ISSN 0105-8517

A method to compare the descriptive power of different types of Petri nets

by

Kurt Jensen

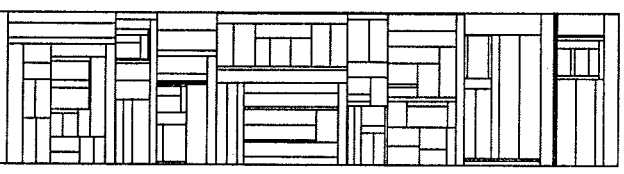
Abstract

This paper proposes a new framework to compare the descriptive power of different types of Petri nets. An extension of condition/event-nets is defined, and in the proposed framework it is shown that this extension has the same descriptive power as condition/event-nets.

This paper has also been published in: Mathematical Foundations of Computer Science 1980, P. Dembiński (ed.), Lecture Notes in Computer Science vol. 88, Springer Verlag 1980, 348-361.

DAIMI PB-108
November 1981 (second edition)

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



1. INTRODUCTION

The purpose of this paper is to show how the descriptive power of different types of Petri nets can be compared, without the use of Petri net languages. Moreover the paper proposes an extension of condition/event-nets and it is shown that this extension has the same descriptive power as condition/event-nets.

During the last decade the role of concurrency has changed drastically. Concurrency was recognized first as a way to speed up throughput, next as a central concept in operating systems, and then as a fundamental concept for description, analysis and comprehension of systems.

Today asynchronous concurrent actions play a central conceptual and practical rôle in hardware, operating systems, computer networks and in many programming and system description languages. To describe, analyse and design such systems, Petri nets have proved to be a valuable tool, [Misunas 73], [Petri 75], [Keller 76], [Baer & Ellis 77], [Mazurkiewicz 77], [Kotov 78], [Schiffers & Wedde 78], [Thiagarajan & Shapiro 78], [Genrich & Lautenbach 79], [Jensen, Kyng & Madsen 79a, b], [Lautenbach & Thiagarajan 79], [Noe 79], and [Zuse 79].

In many of the applications cited above the basic Petri net formalism has been augmented in different ways. Each of these extensions corresponds to a subclass of the very general "transition systems" defined in [Keller 76]. The descriptive power of such subclasses can be compared by use of formal language theory, where each transition is given a name and the set of possible firing sequences is considered, [Hack 76]. In the present paper it is, however, proposed to compare the descriptive power in a more direct way, which is closely connected to the idea of simulation.

In section 2 we define condition/event-nets and an extension of them called "testing Petri nets". Testing Petri nets were introduced in [Jensen 78], (there they were called "extended Petri nets"), and they have been used to define a formal semantics for a system description language in [Jensen, Kyng & Madsen 79a]. Similar primitives are described in [Zuse 79]. The new primitives allow a transition to test some of its conditions without altering their markings.

In section 3 we give a transformation mapping each testing Petri net into a condition/event-net. Moreover we construct a function mapping markings for a testing Petri net into markings for the corresponding condition/event-net. We prove that the transformation satisfies three equations, all of them dealing with reachability.

In section 4 we define "transition systems" (from [Keller 76]) and "simulations" between them. The definition of simulation is directly inspired by the three equations proved for the transformation in section 3. We then prove a close connection between simulation and "strict reduction" (from [Kwong 77]) and this allows us to translate results, obtained for strict reduction by Kwong, to our situation where testing Petri nets are simulated by condition/event-nets. These results show that properties such as existence of a home state, Church-Rosser, non-haltingness and determinacy are preserved by simulation (i.e. the simulated system has the properties iff the simulating system has).

In section 5 we replace transition systems by "named transition systems" (from [Keller 76]) and simulation by "simulation induced by consistent homomorphisms". The latter definition is inspired by "strict reduction induced by homomorphisms"(from [Kwong 77]) and by "consistent" homomorphisms (from [Roucairol & Valk 79]). This allows us to translate a result obtained by Roucairol and Valk concerning liveness.

In section 6 we conclude that the method, used in sections 3, 4 and 5 to compare testing Petri nets and condition/event-nets, can be used to compare other types of Petri nets. We define "equivalence with respect to descriptive power". We discuss how to find the transformations necessary to compare two types of Petri nets, and we give references to papers where such transformations have been sketched.

2. CONDITION/EVENT-NETS AND TESTING PETRI NETS

Definition A condition/event-net is a 5-tuple CEN = $(P, T, PRE, POST, m_0)$ where

1) P is a set of places
2) T is a set of transitions
3) $P \cap T = \emptyset$, $P \cup T \neq \emptyset$
4) $PRE, POST \in [T \to \mathbb{P}(P)]$ where $[\ldots]$ and $\mathbb{P}$ denote total functions and powersets respectively
5) $\forall\, t \in T\ [PRE(t) \cap POST(t) = \emptyset]$
6) $m_0 \in [P \to \{0, 1\}]$ is the initial marking.

A marking is a function $m \in [P \to \{0, 1\}]$. A place p is marked iff m(p)=1 and unmarked iff m(p) = 0. p is a condition for a transition t iff $p \in COND(t) = PRE(t) \cup POST(t)$. It is a precondition iff $p \in PRE(t)$ and a postcondition iff $p \in POST(t)$.

Functions defined on P or T will in this and the following sections be extended to $\mathbb{P}(P)$ or $\mathbb{P}(T)$ in the usual way. As an example $PRE(X) = \bigcup_{t \in X} PRE(t)$, for all $X \subseteq T$.

Condition/event-nets can be represented as directed graphs with two kinds of nodes. Circles represent places, while squares represent transitions. Each transition has ingoing arcs from its preconditions and outgoing arcs to its postconditions. The initial marking is represented by tokens (solid dots) on the marked places.

