<table>
<thead>
<tr>
<th>Title</th>
<th>On Computational Power of Jumping Petri Nets (Semigroups, Formal Languages and Computer Systems)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author(s)</td>
<td>TIPLEA, Ferucio Laurentiu</td>
</tr>
<tr>
<td>Citation</td>
<td>数理解析研究所講究録 (1996), 960: 165-177</td>
</tr>
<tr>
<td>Issue Date</td>
<td>1996-08</td>
</tr>
<tr>
<td>URL</td>
<td><a href="http://hdl.handle.net/2433/60495">http://hdl.handle.net/2433/60495</a></td>
</tr>
<tr>
<td>Type</td>
<td>Departmental Bulletin Paper</td>
</tr>
</tbody>
</table>

Kyoto University
On Computational Power of Jumping Petri Nets\textsuperscript{1}

Ferucio Laurenţiu TIPLEA

Faculty of Informatics
"Al. I. Cuza" University, 6600 Iaşi, Romania
E-mail: fltiplea@infoiasi.ro

Abstract
A Jumping Petri Net ([18], [12]), $JPTN$ for short, is defined as a classical net which can spontaneously jumps from a marking to another one. In [18] it has been shown that the reachability problem for $JPTN$'s is undecidable, but it is decidable for finite $JPTN$'s ($FJPTN$). In this paper we investigate the computational power of such nets, via the interleaving semantics. Thus, we show that the non-labelled $JPTN$'s have the same computational power as the labelled or $\lambda$-labelled $JPTN$'s. When final markings are considered, the power of $JPTN$'s equals the power of Turing machines. Languages generated by $FJPTN$'s can be represented in terms of regular languages and substitutions with classical Petri net languages. This characterization result leads to many important consequences, e.g. the recursiveness (context-sensitiveness, resp.) of languages generated by arbitrarily labelled (labelled, resp.) $FJPTN$'s. A pumping lemma for nonterminal jumping net languages is also established. Finally, some comparisons between families of languages are given, and a connection between $FJPTN$'s and a subclass of inhibitor nets is presented.

1 Introduction and Preliminaries

It is well-known that the behaviour of some distributed systems cannot be adequately modelled by classical Petri nets. Many extensions which increase the computational and expressive power of Petri nets have been thus introduced. One direction has led to the modification of the firing rule of the nets ([2], [3], [5], [6], [7], [8], [12], [16], [17], [18], [19], [20], [21], [22]).

In this paper we investigate the computational power of jumping Petri nets as introduced in [18]. The paper is organized as follows. In the rest of this section we will establish the basic terminology, notations, and results concerning Petri nets in order to give the reader the necessary prerequisites for the understanding of this paper (for details the reader is referred to [1], [9], [11], [13], [14]). In Section 2 we show that non-labelled $JPTN$'s have the same computational power as labelled or $\lambda$-labelled $JPTN$'s and, in the case of final markings, their power equals that of Turing machines. Section 3 gives some characterization results for $FJN$'s in terms of regular languages and substitutions with $\lambda$-free languages. Then some important consequences are derived and a pumping lemma for nonterminal jumping net languages is established. In Section 4 some comparisons between families of languages are given. The last section presents a connection between $FJPTN$'s and a subclass of inhibitor nets.

\textsuperscript{1}a complete final version of this paper will be published elsewhere
The empty set is denoted by $\emptyset$; for a finite set $A$, $|A|$ denotes the cardinality of $A$ and $\mathcal{P}(A)$ denotes the set of all subsets of $A$. Given the sets $A$ and $B$, $A \subseteq B$ ($A \subset B$, resp.) denotes the inclusion (strict inclusion, resp.) of $A$ in $B$. If $R \subseteq A \times B$ then $\text{dom}(R)$ and $\text{cod}(R)$ denote the sets $\text{dom}(R) = \{a \in A \mid \exists b \in B : (a, b) \in R\}$ and $\text{cod}(R) = \{b \in B \mid \exists a \in A : (a, b) \in R\}$. The set of integers (nonnegative integers, positive integers, resp.) is denoted by $\mathbb{Z}$ ($\mathbb{N}$, $\mathbb{N}^+$, resp.).

