7. High Level Petri-Nets

Definition 7.1

A **Net Type** is determined if the following specification is given:

1. **sort of tokens** (e.g. indistinguishable (black), coloured, structured, ...),
2. **sort of labelling of net elements** (e.g. multiplicities for arcs, logical expressions for firing of transitions, ...),
3. **sort of evaluation of net elements** (e.g. capacities for places, priorities for transitions, ...),
4. **firing rule** (e.g. with (strong firing rule) or without (weak firing rule) consideration of capacities of post-places, ...),
5. **firing strategy** (e.g. single or more (maximal step) transitions fire simultaneously, firing no facts, ...)

In the following chapters we learn something about new net types which can be distinguished from condition event nets and place transition nets by a higher modelling power (null test of unbounded places).

7.1 Inhibitor Nets

Until now we have modelled the occurrence of an event under a side condition using an additional place. So in the following net the place $p_2$ is used for a side condition for transition $t$. Transition $t$ can only fire if place $p_2$ contains a token.

![Diagram of Inhibitor Nets](image)

**Abb. 7.1.1:** Firing of a transition with side condition
For many applications often used is the "dual" side condition:

"t is enabled if the pre-place of t contains not a token."

(see Zuse, Konrad: Petri-Netze aus der Sicht des Ingenieurs, Vieweg-Verlag, Braunschweig, 1980).

According to Jensen this is called an "unmarked" side condition. For the graphical representation of this side condition are used an inhibitor arc. Instead of an arrowhead it is used a little circle: 〇.

\[
\begin{align*}
\text{t is not enabled} & \quad \text{t is enabled} \\
& \quad \text{because p2 contains a token} \quad \text{because p2 contains not a token}
\end{align*}
\]

Fig. 7.1.2: Firing of a transition with dual side condition

Such nets with inhibitor arcs are called inhibitor nets.

To define the dynamic behaviour of these nets additional to the new net structure a special firing rule is needed which considers the dual side condition.

It is possible to proof that there doesn’t exist a finite Petri Net which models the reader writer problem if the number of involved processes is unknown or unbounded.

(We already know that a null test is not possible for unbounded places in a place transition net.)

We show now that an inhibitor net allows such a modelling.
Fig. 7.1.3: Reader-Writer-Problem as inhibitor net

Remark: Inhibitor nets with inhibitor arcs of bounded pre-places can be transformed in a place transition net introducing co-places:

Fig. 7.1.4: Transformation of a bounded inhibitor net in a place transition net
7.2 Selfmodifying Nets (SM-Nets)

A next important class of Petri Net for the modelling are the so called selfmodifying nets.

**Definition 7.2.1**

Selfmodifying Nets are place transition nets with extended inscription of arcs. As a inscription is allowed not only a positive natural number but also place names of the net. Different copies of such arcs are allowed. An arc with an inscription of a place p has the multiplicity m(p) if place p contains m(p) tokens. With such an extension of definition of multiplicity of arcs holds the common firing rules.

Hence a SM-net can be considered as a PT-net which dynamically change its structure. Therefore exists a causal relation between places and transitions which are not reflected by arcs but by inscriptions. Selfmodifying nets violate principles of net theory. The advantage of such nets however is the comfortable modelling and the power of this net class. Also with these nets is it possible to implement the null test for unbounded places.

Our well-known reader writer problem for arbitrary many reader processes can be modelled as SM-net, and it looks then like:
Fig. 7.2.1: Reader writer problem as selfmodifying net

If we compare together both solutions of the reader writer problem for inhibitor and selfmodifying nets ones detect easily the differences. We want now to present a general method for this.

To transform an inhibitor net in a selfmodifying net with the same power make following steps:

1) Create an additional place $p_0$ which is unmarked (clean) in the initial marking and no transition fires a token to $p_0$ thus it stays always clean.

2) For each inhibitor arc from place $p$ to transition $t$ introduce a SM-arc from $p_0$ to $t$ and inscribe this arc with multiplicity $p$.

3) Erase all inhibitor arcs.

This method is demonstrated in figure 7.2.2:
It is possible to prove that the power of inhibitor nets is equivalent with the power of selfmodifying nets and also with the power of priority nets which are discussed in chapter 7.4.

### 7.3 Place Transactor Nets

**Definition 7.3.1**

*Place transactor nets* are selfmodifying nets which are using the *conservative firing rule*:

A transition is only enabled if after firing of this transition the number of tokens is not changed.

**Remark:** Therefore all place transactor nets are a priori bounded.
7.4 Priority Nets

Nets with priorities have a priority relation $\gg$ over the set of transitions. This relation is irreflexive and transitive (a strict or strong order). If holds $t1 \gg t2$ then it is said that $t1$ has a higher priority than $t2$. The firing rule is modified such that in every marking only the enabled transition with the highest priority can fire (or if the priority of some transitions is equal than one of they is randomly chosen). Also with this net type it is possible to implement the null test for unbounded places.