Two transitions are independent iff their conditions are disjoint. A nonempty set of mutually independent transitions X has concession (and may fire) in a marking m iff all places in PRE(X) are marked and all places in POST(X) are unmarked. If X fires, a new marking m' is reached where all places in PRE(X) are unmarked and all places in POST(X) are marked. We then say that m' is directly reachable from m, which we write as $m \to m'$ or $m \xrightarrow{X} m'$. The transitive closure of direct reachability is written as $m \xrightarrow{+} m'$ or $m \xrightarrow{X_1 X_2 \ldots X_n} m'$, where

$\{X_i \mid i \in 1..n\}$ with $1 \leq n < \infty$, are the sets of transitions, which are fired to reach m' from m. The transitive and reflexive closure is written as $m \rightarrow^* m'$ or $m \xrightarrow{X_1 X_2 \ldots X_n} m'$ with $0 \leq n < \infty$. A marking m is reachable iff $m_0 \rightarrow^* m$.

The firing rule can be formalized as follows:

$$m \xrightarrow{X} m'$$

$$\Updownarrow$$

$$\begin{cases} X \neq \emptyset \\ \forall\, t_1, t_2 \in X \,[COND(t_1) \cap COND(t_2) = \emptyset] \\ \forall\, p \in P \begin{bmatrix} p \in PRE(X) \Rightarrow m(p) = 1 \wedge m'(p) = 0 \\ p \in POST(X) \Rightarrow m(p) = 0 \wedge m'(p) = 1 \\ p \notin COND(X) \Rightarrow m(p) = m'(p) \end{bmatrix} \end{cases}$$

Definition A testing Petri net is a 7-tuple $TPN = (P, T, PRE, POST, TM, TU, m_0)$, where

1) P, T, PRE, POST, and m_0 are defined exactly as for condition/event-nets.
2) $TM, TU \in [T \rightarrow \mathbb{P}(P)]$.
3) $\forall\, t \in T$ [PRE(t), POST(t), TM(t), and TU(t) are mutually disjoint].

Now the different definitions for condition/event-nets can be repeated, except that conditions are defined as $COND(t) = PRE(t) \cup POST(t) \cup TM(t) \cup TU(t)$, where TM(t) and TU(t) are testmarked-conditions and testunmarked-conditions respectively. The firing rule has the form:

$$m \xrightarrow{X} m'$$

$$\Updownarrow$$

$$\begin{cases} X \neq \emptyset \\ \forall\, t_1, t_2 \in X \,[COND(t_1) \cap COND(t_2) = \emptyset] \\ \forall p \in P \begin{bmatrix} p \in PRE(X) & \Rightarrow m(p) = 1 \wedge m'(p) = 0 \\ p \in POST(X) & \Rightarrow m(p) = 0 \wedge m'(p) = 1 \\ p \in TM(X) & \Rightarrow m(p) = 1 \wedge m'(p) = 1 \\ p \in TU(X) & \Rightarrow m(p) = 0 \wedge m'(p) = 0 \\ p \notin COND(X) & \Rightarrow m(p) = m'(p) \end{bmatrix} \end{cases}$$

Testing Petri nets can be represented graphically in the same way as condition/event-nets, except that transitions are connected to their testmarked-conditions by unbroken (undirected) arcs and to testunmarked-conditions by dashed (undirected) arcs.

3. TRANSFORMATION FROM TESTING PETRI NETS TO CONDITION/EVENT-NETS

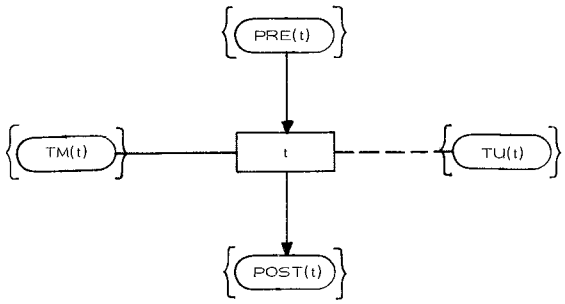
In this section we define a transformation, which maps each testing Petri net into a condition/event-net with a "similar behaviour". We first define the transformation. Then we give a formal meaning to the term "similar behaviour", and we show that the transformation is consistent with this definition.

Let TPN = $(P, T, PRE, POST, TM, TU, m_0)$ be a testing Petri net. We shall transform TPN into a condition/event-net CEN = $(P', T', PRE', POST', m_0')$.

Transformation A

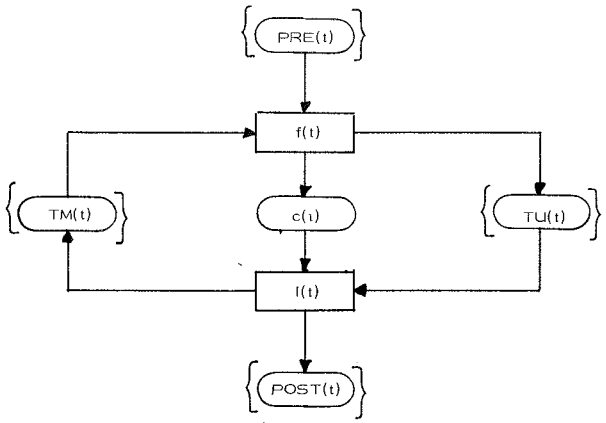
A straightforward first idea is to split each transition t from TPN into two transitions denoted by f(t) and l(t) and connected by a place c(t). Each subnet in TPN of the form

[3.1]



is replaced by a condition/event-subnet of the following form, where {...} indicate subsets of conditions.