For a (finite) alphabet $V$, $V^*$ denotes the free monoid generated by $V$ (under the concatenation operation) with the empty word $\lambda$. Given a word $w \in V^*$, $|w|$ denotes the length of $w$, and $\text{alph}(w)$ denotes the set of all letters occurring in $w$. $\mathcal{L}_3$ ($\mathcal{L}_2$, resp.) denotes the family of regular (context-free, resp.) languages and $\mathcal{L}_{3,\text{pref}}$ denotes the family of regular prefix languages (i.e., those languages containing all prefixes of its words).

A (finite) Petri net, abbreviated $PN$, is a 4-tuple $\Sigma = (S, T; F, W)$ where $S$ and $T$ are two finite sets, of places and transitions, such that $S \cap T = \emptyset$ and $S \cup T \neq \emptyset$, $F \subseteq (S \times T) \cup (T \times S)$ is the flow relation and $W : (S \times T) \cup (T \times S) \rightarrow \mathbb{N}$ is the weight function of $\Sigma$ verifying $W(x, y) = 0$ iff $(x, y) \notin F$. A function $M : S \rightarrow \mathbb{N}$ is called a marking of $\Sigma$ and it will be sometimes identified with a vector $M \in \mathbb{N}^{|S|}$. The set of all markings of $\Sigma$ is denoted by $N^S$. The relation $\leq$ and the operations $+$ and $-$ on nonnegative integers are componentwise extended to vectors in $N^S$.

A marked Petri net, abbreviated $mPN$, is a 2-tuple $\gamma = (\Sigma, M_0)$, where $\Sigma$ is a $PN$ and $M_0$ is a marking of $\Sigma$ called the initial marking of $\Sigma$. An $mPN$ with final markings, abbreviated $mPNf$, is a 3-tuple $\gamma = (\Sigma, M_0, M)$, where $(\Sigma, M_0)$ is an $mPN$ and $M$, called the set of final markings of $\gamma$, is a finite set of markings of $\Sigma$. A labelled $mPN$ ($mPNf$, resp.), abbreviated $lmPN$ ($lmPNf$, resp.), is a 3-tuple (4-tuple, resp.) $\gamma = (\Sigma, M_0, l)$ ($\gamma = (\Sigma, M_0, M, l)$, resp.), where $(\Sigma, M_0)$ ($\Sigma, M_0, M$, resp.) is an $mPN$ ($mPNf$, resp.) and $l$, called the labelling function of $\gamma$, is a function from the set of transitions of $\Sigma$ into an arbitrary alphabet $V$. A $\lambda$-labelled $mPN$ ($mPNf$, resp.), abbreviated $lmPN$ ($lmPNf$, resp.), is a 3-tuple (4-tuple, resp.) $\gamma = (\Sigma, M_0, l)$ ($\gamma = (\Sigma, M_0, M, l)$, resp.), where $(\Sigma, M_0)$ ($\Sigma, M_0, M$, resp.) is an $mPN$ ($mPNf$, resp.) and $l$, called the $\lambda$-labelling function of $\gamma$, is a function from the set of transitions of $\Sigma$ into $V \cup \{\lambda\}$, where $V$ is an arbitrary alphabet and $\lambda$ is the unity of $V^*$.

In the sequel we shall often use the term "net" whenever we refer to a structure $\gamma$ as those above. The notations $x^* = \{(y, x) \mid y \in S \cup T, (y, x) \in F\}$ and $x^+ = \{(x, y) \mid y \in S \cup T, (x, y) \in F\}$, for all $x \in S \cup T$, will be intensively used. Pictorially, a net $\gamma$ will be represented by a graph. Then places are denoted by circles, transitions are denoted by boxes; the flow relation is represented by direct edges: there is a direct edge from $x$ to $y$ iff $(x, y) \in F$. The initial marking is given by putting $M_0(s)$ tokens in the circle representing the place $s$. The labelling function is denoted by placing letters into the boxes representing transitions and final markings are explicitly listed.