We consider now the net from figure 7.4.1. If this partial net have to implement the null test then the token from place $p$ should only be transferred to place $p1$ if place $test$ contains a token. In the considered net the transition $t0$ is also enabled. Because transition $t1$ has a higher priority than transition $t0$ the token in test would be transferred to place $p1$. If place test is not marked then only transition $t0$ is enabled and the token arrive to place $p0$. The net in fact implements the null test.

![Figure 7.4.1: Null test for priority nets](image)

From this facts follows that every nontrivial problem for priority nets is undecidable. For instance the query for boundedness is not decidable. The general method for analysing such nets is to enumerate the reachability set and to hope that this set is finite. The universality of a description language can provide to a situation where a priori it is not possible to analyse something in an effective way. How to model the reader writer problem using priorities shows the following figure:
Fig. 7.4.2: Reader writer problem as priority net
7.5 Timed Petri Nets
In the current and the following chapter we want concern us with nets which are using 
time. Now we assume that there exists in the modelled system a time scale. 
The concurrency of the occurrence of events should be detected with sufficient exactness. 
Timed Petri nets were introduced for simulation reasons. 
The time for different net elements (like places, transitions, arcs and tokens) represents 
certain time for waiting, time for processing, time for transport etc. 
A special relevance plays the time nets for modelling of communication protocols. With 
respect to the additional evaluation of net elements with time the firing rule for transitions is 
also changed. 
At the beginning we study nets where only the transitions are labelled with time. The The 
resulting net type was primarily studied by C. RAMCHANANDAI (Analysis of Asynchronous 
1974.) and lead us to the timed Petri Nets.

Definition 7.5.1
A Timed Petri-Net (germ. Schaltddauer-Netz) (N,D) (abbreviated as TD-net) is a Petri net 
N where N=(P,T,F,W,m0) is a Petri net and D is a mapping which assigns every transition 
t a positive rational number the so called time duration of t. 
The firing of a transition occurs using a firing rule which fires a transition instantly i.e. if a 
transition is enabled in a moment x (by changing marking) then the transition must fire in 
this moment. Only if the transition is in a dynamic conflict or if it fires then it isn't done. In 
every point of time it is fired a so called maximal step.

Definition 7.5.2
A maximal step is a set U of transitions which are conflict free enabled in marking m and 
there doesn't exist a set of transitions V which are also conflict free enabled and contains 
U as a proper subset. The collective firing of a conflict free enabled set U of transitions is 
defined by firing every transition which belongs to U. 
The introduced definition of firing in maximal steps is explained in the following example.

![Diagram](attachment:image.png)

Fig. 7.5.1: Firing in maximal steps
In this example it is obvious that all transitions are enabled therefore $U_1 = \{t_1\}$, $U_2 = \{t_2\}$, and $U_3 = \{t_3\}$ are enabled in the given marking. This also holds for $U_4 = \{t_1, t_3\}$ but not for $\{t_1, t_2\}$, $\{t_2, t_3\}$ or $\{t_1, t_2, t_3\}$. Hence there exists two maximal steps namely $U_2 = \{t_2\}$ and $U_4 = \{t_1, t_3\}$. In a marking which differs from the marking above such that in place $p_2$ are contained 2 tokens then the maximal steps are $\{t_1, t_2\}$ and $\{t_1, t_3\}$.

Without restriction of generality we can assume that the times $D(t)$ are natural numbers. It needs only a multiplication of these times with the least common multiple of the denominators of the fractions which results from the corresponding rational numbers. Such a multiplication denotes a dilation of the time scale by a determined factor.

**Definition 7.5.3**

Let $(N,D)$ a TD-net. A pair $[m,u]$ is called a **state of $(N,D)$** if $m$ is a marking of $N$ and $u: T \rightarrow \mathbb{N}$ a mapping which every transition $t$ of $N$ assigns a natural number $u(t) < D(t)$ its **switching state**. $[m_0,u_0]$ is an **initial state of $(N,D)$** where $u_0(t) = 0$ holds for all $t \in T$.

Thus a state of a timed Petri net is given by the actual marking and the switching states of the transitions.

The switching state of a transition is described by the vector $u$ which is also called **clock positions** (germ.: *Uhrenstellung*). This state defines for every transition the time which was passed since the beginning of the firing operation or it is zero if this transition doesn’t fire in this moment.

Interesting for the analysis are only such points of time where the state of $[N,D]$ can be changed i.e. the moments if tokens are added to places or subtracted from places by firing of transitions.

The behaviour of timed Petri nets is demonstrated using the following example.

![Timed Petri net](image)

**Fig 7.5.2:** Timed Petri net

The transitions of the net from figure 7.5.2 have the following time durations according to table 7.5.1:

<table>
<thead>
<tr>
<th>$t$</th>
<th>$t_1$</th>
<th>$t_2$</th>
<th>$t_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D(t)$</td>
<td>4</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

**Table 7.5.1:** Time durations for transitions of net from fig. 7.5.2
The initial state is \([m_0,u_0]=[(0,0),(0,0,0)]\). The value in column \(x\) of the following table characterises the elapsed time since the initial state.