[3.2]



Moreover we define a function h, which maps each marking m of TPN into a marking $m' = h(m)$ of CEN defined by

$$m'(p) = \begin{cases} m(p) & \text{if } p \in P \\ 0 & \text{if } p \in c(T) \end{cases}$$

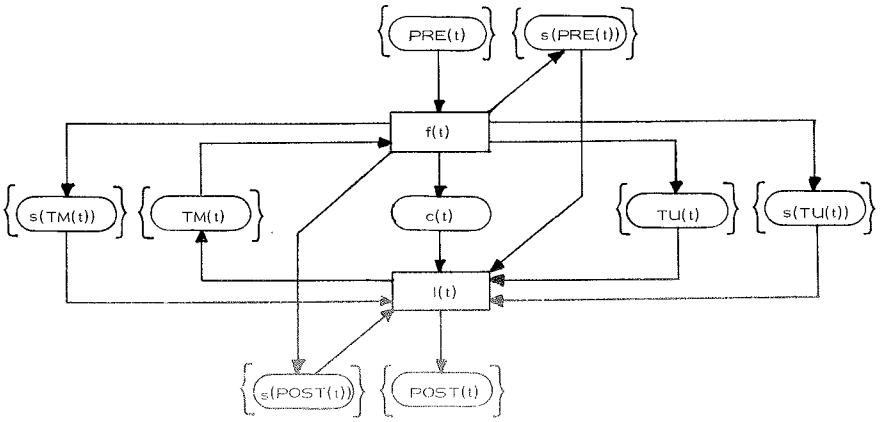
The initial marking of CEN is $m_0' = h(m_0)$.

Unfortunately this simple transformation does not yield a condition/event-net with a "similar behaviour" as the original testing Petri net. The problem is that other transitions may change the marking of COND(t) between the firings of f(t) and l(t).

Transformation B

The problem with transformation A can be solved by splitting each place p into two places denoted by p and s(p). s(p) acts as a binary semaphore controlling the use of p. Then [3.1] is replaced by

[3.3]



and $m' = h(m)$ is defined by

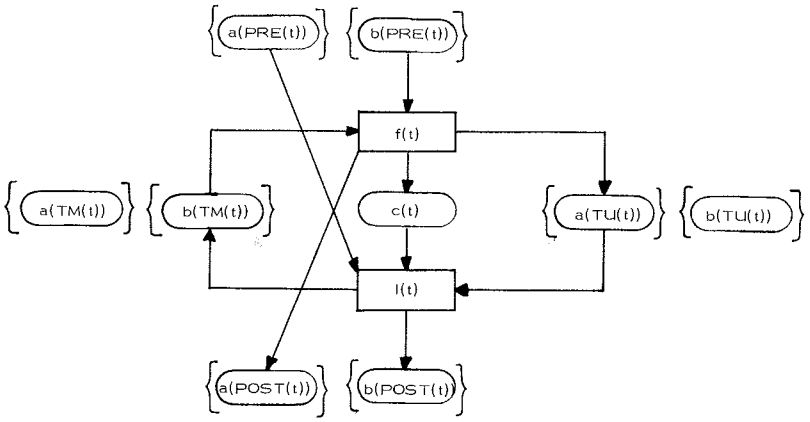
$$m'(p) = \begin{cases} m(p) & \text{if } p \in P \\ 0 & \text{if } p \in s(P) \cup c(T) \end{cases}$$

This transformation yields a condition/event-net with a "similar behaviour" as the original testing Petri net.

Transformation C

However, it is possible to make a more elegant transformation, where the constructed condition/event-net contains the same number of places and transitions, but fewer relations (arcs) between them. Then [3.1] is replaced by

[3.4]



and $m' = h(m)$ is defined by

$$m'(p) = \begin{cases} m(p') \text{ if } p = a(p') \text{ or } p = b(p') \\ 0 \quad \text{if } p \in c(T) \end{cases}$$

The initial marking is $m_0' = h(m_0)$

Similar behaviour

What do we mean, when we say that two Petri nets have a "similar behaviour"? To be very informal we mean that they have the same "basic properties". But then we must ask what the "basic properties" are, and the answer to this question depends heavily on the use of the model.

The answer could be, that the two nets should have a similar concurrency-relation. For transformation C this could be formalised by proving, that two transitions in CEN are concurrent iff the corresponding two transitions in TPN are concurrent.

In this paper we shall, however, focus on properties concerning reachability, liveness, home-states, determinacy etc.

Let TPN, CEN and h be defined as in transformation C. By M_{TPN} and M_{CEN} we denote the sets of markings for TPN and CEN, respectively. A marking $m \in M_{CEN}$ is <u>representative</u> iff $m \in h(M_{TPN})$.

<u>Lemma 1</u> A marking $m \in M_{CEN}$ is representative iff

a) $\forall p \in P \quad [m(a(p)) = m(b(p))]$ and
b) $\forall t \in T \quad [m(c(t)) = 0]$

<u>Proof:</u> Trivial from definition of h. □

<u>Lemma 2</u> For all markings $m'' \in M_{CEN}$, reachable from a representative marking, we have

$$\forall p \in P \quad [m''(a(p)) = m''(b(p)) + \sum_{t \in A(p)} m''(c(t))]$$

where $A(p) = \{ t \in T \mid p \in COND(t) \}$.

<u>Proof:</u> From [3.4] by the invariant-method described in [Lautenbach 75] or by observing that the property is satisfied by any representative marking and kept invariant by the firing of any set of transitions in CEN. □

<u>Lemma 3</u> For all markings $m'' \in M_{CEN}$, reachable from a representative marking, we have

$$\forall t, t' \in T \quad [COND(t) \cap COND(t') \neq \emptyset \Rightarrow m''(c(t)) + m''(c(t')) \leq 1]$$

<u>Proof:</u> Assume that $p \in COND(t) \cap COND(t')$, then $t, t' \in A(p)$ and we conclude from Lemma 2 that $m''(c(t)) + m''(c(t')) \leq 1$. □

In the proof of the following Theorem 1 we shall exclude infinite concurrency in the sense that we demand each set X of transitions, involved in a single step $m \xrightarrow{X} m'$, to be finite. This assumption will allow us to replace $m \xrightarrow{X} m'$ by a finite sequence of individual transitions $m \xrightarrow{x_1 x_2 \cdots x_n} m'$ where $X = \{x_i \mid 1 \leq i \leq n\}$ and $1 \leq n < \infty$. For a proof of the general situation (allowing infinite concurrency) see [Jensen 78].

