. Disjoint copies of the two nets N_0 and N_1 are juxtaposed and extra events of the form (e_0, e_1) are adjoined, for e_0 an event of N_0 and e_1 an event of N_1; an extra event (e_0, e_1) has as preconditions those of its components and can be thought of as an event of synchronisation between two processes one modelled by N_0 and the other as N_1. Copies of the original events, those which are not synchronised with any companion event of the the other process, have the form $(e_0, \underline{0})$ in the copy of N_0 and the form $(\underline{0}, e_1)$ in the copy of N_1.

The product of N_0 and N_1:



Theorem.
The construction $N_0 \times N_1$, with morphisms (π_0, ρ_0) and (π_1, ρ_1), is a product in the category of Petri nets **Net**.

The behaviour of the product of two nets is that which is allowed when projected to the components. Precisely:

Theorem. *The behaviour of a product of nets $N_0 \times N_1$ is related to the behaviour of its components N_0 and N_1 by*

$$N_0 \times N_1 : M \xrightarrow{A} M' \quad \text{iff} \quad (N_0 : \rho_0 M \xrightarrow{\pi_0 A} \rho_0 M' \ \& \ N_1 : \rho_1 M \xrightarrow{\pi_1 A} \rho_1 M').$$

A marking M is reachable in $N_0 \times N_1$ iff $\rho_0 M$ is reachable in N_0 and $\rho_1 M$ is reachable in N_1.

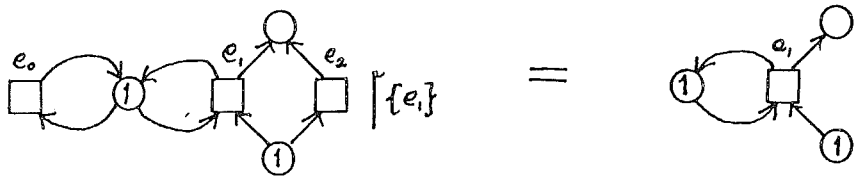
Such product constructions are important when modelling the kind of parallel compositions present in languages like CSP, CCS, SCCS and OCCAM which are based on the idea that processes communicate by events of synchronisation (see [H], [M1,2]). Imagine two processes, modelled as nets, set in parallel. Whether or not they communicate, to form events of synchronisation, depends on the what kinds of events they are prepared to do. The *product* of two nets allows arbitrary synchronisations. Forbidden synchronisations can be removed by another operation of restriction.

Restriction

Let N be a net and $E' \subseteq E$.

Define $N\lceil E'$ to be the net with events E', and conditions B' the remaining *nonisolated* conditions, and pre and post condition maps the restrictions of those of N.

Example.



The behaviour of a net restricted to a set of events is a restriction of the behaviour of the original net.

Proposition.
Let M and M' be markings of $N\lceil E'$. Then

$$N\lceil E' : M \xrightarrow{A} M' \text{ iff } N : M \xrightarrow{A} M' \ \& \ A \in \mu E'.$$

As described restriction is not yet a categorical notion—I am not yet sure how best to do this.

Parallel compositions

Parallel compositions are obtained by restricting the product of nets to a set of allowed synchronisations.

Theorem. *The behaviour of the parallel composition of nets*

$$N_0 \|_S N_1 =_{def} N_0 \times N_1 \lceil S$$

is related to the behaviour of N_0 *and* N_1 *by*

$$N_0 \|_S N_1 : M \xrightarrow{A} M' \text{ iff } A \in \mu S \ \& \ N_0 : \rho_0 M \xrightarrow{\pi_0 A} \rho_0 M' \ \& \ N_1 : \rho_1 M \xrightarrow{\pi_1 A} \rho_1 M',$$

for markings M, M' *of* $N_0 \|_S N_1$.

Synchronous product

As an important example of a parallel composition, we obtain the product of nets in the category $\mathbf{Net}_{syn}$. It is obtained by restricting the product of nets in **Net** to synchronisations the Cartesian product of events of the component nets.

