PB - 180

Nielsen & Thiagarajan: Non-determinism and Concurrency

TRYK: DAIMI/RECAU

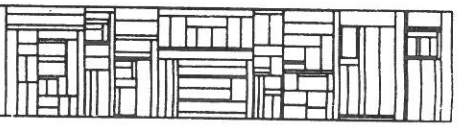
ISSN 0105-8517

Degrees of Non-determinism and Concurrency: A Petri Net View

M. Nielsen
P.S. Thiagarajan

DAIMI PB - 180
October 1984

Computer Science Department
AARHUS UNIVERSITY
Ny Munkegade - DK 8000 Aarhus C - DENMARK
Telephone: 06 - 12 83 55



DEGREES OF NON-DETERMINISM AND CONCURRENCY: A PETRI NET VIEW

by

M. Nielsen and P.S. Thiagarajan
Computer Science Department
Aarhus University
DK-8000 Aarhus C
Denmark

0. INTRODUCTION

The aim of this paper is to present an introduction to the theory of Petri nets. The subject matter of this theory is distributed systems and processes. In our presentation, we shall emphasise concepts at the expense of specific results and techniques. Applications of the theory, though many and varied, will not be dealt with here. Even in dealing with the concepts, we shall focus on those that we believe are relevant to the study of distributed systems in general (independent of the specific framework one might choose). In the concluding part, we will attempt a broader sketch of the scope and contents of net theory.

A main feature of this theory is that, in the study of systems, both states and changes-of-states are assigned equal importance. Moreover, both states and changes-of-states are viewed as distributed entities. A marked net (we prefer to term Petri nets as marked nets) looked upon a system model reflects these concerns. A net may be considered to be a directed bipartite graph with two kinds of nodes called S-elements and T-elements. S-elements denote the local atomic states and T-elements denote the local atomic changes-of-states (transitions). The directed arcs capture the neighbourhood relationship between S-elements and T-elements. Markings are used to represent the states of a system whose structure is modelled by a net.

A marking of a net is a distribution of objects called tokens over the S-elements. In this sense a global state is composed out of local states. In general, tokens can have a complex internal structure. This fact leads to a variety of powerful system models [13, 22, 45] that are at the forefront of applications. In this paper, given our purposes, it is sufficient to just consider marked nets in which the tokens do not have any internal structure and hence are indistinguishable from

each other.

The dynamics of a marked net are captured by a firing rule. It states when and how the transitions associated with the T-elements can transform the token distribution. In general, a number of (local) transitions may proceed independent of each other at a state. In this sense, change-of-state is also a distributed entity. The chief advantage of marked nets is that they provide the means for clearly distinguishing between the three fundamental relationships that can exist between the occurrences of two transitions t_1 and t_2 at a state:

1) t_1 followed by t_2 (sequence, causal dependence)

2) t_1 or t_2 but not both (choice, conflict, non-determinism)

3) t_1 and t_2 but with no order (concurrency, non-sequentiality, causal independence)

This ability of net theory to cleanly separate - in particular - choice and concurrency has at least one important consequence. It is possible to define various mixtures of non-determinism and non-sequentiality and study the resulting sub-classes. It is this aspect of net theory which we wish to bring out in our survey.

In the next section, the basic terminology concerning marked nets is introduced. One can then identify a class of marked nets called safe marked nets. Our review of net theory is essentially confined to this sub-class. In Section 2, we discuss with the help of simple net diagrams the fundamental situations that can arise in the history of a distributed system. Sections 3 and 4 are the heart of the paper. We identify, by syntactic means, a hierarchy of safe marked nets. We briefly indicate the theories of well-known members of this hierarchy. Our aim is to argue that this hierarchy represents one way of obtaining systems that exhibit increasingly complex mixtures of choice and concurrency in their behaviour. To establish this, in Section 4, we first review the various ways of defining the behaviour of a marked net. We then adapt Milner's notion of behavioural equivalence [32] for our purposes. Finally we show that the syntactically defined hierarchy of the previous section indeed agrees with our chosen notion of behavioural equivalence. As mentioned earlier, in the concluding part we indicate the various portions of net theory that have not been dealt with in this paper.

1. TERMINOLOGY

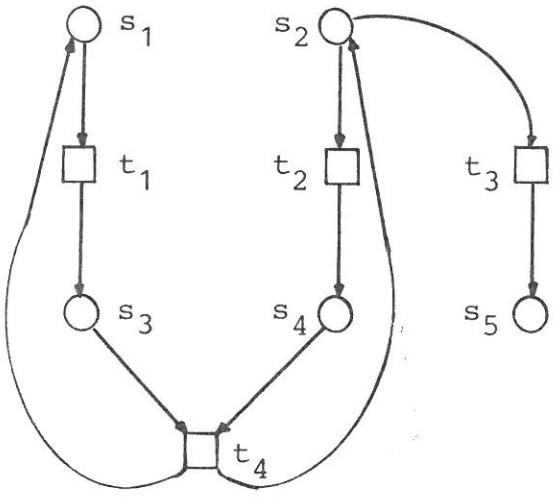
A (directed) net is a triple $N = (S,T;F)$ where:

1) $S \cup T \neq \emptyset$; $S \cap T = \emptyset$.

2) $F \subseteq (S \times T) \cup (T \times S)$ such that $dom(F) \cup range(F) = S \cup T$.

S is the set of S-elements, T is the set of T-elements and F is the flow relation. Depending on the application, various interpretations can be attached to these three components of a net (see [41]). Here, we shall use S-elements to denote the local atomic states, the T-elements to denote the local transitions and the flow relation to denote the extent of changes caused by the local transitions. In what follows we will refer to S-elements as places and T-elements as transitions or as done in the next two sections, refer to S-elements as conditions and to T-elements as events.

In diagrams S-elements will be drawn as circles, T-elements as boxes and the flow relation will be represented by directed arcs. The following is an example of a representation of a net.



For the net N we use S_N (T_N, F_N) to denote its set of S-elements (T-elements, flow relation); $X_N = S_N \cup T_N$ is the set of elements of N. The subscript N will be dropped if N is clear from the context.

It will be very convenient to work with a 'local' representation of the flow relation. To this end let N be a net and $x \in X_N$. Then

$$^\bullet x = \{y \in X \mid (y,x) \in F_N\} \text{ - the pre-set of } x$$
$$x^\bullet = \{y \in X \mid (x,y) \in F_N\} \text{ - the post-set of } x.$$

In our example $^\bullet s_1 = \{t_4\}$, $t_4^\bullet = \{s_1, s_2\}$.

This dot notation is extended to sub-sets of X_N in the obvious way. Now it is possible to identify various sub-classes of nets by suitably (and locally) restricting the dot relation. For example the net N is said to be pure iff $\forall x \in X_N: {}^\bullet x \cap x^\bullet = \emptyset$. N is said to be simple iff $\forall x,y \in X_N: {}^\bullet x = {}^\bullet y \wedge y^\bullet = x^\bullet \Rightarrow x = y$. Our example above is both pure and simple.

In the two sections to follow we will encounter more interesting sub-classes of nets.

A marking of the net $N = (S,T;F)$ is a function $M: S \to \mathbb{N}_0 = \{0,1,2,\ldots\}$. In diagrams, M is represented by placing M(s) tokens (small dark dots) on each s. The transition t is enabled at the marking M iff for each $s \in {}^\bullet t$, $M(s) > 0$; in other words each input place of t should carry at least one token at M. The fact that t is enabled at M will be denoted by $M[t>$. An enabled transition may fire (occur). When t fires at M, a new marking M' is reached which is given by:

$$\forall s \in S: M'(s) = \begin{cases} M(s) - 1, & \text{if } s \in {}^\bullet t \setminus t^\bullet \\ M(s) + 1, & \text{if } s \in t^\bullet \setminus {}^\bullet t \\ M(s), & \text{otherwise.} \end{cases}$$

The transformation of M into M' by the firing of t at M is denoted as $M[t>M'$. Consider our example above with marking M_1, given by

$$M_1(s_1) = M_1(s_2) = 1$$
$$M_1(s_3) = M_1(s_4) = M_1(s_5) = 0$$

We then have $M_1[t_1>M_2$ where

$$M_2(s_2) = M_2(S_3) = 1$$
$$M_2(s_1) = M_2(s_4) = M_2(s_5) = 0$$

The set of markings one can reach in this way, starting from M is called the forward marking class of M. More precisely, for the net N and a marking M of N, the forward marking class of M is denoted as $[M>$ and is the smallest set of markings of N satisfying:

1) $M \in [M>$
2) If $M' \in [M>$, $t \in T$ and M" is a marking of N such that $M'[t>M"$, then $M" \in [M>$.