Theorem 1 The function $h: M_{TPN} \to M_{CEN}$ is an injection and has the following properties:

(1) $h(m_0) = m_0'$

(2) $\forall\, m, m' \in M_{TPN} \left[m \xrightarrow[TPN]{}{}^{+} m' \Leftrightarrow h(m) \xrightarrow[CEN]{}{}^{+} h(m') \right]$

(3) $\forall\, m \in M_{TPN}\, \forall\, m' \in M_{CEN} \left[h(m) \xrightarrow[CEN]{}{}^{*} m' \Rightarrow \exists\, m'' \in M_{TPN} \left[m' \xrightarrow[CEN]{}{}^{*} h(m'') \right] \right]$

Proof: Injectivity and equation 1 follows directly from the definition of h, TPN and CEN.

Equation 2: Assume that $m \xrightarrow[TPN]{}{}^{+} m'$ for two markings $m, m' \in M_{TPN}$. From our exclusion of infinite concurrency this implies the existence of a sequence of transitions $t_1 t_2 \ldots t_r$, with $1 \leq r < \infty$, such that

$$m \xrightarrow[TPN]{t_1 t_2 \cdots t_r} m' .$$

Then

$$h(m) \xrightarrow[CEN]{f(t_1)\, l(t_1)\, f(t_2)\, l(t_2) \ldots f(t_r)\, l(t_r)} h(m')$$

Next assume that $h(m) \xrightarrow[CEN]{}{}^{+} h(m')$ and

$$h(m) \xrightarrow[CEN]{t_1 t_2 \cdots t_s} h(m')$$

where $1 \leq s < \infty$.

We have to prove that

$$m \xrightarrow[TPN]{}^{+} m'.$$

The proof is by induction on s:

s = 1: Impossible since firing of only one transition $f(t_1)$ would lead to a marking with $c(t_1)$ marked and by Lemma 1 such a marking cannot be representative.

s = 2: By an argument similar to case s = 1, we conclude that there exists a transition $t \in T$ such that $t_1 = f(t)$ and $t_2 = l(t)$, but then

$$m \xrightarrow[TPN]{t} m'$$

s > 2: By Lemma 1 all places in c(T) are unmarked in h(m). Thus no transition in l(T) has concession and we conclude that $t_1 = f(t)$ for some $t \in T$. Since c(t) is marked by t_1 and unmarked in h(m'), there exists at least one occurrence of l(t) in the sequence $t_2 \ldots t_s$. Let t_u with $u \in 2..s$ be the first such occurrence. Let $t' \in T$ be any transition with $COND(t) \cap COND(t') \neq \emptyset$. From Lemma 3 it follows that c(t') is unmarked in all markings between the firings of t_1 and t_u and hence neither f(t') nor l(t') can be contained in $FS = t_2 t_3 \ldots t_{u-1}$.

Thus we conclude that no transition in FS has conditions from a(COND(t)) or b(COND(t)), but this means that t_u can be moved just behind t_1 without altering the total effect of the firing sequence:

$$h(m) \xrightarrow[CEN]{t_1 t_u} m'' \xrightarrow[CEN]{t_2 \ldots t_{u-1} t_{u+1} \ldots t_s} h(m')$$

It is easy to check that $m'' = h(m''')$, where m''' is defined by

$$m \xrightarrow[TPN]{t} m'''$$

By the inductive hypothesis we then get

$$m''' \xrightarrow[TPN]{}{}^{+} m'$$

and we conclude

$$m \xrightarrow[TPN]{}{}^{+} m'$$

by transitivity of $\xrightarrow[TPN]{}{}^{+}$.

Equation 3 Assume that $h(m) \xrightarrow[CEN]{}{}^{*} m'$ for two markings $m \in M_{TPN}$ and $m' \in M_{CEN}$.

Let $X = \{t \in T \mid m'(c(t)) = 1\}$. From Lemma 2, Lemma 3 and figure [3.4] it follows that X has concession in m'.

Let $m'' \in M_{CEN}$ be the marking reached from m' by firing X in CEN. From figure [3.4], Lemma 1 and Lemma 2 it follows that m'' is representative. □

In Theorem 1 we showed that h is an injection satisfying three equations. For the moment we will take these equations as our formal definition of similar behaviour. Then Theorem 1 states that transformation C maps each testing Petri net into a condition/event-net with a similar behaviour.

In section 4 we shall see that the three equations imply that TPN has a home-state, is Church-Rosser, non-halting or determinate iff CEN has the corresponding property.

4. TRANSITION SYSTEMS, SIMULATION, AND STRICT REDUCTION

In this section we define "transition systems" (from [Keller 76]) and "strict reduction" (from [Kwong 77]). Moreover we shall see that there is a close connection between simulation (defined in the previous section) and strict reduction. This connection allows us to translate Kwong's results to our situation, where testing Petri nets are simulated by condition/event-nets.

Definition (Keller) A transition system is a triple $TS = (Q, \rightarrow, Q^0)$ where

1) Q is a set of states
2) $\rightarrow \subseteq Q \times Q$ is the transition-relation
3) Q^0 is the set of initial states

We shall write $q \rightarrow q'$ for $(q, q') \in \rightarrow$. The transitive closure is denoted by $\rightarrow^+$, the transitive and reflexive closure by $\rightarrow^*$. A state q' is reachable from a state q iff $q \rightarrow^* q'$. The set of all reachable states is $Q^r = \{q \in Q \mid \exists\, q_0 \in Q^0 [q_0 \rightarrow^* q]\}$.