Let N_0 and N_1 be nets with events E_0 and E_1. Define their *synchronous product*

$$N_0 \otimes N_1$$

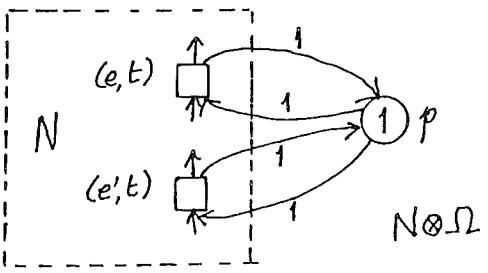
to be the restriction

$$N_0 \times N_1 \lceil (E_0 \times E_1).$$

Theorem. *The synchronous product* $N_0 \otimes N_1$, *with the restrictions of the projections is a product in* $\mathbf{Net}_{syn}$.

Example. A ticking clock is represented by the net Ω:

The synchronous product of a net with Ω serialises the event occurrences of a net:



Sums of nets

Coproducts do not exist in general in the categories **Net** and $\mathbf{Net}_{syn}$. However they do for the subcategory of safe nets.

A Petri net N is *safe* iff $({}^\bullet e)_b \leq 1$ and $(e^\bullet)_b \leq 1$, for all events e and conditions b, and $M_b \leq 1$ for all reachable markings M and conditions b.

For safe nets, for any reachable marking M, if $M \xrightarrow{A} M'$ then ${}^\bullet A$, A, $A^\bullet$, M and M' are all sets in the sense that their multiplicities never exceed 1. In a safe net a condition either holds, with multiplicity 1, or does not hold, with multiplicity 0, which can be thought of as it either being true or false, and similarly events either occur or do not occur.

Let N_0 and N_1 be safe nets. Their *sum* has *events*

$$E = E_0 \uplus E_1,$$

a disjoint union with injections

$$in_k : E_k \to E,$$

conditions,

$$B = \{(b_0, \underline{0}) \mid b_0 \in B_0 - M_0\} \cup \{(\underline{0}, b_1) \mid b_1 \in B_1 - M_1\} \cup (M_0 \times M_1),$$

and *initial marking*

$$M = M_0 \times M_1$$

with injection relations ι_0 and ι_1 where

$$b_0 \iota_0 b \Leftrightarrow \exists b_1 \in B_1 \cup \{\underline{0}\}.\ b = (b_0, b_1),$$
$$b_1 \iota_1 b \Leftrightarrow \exists b_0 \in B_0 \cup \{\underline{0}\}.\ b = (b_0, b_1),$$

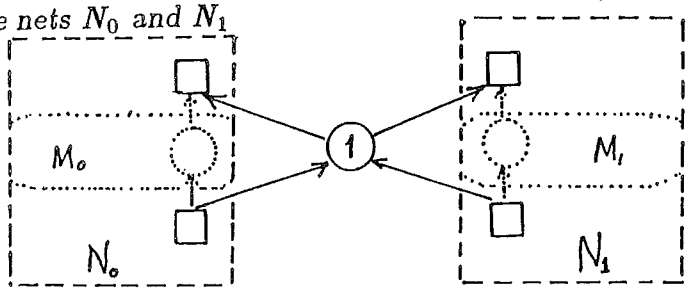
and *pre and post condition maps* given by

$${}^\bullet(in_k e) = \iota_k({}^\bullet e) \text{ and } (in_k e)^\bullet = \iota_k(e^\bullet)$$

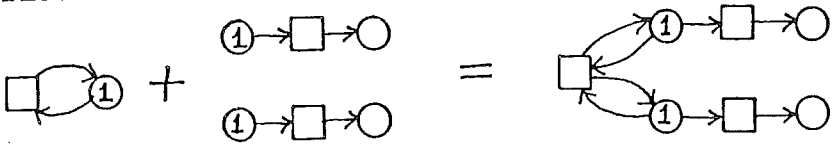
for $k = 0, 1$.

The sum can be described by a simple graphical construction.

The sum of safe nets N_0 and N_1



Example. The sum of two safe nets:



Theorem. *The sum $N_0 + N_1$ with injections (in_0, ι_0) and (in_1, ι_1) is a coproduct in the category of safe Petri nets with (synchronous) morphisms.*

The behaviour of the sum of two safe nets is related to that of its components by the injection morphisms in the following way.