Our system model is a marked net. Formally, a marked net is a quadruple $\Sigma = (S,T;F,M^0)$ where the net $N_\Sigma = (S,T;F)$ is called the underlying net of Σ and M^0 is a marking of N_Σ called the initial marking of Σ. Liveness and safety are two behavioural properties of marked nets which have traditionally received a great deal of attention in net theory. It is possible to define and study various forms of these two properties. Here we choose the 'strongest' versions.

The marked net $\Sigma = (S,T;F,M^0)$ is live iff $\forall M \in [M^0\rangle$, $\forall t \in T$: $\exists M' \in [M\rangle$ such that t is enabled at M'.

Thus in a live net no transition ever loses the possibility of becoming enabled.

The marked net $\Sigma = (S,T;F,M^0)$ is safe iff $\forall M \in [M^0\rangle$, $\forall s \in S$: $M(s) \leq 1$. Consider our example above with initial marking M_1 (exactly one token on s_1 and s_2). This marked net is then safe but not live.

In a safe net no place will ever contain more than one token. As a result, in a safe net each place can be viewed as a propositional variable. A marking is an atomic boolean valuation (1 ~ true, 0 ~ false) which then extends uniquely to a boolean valuation of the formulas built up from the propositional variables in the natural way. This view coupled with the presence of events gives rise to some useful and pleasant links with propositional logic (see [11]) but we digress.

What is of interest here is that the basic concepts of net theory are best brought out at the level of safe marked nets. Consequently, in what follows we shall just concentrate on safe marked nets. To conclude this section, we shall adopt a few conventions. In line with tradition, we will from now on denote a safe marked net as $\Sigma = (B,E;F,M^0)$ and call B the conditions and E the events. The elements of $[M^0\rangle$ are sometimes referred to as cases. Since for each $M \in [M^0\rangle$ $M: B \to \{0,1\}$ we can and shall represent M by the set of conditions that hold at M, i.e. $\{b \in B \mid M(b) = 1\}$. Accordingly, we say that the event e is enabled at M iff ${}^\bullet e \subseteq M$ and so on. We conclude this section with an example of a live and safe marked net (shown in fig. 1).

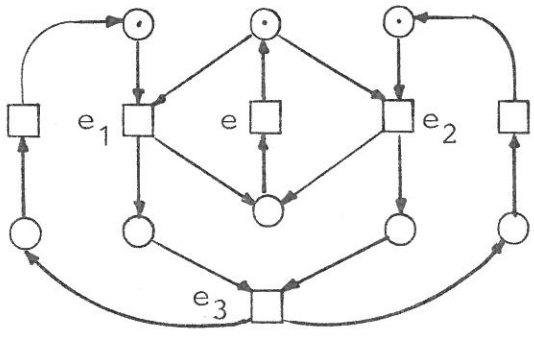
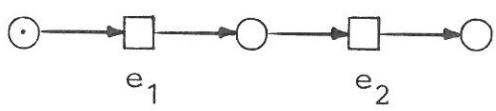


Fig. 1

2. FUNDAMENTAL SITUATION

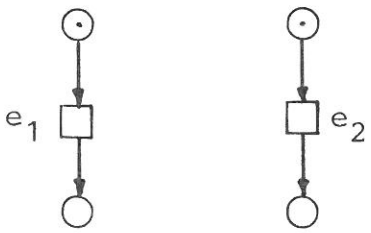
Causality, concurrency, conflict and confusion are four basic notions of net theory. They can be brought out with the help of safe marked nets as follows.

Causality



At the marking shown the occurrence e_2 must be preceded by the occurrence of e_1.

Concurrency



e_1 and e_2 can both occur at the marking. More importantly they can occur without 'interfering' with each other. No order is specified over their occurrences. Thus in general the occurrences of events and the resulting holdings of conditions will be partially ordered; our systems can exhibit non-sequential behaviour. One way to bring in the flavour of concurrency in the firing rule is to introduce the notion of a step.

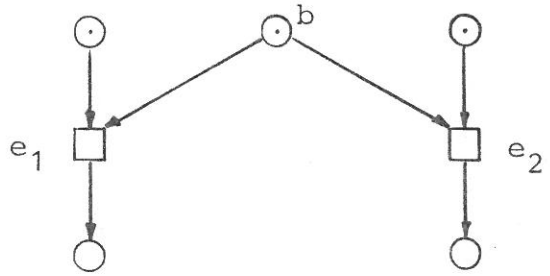
Let $\Sigma = (B,E;F,M^0)$ be a safe net $M \in [M^0\rangle$ and $\emptyset \neq u \subseteq E$. Then u is a step at M (denoted by $M[u\rangle$) iff

1) $\forall e \in u: {}^\bullet e \subseteq M$ (or equivalently $M[e\rangle$)

2) $\forall e_1,e_2 \in u: e_1 \neq e_2 \Rightarrow {}^\bullet e_1 \cap {}^\bullet e_2 = \emptyset$. ($e_1$ and e_2 can carry out the changes-of-states attributed to them without interfering with each other).

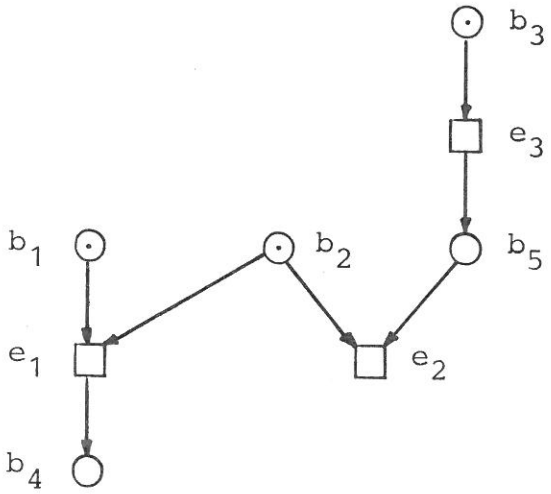
We say the events in u occur concurrently at M. As might be expected, $M[u\rangle M'$ iff $M[u\rangle$ and $M' = (M \setminus \bigcup_{e \in u} {}^\bullet e) \bigcup_{e \in u} e^\bullet$.

Conflict



At the marking shown e_1 and e_2 can occur individually. But $\{e_1,e_2\}$ is not a step due to the shared condition b. We say e_1 and e_2 are in conflict at this marking. Non-determinism enters the picture at this stage because the choice as to whether e_1 will occur or e_2 will occur is left unspecified. One way to explain how conflict is resolved is to postulate that the environment will supply the system with the required bit of information. Conflicts and their resolutions may be thought of as the means for modelling the flow of information across the border between the system and its environment.

Confusion



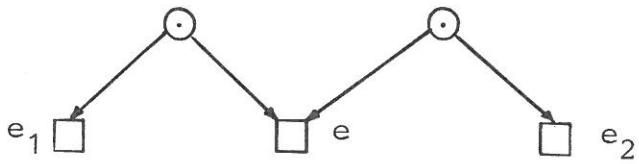
Let $M_0 = \{b_1,b_2,b_3\}$, $M_1 = \{b_4,b_5\}$ so that $M_0[\{e_1,e_3\}>M_1$. Here there could be disagreement over whether or not a conflict was resolved in going from M_0 to M_1. Two honest sequential observers O_1 and O_2 could report:

O_1 e_1 occurred first without being in conflict with any other event. And then e_3 occurred.

O_2 e_3 occurred. e_1 and e_2 got into conflict. The conflict was resolved in favour of e_1.

This is a confused situation. Confusion arises whenever conflict and concurrency 'overlap'. This phenomenon appears to be basic in nature and can be at best swept under the carpet (i.e. to a lower level of description) through temporal assumptions. In asynchronous switching circuits confusion is called the glitch problem or more appropriately the synchronisation failure problem [47].

There is a second form of confusion known as symmetric confusion. Here is an example.

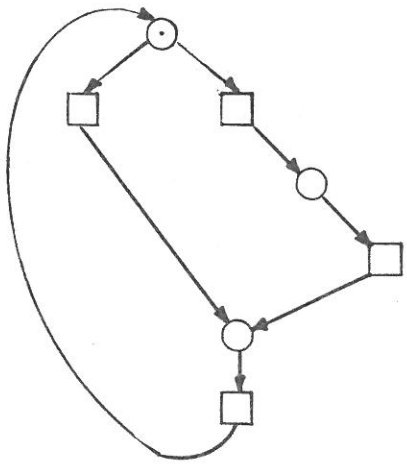


Here e_1 is in conflict with e. $\{e_1,e_2\}$ is a step. If e_2 occurs then e_1 is no longer in conflict with e. In other words e_1 gets out of conflict because of the occurrence of e_2. The whole argument of course applies to e_2 w.r.t. e_1. Hence the term symmetric confusion. Note that if the step $\{e_1,e_2\}$ occurs then there is confusion over which conflict (between e_1 and e or e_2 and e) was resolved.