Let $TS_1 = (Q_1, \xrightarrow{1}, Q_1^0)$ and $TS_2 = (Q_2, \xrightarrow{2}, Q_2^0)$ be transition systems and $h: Q_1 \rightarrow Q_2$ an injection. Let $h(TS_1)$ denote the transition system defined by $h(TS_1) = (h(Q_1), \xrightarrow{h}, h(Q_1^0))$ where $\xrightarrow{h}$ is defined by

$(*)$ $\quad \forall\, q, q' \in Q_1 \; [q \xrightarrow{1} q' \Leftrightarrow h(q) \xrightarrow{h} h(q')]$

Q_1^r, Q_2^r and Q_h^r are the set of reachable states for TS_1, TS_2 and $h(TS_1)$, respectively.

The following definition of "simulation" is motivated by the three equations shown for transformation C in Theorem 1 (section 3).

Definition TS_1 is simulated by TS_2 with respect to h iff

(S1) $h(Q_1^0) = Q_2^0$

(S2) $\forall q, q' \in Q_1^r \; [q \xrightarrow[1]{}{}^{+} q' \Leftrightarrow h(q) \xrightarrow[2]{}{}^{+} h(q')]$

(S3) $\forall q \in Q_1^r \; \forall q' \in Q_2 \; [h(q) \xrightarrow[2]{}{}^{*} q' \Rightarrow$
$\exists q'' \in Q_1 \; [q' \xrightarrow[2]{}{}^{*} h(q'')]]$

Definition (Kwong) TS_2 strictly reduces to TS_1 iff

(1) $Q_1 \subseteq Q_2, \quad Q_1^0 = Q_2^0$

(2) $\forall q_0 \in Q_1^0 \forall q' \in Q_2 \; [q_0 \xrightarrow[2]{}{}^{*} q' \Rightarrow$
$\exists q'' \in Q_1 \; [q' \xrightarrow[2]{}{}^{*} q'' \wedge q_0 \xrightarrow[1]{}{}^{*} q'']]$

(3) $\forall q, q' \in Q_1^r \; [q \xrightarrow[1]{} q' \Rightarrow q \xrightarrow[2]{}{}^{+} q']$

(4) $\forall q, q' \in Q_1^r \; [q \xrightarrow[2]{}{}^{+} q' \Rightarrow q \xrightarrow[1]{}{}^{+} q']$

Theorem 2 TS_1 is simulated by TS_2 with respect to h iff TS_2 strictly reduces to $h(TS_1)$.

Proof: TS_1 is simulated by TS_2 iff (S1)-(S3) are satisfied.

TS_2 strictly reduces to $h(TS_1)$ iff

(R1) $h(Q_1) \subseteq Q_2, \quad h(Q_1^0) = Q_2^0$

(R2) $\forall h(q_0) \in h(Q_1^0) \forall q' \in Q_2 \; [h(q_0) \xrightarrow[2]{}{}^{*} q' \Rightarrow$
$\exists h(q'') \in h(Q_1) \; [q' \xrightarrow[2]{}{}^{*} h(q'') \wedge h(q_0) \xrightarrow[h]{}{}^{*} h(q'')]]$

(R3) $\forall h(q), h(q') \in h(Q_1^r) \; [h(q) \xrightarrow[h]{} h(q') \Rightarrow h(q) \xrightarrow[2]{}{}^{+} h(q')]$

(R4) $\forall h(q), h(q') \in h(Q_1^r) \; [h(q) \xrightarrow[2]{}{}^{+} h(q') \Rightarrow h(q) \xrightarrow[h]{}{}^{+} h(q')]$

Assume (S1)-(S3)

(R1) follows from the functionality of h and from (S1).

(R3) and (R4) follow from (S2) and (*) in the definition of $h(TS_1)$.

In (R2) the first part, $q' \xrightarrow{2}{}^{*} h(q'')$, follows from (S3). The second part, $h(q_0) \xrightarrow{h}{}^{*} h(q'')$, follows from transitivity of $\xrightarrow{2}{}^{*}$ (used on $h(q_0) \xrightarrow{2}{}^{*} q'$ and $q' \xrightarrow{2}{}^{*} h(q'')$) and from (R4).

Assume (R1)-(R4)

(S1) follows from (R1).

(S2) follows from (R3) and (R4).

(S3) follows from (R2), transitivity of $\xrightarrow{2}{}^{*}$ and (S2). □

From the above proof observe that the second part of (R2) is implied by the first part, transitivity of $\xrightarrow{2}{}^{*}$ and (R4). By a similar argument it can be proved that omission of $'' \wedge q_0 \xrightarrow{1}{}^{*} q'' \,''$ from (2) would yield an equivalent definition of strict reduction.

Let $TS = (Q, \rightarrow, Q^0)$ be a transition system. A state $q \in Q$ is

dead iff there does not exist a state q' such that $q \rightarrow q'$
home iff $q' \rightarrow^* q$ for all $q' \in Q^r$

The transition system TS is

non-halting iff all reachable states are non-dead
determinate iff for any $q_0 \in Q^0$ and any $q, q' \in Q$ reachable from q_0
$(\text{dead}(q) \wedge \text{dead}(q')) \Rightarrow (q = q')$
Church-Rosser iff for any $q, q', q'' \in Q^r$
$(q \rightarrow^* q' \wedge q \rightarrow^* q'') \Rightarrow \exists q''' \in Q \,[q' \rightarrow^* q''' \wedge q'' \rightarrow^* q''']$

Corollary 1 If TS_1 is simulated by TS_2 then TS_1 has a home-state, is Church-Rosser, non-halting or determinate iff TS_2 has the corresponding property.