Theorem. *Let $N_0 + N_1$ be the sum of safe nets with injections (in_0, ι_0) and (in_1, ι_1). Then X is a reachable marking of $N_0 + N_1$ and $X \xrightarrow{A} X'$ iff*

$\exists$ *reachable marking* X_0, A_0, X_0'.

$$N_0 : X_0 \xrightarrow{A_0} X_0' \ \& \ A = in_0 A_0 \ \& \ X = \iota_0 X_0 \ \& \ X' = \iota_0 X_0'$$

or

$\exists$ *reachable marking* X_1, A_1, X_1'.

$$N_1 : X_1 \xrightarrow{A_1} X_1' \ \& \ A = in_1 A_1 \ \& \ X = \iota_1 X_1 \ \& \ X' = \iota_1 X_1'.$$

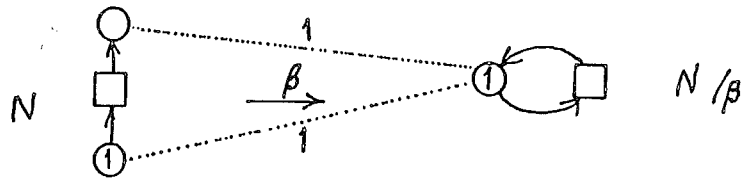
Quotients and loops.

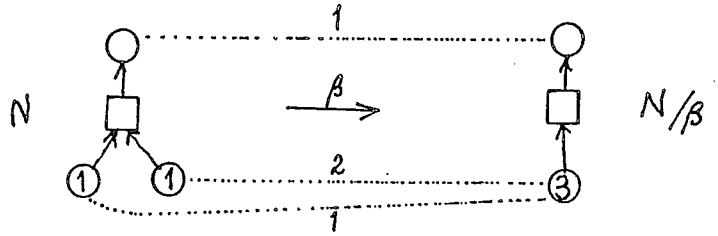
Here we describe an operation which can be used to introduce loops into a Petri net. Let N be a net, conditions B, events E. Let $\beta : B \to_\mu C$ such that βM_0, $\beta({}^\bullet e)$, $\beta(e^\bullet) \neq \underline{0}$ for all events e and initial marking M_0. Define the *quotient* N/β to be the unique net with conditions C, events E, such that

$$(1_E, \beta) : N \to N/\beta$$

is a morphism.

Examples.





Because of the properties of morphisms we see

$$N : M \xrightarrow{A} M' \Rightarrow N/\beta : \beta M \xrightarrow{A} \beta M'.$$

But the converse does not hold in general, and I do not know the full story of how the behaviour of a quotient N/β is related to the behaviour of N.

In this section we have presented a variety of constructions on Petri nets. Starting with a some basic atomic nets—or basic constructions like the "guarding" operation in [W4] which prefixes a net by an event—the constructions can be used to build-up more complicated nets. In order to build-up infinite nets in a sensible way we would need some way to construct nets recursively. This has not yet be done in a completely satisfactory manner. Certainly one can construct such nets by inductive definitions—but this is not categorical and depends rather crucially on the precise set-theoretic constructions used. For safe nets there seems to be a satisfactory method using functors which are continuous on ω-chains of certain kinds of morphisms, though even here I do not know simple and useful, local, sufficient conditions on functors which ensure they are continuous. It is not yet clear how to generalise this to arbitrary Petri nets.

4. Net invariants.

Of course we would like to have methods for reasoning about complicated nets in terms of the nets from which they are built-up. Here we indicate how this can be done, at least for a limited class of safety properties on finite nets. (The work of this section was done with Mogens Nielsen, University of Aarhus, and is still in a provisional state.) We first exhibit a contravariant functor from the categories of Petri nets to $\mathbf{Z}$-modules; it associates each net with a space of *invariants*, weighted sums of conditions which stay constant throughout the net behaviour. (See [R] and [Pe] for an introduction to invariants and some simple uses.)

Let N be a finite Petri net with conditions B and initial marking M_0. An *invariant* of N is a weighted sum of its conditions, *i.e.* a row-matrix $(I_b)_{b\in B}$ with entries $I_b \in \mathbf{Z}$, such that

$$I(M) = I(M_0)$$

for every reachable marking M.

Write $\mathrm{Inv}N$ for the set of invariants of N.

