Internal Report 2011-11 august 2011

Universiteit Leiden
Opleiding Informatica

The beginning of a theory for set-nets

Bas van Stein

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

BACHELOR THESIS
August 30, 2011
Supervisor: Jetty Kleijn

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

The beginning of a theory for set-nets

Bas van Stein

LIACS, Leiden University
Leiden, 2300 RA The Netherlands
bvstein@liacs.nl

September 1, 2011

Abstract. We create the beginning of a theory for a new kind of Petri Nets: set-nets. The key difference between standard Petri Nets and set-nets is that the former uses a quantitative approach while the latter uses a qualitative approach. This means that set-nets only look if some resources exist while other Petri Net systems want to know how many instances of a resource exist.

We investigate to what extent the existing theories for PT and EN systems can be applied to set-nets. We first investigate set-nets without inhibitor and activator arcs. For these we modify the existing concepts for PT and EN systems to create new fundamental concepts for set-nets. We create new concepts for things we only see in set-nets and not in PT or EN systems. We then introduce a partial ordering semantics for set-nets based on occurrence nets and processes. After that we introduce the extended set-nets with inhibitor arcs and activator arcs. We show that these arcs make our model more powerful and that there exists extended set-nets that cannot be simulated by non-extended set-nets.

1 Introduction

This work is created as a bachelor thesis within LIACS at the University of Leiden. Set-nets [6] are a new Petri Net (PN) model inspired by processes in biology [5] such as the chemical reactions taking place in cells where molecules can be consumed or created by, facilitate or prohibit reactions. These nets were introduced by the Theoretical Computer Science Group at LIACS in the year 2011.

The major difference with other PN models is that this model does not care how much of a resource is present but only that a resource is present or absent. This allows several transitions (reactions) to share resources, while in other models this would be a conflict. This model can be compared with other related models like EN systems and PT systems in order to initiate a basic theory for set-nets.

We start this work by studying other related work.
Most definitions and proofs assume a working knowledge of EN and PT systems [9, 4].

We will cover several aspects in this paper starting by the basic notions. We look at the basic notions *conflict*, *concurrency* and *causality*. We take these notions for EN and PT as our starting point and look to what extent we can apply them to set-nets. We then investigate how configuration graphs of set-nets differ from configuration graphs of PT and EN systems. We create new notions for behaviour in these configuration graphs that is specific to set-nets. We then look at another analytical tool: Occurrence nets. In what extent can we create occurrence nets in the same way like we do for EN and PT occurrence nets. How can we modify this method to create partial ordering semantics based on occurrence nets and processes to be correct. After this we look to set-nets extended with inhibitor arc and activator arc. We show that these arcs add something useful to our set-net model. We show that there exists extended set-nets that cannot be simulated by non-extended set-nets.

2 Preliminaries

In this section basic principles of net theory are outlined. First we explain the structure of nets. Next we define their dynamics and give semantics.

2.1 Structure

Let us begin by giving the definitions of the different net types and their underlying structure.

Definition 1 (Net).
A net *is a tuple:*

$$N = (P, T, F)$$

P and T are finite disjoint sets of respectively places *and* transitions *and* $F \subseteq (P \times T) \cup (T \times P)$ *is the* flow relation *of N.*

□

We denote the input places of a transition $t \in T$ as ${}^\bullet t$ and the output places of t as $t^\bullet$. For places we do the same, ${}^\bullet p$ meaning all input transitions of place $p \in P$ and $p^\bullet$ all output transitions of place p.

Definition 2 (Self-loop free). *A net $N = (P, T, F)$ is* self-loop free *if:*

$$\forall t \in T, t^\bullet \cap {}^\bullet t = \emptyset$$

□

Definition 3 (Isolated places). *A net $N = (P, T, F)$ contains no* isolated places *if:*

$$\forall p \in P, p^\bullet \cup {}^\bullet p \neq \emptyset$$

□

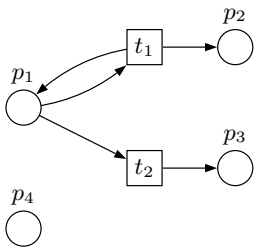


Fig. 1. A net with *places* (circles) and *transitions* (rectangles). An *isolated* place p_4 and a *self-loop* on t_1, p_1.

A place in a net can hold tokens denoted by black dots in the place. A token distribution over the places is said to be a configuration (marking) of the net. Markings will be the states in the operational dynamics of the net system. In set-nets and EN systems a marking is just a subset of all places, i.e. each place is either marked or empty. In a PT system the marking is a function $C : P \to \mathbb{N}$ (where $\mathbb{N}$ is the set of natural numbers $(0, 1, 2, \ldots)$ and $\mathbb{N}_+$ is the set of all natural numbers higher than zero $(1, 2, 3, 4 \ldots)$) that allocates a natural number to each place. So places in PT systems can hold more tokens at the same time and therefore these systems can be seen as "counting" systems.

Definition 4 (Elementary Net system).
An Elementary Net system *is a tuple:*

$$EN = (P, T, F, M_0)$$

where (P, T, F) *is a self-loop free net and*
$M_0 \subseteq P$ *the initial marking of EN.* □

Note that the net of figure 1 is not the underlying net of an EN system because of the self-loop.

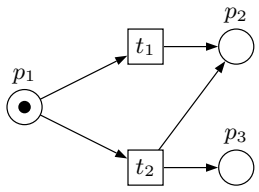


Fig. 2. EN system with *initial marking* $\{p_1\}$, *tokens* are small black dots or sometimes a number to represent that number of small black dots.

Definition 5 (Place/Transition System).
A P/T system is a tuple:

$$MN = (P, T, F, W, M_0)$$

where (P, T, F) *is a net*
$W : F \to \mathbb{N}_+$ *is the weight function, and*
$M_0 : P \to \mathbb{N}$ *is the initial marking.* □

Definition 6 (Set-net).
A set-net (SN) is a tuple:

$$SN = (P, T, F, M_0)$$

where (P, T, F) *is a net*
$M_0 \subseteq P$ *is the initial marking.* □

Note that PT systems and set-nets can contain self-loops.

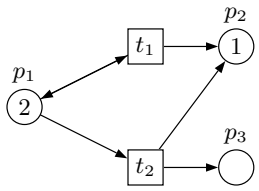


Fig. 3. PT net with initial marking $M(p_1) = 2, M(p_2) = 1$.

2.2 Dynamics

Transitions are the active elements in net models. Whether or not a transition can occur is determined by the current marking. There are different kinds of enabledness depending on the type of net class (EN, PT or SN). In every definition, lemma and theorem from now on we will notate the net class for which this definition, lemma or theorem applies at the start (`EN, PT, SN`) or if it applies to every one of them we denote it by a `N` where `N`$= (P, T, F, M_0)$ or $= (P, T, F, W, M_0)$ a general net system where W is the weight function of a PT system that is only present if we want to apply this definition for PT.

Definition 7 (`EN` enabled). *Let $EN = (P, T, F, M_0)$ be an EN system, let $C \subseteq P$ be the current marking of EN. A transition $t \in T$ is* enabled *at C (t **con** C) if:*
${}^\bullet t \subseteq C \wedge t^\bullet \cap C = \emptyset$. □

Definition 8 (`PT` enabled). *Let $PT = (P, T, F, W, M_0)$ be a PT system, let $C : P \to \mathbb{N}$ be the current marking of PT. A transition $t \in T$ is* enabled *at C (t **con** C) if:*
$\forall p \in {}^\bullet t : W(p, t) \leq C(p)$. □

Definition 9 (`SN` enabled). *Let $SN = (P, T, F, M_0)$ be a SN system, let $C \subseteq P$ be the current marking of SN. A transition $t \in T$ is* enabled *at C (t **con** C) if:*
${}^\bullet t \subseteq C$. □

Having the concept of enabled nets, we want to do something with it. If a transition t is enabled at the marking C, then t can occur at C $C[t\rangle$. What happens when t fires (occurs) at a given marking is determined by the firing rule which in turn is dependent on the chosen net model.

Definition 10 (`EN` firing rule). *Let $EN = (P, T, F, M_0)$ an EN system. If a transition $t \in T$ is* enabled *at configuration C, then it may occur and lead to a new configuration. This is D denoted by $C[t\rangle D$ and defined by:*
$D = (C - {}^\bullet t) \cup t^\bullet$. □

Definition 11 (PT firing rule). *Let $PT = (P, T, F, W, M_0)$ a PT system. If a transition $t \in T$ is* enabled *at configuration C, then it may occur and lead to a new configuration. This is D denoted by $C[t\rangle D$ and defined by:*
$\forall p \in P$,
$D(p) = C(p) - W(p,t) + W(t,p)$. □

Definition 12 (SN firing rule). *Let $SN = (P, T, F, M_0)$ a SN system. If a transition $t \in T$ is* enabled *at configuration C, then it may occur and lead to a new configuration. This is D denoted by $C[t\rangle D$ and defined by:*
$D = (C - {}^\bullet t) \cup t^\bullet$. □

Because of these firing rules we normally do not need isolated places in nets because they are not affected by transitions and hence not relevant to the dynamic behaviour of the net. We want no isolated transitions and we even want them to always contain at least one input place and one output place. We will see later on that it does not matter for the behavior whether we apply these restrictions or not but it makes it easier to analyze and proof things.

2.3 Steps

When two or more transitions are enabled at the same marking, they may fire simultaneously provided that they do not interfere.
What interference means depends on the type of net as we will see shortly.