Proof: Theorem 2 and Theorems 4.4, 4.5, 4.6 and 4.7 in [Kwong 77]. □

The basic idea of this (and the following sections) is to consider Petri nets (of different types) as special instances of transition systems. Then Q is the set of all markings, $\rightarrow$ is direct reachability and Q^0 has only one element, the initial marking.

Now let TPN be a testing Petri net and CEN the condition/event-net constructed from TPN (by transformation C in section 3). Let TPN' and CEN' be the corresponding transition systems. From Theorem 1 it then immediately follows that TPN' is simulated by CEN' and from Corollary 1 we get:

Corollary 2 A testing Petri net has a home state, is Church-Rosser, non-halting or determinate iff the condition/event-net constructed from it has the corresponding property.

In this section we have compared our own definition of simulation (inspired by Theorem 1) with Kwong's definition of strict reduction. It turned out that the two definitions are mathematically equivalent. However, the purpose of the two formalisms is quite different. Kwong uses strict reduction to analyse complicated system descriptions. He starts with a "large" transition system. Then he gradually decreases the number of system states, without altering the basic properties. We use simulation to compare the descriptive power of different types of Petri nets. We start with a "small" transition system corresponding to a description containing some Petri net primitives. Then we translate this description into another set of primitives, and to do this we normally have to enlarge the number of system states. In our situation there is often a nontrivial correspondence between the states in the two transition systems. For this reason we have modified Kwong's definition to make the correspondence between states explicit via the injection h. This is similar to the injection i used in [Roucairol & Valk 79].

5. NAMED TRANSITION SYSTEMS AND SIMULATION INDUCED BY HOMOMORPHISMS

In transformation C (defined in section 3) there is a very close connection between the firing of a transition t in TPN and the firings of f(t) and l(t) in CEN. In the formalisation of simulation (and strict reduction) this connection is not captured, because transition systems do not allow us to attach names to the elements of the transition-relation. To remedy this we define in this section "named transition systems" (from [Keller 76]). Moreover we augment the definition of simulation by adding homomorphisms, which connect the transitions of the two systems.

In Theorem 2 (section 4) we proved a close connection between simulation and strict reduction. A similar connection exists between "simulation induced by homomorphisms" and "strict reduction induced by homomorphisms" (from [Kwong 77], generalised in [Roucairol &Valk 79]). This connection allows us to translate results obtained by Roucairol and Valk.

Definition (Keller) A named transition system is a quadruple $(Q, \Sigma, \rightarrow, Q^0)$ where

1) Q is a set of states
2) Σ is a set of transitions
3) $\rightarrow \subseteq Q \times \Sigma \times Q$ is the transition-relation
4) $Q^0 \subseteq Q$ is the set of initial states.

We shall write $q \xrightarrow{t} q'$ for $(q, t, q') \in \rightarrow$. This represents a single step t transforming q to q'. It is generalised to finite sequences of steps by the following definition, where Σ^* is the set of finite strings over Σ, and Λ the empty string:

For any $q, q' \in Q$

(i) $q \xrightarrow{\Lambda} q' \Leftrightarrow q = q'$

(ii) $\forall\, y \in \Sigma^*\ \forall\, t \in \Sigma\ [q \xrightarrow{yt} q' \Leftrightarrow$

$\exists q'' \in Q\ [q \xrightarrow{y} q'' \wedge q'' \xrightarrow{t} q']]$

The set of reachable states is defined by $Q^r = \{q \in Q \mid \exists q_0 \in Q \; \exists x \in \Sigma^* \; [q_0 \xrightarrow{x} q]\}$.

Let $TS_1 = (Q_1, \Sigma_1, \xrightarrow[1]{}, Q_1^0)$ and $TS_2 = (Q_2, \Sigma_2, \xrightarrow[2]{}, Q_2^0)$ be named transition systems. $\Sigma^+ = \Sigma^* - \{\Lambda\}$. Let $A_1: \Sigma_2 \to \mathbb{P}(\Sigma_1) \cup \{\{\Lambda\}\}$ and $A_2: \Sigma_1 \to \mathbb{P}(\Sigma_2^+)$ be arbitrary functions and $h: Q_1 \to Q_2$ an injection. Q_1^r and Q_2^r are the set of reachable states for TS_1 and TS_2 respectively.

Definition TS_1 is simulated by TS_2 with respect to $<h, A_1, A_2>$ iff

(1) $h(Q_1^0) = Q_2^0$

and extending A_1 and A_2 to homomorphisms in the usual way

(2a) $\forall q, q' \in Q_1^r \; \forall x \in \Sigma_1^+ \; [q \xrightarrow[1]{x} q' \Rightarrow \exists y \in A_2(x) \; [h(q) \xrightarrow[2]{y} h(q')]]$

(2b) $\forall q, q' \in Q_1^r \; \forall x \in \Sigma_2^+ \; [h(q) \xrightarrow[2]{x} h(q') \Rightarrow \exists y \in A_1(x) \; [q \xrightarrow[1]{y} q' \wedge y \neq \Lambda]]$

(3) $\forall q \in Q_1^r \; \forall q' \in Q_2 \; \forall x \in \Sigma_2^* \; [h(q) \xrightarrow[2]{x} q' \Rightarrow \exists q'' \in Q_1 \; \exists y \in \Sigma_2^* \; [q' \xrightarrow[2]{y} h(q'')]]$

With respect to $<A_1, A_2>$ define $\Sigma_2' = \{x \in \Sigma_2 \mid A_1(x) \neq \{\Lambda\}\}$.