3. A HIERARCHY OF SAFE NETS

We now wish to combine choice and concurrency by syntactic means and examine some of the resulting sub-classes. To this end it will be convenient to assume that our nets are finite (i.e. the set of elements is finite) and connected (in the graph theoretic sense).

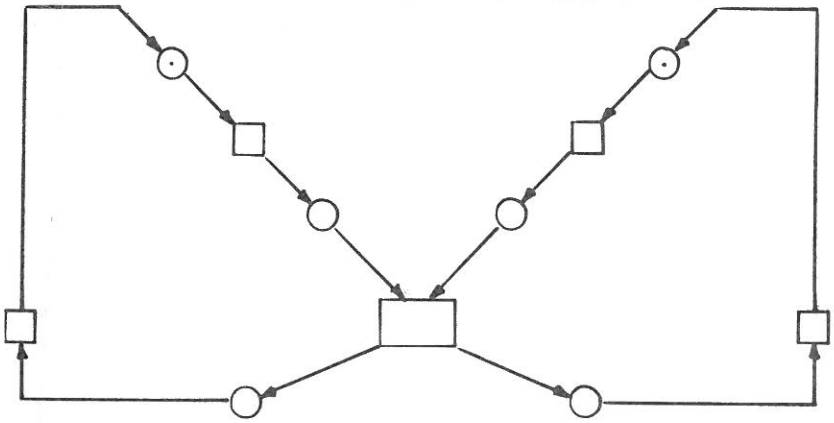
S-graphs can be used to capture the structure of non-deterministic sequential systems. An S-graph is a net $N = (S,T;F)$ in which $\forall t \in T: |{}^\bullet t|, |t^\bullet| \leq 1$. A marked S-graph is a marked net whose underlying net is an S-graph. It is easy to verify that the marked S-graph $\Sigma = (S,T;F,M^0)$ is live and safe iff N_Σ is strongly-connected and $\sum_{s\in S} M^0(s) = 1$. Here is an example of an ls (live and safe) S-graph. Since liveness and safety make sense only in the presence of markings, we will from now on drop the term "marked" whenever possible.



A safe S-graph can get into conflict situations; it can exhibit non-deterministic behaviour. However no two events can ever occur concurrently. In this sense safe S-graphs model non-deterministic sequential systems. Automata theory has a good deal to say about these

systems. Viewed as a sub-class of distributed systems, a more appropriate theory of this class is the one constructed by Milner [32]. Note that due to the absence of concurrency, safe S-graphs are free of confusion.

It is a happy circumstance in net theory that, structurally speaking, there is a duality relation between non-deterministic sequential systems and deterministic non-sequential systems. A T-graph is a net $N = (S,T;F)$ in which $\forall s \in S: |{}^\bullet s|, |s^\bullet| \leq 1$. A marked T-garph is a marked net whose underlying net is a T-graph. Marked T-graphs are often called marked graphs and sometimes synchronisation graphs. Below we show an example of an ls T-graph.



The theory of marked T-graphs is well-understood [6, 10 , 23 ,24]. Here we will just mention a characterisation of ls T-graphs. Details of the proof and other results can be found in [6 , 10].

The marked T-graph $\Sigma = (S,T;F,M^0)$ is live iff $\forall s \in S: {}^\bullet s \neq \emptyset$ and every directed circuit of N_Σ contains at least one S-element which is marked (i.e. carries at least one token) under M^0.

The live T-graph $\Sigma = (S,T;F,M^0)$ is safe iff every S-element of N_Σ is contained in a directed elementary circuit Π that carries exactly one token under M^0. In other words if S' is the set of S-elements that Π passes through then $\sum_{s \in S'} M^0(s) = 1$.

In a safe T-graph two events may occur concurrently; the behaviour can be non-sequential. But no two events can ever be in conflict. Thus safe T-graphs model deterministic non-sequential systems. Due to the absence of choice, safe T-graphs are also free of confusion. The class of systems represented by safe T-graphs is an interesting one. This class has appeared under vary many disguises - with some variations on the expressive power - in the literature and will probably continue to

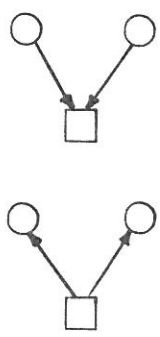
do so. To mention just a few here, Muller [33] in his work on speed independent switching circuits was the first to identify this class. This was followed by Karp and Miller [23] who explicitly used the term 'determinate'. The well-known stream-processing networks of Kahn [54] are one more appealing manifestation of this class and in Milner's CCS [31] they are christened confluent systems. Finally in the land of VLSI systems, they travel under the name of systolic arrays. Clearly the reason for this commonality is that, in the presence of concurrency, deterministic non-sequential systems represent the most elementary step of departure from sequential systems. Indeed we would claim that a good test for a formalism dealing with distributed systems is that it should be able to identify this sub-class in a natural way.

Systems that are both non-deterministic and non-sequential are difficult to analyse. Where confusion is present they are also difficult to synthesise. In net theory there is one particular way of combining choice and concurrency that leads to a class of non-trivial and yet manageable systems. The idea - due to commoner as far as we know - is to find a common generalisation of T-graphs and S-graphs in which choice and concurrency do not 'interfere' with each other.

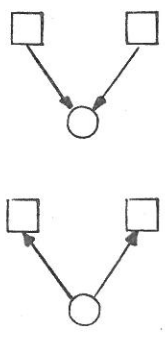
A free choice net (fc net) is a net N = (S,T;F) in which

$$\forall s \in S\ \forall t \in T:\ (s,t) \in F \Rightarrow s^\bullet = \{t\} \ \vee\ \{s\} = {}^\bullet t.$$

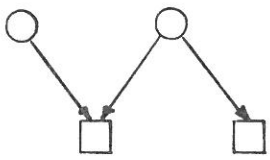
Stated differently, in an fc net, if two transitions share an input place then it is the only input place for both the transitions. Thus by definition, every S-graph (T-graph) is also an fc net. And the converse is clearly not true. For easy reference we show below the sub-structures that are not allowed in the three sub-classes.



S-graph

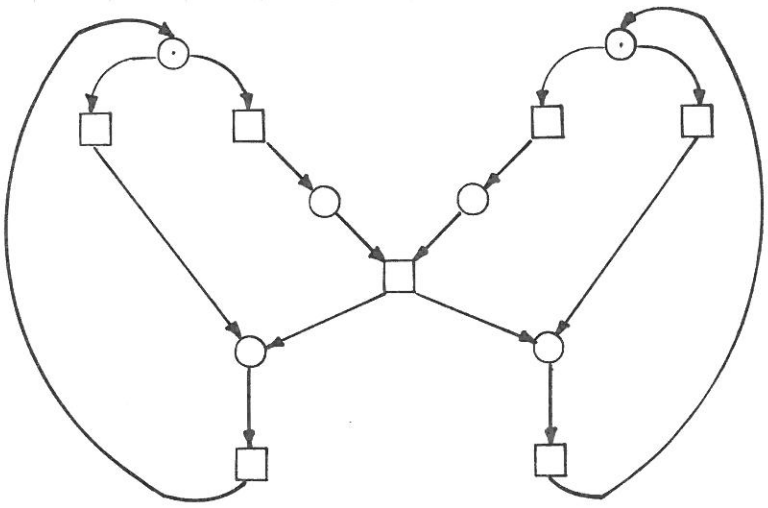


T-graph



fc net

A marked fc net is a marked net whose underlying net is an fc net. An example of an lsfc net is shown below.



Note that a safe fc net can exhibit both non-deterministic and non-sequential behaviours. However their very structure guarantees the absence of confusion. The interested reader might wish to verify this claim.

The theory of lsfc nets is also well-developed. Indeed it is the largest sub-class of safe nets which has a relatively complete theory. Here we shall bring out a characterisation of liveness and safety. For dealing with liveness we need to identify two structural notions called deadlocks and traps. Let $N = (S,T;F)$ be a net and $S' \subseteq S$. Then S' is a deadlock iff ${}^\bullet(S') \subseteq (S')^\bullet$. Thus in a deadlock every T-element which could increase the token count on S' must, for doing so, remove at least one token from S'. The 'opposite' notion is called a trap. $S' \subseteq S$ is a trap iff $(S')^\bullet \subseteq {}^\bullet(S')$. Every T-element which would decrease the token count on S' must, for doing so, put at least one token on S'. A deadlock which is free of tokens can never acquire a token again. A trap which has acquired a token can never become free of tokens again. The liveness theorem for fc nets is:

The marked fc net $\Sigma = (S,T;F,M^0)$ is live iff every deadlock $S' \subseteq S$ contains a trap S'' which is marked under M^0. In other words $S'' \subseteq S'$ and $\sum_{s \in S''} M^0(s) > 0$.