Let $\gamma$ be a net and $M \in N^S$. A transition $t \in T$ is enabled at $M$, denoted $M[t]$, if $M(s) \geq W(s, t)$ for all $s \in S$. If $t$ is enabled at $M$ then $t$ may occur yielding a new marking $M'$, abbreviated $M[t]M'$, given by $M'(s) = M(s) - W(s, t) + W(t, s)$ for all $s \in S$. The above definition can be naturally extended to sequences of transitions by: $M[\lambda]M$ and $M[ut]M'$ iff there is a marking $M''$ such that $M[w]M''[t]M'$ ($M$ and $M'$ are markings, $u \in T^*$ and $t \in T$). Moreover, if $M_0[w]M$ then we say that $w$ is a firing or transition
sequence of $\gamma$ (leading from $M_0$ to $M$), and $M$ is reachable (from $M_0$) in $\gamma$. The set of all reachable markings of $\gamma$ is denoted by $[M_0]$.

Petri nets can be viewed as language generators. First, we extend the labelling function to morphisms in the usual way. Then, let $\gamma_1$ be an $mPTN$, $\gamma_2$ either an $lmPTN$ or an $l^mPTN$, $\gamma_3$ an $mPTNF$, and $\gamma_4$ either an $lmPTNF$ or an $l^mPTNF$. The languages generated by these nets are $P(\gamma_1) = \{\ell(w) | w \in T^* \land (\exists M \in \mathbf{N}^S : M_0[w]_{\gamma_1} M)\}$; $P(\gamma_2) = \{l(w) | w \in T^* \land (\exists M \in \mathbf{N}^S : M_0[w]_{\gamma_2} M)\}$, $L(\gamma_3) = \{w | w \in T^* \land (\exists M \in \mathcal{M} : M_0[w]_{\gamma_3} M)\}$, $L(\gamma_4) = \{l(w) | w \in T^* \land (\exists M \in \mathcal{M} : M_0[w]_{\gamma_4} M)\}$. The languages generated by $mPTN$ ($lmPTN$, $l^mPTN$, resp.) are called free $P$-type languages ($P$-type languages, arbitrary $P$-type languages, resp.) and the family of these languages is denoted by $P^f (P, P^\lambda, \text{resp.})$. For Petri nets with final markings, the terminology is as above, by changing "P" into "L". These languages are usually referred to as Petri net languages or Petri languages.

A jumping $P/T$-net ([18]), abbreviated $JPTN$, is a pair $\gamma = (\Sigma, R)$, where $\Sigma$ is a PTN and $R$, the set of spontaneous jumps of $\gamma$, is a binary relation on the set of markings of $\Sigma$. In what follows the set $R$ of spontaneous jumps of $JPTN$’s will be assumed recursive, that is for any couple $(M, M')$ we can effectively decide whether or not $(M, M')$ is a member of $R$. If $\gamma$ has finitely many jumps then we say that $\gamma$ is a finite $JPTN$, abbreviated $FJPTN$.

Let $Y \in \{JPTN, FJPTN\}$. As usual one can define an $mY$ ($mYf$, $lmY$, $lmYf$, $l^mY$, $l^mYf$, resp.) by adding an initial marking (a set of final markings, a labelling, a $\lambda$-labelling, resp.) to $Y$. In fact, all remarks about Petri nets equally hold for jumping Petri nets. Pictorially, a jumping net will be represented as a classical net. Moreover, the relation $R$ will be separately listed.