Definition (Valk and Roucairol) The homomorphisms $<A_1, A_2>$ are consistent iff

(C1) $\bigcup_{x \in \Sigma_2} A_1(x) \supseteq \Sigma_1$

(C2) $\forall q \in Q_1^r \; \forall x \in \Sigma_2 \; [\exists y \in A_1(x) \; [y \text{ fireable in } q] \Rightarrow \forall y \in A_1(x) \; [y \text{ fireable in } q]]$

(C3) $\forall x \in \Sigma_2' \; \forall y \in \Sigma_2^+ \; [y \in A_2(A_1(x)) \Rightarrow y \text{ contains } x]$

The definitions above define the range of A_1 and A_2 as powersets. Often we do not need this generality and we have simulations satisfying one or more of the following equations, where the cardinality of a set B is denoted by $|B|$:

(C4) $\forall x \in \Sigma_2 \, [|A_1(x)| = 1]$

(C5) $\forall x \in \Sigma_1 \, [|A_2(x)| = 1]$

(C6) $\forall x \in \Sigma_1 \, [\{x\} = A_1(A_2(x))]$

The simulation defined immediately below (transformation C) satisfies all three equations (C4)-(C6). Two of the transformations sketched in [Jensen 78, 79] satsify (C4) and (C6), but not (C5). It should be noted that (C4) implies (C2), while (C6) implies (C1).

Simulation induced by consistent homomorphisms is transitive (by functional composition of the involved homomorphisms and injections). This would not be the case if consistent homomorphisms are replaced by strictly consistent homomorphisms as they are defined in [Roucairol & Valk 79].

We again exclude infinite concurrency (see section 4). We can then consider each Petri net (of some type) as a special instance of named transition systems. Then Q is the set of all markings, Σ is the set of transitions, $\rightarrow$ is direct reachability by firing of only one transition, and Q^0 has only one element, the initial marking.

Now let TPN be a testing Petri net and CEN the condition/event-net constructed from TPN (by transformation C in section 3). Let TPN'' and CEN'' be the corresponding named transition systems, with transitions T and $T' = f(T) \cup l(T)$, respectively. Define $A_1: T' \rightarrow \mathbb{P}(T) \cup \{\{\Lambda\}\}$ and $A_2: T \rightarrow \mathbb{P}((T')^+)$ by

$$A_1(t) = \begin{cases} \{t'\} & \text{if } t = f(t') \\ \{\Lambda\} & \text{if } t = l(t') \end{cases}$$

$$A_2(t) = \{f(t)\, l(t)\}$$

From these definitions it immediately follows that $\langle A_1, A_2 \rangle$ is consistent. Repeating the proof for Theorem 1, while keeping track of the names attached to fired transitions, we get:

Theorem 3 TPN'' is simulated by CEN'' with respect to $<h, A_1, A_2>$. $<A_1, A_2>$ is consistent.

Let TS = $(Q, \Sigma, \rightarrow, Q^0)$ be a named transition system with reachable states Q^r. TS is live iff

$$\forall\, q \in Q^r \forall\, x \in \Sigma \exists q' \in Q \exists\, y \in \Sigma^* \; [q \xrightarrow{y} q' \wedge x \text{ fireable in } q']$$

Now we can proceed as in section 4. We can define "strict reduction induced by consistent homomorphisms" (from [Kwong 77] and [Roucairol & Valk 79]) and we can relate it to "simulation induced by consistent homomorphisms" (by a Theorem analogous to Theorem 2).

Corollary 3 A testing Petri net is live iff the condition/event-net constructed from it is live.

Proof: Theorem 3, Theorem 1.2 (a) in [Roucairol & Valk 79] and the fact that each transition $l(t) \in l(T)$ can be fired immediately after the corresponding transition $f(t) \in f(T)$ (cf. the proof for equation 3 in Theorem 1). □

In section 4 we have considered the inverse of transformation C as a "strict reduction". In this section we have considered it as a "strict reduction induced by consistent homomorphisms". It could also be considered as an "algebraic simulation" (from [Milner 71] and [Brand 78]) or as a "contraction" (from [Gourlay, Rounds & Statman 79]).

6. CONCLUSION

In section 2 we defined condition/event-nets and testing Petri nets. In section 3 we gave a transformation mapping testing Petri nets into condition/event-nets, and we proved that this transformation satisfies three equations (Theorem 1). In section 4 we then defined transition systems and simulation between them. The definition of simulation was directly inspired by the three equations proved in Theorem 1. We proved a close connection between simulation and strict reduction (Theorem 2) and this allowed us to translate results obtained for strict reduction to our situation, where testing Petri nets are simulated by condition/event-nets (Corollary 2).

In section 5 we replaced transition systems by named transition systems and we replaced simulation by simulation induced by consistent homomorphisms. This allowed us to translate results obtained for strict reduction induced by consistent homomorphisms (Corollary 3).

This approach, described above for testing Petri nets and condition/event-nets, are general enough to be used to compare the descriptive power of other types of Petri nets.

We say that a Petri net PN_1 is <u>simulated</u> by another Petri net PN_2 iff the transition system corresponding to PN_1 is simulated by the transition system corresponding to PN_2. The two Petri nets need not be of the same type. An analogous definition is made for <u>simulation induced by consistent homomorphisms</u>.

Let A and B be two types of Petri nets.

<u>Definition</u> A is <u>(strongly) not weaker</u> than B iff each net in B can be simulated (induced by consistent homomorphisms) by a net in A. A and B are <u>(strongly) equivalent with respect to descriptive power</u> iff A is (strongly) not weaker than B and vice versa.

As mentioned earlier simulation (induced by consistent homomorphisms) is transitive and thus (strong) equivalence with respect to descriptive power is an equivalence-relation.

Theorem 4 Condition/event-nets and testing Petri nets are strongly equivalent with respect to descriptive power.