Definition 13 (Neighbourhood). *Let $N = (P, T, F)$ a net. The neighbourhood of a transition $t \in T$ consists of its input and output places.*
$nbh(t) = t^\bullet \cup {}^\bullet t$. □

The most common interpretation of non-interference is when transitions have disjoint neighbourhoods. This is the assumption underlying concurrent steps, that is, sets of simultaneously occurring transitions in EN systems.

Definition 14 (EN concurrent step). *Let $EN = (P, T, F, M_0), C \subseteq P$. A nonempty set $U \subseteq T$ is a* concurrent step *occurring at C if:*
(1) for every two distinct transitions $t_1, t_2 \in U : nbh(t_1) \cap nbh(t_2) = \emptyset$.
*(2) for every $t \in U : t$ **con** C, denoted by U **con** C.*
(3) if U occurs at C then this leads to a new configuration D, denoted by $C[U\rangle D$ where $D = (C - {}^\bullet U) \cup U^\bullet$.

□

Simultaneously is also called independence in EN systems, since transitions with disjoint neighbourhoods are completely independent of one another.

Definition 15 (PT concurrent step). *Let $PT = (P, T, F, W, M_0), C : P \to \mathbb{N}$. A non-empty set $U \subseteq T$ is a* concurrent step *occurring at C if:*
$\forall p \in {}^\bullet U : \sum_{\forall t \in U} W(p,t) \leq C(p)$
There are enough tokens in every input place for all the transitions together.

if U occurs at C then this leads to a new configuration D, denoted by $C[U\rangle D$ where:
$\forall p \in P$,
$D(p) = C(p) - (\sum_{\forall t \in U} W(p,t)) + (\sum_{\forall t \in U} W(t,p))$. □

Note that also "auto-concurrency" is possible with PT systems but that is not considered here.
Non-interference for PT systems is a dynamic property instead of a structural property. It needs enough tokens in every input place for a step to be enabled.

Definition 16 (SN concurrent step). *Let $SN = (P, T, F, M_0), C \subseteq P$. A non-empty set $U \subseteq T$ is a concurrent step occurring at C if:*
*for every $t \in U : t$ **con** C, denoted by U **con** C.*
if U occurs at C, then this will lead to D, denoted by $C[U\rangle D$ where $D = (C \backslash {}^\bullet U) \cup U^\bullet$.

□

Note that for SN systems transitions never interfere with each other. Every set of *enabled* transitions in a set-net can fire simultaneously.
When sequences of transitions and steps occur from the initial marking we get a lot of new configurations. We may want to see all these configurations in one graph. If we do so we call this graph the configuration graph. If we leave the steps out of this graph, then we get the sequential configuration graph. The effect of a step consisting only one transition $\{t\}$ is the same as the effect of the single transition t.

Definition 17 (N Step-reachable markings). *Let $N = (P, T, F, M_0)$ a net system. The* step-reachable markings *C_{sM} of N are all the possible markings that we can create by firing sequences of* steps *starting from the initial marking M_0.*

□

note that a step of only one transition is still a step.

Definition 18 (Reachable markings). *Let $N = (P, T, F, M_0)$ a net system. The* Reachable markings *C_M of N are all the possible markings that we can create by firing sequences of* transitions *starting from the initial marking M_0.*

□

Definition 19 (N configuration graph).
Let $N = (P, T, F, M_0)$ a net system. A configuration graph (CG) of N contains all the step-reachable markings *C_{sM} in the net from the initial marking M_0 where we construct the graph as follows:*
let C be the current marking beginning with $C = M_0$
for every step U that can occur at C and leading to a configuration D we create a new node D and a directed edge from the node C to D labeled with U. □

Definition 20 (N sequential configuration graph).
Let $N = (P, T, F, M_0)$ a net system. A sequential configuration graph (SCG) of a net contains all the reachable markings *C_M in the net from the initial marking M_0 where we only allow single transitions to occur:*

let C be the current marking beginning with $C = M_0$

for every transition t that can occur at C and leading to a configuration D we create a new node D and a directed edge from the node C to D labeled with t. □

3 Set-nets : Basic notions

Now that we have these basic definitions and see the differences between the systems we can go into more detail. In this section we will investigate the basic notions *conflict*, *concurrency* and *causality* for set-nets. We look to what extend we can apply these notions already known for EN and PT systems.

3.1 Conflict

Conflict normally occurs when two or more transitions interfere.

Definition 21 (N conflict). *Let $C \subseteq P$ and $t_1, t_2 \in T$. Two transitions t_1, t_2 are in conflict at C if:*

t_1 ***con*** $C \wedge t_2$ ***con*** $C \wedge \neg\{t_1, t_2\}$ ***con*** C.

□

This is conflict because the transitions cannot fire simultaneously. In set-nets conflicts as just defined cannot occur. If t_1 **con** C and t_2 **con** C then $\{t_1, t_2\}$ **con** C (Definition 16). However there are situations in which the word conflict would be appropriate for set-nets.
Consider Figure 4.

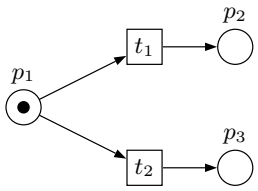


Fig. 4. A set-net with conflict?

If t_1 occurs in its initial marking, t_2 cannot occur after since its input is consumed by t_1. However initially t_2 could occur simultaneously with t_1. There are two ways of firing transitions: Either *sequentially* or *simultaneously*. If we only look at the sequential way we can say in this situation that there is a conflict.
Let us introduce a new kind of conflict called the sequential conflict.

Definition 22 (N sequential conflict). *Let $N = (P, T, F, M_0)$, $t_1, t_2 \in T$ and C a configuration of N. t_2 is in sequential conflict with t_1 at C if:*

t_1 ***con*** $C \wedge t_2$ ***con*** $C \wedge \neg t_2$ ***con*** D *where* $D = C[t_1\rangle D$.

□

Now consider Figure 5.

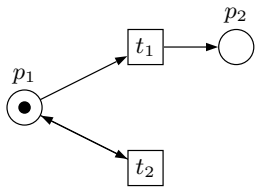


Fig. 5. Another conflict type?

In this set-net t_1 **con** C and t_2 **con** C and t_1 **con** D where D is $C[t_2\rangle D$. If we look at Definition 22 this says there is no sequential conflict between t_2 and t_1, but t_2 cannot occur after t_1, so t_1 is in sequential conflict with t_2. We can distinguish symmetric sequential conflict and a-symmetric sequential conflict.

Definition 23 (N symmetric sequential conflict).
Let $N = (P, T, F, M_0)$, $t_1, t_2 \in T$ and C a configuration of N. t_1 and t_2 are in a symmetric sequential conflict at C if:

t_1 is in sequential conflict with t_2 at C and t_2 is in sequential conflict with t_1 at C. □

The asymmetric sequential conflicts are caused by self-loops. Because of these self-loops it can be that a transition outputs his own input and therefore does not disable other transitions connected to this input place.

Theorem 1. *Let $SN = (P, T, F, M_0)$. If SN is self-loop free then all sequential conflicts are symmetric.*

Proof. Let $t_1, t_2 \in T$, $C \subseteq P$ and t_1 is in *sequential conflict* with t_2 at C
Since t_1 is in sequential conflict with t_2, both transitions are enabled at C. If t_1 occurs then t_2 becomes disabled. This can only be the case if ${}^\bullet t_1 \cap {}^\bullet t_2 \neq \emptyset$. Since SN has no self-loops we know that ${}^\bullet t_2 \cap t_2^\bullet = \emptyset$. Hence if t_2 occurs then t_1 becomes disabled.
So t_2 is in *sequential conflict* with t_1

□

Now we have seen the situation that transitions share an input place without having this place as output place and the situation that one of the transitions had this shared input place as output place. There is one situation left we did not do yet. Two transitions can also share input places that they also have as output places (with self-loops). In this case we do not speak of any type of conflict since they can fire in every order.
A concurrent step can also be in sequential conflict with another step.

Definition 24 (N sequential conflict for steps).
Let $N = (P, T, F, M_0)$, $U \subseteq T, V \subseteq T$ and C a configuration of N. U is in sequential conflict with V at C if:
*U **con** $C \wedge V$ **con** $C \wedge \neg V$ **con** D where $D = C[U\rangle D$.*

□

3.2 Concurrency

We have called two transitions concurrent if they can fire "simultaneously" at a given configuration. In set-nets all transitions that are enabled, are not in conflict and can fire simultaneously. We will now look at differences between *simultaneously*, *independence* and *unordered* firing of transitions. Then the question arises if the concurrent step has the so called diamond structure in its configuration graph.

Definition 25 (N diamond structure). *Let $U \subseteq T, C \subseteq P$. A step U has a* diamond *structure in the* configuration graph *of N if:*
for every distinct two subsets $U_1, U_2 \subseteq U$ where $U_1 \neq \emptyset \wedge U_2 \neq \emptyset$ and $U_1 \cup U_2 = U, \forall C \in C_{sM} : C[U\rangle \Rightarrow \exists (C, D, D', F \subseteq P)$ so that $C[U_1\rangle D, C[U_2\rangle D', D'[U_1\rangle E$ and $D[U_2\rangle E$.
*And there must be at least one $C \in C_{sM}$ where U **con** C.* □

Definition 26 (N substep property). *Let $U \subseteq T, C \subseteq P$. If a step U is enabled at C then*
every substep V of U is enabled at C

□

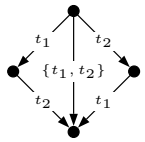


Fig. 6. diamond structure in CG.