Proposition. *Let N be a net. Then $\mathrm{Inv}N$ forms a $\mathbf{Z}$-module under matrix addition and scalar multiplication.*

Theorem. *There is a contravariant functor from the category of finite Petri nets with finitary homomorphisms to the category of $\mathbf{Z}$-modules with linear maps; on objects it acts as*

$$N \mapsto \mathrm{Inv}N,$$

and takes $(\eta,\beta) : N_0 \to N_1$ to the linear map

$$\beta^* : \mathrm{Inv}N_1 \to \mathrm{Inv}N_0, \text{ where } \beta^*(I) = I\beta,$$

on $\mathbf{Z}$-modules.

Remark. With a little more effort, to deal with nonconvergent sums of integers correctly, this result also holds for infinite Petri nets.

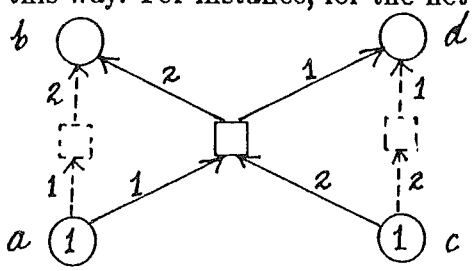
Morphisms on nets are an aid in producing a calculus for invariants. Consider:

$$\begin{array}{ccccc} & & N_0 \|_S N_1 & & \\ & & \downarrow & & \\ N_0 & \xleftarrow{(\pi_0,\rho_0)} & N_0 \times N_1 & \xrightarrow{(\pi_1,\rho_1)} & N_1 \end{array}$$

Because morphisms preserve invariants in a contravariant way it is easy to see that:

$$I_0 \in \mathrm{Inv}N_0 \;\&\; I_1 \in \mathrm{Inv}N_1 \Rightarrow I_0\rho_0 + I_1\rho_1 \in \mathrm{Inv}(N_0 \|_S N_1).$$

But not all invariants are got this way. For instance, for the net



we have $a^* + b^* + c^* + d^* \in \mathrm{Inv}N_0 \|_S N_1$ although $a^* + b^* \notin \mathrm{Inv}N_0$ & $c^* + d^* \notin \mathrm{Inv}N_1$.
(We use b^* for the row vector with 1 in row b and 0 elsewhere.)

We must use a more general notion. Let N be a finite net. Let ϕ be a weighted sum of conditions and $k \in \mathbf{Z}$. Let E be a subset of events which may possibly contain $\underline{0}$—the inclusion of $\underline{0}$ is important to get rules to reason about parallel compositions. Define

$$N \models [E]^k \phi \text{ iff } \forall e \in E.\ \phi(e^\bullet - {}^\bullet e) = k.$$

This relation is related to invariants as follows.

Proposition. *Let N be a finite net with events E. Let ϕ be a weighted sum of conditions. If every event of E can occur at some reachable marking then*

$$N \models [E]^0 \phi \text{ iff } \phi \in \mathrm{Inv} N.$$

We can see how the notion is respected by morphisms in the next lemma.

Lemma. *Let $(\eta, \beta) : N \to N'$ be a morphism between finite nets. Let ϕ be a weighted sum of conditions of N'. Then:*

(i) $N' \models [E']^k \phi \Rightarrow N \models [\eta^{-1} E']^k \phi\beta$, *for a subset E' of events of N'.*

(ii) $N \models [E]^k \phi\beta \Rightarrow N' \models [\eta E]^k \phi$ *for E a subset of events of N.*

Now we can present results which show how a relation holding for a parallel composition is equivalent to relations holding of its components, and similarly for the other constructions. The results reduce proving an assertion about a constructed net to proving assertions about its components.