Proof: It is trivial that testing Petri nets are strongly not weaker than condition/event-nets. The other direction follows from Theorem 3. □

Having made these formal definitions, an important question arises. Given two types of Petri nets. How do we construct the transformation(s) necessary to compare their descriptive power?

The construction-method from section 3 can often be used: Take a single transition together with its conditions and replace this subnet by a simulating subnet of the other type. Do this for each transition.

The method described in this paper has been used to compare the descriptive power of "testing Petri nets" and "hyper Petri nets" and the descriptive power of "place/transition-nets" and "coloured Petri nets". For sketches of the involved transformations see [Jensen 78, 79]. The transformations used there, are simple in the sense that firing of a single transition is simulated by firing only a single transition in the other net. It is the absence of this property, which makes transformation C and the proof of Theorem 1 non-trivial.

Acknowledgments

Valuable criticism and comments have been made by Morten Kyng, Antoni Mazurkiewicz, Brian Mayoh, Robin Milner, Mogens Nielsen, and Erik Meineche Schmidt.

References

[Baer & Ellis 77]

Baer, J.L. & Ellis, C.S.: Model, design and evaluation of a compiler for a parallel processing environment. IEEE transactions on Software Engineering, Vol. SE-3, No. 6 (November 1977), 394-405.

[Brand 78]

Brand, D.: Algebraic simulation between parallel programs. IBM Research Report RC 7206 (June 1978).

[Genrich & Lautenbach 79]

Genrich, H.J. and Lautenbach, K.: The analysis of distributed systems by means of predicate/transition-nets. Semantics of Concurrent Computation, Evian 1979, G. Kahn (ed.), Lecture Notes in Computer Science vol. 70, Springer Verlag 1979, 123-146.

[Gourlay, Rounds & Statman 79]

Gourlay, J.S., Rounds, W.C. and Statman, R.: On properties preserved by contractions of concurrent systems. Semantics of Concurrent Computation, Evian 1979, G. Kahn (ed.), Lecture Notes in Computer Science vol. 70, Springer Verlag 1979, 51-63.

[Hack 76]

Hack, M.: Petri net languages. Technical Report 159, Laboratory for Computer Science, Massachusetts Institute of Technology, March 1976.

[Jensen 78]

Jensen, K.: Extended and hyper Petri nets. DAIMI TR-5, Computer Science Department, Aarhus University, August 1978.

[Jensen, Kyng & Madsen 79a]

Jensen, K., Kyng, M. and Madsen, O.L.: Delta semantics defined by Petri nets. DAIMI PB-95, Computer Science Department, Aarhus University, March 79.

[Jensen, Kyng & Madsen 79b]

Jensen, K., Kyng, M. and Madsen, O.L.: A Petri net definition of a system description language. Semantics of Concurrent Computation, Evian 1979, G. Kahn (ed.), Lecture Notes in Computer Science vol. 70, Springer Verlag 1979, 348-368.

[Jensen 79]

Jensen, K.: Coloured Petri nets and the invariant-method. Theoretical Computer Science 14 (1981), 317-336.

[Keller 76]

Keller, R.M.: Formal verification of parallel programs. Comm. ACM 19, 7 (July 1976), 371-384.

[Kotov 78]

Kotov, V.E.: An algebra for parallelism based on Petri nets. Mathematical Foundations of Computer Science 1978, J. Winkowski (ed.), Lecture Notes in Computer Science vol. 70, Springer Verlag 1978, 39-55.

[Kwong 77]

Kwong, Y.S.: On reduction of asynchronous systems. Theoretical Computer Science 5 (1977), 25-50.

[Lautenbach 75]

Lautenbach, K.: Liveness in Petri nets. Interner Bericht ISF-75-02. 1, GMD Bonn, July 1975.

[Lautenbach & Thiagarajan 79]

Lautenbach, K. and Thiagarajan, P.S.: Analysis of a resource allocation problem using Petri nets. 1st European Conference on Parallel and Distributed Processing, Toulouse 1979, J.C. Syre (ed.), Cepadues Editions Toulouse 1979, 260-266.

[Mazurkiewicz 77]

Mazurkiewicz, A.: Concurrent program schemes and their interpretations. DAIMI PB-78. Computer Science Department, Aarhus University, July 1977.

[Milner 71]
Milner, R.: An lagebraic definition of simulation between programs. Second Int. Joint Conf. on Artificial Intelligence, British Computer society, 1971, 481-489.

[Misunas 73]
Misunas, D.: Petri nets and speed independent design. Comm. ACM 16, 8 (August 1973), 474-481.

[Noe 79]
Noe, J.D.: Nets in modelling and simulation. Net theory and applications, W. Brauer (ed.), Lecture Notes in Computer Science vol. 84, Springer Verlag 1980, 347-368.

[Petri 75]
Petri, C.A.: Interpretations of net theory. Interner Bericht 75-07, GMD Bonn, July 1975.

[Roucairol & Valk 79]
Berthelot, G., Roucairol, G. and Valk, R.: Reductions of nets and parallel programs. Net theory and applications, W. Brauer (ed.), Lecture Notes in Computer Science vol. 84, Springer Verlag 1980, 277-290.

[Schiffers & Wedde 78]
Schiffers, M. and Wedde, H.: Analyzing program solutions of coordination problems by CP-nets. Mathematical Foundations of Computer Science 1978, J. Winkowski (ed.), Lecture Notes in Computer Science vol. 64, Springer Verlag 1978, 462-473.

[Thiagarajan & Shapiro 78]
Thiagarajan, P.S. and Shapiro, R.M.: On the maintenance of distributed copies of a data base. Interner Bericht ISF-78-04, GMD Bonn, July 1978.

[Zuse 79]
Zuse, K.: Petri-nets from the engineer's viewpoint. Net theory and applications, W. Brauer (ed.), Lecture Notes in Computer Science vol. 84, Springer Verlag 1980, 441-479.