<table>
<thead>
<tr>
<th>(x)</th>
<th>state ([m,u])</th>
<th>maximal steps</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(S_0= [(0,0),(0,0,0)])</td>
<td>({t_1})</td>
</tr>
<tr>
<td>1</td>
<td>([(0,0),(1,0,0)])</td>
<td>∅</td>
</tr>
<tr>
<td>2</td>
<td>([(0,0),(2,0,0)])</td>
<td>∅</td>
</tr>
<tr>
<td>3</td>
<td>([(0,0),(3,0,0)])</td>
<td>∅</td>
</tr>
<tr>
<td>4</td>
<td>(S_4= [(1,1),(0,0,0)])</td>
<td>({t_1,t_2,t_3})</td>
</tr>
<tr>
<td>4.5</td>
<td>([(0,0),(0.5,0.5,0.5)])</td>
<td>∅</td>
</tr>
<tr>
<td>5</td>
<td>([(1,0),(1,1,0)])</td>
<td>∅, (t_1) and (t_2) are still firing</td>
</tr>
<tr>
<td>6</td>
<td>(S_6= [(1,0),(2,0,0)])</td>
<td>({t_2})</td>
</tr>
<tr>
<td>7</td>
<td>([(0,0),(3,1,0)])</td>
<td>∅</td>
</tr>
<tr>
<td>8</td>
<td>(S_8= [(1,1),(0,0,0)]=S_4)</td>
<td>({t_1,t_2,t_3})</td>
</tr>
</tbody>
</table>

Table 7.5.2: States and maximal steps for example from fig. 7.5.2

7.6 Time Nets

The time nets were introduced by P.M. Merlin in the article: A Study of the Recoverability of Computing Systems. Irvine; Univ California. Dept. of Information and Computer Science, TR 58 (1974).

Definition 7.6.1
A **Time Net** (germ.: **Zeit-Netz** or **Schaltintervall-Netz**) \((N,I)\) is a Petri net \(N=(P,T,F,W,m_0)\) where using a mapping \(i\) every transition \(t\) is assigned a rational interval \(i(t)=[eft(t),lft(t)]\). Thereby the lower interval endpoint \(eft(t)\) (the earliest fire time) is the time period which at least must elapse between the enabling of \(t\) and firing of \(t\) and \(lft(t)\) (the last fire time) is the time period which at most must elapse between the enabling of \(t\) and firing of \(t\). Firing a transition makes no costs of time. The values \(eft(t)\) and \(lft(t)\) are rational numbers with property \(0 <= eft(t) <= lft(t)\).

Definition 7.6.2
A **state** \([m,c]\) of a time net is given by the actual marking \(m\) and a T-vector \(c\), the **clock status** (germ.: **Uhrenstellung**) which has the value * for every transition \(t\) which is not enabled in \(m\) and for enabled transitions it shows the time which was elapsed since enabling.
Fig. 7.6.1: Time net

Fig. 7.6.2: Reachability graph for net from fig. 7.6.1

Remark: Liveness and boundedness of time nets are undecidable.
7.7 Coloured Petri Nets

This net type will be discussed only informative. Now exist individual, so called *coloured tokens* and the inscriptions are *mappings*. The transitions can fire in different *modi*.

As an example for a coloured Petri net we take our well-known reader writer problem:

![Diagram of a coloured Petri net](image_url)

**Fig. 7.7.1:** Reader writer problem as coloured Petri net

- **pool**
- **to cs**
- **wait**
- **take**
- **help**
- **cs**
- **put**

**t**: function take
- if color □ in wait then consume 1 token (●) from help
- if color ◦ in wait then consume 2 tokens (●) from help

**u**: function put
- if color □ in cs then put 1 token (●) to help
- if color ◦ in cs then put 2 tokens (●) to help

*It is possible that two readers can read in parallel.*
- □ - reader
- ◦ - writer

**cs** - critical section

**id**: identity function (transport of a coloured token)
7.8 Predicate Transition Nets

A predicate transition net has also individual tokens like a coloured Petri net. They are the so called attributes (colours) (e.g. (a,b)). Hereby every place consist a number (multiplicity) for every kind of attributes. The arc inscriptions are formal sums (e.g. \((x_1,x_2)+(x_2,x_3)\)) and characterises the type of subtracted or added attributes. Additional exist inscription for the transitions the so called predicates.

It holds the following firing rule:

If enough attributes of needed sorts are contained in the input places, and if the predicate of the transition is true then it is realized the firing accordingly to the arc inscriptions. Thereby the formal sums are allocated with concrete attributes.

Fig. 7.8.1: Firing of a predicate transition net

Fig. 7.8.2: n philosophers problem as predicate transition net