Theorem. *Let N_i be nets for $i = 0, 1$. Let E_i be a subset of events of N_i and ϕ_i a weighted sum of its conditions, for $i = 0, 1$. Then, for $k \in \mathbf{Z}$,*

$$\exists k_0, k_1.\ k_0 + k_1 = k \ \ \&\ \ N_0 \models [E_0]^{k_0} \phi_0 \ \ \&\ \ N_1 \models [E_1]^{k_1} \phi_1$$

iff

$$N_0 \times N_1 \models [E_0 \times E_1]^k \phi_0 \rho_0 + \phi_1 \rho_1.$$

Proposition. *Let ϕ be a weighted sum of conditions of net N with a family of subsets of events $\{E_i \mid i \in I\}$. Then, for $k \in \mathbf{Z}$,*

$$(\forall i \in I.\ N \models [E_i]^k \phi) \text{ iff } N \models [\textstyle\bigcup_{i \in I} E_i]^k \phi.$$

Proposition. *For a net N with weighted sum ϕ, integer k and subsets of events E, E',*

$$N\lceil E' \models [E]^k \phi \text{ iff } N \models [E]^k \phi \ \ \&\ \ E \subseteq E'.$$

The result for sums is:

Theorem. *Let N_i be safe nets for $i = 0, 1$. Let E_i be a subset of events of N_i and Let ϕ be a weighted sum of conditions in $N_0 + N_1$. Then, for $k \in \mathbf{Z}$,*

$$N_0 \models [E_0]^k \phi\iota_0 \ \ \&\ \ N_1 \models [E_1]^k \phi\iota_1 \text{ iff } N_0 + N_1 \models [(in_0 E_0 \cup in_1 E_1)]^k \phi.$$

To deal with quotients (and so loops) we have the result:

Theorem. *Let N/β be a quotient of a net N. Let ϕ be a weighted sum of conditions of the quotient. Then for $k \in \mathbb{Z}$*

$$N/\beta \models [E]^k\phi \text{ iff } N \models [E]^k\phi\beta.$$

Of course invariants express the property that a situation holds in all reachable markings, to be thought of as the states a process can go into. Such properties are often called *safety properties* because in practice they often express that "something bad never happens". On the other hand a *liveness property* is one which expresses that "something good must eventually happen". To make these ideas precise one considers properties expressed by a modal logic, and these can be designed to be closely associated with a net. Roughly it appears that liveness properties are preserved in the direction of morphisms while safety properties are preserved in the opposite direction.

5. Another model: trees.

Trees in the form of synchronisation trees, in which arcs are labelled, are a model which underpin much of the work in the semantics of parallel computation (see *e.g.* [M1], [B], [W2]). The nodes are thought of as states and the arcs as events with branching representing nondeterminism.

For precision, a *tree* is a subset $T \subseteq A^*$ of finite sequences of some set A, called *events*, such that

$$\langle\rangle \in T \quad \text{and,}$$
$$\langle a_0, a_1, \ldots a_n, \ldots\rangle \in T \Rightarrow \langle a_0, a_1, \ldots a_n\rangle \in T.$$

(So a tree is a non–null subset of sequences closed under initial subsequences.)

Define

$$t \rightarrow_T t' \Leftrightarrow_{def} \exists a.\ t' = t\langle a\rangle.$$

(We use st to stand for the concatenation of sequences s and t.)

A *morphism* of trees from S to T is a map $f : S \rightarrow T$ such that

$$f(\langle\rangle) = \langle\rangle \quad \text{and,}$$
$$s \rightarrow_S s' \Rightarrow f(s) = f(s') \text{ or } f(s) \rightarrow_T f(s').$$

So intuitively a morphism between trees is a map on states which preserves the initial state and respects the nature of events in the same way as morphisms on nets. Now we have a category of trees **Tr** . Much more is said about the category in [W2], and about synchronisation trees in general in [M1].

We describe some categorical constructions on trees

Coproduct of trees

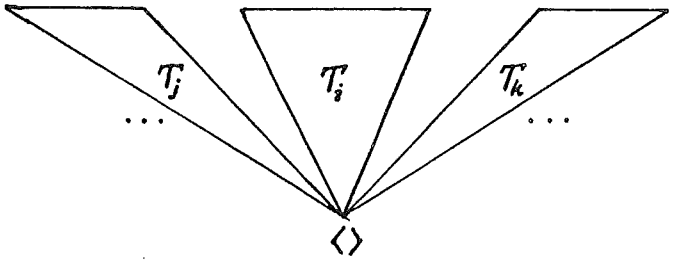
Let $\{T_i \mid i \in I\}$ be an indexed set of trees. Their *coproduct*

$$\sum_{i\in I} T_i = \bigcup_{i\in I}\{\langle(i, a_0), \ldots, (i, a_{n-1})\rangle \mid \langle a_0, \ldots, a_{n-1}\rangle \in T_i\}.$$