EN and PT systems have the *substep property*, SN systems have this property as well. Moreover in SN systems it is also the case that if a number of substeps are enabled at C that form U, U is enabled at C as well. This is not the case with EN or PT systems, since EN or PT transitions can be in conflict.

For EN we only need the sequential configuration graph to create the complete (concurrent) configuration graph (CG). For SN this is not the case. The concurrent steps in SN can create complete new configurations that do not appear in the sequential configuration graph and can lead to different behaviour. In EN systems the diamond property is something very structural, we can see if a step has the diamond structure by looking at the structure of the substeps. If these substeps are disjoint then there is a diamond structure in EN systems. Can we see if a step has the diamond structure in SN at the net structure as well?

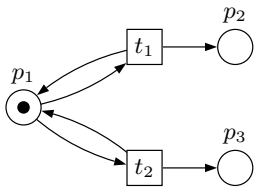


Fig. 7. A set-net where $U = t_1, t_2$ creates a diamond structure in the CG.

To see what kind of steps create this diamond structure we will split concurrent steps into four different groups.

Definition 27 (SN independent). *Let* $SN = (P, T, F, M_0)$, $U \subseteq T$. U *is* independent *if:*

$\forall$ *distinct pair of transitions* $t_1, t_2 \in U, nbh(t_1) \cap nbh(t_2) = \emptyset$

. □

Note that these independent steps are just the ones we have seen earlier (for EN). We added them here again for set-nets to explain the differences between these steps and other step types.

Definition 28 (SN semi-unordered). *Let $SN = (P, T, F, M_0)$, $U \subseteq T$. U is a* semi-unordered *step if:*

(1) $\exists t_1, t_2 \in U$, $(\exists p \in P : p \in {}^\bullet t_1 \rightarrow p \in t_2^\bullet \wedge p \notin {}^\bullet t_2) \vee$
$(\exists q \in P : q \in {}^\bullet t_2 \rightarrow q \in t_1^\bullet \wedge q \notin {}^\bullet t_1)$

(2) $\forall$ *distinct pair of transitions* $t_1, t_2 \in U, {}^\bullet t_1 \cap {}^\bullet t_2 \subseteq t_1^\bullet \cap t_2^\bullet$

□

Semi-unordered steps are steps where we can fire the substeps in every order we want, but the order in which we fire them results in different end configurations.

Definition 29 (SN unordered). *Let $SN = (P, T, F, M_0)$, $U \subseteq T$. U is an* unordered *step if:*

($\forall$ distinct pair of transitions $t_1, t_2 \in U, {}^\bullet t_1 \cap {}^\bullet t_2 \subseteq t_1^\bullet \cap t_2^\bullet$ *)*

□

To let these steps occur they of course need to be enabled at the current marking. Recall that every enabled step is *simultaneous* as well.
The possibility to be a semi-unordered, unordered or simultaneous step is structural. However, to know that a step occurs in such way we have to look at the behaviour.

Lemma 1 (SN semi-unordered step in CG). *Let $SN = (P, T, F, M_0)$, $U \subseteq T, C \subseteq P$. If U is a* semi-unordered *step at a configuration C then*

U together with all fire sequences consisting of non empty distinct subsets of U: $(U_1 \cup U_2 \cup \ldots \cup U_n = U)$ at C give a structure in the configuration graph of SN that leads to at least two different end configurations. So $D \neq D'$ where $C[U\rangle D$ and $C[U_i U_k \ldots U_l\rangle D'$ and $U_i \cup U_k \cup \ldots \cup U_l = U$.

In PT systems *semi-unordered* steps would create a diamond structure and in EN systems they would not be a step because of contact. The structure in the configuration graph gives for each order of firing a possible different configuration. That is why we cannot call them only unordered. We are still able to fire them in every order, so in that respective we could call them unordered.
These semi-unordered steps do not have the diamond structure in its configuration graph (Definition 25). That is why this new name is created for set-nets. All other names already exists for PT systems.

We can divide these *semi-unordered* steps in two categories: *symmetric* and *asymmetric*. They create either three different configurations (b) or two different configurations (a) when firing in all possible orders (figure 6):

Fig. 8. SN asymmetric semi-unordered (a), symmetric semi-unordered (b).

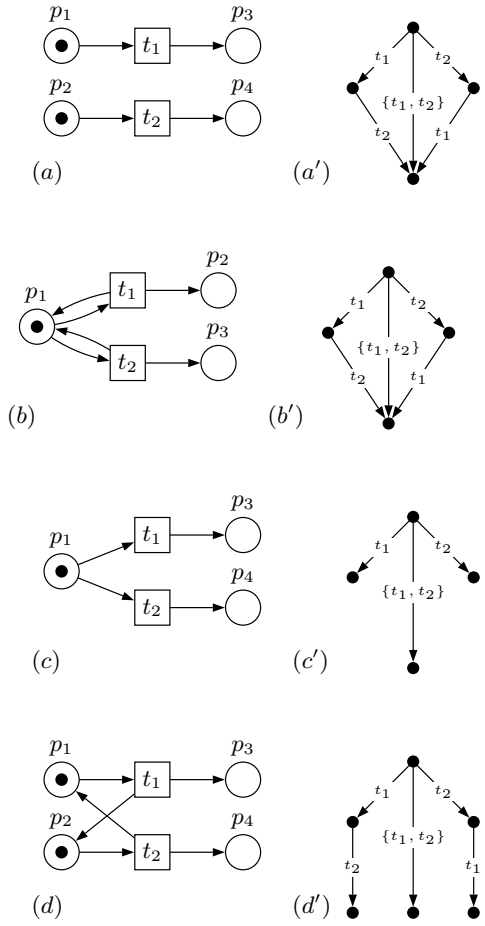


Fig. 9. SN Independent (a) with CG (a'), unordered (b) with CG (b'), simultaneous (c) with CG (c'), semi-unordered (d) with CG (d')

When a semi-unordered symmetric step (with two transitions) occurs, three different configurations dependent of the order of firing $(t_1t_2), (t_2t_1), \{t_1, t_2\}$ (last one is simultaneous) could exist. This is unique in set-nets and cannot happen in EN and PT systems. This occurs when places already filled will be "overwritten" when a transition fires that has that place as an output place and then get emptied by the other transition. In EN this is called contact and makes the step disabled.

Theorem 2 (SN diamond property). *Let $SN = (P, T, F, M_0)$, $U \subseteq T$ and CG the configuration graph of SN.*
The diamond property *of SN systems: U has the* diamond structure *in CG iff:*
U is unordered *and not* semi-unordered *and*
U is enabled at a configuration $C \in C_{sM}$.

Proof. $\leftarrow$
Assumption: U is *unordered* and not *semi-unordered* and can occur at $C \in C_{sM}$.
To prove: U has the *diamond structure* in the CG.
Not semi-unordered means$\forall t_1, t_2 \in U : ({}^\bullet t_1 \cap t_2^\bullet = \emptyset) \wedge ({}^\bullet t_2 \cap t_1^\bullet = \emptyset)$.
$C[t_1, t_2\rangle = C[t_2, t_1\rangle$ because the output of t_1 is not used as input for t_2
and vice versa. So they both create output that will not be affected by the other.
Because U is unordered we know that we can fire $C[t_1, t_2\rangle$ only if we can fire $C[t_2, t_1\rangle$ so U has the diamond structure in CG.
$\rightarrow$
Assumption: U has the *diamond structure* in the CG.
To prove: U is *unordered* and not *semi-unordered* and can occur at $C \in C_{sM}$.
U can occur at a $C \in C_{sM}$ because if he could not U would not be in the CG and therefor can never has a diamond structure in the CG.
The substeps of U have either shared input or not. If they have no shared input then U is unordered since no transition can disable others within this step (they cannot consume the input of other substeps).
If the substeps have shared input then U is unordered or cannot fire sequentially: $\exists t_1, t_2 \in U : \neg(t_1t_2 \textbf{ con } C) \vee \neg(t_2t_1 \textbf{ con } C)$. In the latter case U is not unordered and has no diamond structure so U has to be unordered for our assumption. U cannot be semi-unordered since then there $\exists t_1, t_2 \in U, p \in P : p \in {}^\bullet t_2 \wedge p \notin {}^\bullet t_1 \wedge p \in t_1^\bullet$. If t_1 fires the token produced in place p, the token will "disappear" because there already was a token (otherwise t_2 was not enabled). Firing t_1t_2 gives another configuration then t_2t_1 and so has no diamond structure. U has to be unordered and not semi-unordered. □

Now we have four groups of concurrent steps that cover all the possible configuration structures occurring in set-nets. All steps are *simultaneous*, those that are not in sequential conflict are *unordered*. We have a subset of *unordered* steps that create new and special structures in the configuration graph: *semi-unordered* (unordered steps with contact between substeps). And last but not

least we got a subset of *unordered* that would have a diamond property in EN systems as well: *independent*.

3.3 Causality

Causality is the notion that transitions are the cause of other transitions. In other words, to enable a transition that is not yet enabled at a given configuration another transition has to fire first.