Let $\gamma$ be a jumping net. The transition $t$ is $j$-enabled at a marking $M$ (in $\gamma$), abbreviated $M[t]_{\gamma,j}$, if there exists a marking $M_1$ such that $MR^*M_1[t]_{\Sigma} (\Sigma$ being the underlying net of $\gamma$ and $R^*$ the reflexive and transitive closure of $R$); if $t$ may occur at $M$ then it can yield a new marking $M'$, abbreviated $M[t]_{\gamma,j}M'$, given by $MR^*M_1[t]_{\Sigma}M_2R^*M'$, where $M_1, M_2$ are markings of $\gamma$. The notions of transition $j$-sequence and $j$-reachable marking are similarly defined as for Petri nets (we set $M[\lambda]_{\gamma,j}M'$ whenever $MR^*M'$). The set of all $j$-reachable markings of a marked JN $\gamma$ is denoted by $[M_0]_{\gamma,j}$ ($M_0$ being the initial marking of $\gamma$). The notation "$[\cdot]_{\gamma,j}$" will be simplified to "$[\cdot]_j$" whenever $\gamma$ is understood from the context.

Jumping nets can be considered as generators of languages in the same way as classical nets, by changing "$[\cdot]$" into "$[\cdot]_j$". For example, if $\gamma = (\Sigma, R, M_0, \mathcal{M}, l)$ is an $l^mJPTNf$, then the language generated by $\gamma$ is $L(\gamma) = \{l(w) | w \in T^* \land (\exists M \in \mathcal{M} : M_0[w]_{\gamma} M)\}$. Thus, $RX^f (RX, RX^\lambda, \text{resp.})$ will denote the family of free $X$-type jumping Petri net languages ($X$-type jumping Petri net languages, arbitrary $X$-type jumping Petri net languages, resp.), for any $X \in \{P, L\}$. For finite jumping nets, the corresponding family of languages will be denoted by $RX^f_{\text{fin}} (RX_{\text{fin}}, RX^\lambda_{\text{fin}}, \text{resp.})$. For any $X \in \{P, L\}$ we have $X^f \subseteq RX^f_{\text{fin}} \subseteq RX^f, X \subseteq RX_{\text{fin}} \subseteq RX, X^\lambda \subseteq RX^\lambda_{\text{fin}} \subseteq RX^\lambda$.

Some jumps of a $FJN$ can be never used. Thus we say that a marked finite jumping net $\gamma$ is $R$-reduced if for any jump $(M, M')$ of $\gamma$ we have $M \neq M', M \in [M_0]_{\gamma,j}$, and there are some final markings of $\gamma$ which are j-enabled from $M'$ (if $\gamma$ has final markings). As
the reachability problem for \textit{FJN}'s is decidable ([18]), for any marked \textit{FJN} \( \gamma \) we can effectively construct (modifying only the set of jumpings of \( \gamma \)) a marked \textit{FJN} \( \gamma' \) such that \( \gamma' \) is \( R \)-reduced and it has the same computational power as \( \gamma \). All finite jumping nets in this paper will be considered \( R \)-reduced.

\section{Jumps and Labellings}

In this section we show that the jumps can "simulate" the labelling of nets. Then we use this result to prove that the power of \textit{JPTN}'s equals the class of recursively enumerable languages. In the case of jumping nets with finite state space the connection with regular languages is made.

\textbf{Theorem 2.1} For any \( X \in \{ P, L \} \) we have \( RX^f = RX = RX^\lambda \).

\textbf{Sketch of Proof} We prove only the case \( X = L \), the other one being similar to this. The inclusion \( RL^f \subseteq RL \) follows from definitions. Conversely, let \( L \in RL \). There is an \( lmJPTN \) \( \gamma = (\Sigma, R, M_0, M, l) \) such that \( L = L(\gamma) \). Let \( \Sigma = (S, T; F, W) \). Without loss of the generality we may assume \( T \cap \{ l(t) \mid t \in T \} = \emptyset \). Let \( T_1 = \{ t \in T \mid \forall t' \in T, t \neq t' \implies l(t) \neq l(t') \} \subseteq T \).