Define the obvious injections $in_i : T_i \rightarrow \sum_{i\in I} T_i$ by

$$in_i(\langle a_0, \ldots, a_{n-1}\rangle) = \langle(i, a_0), \ldots, (i, a_{n-1})\rangle$$

for $i \in I$. The coproduct corresponds to gluing the trees together at their roots:

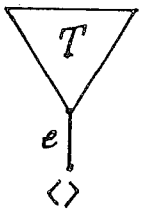


Product of trees

First we define a prefixing operation on trees. Let T be a tree and e an element. Define

$$eT = \{\langle e\rangle t \mid t \in T\},$$

which prefixes an event e onto the tree T. In a picture:

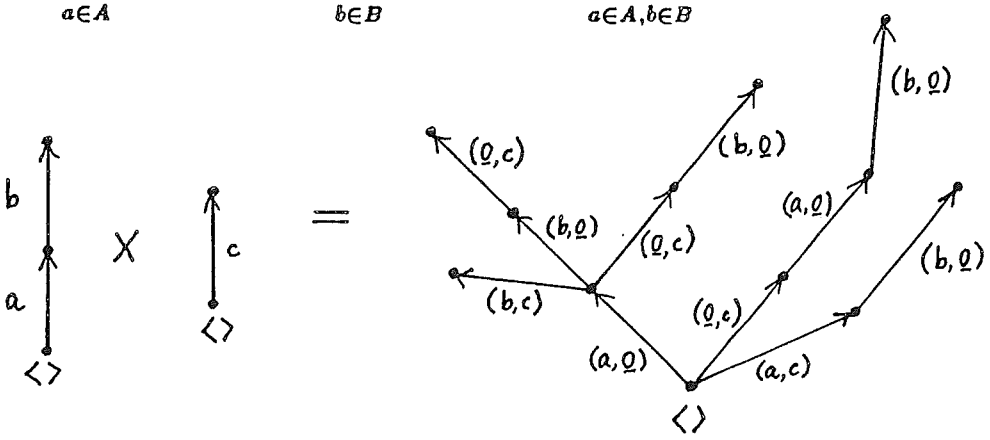


We now characterise the product (object) of two trees (*cf.* the expansion theorem in [M1]).

Theorem.
Suppose $S \cong \sum_{a\in A} aS_a$ *and* $T \cong \sum_{b\in B} bT_b$. *Then*

$$S \times T \cong \sum_{a\in A}(a,\underline{0})S_a \times T \;+\; \sum_{b\in B}(\underline{0},b)S \times T_b \;+\; \sum_{a\in A, b\in B}(a,b)S_a \times T_b.$$

Example.



6. A coreflection between safe Petri nets and trees.

Given that a net N describes a computation, what tree best describes that computation? We take the associated tree to be $\mathcal{T}N$ where $\mathcal{T}N$ consists of sequences of events $\langle e_0, e_1, \ldots, e_{n-1}\rangle$ such that

$$M_0 \xrightarrow{e_0} M_1 \xrightarrow{e_1} \cdots \xrightarrow{e_{n-1}} M_n.$$

We extend $\mathcal{T}$ to morphisms $(\eta,\beta) : N \to N'$ on nets by defining

$$[\mathcal{T}(\eta,\beta)](t\langle e\rangle) = \begin{cases} \eta(t)\langle\eta(e)\rangle & \text{if } \eta(e) \text{ defined,} \\ \eta(t) & \text{otherwise.} \end{cases}$$

This yields a functor $\mathcal{T}$: **Net** $\to$ **Tr** from nets to trees which clearly cuts down to a functor $\mathcal{T}$: **Net** $\to$ **Tr** from safe Petri nets to trees.