Definition 30 (EN causality). *Let* $EN = (P, T, F, M_0)$, $t_1, t_2 \in T, C \subseteq P$.
Transition t_2 *is caused by transition* t_1 *at* C *if:*
t_2 ***con*** C *does not hold but* $t_1 t_2$ ***con*** C *does hold.*

□

In EN systems we have two kinds of concession, input and output concession.
Input Concession: $p \in (t_1^\bullet \cap {}^\bullet t_2)$ where p contains no token
Output Concession: $p \in ({}^\bullet t_1 \cap t_2^\bullet)$ where p contains a token
In SN we only have input concession.
The EN causality does not work in the same way for several situations with set-nets:

Transitions can be caused by a step rather than caused by one transition. Steps can have different behaviour as firing these transitions sequentially. See Figure 10.
Output concession is based on the notion of contact and this does not exist for set-nets.

Definition 31 (SN causality). *Let* $SN = (P, T, F, M_0)$, $t \in T$, *a non-empty set* $U \subseteq T$, $t \notin U$, $C \subseteq P$.
t *is* caused by *a set of transitions* U *at* C *if:*
$U \subseteq T, t \in T$, t ***con*** C *does not hold*
but Ut ***con*** C *does hold*
Moreover U *is the* minimal set *that causes* t *if in addition:*
there does not exist a $V \subset U$ *where* Vt ***con*** C *does hold. (*U *is the* minimal set*)*

□

In EN systems transitions can also be the cause of more than one other transition. If this is the case then this transition is the cause of each single transition in the step at a specific C. In set-nets it may happen that because of self-loops like in Figure 10 that a transition is the cause of a step U but not the cause of each individual transition in U (at any C). But even without self-loops this is possible as we see in Figure 11.

See Figure 10 for a set-net in which this set causality occurs.

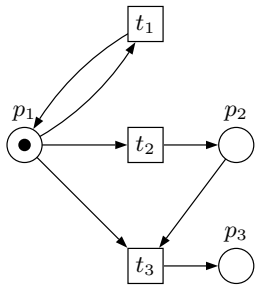


Fig. 10. t_3 is caused by $\{t_1, t_2\}$

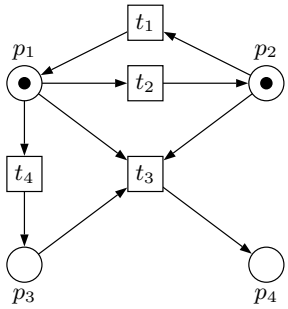


Fig. 11. t_3 is caused by $\{t_1, t_2, t_4\}$

We can even be more general and say that a *set* of transitions can be caused by a set of transitions.

Definition 32 (SN Causality for steps). *Let* $SN = (P, T, F, M_0), C \subseteq P, U, V \subseteq T$, $V \cap U \neq U$.
U is the cause *of another* step *V at C if:*
*$U \subseteq T, V \subseteq T, V$ **con** C does not hold*
*but UV **con** C does hold*
Moreover U is the minimal step cause of V if in addition:
*there does not exist a $W \subset U$ where WV **con** C does hold.* □

This however is because for all $t \in V$ there is a minimal subset $W \subseteq U$ where t is caused by W or t was already enabled before U. Because set-nets have the subset property (definition 26) we know that each transition in V is enabled after we fired U.
We do not need this extra definition. Let SN be a set-net and V, U steps as above.
A transition in V was not enabled or already enabled before the firing of U. If

the transition $t \in V$ was already enabled in U then there exists a $V - t$ that is caused by U. If the transition $t \in V$ is not enabled before U then there is a $W \subseteq U$ that enables t because t is enabled after firing U. So when steps are caused by other steps this is just because all the transitions inside this step are caused by the first step.

3.4 Contact-freeness and complement

In this section we will investigate if we can *complement* set-nets to make them *contact-free*. We do so because this would be very useful when we look at occurrence nets and processes. *Contact-free* means that transitions only occur when their output places are empty.

Definition 33 (EN,SN contact-free). *Let $M = (P, T, F, C_{in})$ be an EN or SN system.*
M is contact-free if for all $t \in T$ and $C \in C_M$: if ${}^\bullet t \subseteq C$ then $t^\bullet \cap C = \emptyset$. □

To create contact-free EN systems we use to complement places.

Definition 34 (EN, SN complement). *Let M be an EN or SN system and let $p, q \in P_M$ and M_0 the initial marking of M.*
Then p and q are complementary, denoted by p com q, if $p^\bullet = {}^\bullet q$ and ${}^\bullet p = q^\bullet$ and $M_0(p) = 1 - M_0(q)$. □

We may want to use this complementing for set-nets as well. We can just construct this complementing in the same way as for EN systems. If we complement a place we want to know if the behaviour of the net system is not affected. Lets find out if we can use complementing for set-nets as well.

Definition 35 (SN complement construction).
Let M be a set-net (P_M, T_M, F_M, M_{0M}), $p \in P_M$ and $K = cpl(M, p, q)$ (M with p complemented by a new place q).
K will then be (P_K, T_K, F_K, M_{0K}) where $P_K = P_M \cup \{q\}$, $T_K = T_M$.
$F_K = F_M$ with in addition:
$\forall e \in P_M$: if $(e, p) \in P_M$ then $(q, e) \in F_K$, if $(p, e) \in P_M$ then $(e, q) \in F_K$.
$M_{0K} = M_{0M}$ with in addition $M_{0K}(q) = 1 - M_{0M}(p)$. □

Theorem 3 (Complementing in set-nets). *Let M a set-net and $M' = cpl(M, p, q)$ as in definition 35, then M can be simulated by M' (M' accepts the same step firing sequences of M),*
iff $\forall U \subseteq T_M : p \notin {}^\bullet U \cap U^\bullet$.

Proof. Let M be a set-net and $p \in P_M$. We create a new set-net M' by taking the complement of p: p'. If the behaviour is unaffected then all transition steps should have the same effect on either M and M' with the difference that p' will

contain a token when p does not and vice versa.

Let U be non-empty a step of transitions. $p \in nbh(U)$ or $p \notin nbh(U)$. If $p \notin nbh(U)$ in M then $p \notin nbh(U) \wedge p' \notin nbh(U)$ in M'.

If $p \in nbh(U)$ in M then this can occur in three ways:

- $p \in {}^\bullet U - U^\bullet$
  then there is a $t \in U$ so that $p \in {}^\bullet t$ and $p' \in t^\bullet$. and there does **not** exists a t where $p' \in {}^\bullet t$ and $p \in t^\bullet$. When U fires in M the token from p will be removed. When this happens in M' the token will also be removed from p and a token will be added in p'. The complemented place is still working since it contains a token while the original place does not.

- $p \in U^\bullet - {}^\bullet U$
  then there is a $t \in U$ so that $p' \in {}^\bullet t$ and $p \in t^\bullet$. and there does **not** exists a t where $p \in {}^\bullet t$ and $p' \in t^\bullet$. When this step happens in M, p gets a new token. When this happens in M' the token will also be added to p and a token will be removed from p'. The complemented place is still working since it contains no token while the original place does.

- $p \in U^\bullet \cap {}^\bullet U$
  then there is a $t \in U$ so that $p \in {}^\bullet t$ and $p' \in t^\bullet$. and there also exists a t where $p' \in {}^\bullet t$ and $p \in t^\bullet$. When this step happens in M the token from p will be removed and another token will be putted back in. When this step wants to fire in M' it requires a token from p and p' witch may never occur so the new set-net with a complement of p has different behaviour then the original one.

Conclusion: complementing places in set-nets does not affect the behaviour if and only if the complemented places are only used as input or only used as output place. Contact-free is not a normal form for set-nets.

□

4 Configuration graphs

4.1 Complexity

It is rather easy to create a CG for a given set-net. We can just check all possible transitions and combinations of transitions that can fire in the initial configuration and we can calculate what new configuration these transitions or set of transitions lead to.
But, there is one little problem with set-nets, since the firing of steps (a set of transitions) could have different behaviour then firing them sequentially, different configurations can arise from those steps. So we **need** to include **all** steps to

see what configurations are reachable.
If a certain configuration allows n transitions to fire, there are n^2 different steps that can fire as well. All possible sets of enabled transitions can fire from that configuration. Creating the full configuration graph can be time consuming depending on the amount of enabled transitions per configuration.

4.2 Finite

A set-net has always a finite configuration graph because:
(1) a set-net has a finite number of places
(2) every place has one token or no token at all,
so for every place there exists two possible markings.
(3) a finite number of places with a maximum of 2 configurations per place gives us a maximum of 2^n where $n = \#P$ (number of places in P) possible configurations in the graph.

4.3 Special configurations

There are a few special configurations that can occur in set-nets. Deadlocks are the dead ends of the CG, because the CG is finite we can always detect deadlocks.
Another very special configuration is a configuration in which transitions can fire, but if they fire the configuration does not change. Let us call these configurations 'stable configurations'.

Definition 36 (N stable configuration). *Let* $N = (P, T, F, M_0)$, $C \subseteq P$.
A configuration C *is* stable *if:*
for all $U \subseteq T$, U ***con*** $C \Rightarrow C[U\rangle C$.

□

Deadlocks are a special case of these stable configurations. In deadlocks no transitions can occur.

Definition 37 (N Deadlocks). *Let* $N = (P, T, F, M_0)$, $C \subseteq P$.
A configuration C *is a* deadlock *if:*
C *is a* stable state *and* *for all* $U \subseteq T$, U ***con*** C *implies* $U = \emptyset$.

$C[U\rangle C$ can also be written as ${}^\bullet U \subseteq U^\bullet \wedge U^\bullet \subseteq C$ to show that the possible appearance of these configurations can be detected by looking at the structure of the net rather than the behaviour. A typical set-net with a stable configuration:

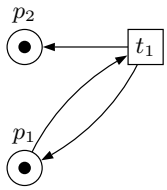


Fig. 12. set-net with a stable configuration.