If \( T_1 = T \) we consider \( \gamma' = (\Sigma', R, M_0, M') \), where \( \Sigma' \) is obtained from \( \Sigma \) renaming each transition \( t \) by \( l(t) \). \( \gamma' \) is an \( mJPTNF \) and \( L(\gamma') = L \). If \( T_1 \subset T \), we construct \( \gamma' = (\Sigma, R', M'_0, M', l) \) as described below. We partition the set \( T_2 = T - T_1 \) in \( k \geq 1 \) subsets, \( T_2 = T_2^1 \cup \cdots \cup T_2^k \), such that for any \( i, 1 \leq i \leq k \), the set \( T_2^i \) contains those transitions of \( \Sigma \) which have the same label; let \( a_i \) be this label. We have \( a_i \neq a_j \) for any \( i \neq j \). The set of transitions of \( \Sigma' \) will be \( T' = l(T_1) \cup T_2 \cup \{ a_1, \ldots, a_k \} \). The basic idea is: when a transition \( t \in T_1 \) occurs in \( \gamma \), its effect is simulated in \( \gamma' \) by the transition \( l(t) \in l(T_1) \); when a transition \( t \in T_2^i, 1 \leq i \leq k \), occurs in \( \gamma' \), its effect is simulated in \( \gamma' \) by the relation \( R' \) and the transition \( a_i \). The transitions of \( T_2 \) will be blocked forever in the net \( \gamma' \).

We consider now the second equality. The inclusion \( RL \subseteq RL \lambda \) follows from definitions. Conversely, let \( L \in RL \lambda \). There is an \( \lambda mJPTNF \gamma = (\Sigma, R, M_0, M, l) \) such that \( L = L(\gamma) \). Without loss of the generality we may assume that \( T \cap \{ l(t) \mid t \in T \} = \emptyset \), \( T \) being the set of transitions of \( \gamma \). Let \( T_1 = \{ t \in T \mid l(t) \neq \lambda \} \). We have \( T_1 \subseteq T \).

If \( T_1 = T \) then \( \gamma \) is an \( lmJPTNF \) and hence \( L \in RL \). If \( T_1 \subset T \) we construct an \( lmJPTNF \gamma' = (\Sigma', R', M'_0, M', l') \) as follows. \( \Sigma' \) will have the same transitions and places as \( \Sigma \) excepting a new place \( s' \) which will be used to block all transitions in \( T_2 = T - T_1 \); their effect will be simulated by the relation \( R' \). \( \square \)

\textbf{Theorem 2.2} \( RL^f = RL = RL \lambda = L_0 \).

\textbf{Proof} The equalities \( RL^f = RL = RL \lambda \) have been already established. The equality with the set of recursively enumerable languages can be obtained as follows. In [18] it has been proved that jumping Petri nets can simulate inhibitor nets (which have the power of Turing machines). As a consequence, \( L_0 \subseteq RL \lambda \). Now we prove that \( RL^f \subseteq L_0 \).

Let \( \gamma = (\Sigma, R, M_0, M) \) be an \( mJPTNF \). We show that there is an algorithm \( A \) such that for all \( w \in T^* \) we have:

\( w \in L(\gamma) \) iff \( A \) beginning with the input \( w \) it will eventually halt accepting \( w \).
First we have to remark that $w \in L(\gamma)$ iff there is a computation in $\gamma$ of the form:

$$M_0R^*M_0'[w_1]M_1R^*M_1'[\ldots]M_{k-1}R^+M_{k-1}'[w_k]M_kR^+M \in M,$$

where $w_1, \ldots, w_k$ ($k \geq 1$) is a decomposition of $w$ in non-empty words, that is $w = w_1 \cdots w_k$ and none of $w_i$ is empty. All computations as the above one will be called terminal computations in $\gamma$. A terminal computation can be written as a (formal) string

$$(M_0, M_0')w_1(M_1, M_1') \cdots (M_{k-1}, M_{k-1}')w_k(M_k, M),$$

where $(M_0, M_0'), (M_k, M) \in R^*$, $(M_1, M_1'), \ldots, (M_{k-1}, M_{k-1}') \in R^+$ and $w_1, \ldots, w_k \in T^+$ (the empty transition sequence is identified by a string of the form $(M_0R^*M_0')$). It is clear that not any string as above describes a terminal computation in $\gamma$. But if we have such a string we can effectively decide whether or not it describes a terminal computation in $\gamma$. Now we remark that $R^*$ is recursively enumerable ($R$ is recursive) and consequently, we can enumerate $R^*$, $r_0, r_1, \ldots, r_n, \ldots$, (for any $n \geq 0$, $r_n$ is a couple $(M, M')$ satisfying $MR^M$).