Once again we omit all proofs and the details can be found in [17]. For characterising safety, we need the notion of SM-components. Let $\Sigma = (S,T;F,M^0)$ be a marked net and $N' = (S',T';F')$ be a sub-net of N. In other words, N' is a net; $S' \subseteq S$; $T' \subseteq T$; $F' = F \cap (S' \times T' \cup T' \times S')$. Then N' is called an S-component of Σ iff

1) N' is a strongly connected S-graph.

2) The environment of each S-element in N' is complete relative to N_Σ. More precisely, $\forall s \in S'$: ${}^\bullet s \cup s^\bullet$ (in N') = ${}^\bullet s \cup s^\bullet$ (in N_Σ).

The key property of an S-component is that the total number of tokens distributed over its S-elements remains invariant through transition firings in the composite net. Finally, an SM-component of Σ is an S-component N' = (S',T';F') which satisfies $\sum_{s \in S'} M^0(s) = 1$. We can now state when a live fc net is safe.

The live fc net $\Sigma = (S,T;F,M^0)$ is safe iff it is covered by its set of SM-components. In other words $\forall s \in S$ (and hence $\forall t \in T$) there is an SM-component N' = (S'T';F') such that $s \in S'$.

lsfc nets admit an elegant and powerful decomposition theory. The classic work of Hack [17] contains the central results in this area. He identifies S-components and their dual called T-components as the major structural constituents of an lsfc net. One important consequence of this theory is that lsfc nets are, in a behavioural sense, a common generalisation of ls S-graphs and ls T-graphs. A second consequence is that in this sub-class conflict and concurrency are dual notions. In fact since we have used the term 'dual' repeatedly it might be helpful to nail it down properly.

The reverse dual (or just dual) of a net N = (S,T;F) is the net $\hat{N} = (\hat{S},\hat{T};\hat{F})$ where $\hat{S} = T$, $\hat{T} = S$ and $\hat{F} = F^{-1}$. It is easily verified that the reverse dual of an S-graph (T-graph) is a T-graph (S-graph). Interestingly enough, the reverse dual of an fc net is an fc net. Hack's decomposition theory leads to the following beautiful result.

An fc net has a live and safe marking iff its reverse dual has a live and safe marking.

Based on these results a number of additional structural and behavioural results concerning lsfc nets have been obtained in [48]. At present what is lacking is a synthesis theory. There is however a fairly satisfactory synthesis theory for a sub-class of lsfc nets called well-behaved bipolar schemes [14] which properly include ls S-graphs and ls T-graphs. These schemes also admit a computational interpretation which leads to a class of "well formed" concurrent programs [15].

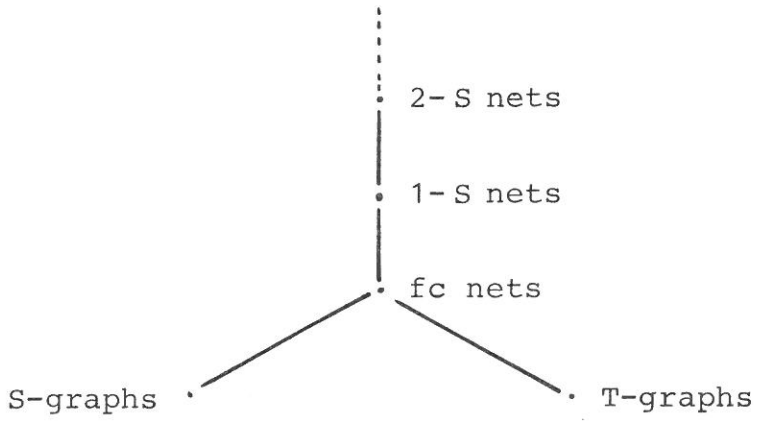
At present not much is known about larger classes of safe nets (see [7, 5] for a few results). The reason we believe is the interplay between choice and concurrency resulting in varying degrees of confusion.

We shall conclude this section with a proposal to classify the remaining safe nets.

For $n \geq 1$, we shall say that a net $N = (S,T;F)$ is an n-S net (pronounced as "n-shared net") iff

$$\forall t \in T: |\{s \in {}^\bullet t \mid |s^\bullet| > 1\}| \leq n.$$

We say a place is shared if it serves as an input place to more than one transition. Note that shared places provide the means for modelling conflict. Thus in an n-S net every transition can have at most n shared input places. It is easy to verify that an fc net is a 1-S net and that the converse is not true. (Consider the marked net of fig. 1.) More generally, one readily obtains the following syntactic hierarchy of nets and hence safe nets (where the ordering relation is inclusion).



In the next section we shall argue that for safe nets, under a reasonable definition of behavioural equivalence, this is also behavioural hierarchy.

4. REPRESENTATIONS OF BEHAVIOUR

A variety of tools are available for representing the behaviour of a safe net. Before we begin discussing these tools, we would like to extend somewhat the notion of a safe net. We do so because it will enable us to provide a uniform framework for discussing behavioural notions such as processes and unfoldings. Moreover, the suppleness of the equivalence notion that we introduce is best brought out by the extended model.

The main change we propose is to consider labelled events. We assume a countable set of actions $A = \{a, a_1, \ldots, b, b_1, \ldots, c, c_1, \ldots\}$ and consider labelled safe nets of the form $\Sigma = (B, E; F, M^0, L)$ where $L: E \to A$ is the labelling function. One question that arises at once is what restrictions, if any, should be placed on L? If L is required to be injective we are back where we started. On the other hand there is something strange about two events carrying the same label occurring concurrently. If nothing else, one will have to use multi-sets (bags) rather than sets to handle steps. Hence we will compromise and demand that the labelling function L should satisfy:

$$\forall e_1, e_2 \in E: L(e_1) = L(e_2) \wedge e_1 \neq e_2 \Rightarrow \forall M \in [M^0\rangle: \{e_1, e_2\} \text{ is } \underline{\text{not}} \text{ a step at } M.$$

One way to ensure this would be to require all the events that carry the same label to lie on an SM-component†. Anyway, in this section we will just consider labelled safe nets whose labelling functions satisfy the above requirement. For convenience we will say just 'safe nets' and drop the term 'labelled'.

The simplest representation of behaviour is in terms of firing sequences. Let Y^* be the free monoid generated by the set Y; λ the null sequence. Then $FS(\Sigma)$, the set of firing sequences of the safe net $\Sigma = (B, E; F, M^0, L)$ is smallest sub-set of E^* given by:

1) $\lambda \in FS(\Sigma)$; $M^0 [\![\lambda \rangle M^0$.

2) Let $\sigma \in FS(\Sigma)$ and $M^0 [\![\sigma \rangle M$.
   If $M[e\rangle M'$ for $e \in E$ then $\sigma e \in FS(\Sigma)$; $M^0 [\![\sigma e \rangle M'$.

Thus $[\![\rangle$ is the three place relation $[\rangle$ extended to sequences of events in the natural way. The language generated by Σ is defined as:

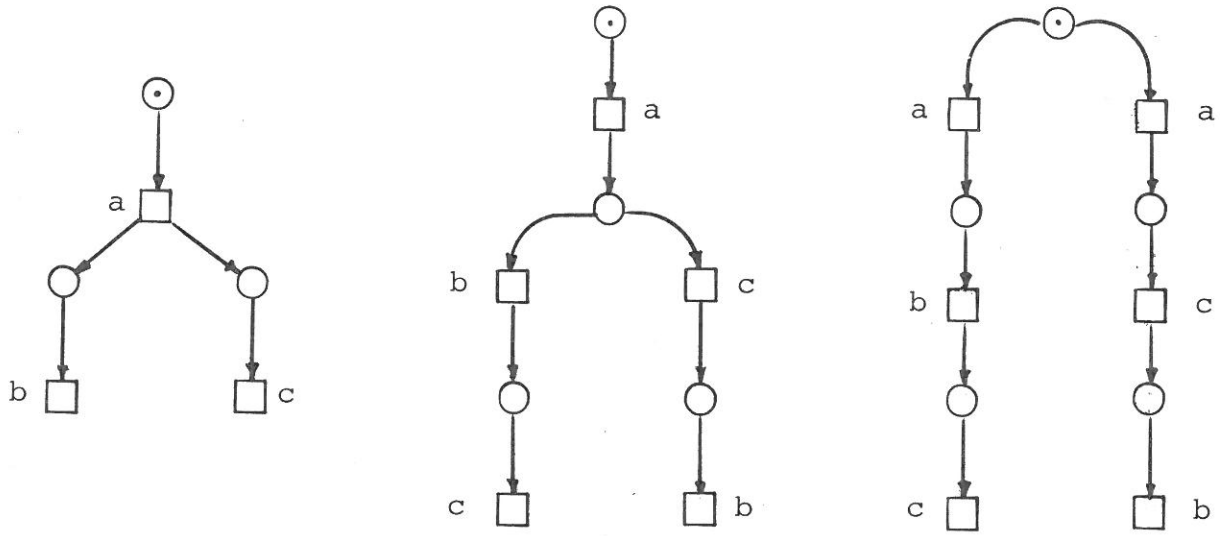
$$L(\Sigma) = \{\tilde{L}(\sigma) \mid \sigma \in FS(\Sigma)\}$$

where $\tilde{L}$ is the obvious extension of L to E^*.