Stable configurations create self-loops in the configuration graph.
This is an interesting property of a configuration as we will see when we try to convert set-nets to other related boolean net systems. We may want to name a configuration with such loops. Not only if all transitions at C have this property but also if only one transition has this property at C.

Definition 38 (N self-loop configuration). *Let $N = (P, T, F, M_0), C$ a configuration of $N, U \subseteq T$.*
C is a self-loop configuration *if:*
*there exists a non-empty U, U **con** $C \Rightarrow C[U\rangle C$.*

□

A net without self-loops can also have a self-loop configuration. This is caused by steps as we can see in Figure 13.

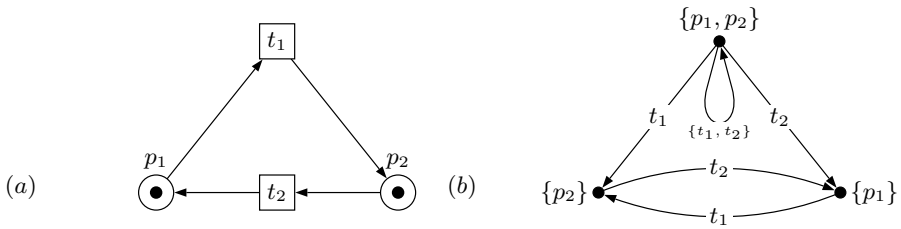


Fig. 13. self-loop configuration (b) without self-loops in net structure (a)

5 Occurrence nets and partial orderings

Occurrence nets are nets that provide insight into the causal relations between occurrences of transitions. They are different from configuration graphs because

they represent a single concurrent run of the net and the relations between the transitions occurring in that run.
Two major problems arise when constructing occurrence nets for set-nets.

We extract occurrence nets from *partial orders* of transitive occurrences.

Definition 39 (Partial Ordering). *Let A be a finite set. A binary relation $\rho \subseteq A \times A$ is a partial order on A if ρ is irreflexive and transitive; (A, ρ) is also called a partially ordered set. A subset B of A is linearly ordered if for all $a, b \in B : a \ \rho \ b$ or $b \ \rho \ a$ or $a = b$.*

□

A partial ordering of transitions in a run of an EN system shows how transitions precedes and succeeds one another.

5.1 Occurrence nets

Definition 40 (Occurrence nets). *A net $N = (B, E, R)$ is an* occurrence net *if:*
(1) N is acyclic. and
(2) $\#({}^\bullet b) \leq 1$ and $\#(b^\bullet) \leq 1$ for all $b \in B$.

□

Labeled occurrence nets for EN systems are created by unfolding the net according to a step firing sequence of the system. We begin with all places invectively labeled according to C_{in} from our EN system in our new labeled occurrence net $OCEN$. Now let us assume that our labeled occurrence net is partly finished and all places in $OCEN$ that do not have output edges correspond to the marking reached in our original EN system (hypothesis). Step U fires and we show how the labeled occurrence net is now extended.
We add $t \in U$ with a new transition labeled "t" to $OCEN$. We then find the places that matches ${}^\bullet t$ in our labeled occurrence net and that have no output transition. We add directed arcs from every labeled place we selected to the new transition. We add new places to the $OCEN$ due for each output place of U and label them accordingly and create directed edges between the labeled transition and all his labeled output places. We repeat this for all steps in the step firing sequence.

When we create processes of SN systems by unfolding the net in the same way, a very strange thing occurs: branching places.
This is in contrast with the second part of Definition 40. In set-nets places can be the input place of more then one transition, where those transitions can fire simultaneously. This creates branching places in our occurrence net.
We could adapt the definition and just say they are cyclic, but we can also choose to change the way we create these occurrence nets, we choice the latter option because cyclic occurrence nets cannot be abstracted to valid partial orders.

If we look at set-nets and how they differ from EN systems, we see that in set-nets sets of transitions can fire simultaneously as a step with shared input. If a step occurs at a configuration and they do share input then we see a branching place in our occurrence net. Now instead of modeling a branching place we just model the set of transitions as one single combined labeled transition. If we do so then there are no branching places anymore. This idea leads to the following definitions and constructions.

Definition 41 (SN Dependent substeps). *Let $SN = (P, T, F, M_0)$ be a set-net, $U \subseteq T$ a step.*
We say that $U_1U_2 \dots U_n$ is a partition of U and are dependent substeps *if:*
For all these substeps U_k and all $t \in U_k$: $\exists t' \in U_k$ where
$nbh(t) \cap nbh(t') \neq \emptyset$ and for all $t \in U_k, \forall l \in \mathbb{N} \leq n, l \neq k : \neg\exists t' \in U_l$ such that $nbh(t) \cap nbh(t') \neq \emptyset$.

This partition of substeps is always unique because every transition is such a substep is dependent on at least one other transition in the step and not connected to any other transition outside this step. There is always at least one dependent substep for a step U.

Lemma 2 (SN Dependent substeps firing). *Let $SN = (P, T, F, M_0)$ be a set-net, $U \subseteq T$ a step, $U_1U_2 \dots U_n$ dependent substeps of U. All the substeps are enabled at C if U is enabled at C and the firing order of the dependent substeps does not matter for the end result after firing all substeps or U as a step.*

Proof. Because of the substep property every substep of U is enabled if U is enabled. Every dependent substep has no shared places with other dependent substeps. Therefore firing a substep would not change the state of the input and output places connected to the other dependent substeps and so the result after firing all substeps will be the same as firing U.

Each dependent substep will be modeled as a single transition. Since the dependent substeps are independent with respect to other substeps, firing the dependent substeps after each other has the same effect as firing them as one big step. This is why we create a new transition for each dependent part of such a big step instead of one transition for the whole step.

Definition 42 (*construction* SN occurrence net). *Let $SN = (P, T, F, M_0)$ be a set-net. Let $OCS_0 = (B_0, E_0, R_0, \Phi_0)$ be our initial labeled occurrence net. Let $V_1V_2 \dots V_n$ be the step firing sequence we use for our construction and OCS_k the labeled occurrence net we have after firing step U_k.*
We start with the occurrence net OCS_0 where Φ is our labeling function between OCS and SN. $\Phi(B) \to P, \Phi(E) \to \mathcal{P}(T)$
$E_0 = R_0 = \emptyset$. $\forall p \in M_0 : B_0 = \{b_{p_0} : p \in M_0\}$ $\Phi(b_{p_0}) = p$.
now for every step $U = V_k$ from our step firing sequence we fire in SN:
(1) $U \subseteq T, U \neq \emptyset$. We first split U

in parts that are independent from each other by using definition 41.
(2) For each dependent substep U_l we add a new *labeled transition e to E and extend Φ with $\Phi(e) = U_l$.*
We then apply for each U the steps below. (3-7)
(3) We find all places b where $\Phi(b) = p, p \in {}^\bullet U_l$ let this be P_1.
(4) We take all places $P_2 \subseteq P_1$ such that $P_2^\bullet = \emptyset$.
(5) $\forall b \in P_2$ we add (b, e) to R (directed edges).
(6) $\forall p \in U_l^\bullet$, we add a new b to B and extend Φ with $\Phi(b) = p$, let this set be P_3.
(7) $\forall b \in P_3$ we add (e, b) to R (directed edges).

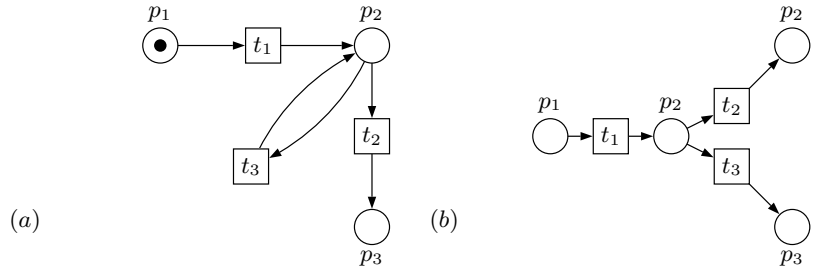


Fig. 14. Set-net (a) with standard occurrence net (b) for $\{t_1\}\{t_2, t_3\}$ with branching places.

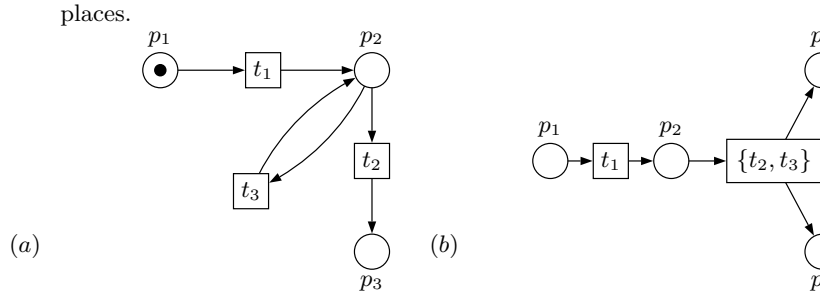


Fig. 15. Set-net (a) with new occurrence net (b) for $\{t_1\}\{t_2, t_3\}$ without branching places.