Conversely, trees can be viewed as special kinds of safe nets. A tree T determines a safe net $\mathcal{N}T$ in which:

Events $E = \{(t,t') \mid t \to_T t'\}$, the arcs of T, with *dependency* relation $<$ and *conflict* relation $\#$ given by

$$(t_0,t_0') < (t_1,t_1') \Leftrightarrow t_0' \to_T^* t_1$$
$$(t_0,t_0') \# (t_1,t_1') \Leftrightarrow (t_0,t_0') \not\leq (t_1,t_1') \ \& \ (t_1,t_1') \not\leq (t_0,t_0').$$

Conditions B have the form

$(\emptyset, C)$ where C is a subset of events in pairwise conflict *i.e.* $e \# e'$ or $e = e'$ for all e, e' in C.

$(\{e\}, C)$ where e is an event such that $e < c$ for all c in C, and C is a set of events in pairwise conflict.

Pre and post condition maps are

$$e^\bullet = \{(\{e\}, C) \mid (\{e\}, C) \in B\}$$
$${}^\bullet e = \{(A, C) \in B \mid e \in C\}.$$

Theorem. *Let T be a tree. Then $\mathcal{N}T$ and the morphism*

$$\theta : T \cong \mathcal{T}\mathcal{N}(T)$$

is free over T w.r.t. $\mathcal{T} : \mathbf{Net}_{safe} \to \mathbf{Tr}$, where

$$\theta\langle a_0, a_1, \ldots, a_{n-1}\rangle = \langle (s_0,s_1),(s_1,s_2),\cdots,(s_{n-1},s_n)\rangle$$

in which $s_i = \langle a_0, \ldots, a_{i-1}\rangle$.

Hence there is a coreflection between safe nets and trees,

$$\mathbf{Net}_{safe} \underset{\mathcal{N}}{\overset{\mathcal{T}}{\rightleftarrows}} \mathbf{Tr}$$

with right adjoint $\mathcal{T}$ and left adjoint $\mathcal{N}$. It can be shown that there is not a coreflection between the category **Net** of *all* nets and $\mathbf{Net}_{safe}$, nor one between **Net** and **Tr**. I do not know of a weaker notion which expresses suitably the relationship between these pairs of categories.

Right adjoints preserve limits, left adjoints colimits (see *e.g.* [Mac]). Hence *e.g.*

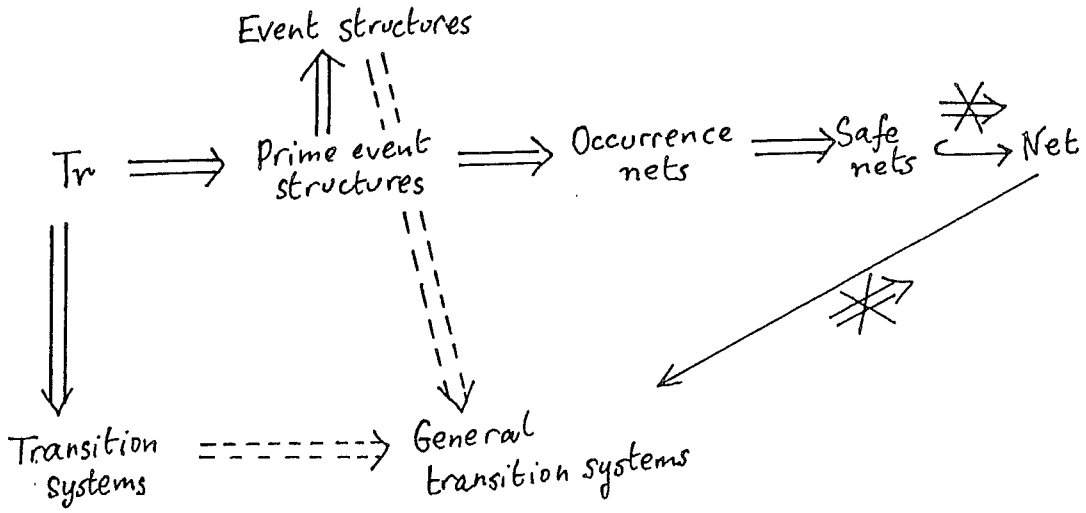
$$\mathcal{T}(N_0 \times N_1) \cong \mathcal{T}N_0 \times \mathcal{T}N_1$$
$$T_0 \times T_1 \cong \mathcal{T}(\mathcal{N}T_0 \times \mathcal{N}T_1)$$
$$T_0 + T_1 \cong \mathcal{T}(\mathcal{N}T_0 + \mathcal{N}T_1),$$

where showing the latter two isomorphisms depends on the natural isomorphism $T \cong \mathcal{T}\mathcal{N}T$ provided by the coreflection. Such facts are clearly useful to relate denotational semantics of a language where the denotations are nets (possibly with some extra structure like labels on the events) to denotational semantics with trees (possibly with extra structure like synchronisation trees).