In general, this method of representing behaviours loses information (about concurrency and conflict). The three systems shown below will have the same set of firing sequences.

---

† In this context, it is better to weaken the notion of an SM-component somewhat; instead of demanding strong-connectedness of the underlying S-graph, just demand connectedness.



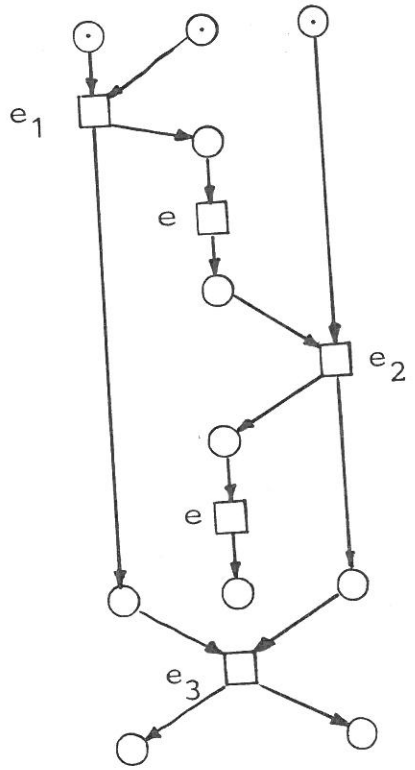
A considerable amount of work has gone into the study of languages generated by marked nets in general. The interested reader may wish to consult [18 , 37 , 49]. Though unsatisfactory as a representation of concurrency, firing sequences are an indispensable tool for proving properties of marked nets.

One generalisation of firing sequences consists of considering sequences of steps. One then obtains what is called a sub-set language. The idea should be clear and we will omit the details. Since a sequence of singletons in the sub-set language can be interpreted as a firing sequence, one gets, using the sub-set language, a finer behavioural representation (in case concurrency is present). The first of the above examples will be distinguished from the two others using this representation. Nevertheless the appraoch is very much rooted in formal language theory. Results concerning sub-set languages can be found in [46].

An elegant generalisation of firing sequences is the notion of a trace due to Mazurkiewicz [29]. Here one retains the power of string manipulating operations and yet obtains a faithful representation of concurrency. Unfortunately it would take us too far afield to explain this notion in more detail. The reader is urged to consult [30] where the full power of this concept is exploited to obtain an algebraic behavioural representation of safe nets.

In net theory corresponding to the notion of a trace, we have a partially ordered set of events. It will however be more convenient to first define the notion of a deterministic process (d-process for short). Loosely speaking, a d-process of a system $\Sigma = (B,E;F,M^0,L)$ is a record of a non-sequential run on Σ where conflicts are resolved as and when they arise. The record will thus consist of partially ordered occurrences of events and conditions. An example of a d-process of the

system of fig. 1 is shown below.



For putting down a definition we need a few notations. A deterministic occurrence net (d-occurrence net, for short) is a T-graph $N = (B,E;F)$ in which F^*, the transitive reflexive closure of F, is a partial ordering relation. In other words N is acyclic. F^* is usually written as $\leq_N$ and as usual the subscript is dropped whenever N is clear from the context.

Let $N = (B,E;F)$ be a d-occurrence net and $x_1, x_2 \in X_N$. Then x_1 co x_2 iff $x_1 \not< x_2$ and $x_2 \not< x_1$. Let $X' \subseteq X$. Then $\downarrow X' = \{y \in X \mid \exists x \in X' \text{ s.t. } y \leq x\}$. Finally, for convenience, we shall relax the definition of a net just enough (drop the demand $S \cup T \neq \emptyset$) to permit the empty net $\emptyset_N = (\emptyset,\emptyset;\emptyset)$ and the corresponding labelled safe net $(\emptyset,\emptyset;\emptyset,\emptyset,\emptyset)$!. Here and in what follows wherever necessary both F and L will be viewed as sets of ordered pairs.

Let $\Sigma = (B,E;F,M^0,L)$ be a safe net. Then the set of d-processes of Σ is denoted $\mathcal{DPR}(\Sigma)$ and is the smallest set of safe nets given by:

1) $\emptyset_d = (\emptyset,\emptyset;\emptyset,\emptyset,\emptyset)$ is a d-process;
   $\text{Rest}(\emptyset_d) = \{(b,\emptyset) \mid b \in M^0\}$.

2) Let $dpr_1 = (B_1,E_1;F_1,M_1^0,L_1)$ be a d-process.

   process is
   (Part

2a) $B_2 = B_1 \cup B_{12} \cup \{(b, \{e\} \cup B_{12} \cup {}^{\bullet}B_{11}) \mid b \in e^{\bullet}\}$.
(Once again $^{\bullet}$ is w.r.t. $\leq_1$)

2b) $E_2 = E_1 \cup \{(e, B_{12} \cup {}^{\bullet}B_{11})\}$.

2c) $F_2 = F_1 \cup \{((b,x_b),(e,x_e)) \mid (b,x_b) \in B_{11} \cup B_{12}, (e,x_e) \in E_2 \setminus E_1\}$
$\cup \{((e,x_e),(b,x_b)) \mid ((e,x_e) \in E_2 \setminus E_1, (b,x_b) \in B_2 \setminus (B_1 \cup B_{12})\}$

2d) $M_2^0 = M_1^0 \cup B_{12}$

2e) $\forall (e',x_{e'}) \in E_2 : L_2((e',x_{e'})) = L(e')$.

2f) $Rest(dpr_2) = Rest(dpr_1) \setminus B_{12}$.

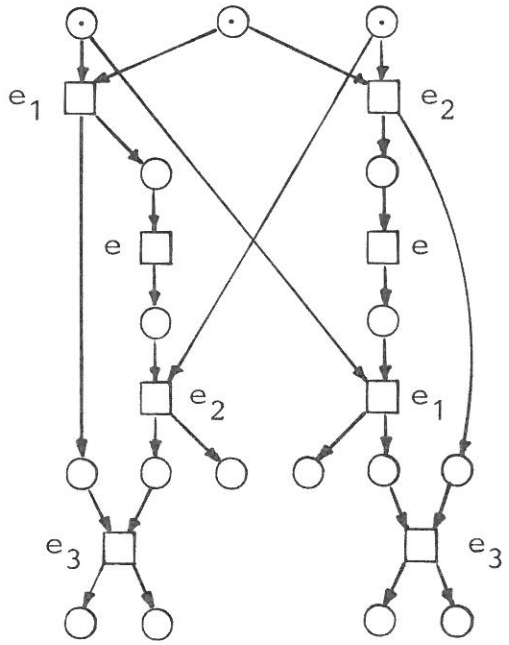
Thus each element of a d-process is an ordered pair. The first component is an element of N_Σ. The second component represents its 'past' in the run modelled by the d-process. In diagrams, the second component can be suppressed as done in the example shown above. We have departed violently from well established conventions in introducing the notion of a d-process. Normally what we call a d-occurrence net is called an occurrence net or a causal net. What we call a d-process is called just a process. The reason for introducing a new name is that we also wish to discuss a more general behavioural representation called the unfolding. And the unfolding of a safe net is based on a type of net which is called - yes! - an occurrence net. Moreover we wish to build up the unfolding through finite approximation which we wish to call processes. So much for terminology.

d-processes are normally defined as a mapping from a d-occurrence net to the underlying net of Σ which preserves the labels and the neighbourhood of events. For details see [12]. d-occurrence nets can be studied in their own right as a model of non-sequential processes. The fundamental ideas in this area are set out in [42] where an intuitively appealing density property called K-density is identified. Results concerning K-density and related density properties can be found in [3]. Notice that in d-occurrence nets, conflict is banished and the causality relation is particularly easy to handle, and its interaction with the concurrency relation is transparent. (Consider our definition of co.) Consequently, one can focus on the concurrency relation. Petri has made an impressive attempt to axiomatise his intuition concerning concurrency in [44]. A study of the properties identified by Petri's axioms is reported in [8]. The relationship between the properties of a system and the properties of its d-processes has been initiated in [16]. In this work, finite marked nets (i.e. the underlying nets are finite but not necessarily safe) are considered and the notion of a

d-process is extended to this larger class. The major result here is that the finiteness of the state space (i.e. the set of reachable markings) of a marked net is equivalent to the K-density of its d-processes. Based on this result, a kind of liveness property of tokens at the system level has been related to a density property called D-continuity (a generalisation, to posets, of Dedekind's definition of continuity of the reals) at the level of d-processes in [4].