This notation of occurrence nets for set-nets is much more natural than the EN occurrence net notation. Not only because it eliminates the problem of branching places but because it gives the right information for contact-free set-nets as well. If we look at Figure 14.a then we see that the configuration $\{p_2, p_3\}$ is caused by the transitions t_2 and t_3. We don't know that p_2 is caused by of t_2

for example. It could be that firing a step leads to different configurations and therefore to different effects.
In the new notation (figure 15.a) we see that the configuration $\{p_2, p_3\}$ is caused by the step $\{t_2, t_3\}$ and that is exactly what we want.

Theorem 4 (SN labeled step occurrence nets). *Let* $SN = (P, T, F, M_0)$. *If* $OCS = (B, E, R, \Phi), \Phi(E) \subseteq \mathcal{P}(T)$, *is a* labeled step occurrence net *of* SN *according to definition 42 then:*
(1) OCS *is acyclic. and*
(2) $\#({}^\bullet p) \leq 1$ *and* $\#(p^\bullet) \leq 1$ *for all* $p \in B$.

Proof. Every step in SN consists of 2 or more transitions sharing places or not sharing places (steps with 1 transition are just single transitions). If they do not share places then we have just two concurrent transitions without any branching at all in our OCS. If they do share places then we can create a *new* combined transition that models the step by just taking the combined effect of the transitions. OCS is acyclic because we add new places every time a transition occurs. We never take places already in OCS as an output place, so no cycles can be made. □

By creating occurrence nets this way (definition 42) we will not get branching places. We also have the nice side effect that a lot of the EN theorems about occurrence nets now also hold for contact-free set-net occurrence nets, because we can unfold them almost the same way. As another side effect we get immediately the information if a step has taken place or whether transitions fired independently.
We want that an occurrence net displays only possible histories of the set-net and one of those histories has to be the one we fired to create the occurrence net. This is now the case for contact-free set-nets. However we know that contact-free is no normal form for set-nets. For non contact-free set-nets this construction does not work.
We do not see the causal relations between the transitions inside the step any longer. Steps are a very basic and an important aspect of set-nets. We could see the firing of steps as an elementary behaviour. We can even say we only fire sets in set-nets where those sets consist of one or more transitions. Recall that those steps could have different behaviour then the single transitions, that's why we want to treat them differently.

When a step with shared places happens in the occurrence net we do not know if the transitions after this step are being caused by the step or a subset of the step. Since we do not know that the step is the minimal set to enable these transitions, we cannot say this step has to happen before the transitions he causes. We therefor introduce an already known relation: *weak causality* [7]. Normal causality (denoted by $U \prec t$) means that U need to occur before t.

Definition 43 (Weak causality). *Let $SN = (P, T, F, M_0)$ a set-net, $U \subseteq T$ a step and $t \in T$ a transition. U precedes t in a weak causal way (denoted by $U \sqsubset t$) if:*

$U \prec t$ or U occurs at the same time as t. □

Definition 44 (SN inside step relations). *If $SN = (P, T, F, M_0)$, $OCS = (B, E, R, \Phi)$ be a* labeled step occurrence net *of SN, U a step in SN with a labeled transition in OCS, v a transition or step preceding U then:*

for all $t \in T$, $t \in U \wedge v \prec U \Rightarrow v \prec t$

for all $t_1 \in U, t_2 \in U, t_1 \neq t_2 : t_1 \sqsubset t_2 \wedge t_2 \sqsubset t_1$

□

Transitions inside steps have a weak causality because we only know that they can fire simultaneously while it might be that they can also fire after one another.

5.2 Occurrence nets for non contact-free set-nets

The semantics of our new occurrence nets (created by construction 42) need to be correct (*soundness* of the occurrence net). This means that if we create an occurrence net by our construction method then we get an occurrence net with only correct histories according to the original set-net. One of the histories has to be the one we fired to create the occurrence net.
Firing a transition in an occurrence net works the same way as firing transitions in set-nets (same rules apply).

Our proposed occurrence net model so far works for contact-free set-nets, however there is a big problem we did not solve yet: contact.
We did not handle contact and that is a problem as we will see in the next examples. (Figure 16)

For EN systems we already know that we can only create an occurrence net if the elementary net is contact-free (Definition: 33). We can make these contact-free EN systems by complementing places. Recall that complementing places does not work in set-nets. We cannot make a given set-net contact-free because the contact situations give us different behaviour of the net.

Contact violates the soundness of our occurrence net. To deal with this problem we need to modify our occurrence net structure. When contact happens in the set-net we need to preserve the specific firing order in our occurrence net without making it to strict. We can do this by creating new, so called, *shadow places*. Places that do not really count to the configuration but who indicate that contact happened due to this particular place. When we then fire a transition from the real place we have to take all tokens from their shadow places as well. This way we force the firing sequence to be correct in the occurrence net.

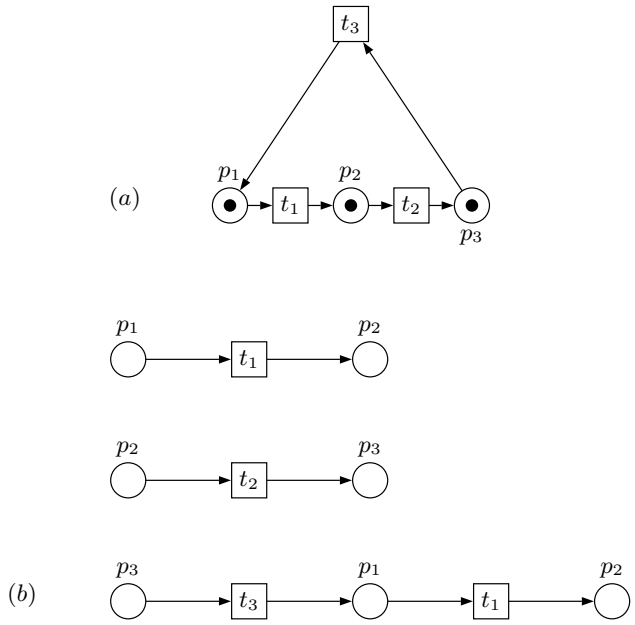


Fig. 16. Proposed occurrence net (b) for $\{t_1\}\{t_2\}\{t_3\}\{t_1\}$ of the set-net (a) with contact is not sound!

Definition 45 (slice). *Let $OCS = (B, E, R, \Phi)$ be a* labeled step occurrence net.
A slice is a maximum set of places not connected to each other by R^*. □

We need the following additional rules for creating our occurrence nets:

- When we fire a step U with $t \in U$ at C with $p \in t^\bullet \cap C$ then we create a shadow place $S(p_k)$ in our labeled occurrence net where $\Phi(S(p_k)) = p$ and add a directed arc from the step to the shadow place. (k is a number that gets increased every time we add a new place.
- When we fire a transition t at C with input places ${}^\bullet t \subseteq P$, t has B_t as input places in the labeled occurrence net with $\Phi(B_t) = {}^\bullet t$ and $S(B_t)$ as shadow input places in the occurrence net denoted by a dashed directed edge.

The construction for our labeled step occurrence nets will now be as follows:

Definition 46 (SN labeled step occurrence net construction). *Let $SN = (P, T, F, M_0)$ be a set-net. Let $OCSN_0 = (B_0, E_0, R_0, \Phi_0)$ be our initial labeled occurrence net. Let $V_1, V_2 \ldots V_n$ be the step firing sequence we use for our construction and $OCSN_k$ the labeled occurrence net we have after firing step U_k*

We start with the occurrence net $OCSN_0$ where Φ is our labeling function between $OCSN$ and SN. $\Phi(B) \to P, \Phi(E) \to \mathcal{P}(T)$.

$E_0 = R_0 = \emptyset$. $\forall p \in M_0 : b_{p_0}, B_0 = \{b_{p,0} : p \in M_0\}$ $\Phi(b_{p_0}) = p$

now for every step $U = V_k$ from our step firing sequence we fire in SN at the current configuration C:

(1) $U \subseteq T, U \neq \emptyset$. We first split U

in parts that are independent from each other by using definition 41.

(2) For each dependent subset U_l we add a new labeled transition e to E and extend Φ with $\Phi(e) = U_l$.

We then apply for all substeps U_l of the partition each step below (3-7).

(3a) We find all places b where $\Phi(b) = p, p \in {}^\bullet U_l$ let this be P_1.

(3b) We find all places $S(b)$ where $\Phi(S(b)) = p, p \in {}^\bullet U_l$ let this be SP_1.

(4a) We take all places $P_2 \subseteq P_1$ such that $P_2^\bullet = \emptyset$.

(4b) We take all places $SP_2 \subseteq SP_1$ such that ${SP_2}^\bullet = \emptyset$.

(5a) $\forall b \in P_2$ we add (b, e) to R (directed edges).

(5b) $\forall S(b) \in SP_2$ we add $(S(b), e)$ to R (dashed directed edges).

(6a) $\forall p \in U_l^\bullet \wedge p \notin C$, we add new b to B.
and extend Φ with $\Phi(b) = p$, let this set be P_3.

(6b) $\forall p \in U_l^\bullet \wedge p \in C$, we add new $S(b)$ to B.
and extend Φ with $\Phi(S(b)) = p$, let this set be P_4.

(7a) $\forall b \in P_3$ we add (e, b) to R (directed edges).

(7b) $\forall S(b) \in P_4$ we add $(e, S(b))$ to R (directed edges).

□

We also need some additional rules for firing transitions in our occurrence nets:

- We can only fire a transition if all its input places and shadow input places are filled. (so we threat shadow places like normal places if it comes to enableness)
- A transition cannot fire to a shadow place $S(p)$ if p is not in the current slice.
- We only allow fire sequences in our process net that begin with the initial marking and fire until the end marking (last possible marking of the occurrence net) has been reached.