7. Categories of models.

There is a criss-cross of functors, often parts of coreflections, bridging different categories of models, as illustrated in the diagram below. Coreflections are represented by double arrows in the direction of the left adjoint, and single functors by single arrow. Where they are suspected but not

worked out is shown by dotted versions of these arrows. Where they have been shown to be absent is indicated by a crossed–out arrow.



A word on the different categories: event structures are a model of processes in which sets of events carry causal dependency and conflict relations, prime event structures are those in which the causal dependency relation is a partial order, transition systems are understood to have transitions corresponding to single events, while in general transition systems they are associated with multisets. There is, for example, a functor from nets to general transition systems expressing the fact that morphisms on nets preserve dynamic behaviour. This diagram improves that in [W4]; the improvements are due to Marek Bednarczyk.

Generally when modelling systems one works not just with nets, or event structures or trees, for example, but with such structures together with some extra structure in the form of a labelling of events to indicate what kind of events they are and so how they interact with the environment. As part of [W1,2] I attempted to incorporate this labelling structure into the categorical set–up. It is not clear that my approach was the right one and more recently several people (Labella and Peterossi, Bednarczyk, Fourman) have proposed other solutions.

Acknowledgements.

I am grateful for discussions with Mogens Nielsen. I would like to thank D. Pitt, A. Poigné and D. Rydeheard for all the work they have put into organising the Surrey workshop on Category Theory and Computer Science and nudging me to write this up.

References

[AM] Arbib, M.A.,and Manes,E.G., Arrows, Structures and Functors, The categorical imperative. Academic Press (1975).

[AM1] Arbib, M.A.,and Manes,E.G., Formal semantics of programming languages. Final version forthcoming, preliminary version (1982).

[B] Brookes, S.D., On the relationship of CCS and CSP. ICALP 1983.

[Br] Brauer, W.(Ed.), Net Theory and Applications, Springer–Verlag Lecture Notes in Comp. Sci., vol.84 (1980).

[H] Hoare, C.A.R., Communicating sequential processes. Comm. ACM 21 (1978).

[HBR] Hoare, C.A.R., Brookes, S.D., and Roscoe, A.W., A Theory of Communicating Processes, Technical Report PRG-16, Programming Research Group, University of Oxford (1981); in JACM (1984).

[Mac] Maclane, S., Categories for the Working Mathematician. Graduate Texts in Mathematics, Springer (1971).

[M1] Milner, R., A Calculus of Communicating Systems. Springer Lecture Notes in Comp. Sc. vol. 92 (1980).

[M2] Milner, R., Calculi for synchrony and asynchrony. Theoretical Computer Science, pp.267–310 (1983).

[Pe] Peterson, J. L., Petri Net Theory and the Modelling of Systems. Prentice–Hall (1981).

[R] Reisig, W., Petri nets. Springer Lecture Notes in Comp. Sc. (1984).

[W1] Winskel, G., Event structure semantics of CCS and related languages, Springer–Verlag Lecture Notes in Comp. Sc. 140 and, expanded, as a report of the Computer Sc. Dept., University of Aarhus, Denmark (1982).

[W2] Winskel, G., Synchronisation trees. In Theoretical Computer Science, May 1985.

[W3] Winskel, G., A New Definition of Morphism on Petri Nets. Springer Lecture Notes in Comp Sc, vol. 166 (1984).

[W4] Winskel, G., Categories of Models for Concurrency. In the proceedings of the workshop on the semantics of concurrency, Carnegie–Mellon University, Pittsburgh, Springer Lecture Notes in Computer Science 197 (July 1984), and appears as a report of the Computer Laboratory, University of Cambridge (1984).

[W5] Winskel, G., Petri nets, algebras, morphisms and compositionality. Report 79 of the Computer Laboratory , University of Cambridge, and to appear in Information and Control (1985).