Given a d-process dpr = $(B_1,E_1;F_1,M_1^0,L_1)$ of the system Σ one can strip away the conditions and obtain the labelled poset of just events $(E_1;\leqq_1',L_1)$ where $\leqq_1'$ is $\leqq_1$ restricted to E_1. Such structures are called elementary event structures. A more general notion is called an event structure. Event structures are obtained by stripping away the conditions from an occurrence net. Occurrence nets can be used to define the processes of a safe net and a fundamental object associated with a safe net called the unfolding. This representation of behaviour is the finest and the last we shall encounter.

In going through the definitions to follow it might be helpful to consider an example of a process of the system shown in fig. 1.



An occurrence net is a net N = (B,E;F) which satisfies:

1) $\forall b \in B: |{}^\bullet b| \leqq 1$ (no backward sharing)

2) $F^* = \leqq$ is a partial ordering relation (N is acyclic)

Thus every d-occurrence net is an occurrence net but the converse is clearly not true.

Let $N = (B,E;F)$ be an occurrence net. The conflict relation $\#_N \subseteq X \times X$ associated with N is given by:

1) $\forall b \in B\ \forall e_1,e_2 \in b^\bullet: e_1 \neq e_2 \Rightarrow e_1 \#_N e_2$.

2) $x_1 \#_N y_1 \wedge x_1 \leq x_2 \wedge y_1 \leq y_2 \Rightarrow x_2 \#_N y_2$. ($\leq = F^*$ as before)

Thus # is an irreflexive, symmetric relation. Note that $x_1 \# x_2 \wedge x_1 < y$ would imply that $x_2 \# y$.

The independence relation $\&_N \subseteq X \times X$ associated with N is given by:

$$x \,\&\, y \quad \text{iff } x \text{ co } y \text{ and } \neg(x \# y).$$

Finally, $X' \subseteq X$ is a &-set iff $\forall x,y \in X': x \,\&\, y$. We can now generate the processes of a safe net. The set of processes of the safe net $\Sigma = (B,E;F,M^0,L)$ is denoted as $PR(\Sigma)$ and is the smallest set of safe nets satisfying:

1) $\emptyset_{pr} = (\emptyset,\emptyset;\emptyset,\emptyset,\emptyset)$ is a process and $Rest(\emptyset_{pr}) = \{(b,\emptyset) \mid b \in M^0\}$.

2) Let $pr_1 = (B_1,E_1;F_1,M_1^0,L_1)$ be a process of Σ. (As before, part of the inductive hypothesis is that every process is a safe net whose underlying net is an occurrence net with the associated relations $\leq$, # and &).
Suppose $e \in E$, B_{11} is a &-set of pr_1 and $B_{12} \subseteq Rest(pr_1)$ such that

$$^\bullet e = \{b \mid (b,x_b) \in B_{11} \cup B_{12}\} \text{ and } (e, B_{12}^{U\downarrow B_{11}}) \notin E_1$$

Then $pr_2 = (B_2,E_2;F_2,M_2^0,L_2)$ is also a process of Σ where:

2a) $B_2 = B_1 \cup B_{12} \cup \{(b,\{e\} \cup B_{12}^{U\downarrow B_{11}}) \mid b \in e^\bullet\}$.

2b) $E_2 = E_1 \cup \{(e, B_{12}^{U\downarrow B_{11}})\}$.

2c) $F_2 = F_1 \cup \{((b,x_b),(e,x_e)) \mid (b,x_b) \in B_{11} \cup B_{12},\ (e,x_e) \in E_2 \setminus E_1\}$.

$\cup \{((e,x_e),(b,x_b)) \mid (e,x_e) \in E_2 \setminus E_1,\ (b,x_b) \in B_2 \setminus (B_1 \cup B_{12})\}$.

2d) $M_2^0 = M_1^0 \cup B_{12}$.

2e) $\forall (e',x_{e'}) \in E_2 : L_2((e',x_{e'})) = L(e')$.

2f) $Rest(pr_2) = Rest(pr_1) \setminus B_{12}$.

Let $pr_1 = (B_1,E_1;F_1,M_1^0,L_1)$ be a process of Σ. Then it is quite easy to verify pr_1 is a safe net whose underlying net is an occurrence net. Moreover pr_1 (assuming that it is not the 'empty' process) has a very pleasant property. Namely $\forall e \in E_1 : \exists M \in [M_1^0\rangle$ such that $M[e\rangle$ and $\forall b \in B_1 : \exists M,M' \in [M_1^0\rangle$ such that $b \in M$ and $b \notin M'$. In other words in a process we just record those events and condition holdings that have an occurrence (and hence the term occurrence net). The unfolding of a safe net is now obtained by 'summing up' all its processes. To nail this down, first note the set of processes of a safe net Σ, as we have defined it, is a countable set: $PR(\Sigma)$: $\{pr_0,pr_1,\ldots\}$). Assume for $i \geq 0$: $pr_i = (B_i,E_i;F_i,M_i^0,L_i)$. Then the unfolding of Σ is denoted as $\hat{\Sigma}$ and is given by:

$\hat{\Sigma} = (\hat{B},\hat{E};\hat{F},\hat{M}^0,\hat{L})$ where

$$\hat{B} = \bigcup_{i=0}^{\infty} B_i, \quad \hat{E} = \bigcup_{i=0}^{\infty} E_i \text{ etc.}$$

A similar route can be followed to obtain (if they exist) the infinite d-processes of a safe net. The notion of unfolding is due to the authors of [35]. Given a process $(B_1,E_1;F_1,M_1^0,L_1)$ one can strip away the conditions and obtain an event structure of the form $(E_1;\leq_1',\#_1',\&_1')$ where $\leq_1'$ $(\#_1',\&_1')$ is $\leq_1$ $(\#_1,\&_1)$ restricted to E_1. Building on the results of [35], Winskel has worked out a substantial theory of event structures and employed them to provide 'non-interleaved' semantics of CCS-like languages [51, 52].

This brings to an end our discussion of representations of behaviour. We can now define an equivalence notion.

Let $\Sigma_1 = (B_1,E_1;F_1,M_1^0,L_1)$ and $\Sigma_2 = (B_2,E_2;F_2,M_2^0,L_2)$ be two safe nets. Then $R \subseteq [M_1^0\rangle \times [M_2^0\rangle$ is called a bisimulation (between Σ_1 and Σ_2) iff

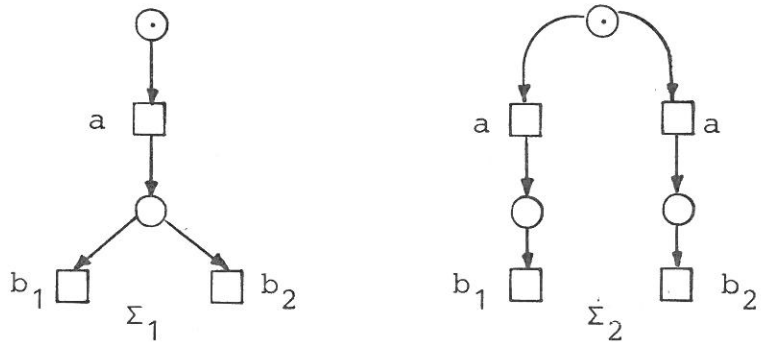
1) $(M_1^0,M_2^0) \in R$

2) $(M_1,M_2) \in R \Rightarrow$ a) $M_1[u_1\rangle M_1'$ (in Σ_1) $\Rightarrow \exists M_2[u_2\rangle M_2'$ (in Σ_2) such that $L_1(u_1) = L_2(u_2)$ and $(M_1',M_2') \in R$.

b) $M_2[u_2\rangle M_2'$ (in Σ_2) $\Rightarrow \exists M_1[u_1\rangle M_1'$ (in Σ_1) such that $L_1(u_1) = L_2(u_2)$ and $(M_1',M_2') \in R$.

We say that Σ_1 and Σ_2 are (bisimulation) equivalent and write $\Sigma_1 \approx \Sigma_2$ iff there is a bisimulation between them. Bisimulation is a refinement discovered by Park [36] of Milner's notion of observational equivalence. The bisimulation relation has very useful properties. Chief among them of course is that it is an equivalence relation.

It is easy but important to verify that $\hat{\Sigma} \approx \Sigma$ where Σ is a safe net and $\hat{\Sigma}$ its unfolding. The crucial feature of bisimulation is that through it one is forced to keep track of all the potential behavioural possibilities (which might lie in the distant future). A simple example might illustrate this point. The two systems shown below are not equivalent though all the behaviours (except processes) we have considered here would identify them.