The shadow places in the occurrence net (figure 17) force us to fire the sequence in the occurrence net that we fired in the original set-net to create the occurrence net. This assures us that this is a correct history of the net. We lose some information about the relations between these transitions, that is why we create dashed edges between the shadow places and the transitions that "depend" on them. The soundness of our occurrence net can now be proved.

Theorem 5 (`SN` occurrence net soundness).
The occurrence net we get from a firing sequence in a set-net is sound.

Proof. Let $SN = (P, T, F, M_0)$ be a set-net and $OCSN = (B, E, F, \Phi)$ a labeled occurrence net for SN. Every configuration of $OCSN$ corresponds to a configuration in SN. $\Phi(B) \to P$. If we fire a transition e in $OCSN$ we can also fire

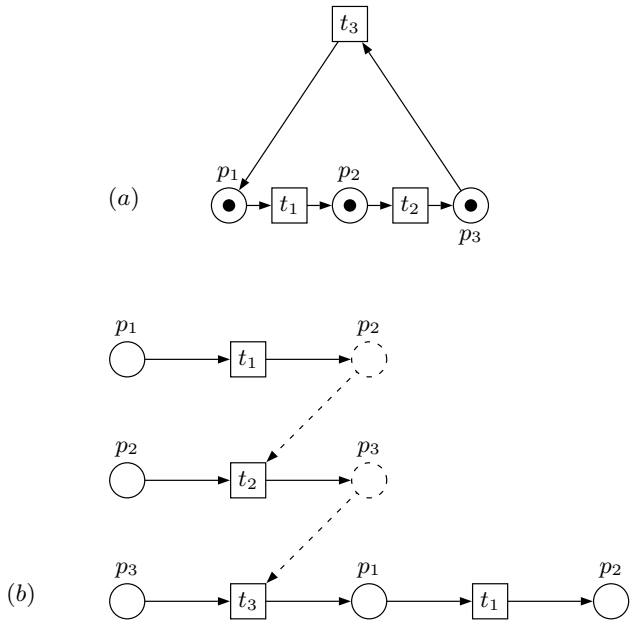


Fig. 17. The set-net (a) with its occurrence net (b) with shadow places.

the corresponding transition or step $\Phi(e)$ in SN. This is because the transition e in $OCSN$ must have all input places (also shadow places) filled and therefor all input places of the corresponding step or transition in SN are filled as well. We can fire the step firing sequence we used to create $OCSN$ from the initial marking because of our construction. This step firing sequence is valid in our $OCSN$ and because of all this the occurrence net is sound. □

5.3 Partial orders

We can create a partial ordering of the SN occurrence net the same way we did by EN occurrence nets. Just by removing the places in the occurrence net. We then need to specify what to do with shadow places. We remove the shadow place and use a dashed directed edge from the transition before the shadow place to the transition after the shadow place. This way we again enforce the fire sequence to be correct while we still have the information that contact did happen and that the causality between these transitions are not strict.

Theorem 6 (SN partial order correctness).
The partial order *we get when we erase all places from our labeled step occurrence net is* correct.

Proof. The fire sequences that can be made from the partial ordering are all valid fire sequences of the occurrence net and set-net. All the fire sequences

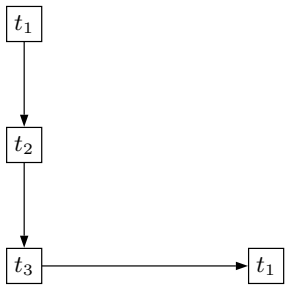


Fig. 18. Partial order of the occurrence net of Figure 17.

in the partial ordering correspond one to one to a sequence of the occurrence net since every place is now exactly connected to only one input transition and only one output transition from the occurrence net. We already proved that our occurrence net was sound for set-nets so our ordering works correct. □

Theorem 7 (SN partial orders to occurrence nets). *Given a partial order from a set-net we can convert the order back to an occurrence net.*

Proof. If we have the partial ordering we can add the input and output places between the arcs again for every transition. Between a dashed arc we create a shadow place with an input edge and a dashed output edge. The produced occurrence net is isomorph to the original occurrence net. □

Given all these we can conclude that the semantics of our occurrence net model are correct for set-nets. But if a set-net has contact somewhere we lose some information about the specific causal relations between the affected transitions. We can however regain this information if more occurrence nets with different sequences are provided or being made.

6 Extended set-nets

Extended set-nets are set-nets with additional types of arcs: *Inhibitor* and *activator* arcs. These arcs added to the normal set-nets to check whether a certain event can occur or not (in biological reactions there are catalysts (activators) and inhibitors). In the next section we define these extended set-nets and prove that they add something useful to the set-net model.

6.1 Definitions

An *extended set-net* is a set-net with *activator* arcs and *inhibitor* arcs.

Definition 47 (extended set-net). *An* extended set-net *is a tuple:*

$$SNIA = (P, T, F, Inh, Act, M_0)$$

where (P, T, F, M_0) *is a set-net and*

$$Inh \subseteq P \times T$$
$$Act \subseteq P \times T$$

are respectively the sets of inhibitor *and* activator *arcs.* □

An inhibitor arc ends with a small open circle, while an activator arc ends with a small black circle.
For a transition t we use:

$$^\circ t = \{q \mid (q,t) \in Inh\} \text{ and } ^\blacklozenge t = \{q \mid (q,t) \in Act\}$$

to denote the inhibitor and activator places of t.

Inhibitor arcs and *activator* arcs have a special effect on the enabling of transitions. Places with an activator arc to a transition t have to be filled and all places with an inhibitor arc to t have to be empty in order for t to be enabled.

Definition 48 (enabled extended set-net).
Let $SNIA = (P, T, F, Inh, Act, M_0)$ *be an extended set-net,* $C \subseteq P$*, a non-empty set* $U \subseteq T$*.* U *is enabled at configuration* C *if*

$$^\bullet U \cup ^\blacklozenge U \subseteq C \quad \textit{and} \quad ^\circ U \cap C = \emptyset;.$$

The firing rule of enabled steps and transitions in extended set-nets are the same as those of normal set-nets, inhibitor and activator arcs do not affect them.
Thus, inhibitor arcs check if there is no token in a place connected to it, while activator arcs check if there is a token in its connected places. Both arcs do not affect the tokens in the connected places.

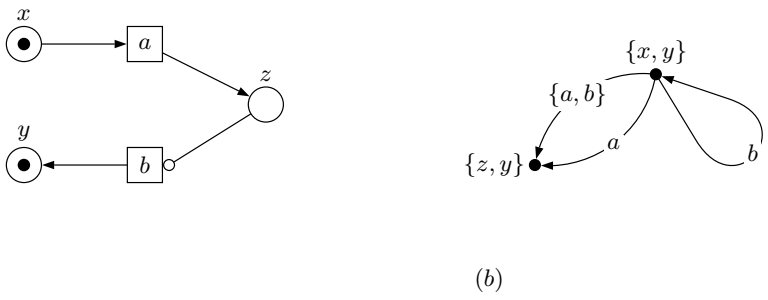
6.2 Expressive power

Why would we use inhibitor arcs and activator arcs?
If we can use normal set-nets only, things would be much simpler as you can imagine. If we can prove that normal set-nets have the same expressive power as extended set-nets we do not need inhibitor arcs and activator arcs.

The first thing we may try is to get rid of all the inhibitor arcs by complementing the places with these arcs and give their complements an activator arc. This works only when complementing does not affect the behaviour of the net. As we have seen in Theorem 3 this only works when the place with an activator or inhibitor arc has only output arcs or only input arcs. Every set-net that has inhibitor or activator arcs only to places that can be complemented, can be replaced by a set-net with respectively activator or inhibitor arcs which can be simulated with the original set-net. This is only a very small set of nets, so not

very useful.

We may try is to see whether there is a $SNIA$ with a configuration graph that does not exist with only normal input and output arcs. This would prove that inhibitor arcs really add something to the net system.



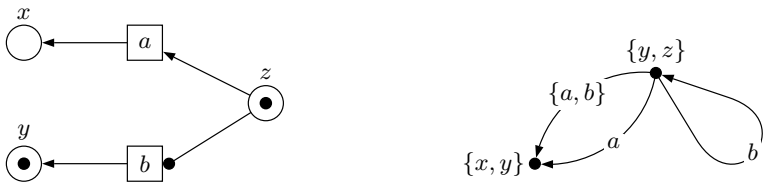
(a) (b)

Fig. 19. an extended set-net (a) with its configuration graph (b).

Figure 19 shows an extended set-net with an inhibitor arc that has a rather special configuration graph. Transition b can fire multiple times from the initial configuration but when a or the set $\{a, b\}$ occurs, b will be disabled. If we try to do this with a normal set-net we fail to achieve this configuration structure. The set in which b itself is an element, could never disable b in a normal set-net. This is because b alone does not disable itself (in a normal set-net b would have a self-loop). This proves that inhibitor arcs really extend set-nets as a model.

We can also give an extended set-net with a special configuration graph to prove that activator arcs just like inhibitors add something to our set-net system (figure 20).

However Figure 19 and Figure 20 can be converted into each other by using complementation since the places that use the inhibitor arc respectively activator arc only use input or only use output arcs.
However it is very simple to create a similar net that cannot be complemented (because the place with inhibitor or activator arc has input and output edges) and still has a configuration graph that normal set-nets cannot have (Figure 21). However this does not prove that both arc types are needed for the expressive power. One extended arc type might be enough but this will not be proven in this paper.