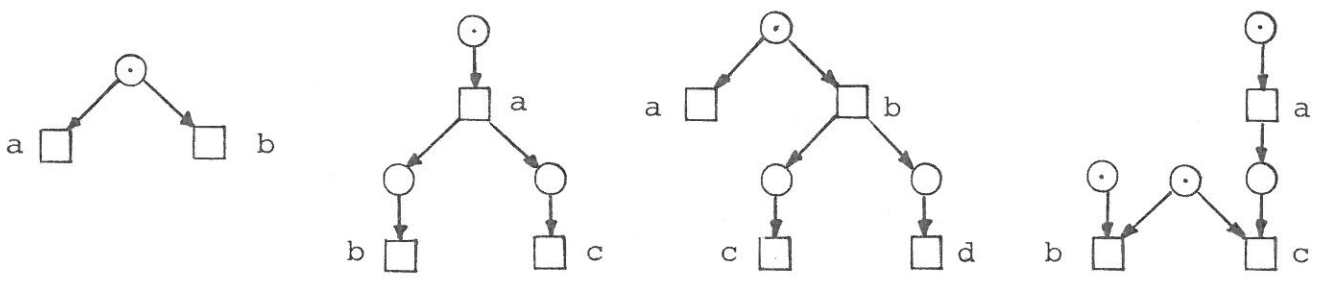
For more interesting and subtle examples the reader should consult [31]. Here we have slightly strengthened the definition in terms of steps in order to block the possibility of representing concurrency through interleaving. Given two classes of safe nets N_1 and N_2, let us define the ordering relation $\preccurlyeq$ as follows:

$$N_1 \preccurlyeq N_2 \quad \text{iff} \quad \forall \Sigma_1 \in N_1 : \exists \Sigma_2 \in N_2 \text{ such that } \Sigma_1 \approx \Sigma_2.$$

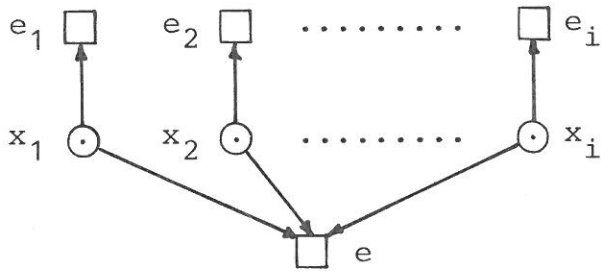
$$N_1 \prec N_2 \quad \text{iff } N_1 \preccurlyeq N_2 \text{ and } N_2 \not\preccurlyeq N_1.$$

For convenience we will let N_T (N_S,N_{fc}) to denote the class of safe T-graphs (safe S-graphs, safe fc nets); and for $n \geq 1$, N_{n-S} will stand for the class of safe n-S nets.

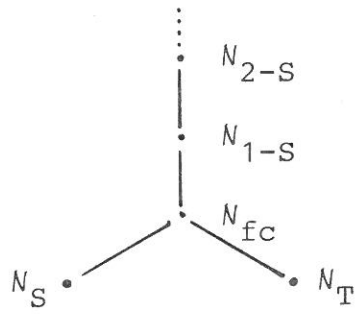
Consider the following systems:



Because of Σ_S (Σ_T) in N_S (N_T) we have $N_S \not\prec N_T$ ($N_T \not\prec N_S$). From the definitions it follows that $N_S, N_T \prec N_{fc} \prec N_{1-S}$. Because of Σ_{fc} we have N_S, $N_T \prec N_{fc}$. And because of Σ_{1-S}, one can obtain $N_{fc} \prec N_{1-S}$. Once again from the definitions it follows that $\forall n \geq 1$, $N_{n-S} \prec N_{n+1-S}$. To show that this ordering is also strict, we consider the sequence of safe nets $\Sigma_2, \Sigma_3, \Sigma_4, \ldots$ where for $i \geq 1$, Σ_i looks as follows.

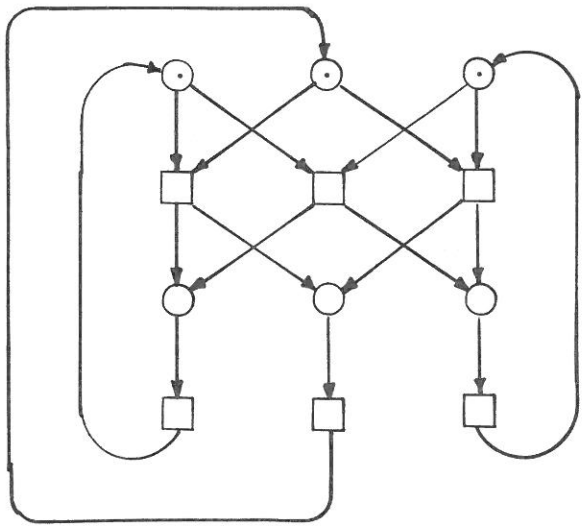


Then it is straightforward to verify that $\forall \Sigma \in N_{i-s}$: $\Sigma \not\approx \Sigma_{i+1}$. Consequently under $\prec$, we can get the following hierarchy.



It would be nice to give some quantitative arguments to support our intuition that this hierarchy represents increasingly complex mixtures of choice and concurrency. Unfortunately, we do not have any results at present along this direction. We do wish to point out though the event structures of the form $(E; \leq, \#, \&)$ 'contained' in the various subclasses will display more and more intricate webs of $\leq$, # and & as one moves up through the hierarchy. And this fact might be helpful in obtaining the kind of result we would like to see.

We are content however that we have exhibited at least one such classification scheme. As stated in the introduction, our main aim has been to bring out some aspects of net theory while using the interplay between choice and concurrency as the topic of discussion. We hope to have succeeded in achieving this aim. We shall conclude this section with an, we think, interesting example. Actually, it is an element of a sequence of examples $\Sigma'_2, \Sigma'_3, \Sigma'_4, \ldots$ and here we show Σ'_2. It should be easy enough to guess how the rest of the sequence looks. This sequence of examples essentially shows that our hierarchy remains intact, if one adds liveness as an additional property.



Σ_2'

5. CONCLUSION

The interplay between causality, conflict and concurrency leads to a bewildering variety of distributed systems. We feel that the chief attraction of net theory lies in its ability to represent - conceptually, mathematically and graphically - these three phenomena and their interactions. The selection of topics treated in this paper has been guided by this theme.

A more ambitious aim is to construct a comprehensive theory of distributed systems and processes based on nets. Petri - with several of his group at GMD, St. Augustin - has been pursuing this aim. Apart from initiating this systems theory [38] he has, over the years, identified a number of fundamental constructs of this theory: Information flow [39]; net morphisms [40]; completions of safe nets to obtain invariants concerning condition-holdings and event occurrences [40]; as already mentioned non-sequential processes [42] and concurrency axioms [44]. The aims and scope of this theory of systems are set out in [43] and the major details are reported in [12]. A second significant attempt to construct such a theory has been made by A.W. Holt, whose influence is less transparent (than Petri's) but nevertheless crucial. The seminal work reported in [19] is still one of the best introductions to net theory and continues to impress with its breadth and depth. Holt was also one of the first to identify confusion as a deep source of difficulties in the study of distributed systems. Finally, Holt has made a valiant - but not entirely successful - attempt to construct a theory of systems in which the continuous ('duration') and the discrete ('instant') hang together gracefully [20].

A good deal of effort has gone into the study of decision problems associated with marked nets. The classic work and results in this area are once again due to Hack [18]. The most famous problem in this area is the reachability problem (given a marked net Σ and a marking M of N_Σ, is M reachable in Σ?) and was solved recently [26,28]. The interested reader may also wish to consult [50] for some additional decidability results.

Marked nets in which the tokens have internal structure are often called high-level Petri nets. Predicate/Transition nets [13] and coloured Petri nets [22] are the most popular versions and a related model has been proposed in [45]. A different but elegant generalisation of marked nets are called FIFO nets where the places are viewed as queues. For an introduction and a nice application see [9]. High-level nets are crucial in applications because marked nets, when used as a modelling tool, yield descriptions that are too detailed and unwieldly.

The best developed analysis tool available for both marked nets and high-level nets is S-invariants (and the related T-invariants). The notion of an S-invariant for marked nets was identified in [27] and later lifted to Predicate/Transition nets [13] and coloured nets [22]. Yet another analysis tool is the so-called reachability tree and a promising extension of this tool for high-level nets has been achieved in [21].

The ability to transform one net description into another in a consistent fashion is a crucial one. Net theory proposes to employ net morphisms for this purpose. So far though, net morphisms have been mostly used for definitional purposes (in particular, for defining d-processes) and is an important unexplored area of research. Winskel has proposed a different notion of a morphism which preserves the token game and hence several crucial behavioural properties [53]. This notion of a net morphism leads to a number of interesting results. In particular one obtains an elegant (category-theoretic) characterisation of the unfolding of a safe net and several useful operations on safe nets.

As mentioned in the introduction, we will not make an attempt to survey applications. A sample of the literature in this area can be found in [1, 2, 34].

Acknowledgement We wish to thank Karen Møller for producing, as usual, an excellent manuscript under difficult circumstances.

REFERENCES

[1] Applications and Theory of Petri Nets. Eds. C. Girault and W. Reisig. Informatik-Fachberichte 52, Springer Verlag, Berlin, Heidelberg, New York (1982).

[2] Applications and Theory of Petri Nets. Eds. A. Pagnoni and G. Rozenberg, Informatik-Fachberichte 66, Springer Verlag, Berlin, Heidelberg, New York (1983).

[3] E. Best: The Relative Strength of K-density. In [34].

[4] E. Best, A. Merceron: Frozen Tokens and D-continuity. Proceedings of the 5th European Workshop on Applications and Theory of Petri Nets, Aarhus, Denmark (1984).

[5] E. Best, M.W. Shields: Some Equivalence Results on Free Choice Nets and Simple Nets and on the periodicity of Live Free Choice Nets, Proceedings of the 8th Colloquium on Trees in Algebra and Programming (CAAP), L'Aquila (1983).

[6] F. Commoner, A.W. Holt, S. Evens, A. Pnueli: Marked Directed Graphs. Journal of Computer and System Sciences 5 (1971), 511-523.

[7] F. Commoner: Deadlocks in Petri Nets. Applied Data Research Inc., CA-7206-2311, Wakefield, Mass., USA (1972).

[8] C. Fernandez, P.S. Thiagarajan: D-Continuous Causal Nets: A Model of Non-sequential Processes. Theoretical Computer Science 28 (1984), 171-196.

[9] M.P. Flé, G. Roucairol: Fair Serializability of Iterated Transactions using FIFO Nets. Proceedings of the 4th European Workshop on Applications and Theory of Petri Nets, Toulouse, France (1983).

[10] H.J. Genrich, K. Lautenbach: Synchronisationsgraphen. Acta Informatica 2 (1973), 143-161.

[11] H.J. Genrich, G. Thieler-Mevisson: The Calculus of Facts. Lecture Notes in Computer Science 45, Springer Verlag, Berlin, Heidelberg, New York (1976), 588-595.

[12] H.J. Genrich, K. Lautenbach, P.S. Thiagarajan: Elements of General Net Theory. In [34].

[13] H.J. Genrich, K. Lautenbach: System Modelling with High-level Petri Nets. Theoretical Computer Science 13, (1981), 109-136.

[14] H.J. Genrich, P.S. Thiagarajan: A Theory of Bipolar Synchronisation Schemes. Theoretical Computer Science 30 (1984), 1-78 (to appear).

[15] H.J. Genrich, P.S. Thiagarajan: Well-Formed Flow Charts for Concurrent Programming. In: Formal Description of Programming Concepts - II, Ed. D. Bjørner, North-Holland Publishing Company, Amsterdam, New York, Oxford (1983), 357-381.

[16] U. Goltz, W. Reisig: The Non-Sequential Behaviour of Petri Nets. Information and Control, 57 (1983), 125-147.

[17] M.H. Hack: Analysis of Production Schemata by Petri Nets. M.S. Thesis, TR-94, Project MAC, Cambridge, Mass., USA (1972).

[18] M.H. Hack: Decidability Questions for Petri Nets. Ph.D. Thesis, TR-161, Project MAC, Cambridge, Mass., USA (1976).

[19] A.W. Holt et al.: Information System Theory Project: Final Report. Applied Data Research Inc., RADC-TR-68-305, Princeton, N.J. (1968).

[20] A.W. Holt: A Mathematical Model of Continuous Discrete Behaviour. Unpublished Manuscript (1980).

[21] P. Huber, A.M. Jensen, L.O. Jepsen, K. Jensen: Towards Reachability Tree for High-level Petri Nets. Proceedings of the 5th European Workshop on Applications and Theory of Petri Nets, Aarhus, Denmark (1984).

[22] K. Jensen: Coloured Petri Nets and the Invariant Method. Theoretical Computer Science 14 (1981), 317-336.

[23] J.R. Jump, P.S. Thiagarajan: On the Equivalence of Asynchronous Control Structures, SIAM Journal on Computing 2, (1973), 67-87.

[24] J.R. Jump, P.S. Thiagarajan: On the Interconnection of Asynchronous Control Structures. Journal of ACM 22, (1975), 596-612.

[25] R.M. Karp, R.E. Miller: Properties of a Model for Parallel Computations: Determinacy, Termination, Queueing. SIAM Journal of Applied Mathematics 14, 6 (1966), 1390-1411.

[26] S.R. Kosaraju: Decidability of Reachability in Vector Addition Systems. Proceedings of the 14th Annual Symposium on Theory of Computing (1982).

[27] K. Lautenbach: Exakte Bedingungen der Lebendigkeit für Eine Klasse von Petri Netzen. (In German.) Berichte der GMD 82, St. Augustin, W. Germany (1973).

[28] E.W. Mayr: An Algorithm for the General Petri Net Reachability Problem. SIAM Journal on Computing 13, 3 (1984), 441-460.

[29] A. Mazurkiewicz: Concurrent Program Schemes and their Interpretations. DAIMI PB-78, Computer Science Department, Aarhus University, Aarhus, Denmark (1979).

[30] A. Mazurkiewicz: Semantics of Concurrent Systems. A Modular Fixed-Point Trace Approach. In: Advances in Applications and Theory of Petri Nets. (To appear.)

[31] R. Milner: A Calculus of Communicating Systems. Lecture Notes in Computer Science 92, Springer Verlag, Berlin, Heidelberg, New York (1980).

[32] R. Milner: A Complete Inference System for a Class of Regular Behaviours. Journal Computer and System Sciences 28 (1984), 39-466.

[33] D.E. Muller, W.S. Bartky: A Theory of Asynchronous Circuits. The Annals of the Computation Laboratory of Harvard University, 29, Harvard University Press, Cambridge, Mass., USA (1959), 204-243.

[34] Net Theory and Applications. Ed. W. Brauer, Springer Lecture Notes in Computer Science, 84, Springer Verlag, Berlin, Heidelberg, New York (1980).

[35] M. Nielsen, G. Plotkin, G. Winskel: Petri Nets, Event Structures and Domains, Part I, Theoretical Computer Science 13 (1981), 85-108.

[36] D.M. Park: Concurrency and Automata on Finite Sequences. Report, Computer Science Department, University of Warwick, Great Britain, (1981).

[37] J.L. Peterson: Petri Nets and the Modelling of Systems. Prentice-Hall, Englewood Cliffs, N.J., USA (1981).

[38] C.A. Petri: Kommunikation mit Automaten. (In German.) Schriften des IIM Nr. 2, Institute für Instrumentelle Mathematik, Bonn, W. Germany (1962).

[39] C.A. Petri: Fundamentals of a Theory of Asynchronous Information Flow, Proceedings of IFIP Congress 62 (1962).

[40] C.A. Petri: Concepts of Net Theory. Proceedings of the Mathematical Foundations of Computer Science Symposium and Summer School, High Tatras, Czechoslovakia (1973).

[41] C.A. Petri: Interpretations of Net Theory. ISF Report 75.07, GMD, St. Augustin, W. Germany (1975).

[42] C.A. Petri: Non-Sequential Processes. ISF Report 77.05, GMD, St. Augustin, W. Germany (1977).

[43] C.A. Petri: Introduction to General Net Theory. In [34].

[44] C.A. Petri: Concurrency. In [34].

[45] W. Reisig: Petri Nets with Individual Tokens. In [2].

[46] G. Rozenberg, R. Verraedt: Subset Languages of Petri Nets. In [2].

[47] C.L. Seitz: System Timing. In: Introduction to VLSI Systems, Eds. Mead, Conway, Addison-Wesley Publishing Company (1980).

[48] P.S. Thiagarajan, K. Voss: A Fresh Look at Free Choice Nets. Arbeitspapiere der GMD, 58, GMD, St. Augustin, W. Germany (1983). (Also to appear in Information and Control.)

[49] R. Valk: Infinite Behaviour of Petri Nets. Theoretical Computer Science 25 (1983), 311-341.

[50] R. Valk, M. Jantzen: The Residue of Vector Sets with Applications to Decidability Problems in Petri Nets. Proceedings of the 4th European Workshop on Applications and Theory of Petri Nets.

[51] G. Winskel: Events in Computation. Ph.D. Thesis, Department of Computer Science, University of Edinburgh, Edinburgh, Scotland (1980).

[52] G. Winskel: Event Structure Semantics for CCS and Related Languages. Lecture Notes in Computer Science 140, Springer Verlag, Berlin, Heidelberg, New York (1982), 561-576.

[53] G. Winskel: A New Definition of Morphism on Petri Nets. Lecture Notes in Computer Science 166, Springer Verlag, Berlin, Heidelberg, New York (1984), 140-150.

[54] G. Kahn: The Semantics of a Simple Language for Parallel Processing. Proceedings of IFIP Congress 74 (1974).