(a) (b)

Fig. 20. An extended set-net (*a*) with its configuration graph (*b*).

We can conclude that there exists extended set-nets with a CG where there is no normal set-net able to simulate this extended set-net.

7 A comparison of set-nets and EN systems

In this section we investigate whether set-nets can be converted into EN systems and extended set-nets into EN systems with inhibitor and activator arcs. This should then make it possible to say more about equivalence and the behaviour of set-nets and their structure with respect to EN and other net systems.

With converting we mean that the set-net or extended set-net can be simulated by an EN system or ENIA (EN with inhibitor and activator) system. With simulation we mean that the valid step firing sequences of the SN or SNIA, correspond one to one with a valid firing sequence of the EN or ENIA system. For example, when a step firing sequence for a SN is $\{t_1\}\{t_2t_3\}\{t_4\}$ then the corresponding sequence that has to be correct for the EN system will be: $t_1t_{23}t_4$ where t_{23} is a single transition corresponding to the step $\{t_2t_3\}$. If this converting is vice versa (from EN to SN) then they have to accept exactly the same step fire sequences.

Theorem 8 (Contact-free set-net to EN). *Let $SN = (P, T, F, M_0)$ be contact-free. Then there exists an EN system that is able to simulate the given set-net.*

Proof. For all $U \subseteq T$ and reachable C in SN we have if ${}^\bullet U \in C$ then ${}^\bullet U \subseteq C \wedge U^\bullet \cap C = \emptyset$. We copy SN as the new EN system and in addition for every U, we have a new transition t_U with the same input and output places as U. The

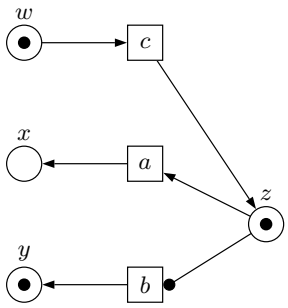


Fig. 21. Extended set-net where the activator arc cannot be replaced by complementing

new EN system simulates the original SN system because every valid step firing sequence $\{U_1\}\{U_2\}\ldots\{U_n\}$ for SN can be converted to a valid firing sequence $t_{U1}t_{U2}\ldots t_{Un}$ for EN. □

Theorem 9 (Contact-free extended set-net to ENIA).
Let $SNIA = (P, T, F, Inh, Act, M_0)$ be contact-free. Then there exists an EN system with inhibitor and activator arcs ($ENIA$) that is able to simulate the given set-net.

Proof. Let us use Theorem 8. Since activator and inhibitor arcs are used in the same manner for both net systems, we can just copy these arcs directly from our extended set-net to our EN system. □

Theorem 10 (Contact-free and conflict-free EN system to SN). *Let $EN = (P, T, F, M_0)$ be contact-free and conflict-free EN system. Then there exists a set-net that is able to simulate the given EN system.*

Proof. We can copy the net into a new set-net that has the same net structure and exactly the same behaviour as the original EN system. Since there is never a conflict between transitions, only those steps can occur that were concurrent steps in the original EN system. And because the net is also contact-free, all enabled transitions in the set-net will be the same as in the original EN system. □

Theorem 11 (Contact-free and conflict-free ENIA to SNIA). *Let $ENIA = (P, T, F, Inh, Act, M_0)$ be contact-free and conflict-free. There exists an extended set-net ($SNIA$) that is able to simulate $ENIA$.*

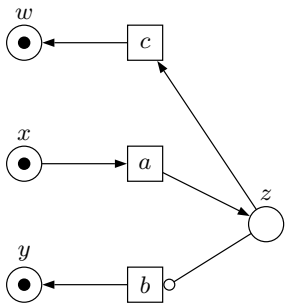


Fig. 22. Extended set-net where the inhibitor arc cannot be replaced by complementing

Proof. Let us use Theorem 10. The inhibitor and activator arcs affect each kind of net system the same way, so we can just copy them as well. □

Thus set-nets can be converted to EN systems if they are contact-free. And if a SN system is contact-free and conflict free it behaves exactly as an EN system.

8 Related boolean net systems

In this section we briefly refer to two boolean net models that appear to be related to set-nets in that they have different enable rules than EN systems.

In the early sixties Petri nets were founded by Carl Adam Petri [8]. Some of the early Petri nets where called condition event nets (C/E). These nets where actually the same as the now known EN systems. The only difference was that they also looked at case classes. A case class is actually a large set of all case graphs (now called configuration graphs (Definition 19) or reachability graphs) of a given net without a specific begin marking. It is unclear why we don't look at these case classes anymore but it is neither clear why they did this before. Because these nets are not used anymore and almost identical to EN systems we did not pay further attention to them in this paper.
Another related net model is the flipflop system [10]. These systems are like EN systems but in addition they have some different type of arcs. They consist of four relations : $+, -, \times, =$ between places and transitions. The $+$ and $-$ are the normal EN relations where $+$ gives the place a token and $-$ removes one token. The $=$ means that there is no arc and the $\times$ either removes a token or adds a token depending if there already is a token or not (the $\times$ gives the behaviour as in a flipflop). These flipflop systems where created to solve the synthesis problem

in polynomial time. These flipflop nets can be seen as an extension of the EN system. Set-nets differ from EN systems because of their enabling and firing rules while these flipflop nets differ from EN systems by structure (and firing rules). Their enabling rule is the same as the rule for EN systems. $\times$ arcs do not affect the enabling since they just swap tokens either if there is a token present or not. So this net system is related to set-nets because it has different enabling rules than EN systems by means of the $\times$ arcs, but they are still very different from set-nets because these flipflop systems do have contact and do not have steps as in set-nets.

9 Concluding remarks

We have investigated the new set-nets, a Petri net model prior introduced to model reaction systems following already existing theories for EN systems inspired by Elementary Net Systems [9]. The main difference with existing net classes is the occurrence of steps with shared input that messes up our notions of conflict and concurrency as we know them from EN and PT systems. Contact is also a big difference because we do not have the notion of contact in set-nets. We have proposed a theory for the concurrent and causal behaviour of set-nets as well as a method to derive semantically correct occurrence nets and partial orderings. We focused mostly on set-nets but also on an extended set-nets with inhibitor and activator arcs. We proved that this extended set-net model has more expressive power then our standard set-net model. The rewriting of our notions and findings with respect to set-nets in this paper for this extended set-net model would be future work.

Set-nets deserve their place in the full Petri net theory because they have proven to be very different from EN and PT systems while very useful for the modeling of biological reactions. Because of the ability to fire steps with shared input and output places, the set-net system is quite powerful. With only a few transitions, many firing combinations arise and all may give different behaviour and configurations.

The phenomenon causing the differences between set-nets and PT/EN systems is *semi-unorderedness*. Semi-unordered steps are proved to be quite useful in understanding set-nets and the differences between set-nets and other Petri net systems. Set-nets without these steps and without self-loops can be simulated by EN systems.

Semi-unordered steps lead to problems for the creation of occurrence nets and partial orderings. We have proposed a model to construct occurrence nets for SN by using the new concept of *shadow places and arcs*. Shadow places should ensure that the model would be sound, in the sense that every occurrence net associated with SN allows only valid step firing sequences. From these sound

occurrence nets we could then derive partial orderings that resemble the causal relations between transitions and steps in our set-net.

While some questions have been answered about set-nets in this paper, there are many more to be answered in the future. For extended set-nets a lot of problems have not yet been solved and interesting questions could be: Has SN with inhibitor arcs or SN with activator arcs the same expressive power as SN with both activator and inhibitor arcs? How can we create processes for these extended set-nets and do the inhibitor arcs and activator arcs cause additional problems?
This and other questions give us a nice challenge in future works. The beginning of a theory for set-nets has been written but it is far from finished.

References

1. E.Badouel and P.Darondeau: Theory of regions. LNCS 1491 (1998) 529–586
2. L.Czaja: A Calculus of Nets. Cybernetics and Systems 29 (1993) 185–193
3. P.De Bra, G.J.Houben and Y.Kornatzky: A Formal Approach to Analyzing the Browsing Semantics of Hypertext. Proc. CSN-94 Conference (1994) 78–89
4. J.Desel, W.Reisig: Place Transition Petri Nets, LNCS 1491 (1998) 122–173
5. A.Ehrenfeucht and G.Rozenberg: Reaction Systems. Fundamenta Informaticae 76 (2006) 1–18
6. J.Kleijn, M.Koutny and G.Rozenberg: Modelling Reaction Systems with Petri Nets Proceedings of the International Workshop on Biological Processes and Petri Nets (2011) 36–52
7. J.Kleijn and M.Koutny: Formal Languages and Concurrent Behaviours, New Developments in Formal Languages and Applications. Studies in Computational Intelligence (2008) Springer Berlin / Heidelberg 125–182
8. C.A.Petri: Kommunikation mit Automaton, Bonn: Institut fur Instrumentelle Mathematik, Schriften des IIM Nr. 2, (1962). In German.
9. G.Rozenberg, J.Engelfriet: Elementary Net Systems. LNCS 1491 Petri Nets (1996) 12–121
10. V.Schmitt: Flip-flop nets, 13th Annual Symposium on Theoretical Aspects of Computer Science Grenoble, France, February 22–24, 1996 Proceedings
11. A.Yakovlev, M.Kishinevsky and A.Kondratyev and L.Lavagno: OR Causality: Modelling and Hardware Implementation. Application and Theory of Petri Nets (1994) 568–587