Any $w \in T^*$ has finitely many decompositions $w = w_1 \cdots w_k$ ($k \geq 1$) with $w_i \in T^+$ for all $i$, and let $d_1, \ldots, d_m$ ($m \geq 1$) be all these decompositions. For any decomposition $d_i$ ($1 \leq i \leq m$), $d_i : w = w_1 \cdots w_k$, we consider the $N$-indexed sequence $S_i$ defined by:

- consider first all strings obtained from $d_i$ and $r_0$ as above (in this case we have only one string $r_0w_1r_0 \cdots r_0w_kr_0$);
- consider then, in an arbitrary but fixed order, all strings as above obtained from $d_i$ and $r_0, r_1$ (for example, $r_0w_1r_0 \cdots r_0w_kr_1$ is such a string);
- and so on.

We obtain, using all decompositions of $w$, $m$ sequences $S_i : c_1^i, c_2^i, \ldots, c_m^i, \ldots, 1 \leq i \leq m$. Now, the activity of the algorithm $A$ on the input $w \in T^*$ can be described as follows:

1. $A$ computes all decompositions of $w$; let $d_1, \ldots, d_m$ ($m \geq 1$) be these decompositions;
2. $A$ searches the sequences $S_1, \ldots, S_m$ (as above) in the order $c_1^1, c_1^2, \ldots, c_1^m, c_2^1, c_2^2, \ldots, c_2^m, \ldots$;
3. for a string $c_j^i$ ($i \geq 1, 1 \leq j \leq m$) the algorithm $A$ can effectively decide whether or not $c_j^i$ describes a terminal computation of $w$ in $\gamma$. If this is the case, then $A$ halts with the answer "$w$ is a member of $L(\gamma)$"; otherwise, $A$ will continue the searching.

It is easy to see that $A$ halts on the input $w$ iff $w \in L(\gamma)$. We conclude that $L(\gamma) \subseteq L_0$ and so, $\text{RL}^f \subseteq L_0$. Combining this inclusion with the other one we obtain the theorem.

\[\square\]

### 3 Characterization Results and Consequences

In this section we focus on finite jumping nets. We shall prove that any language $L \in \text{RL}^f_{\text{fin}}$ (or $\text{RL}^f_{\text{fin}}$, resp.) can be represented as $L = \varphi(L')$, where $L'$ is a regular language and $\varphi$ is a substitution with $\lambda$-free languages. Similar results hold true for $P$-type jumping Petri net languages.

**Theorem 3.1** For any $L \in \text{RL}^f_{\text{fin}}$ (or $\text{RL}^f_{\text{fin}}$, resp.) there exist a language $L' \in L_3$ and a substitution with $\lambda$-free languages $\varphi$ from alph($L'$) into $L^f$ ($L, L^\lambda$, resp.) such that $L = \varphi(L')$.

**Proof** Let $L \in \text{RL}^f_{\text{fin}}$. There is an $mFJPTNF$ $\gamma = (\Sigma, R, M_0, M)$ such that $L = L(\gamma)$. We construct an finite automaton with $\lambda$-moves, $A = (Q, I, \delta, q_0, Q_f)$, as follows:
(i) \( Q = \{ M_0 \} \cup \text{dom}(R) \cup \text{cod}(R) \cup \mathcal{M}, \ q_0 = M_0, \ Q_f = \mathcal{M}; \)

(ii) \( I = \{ a_{M', M} | M', M \in Q \} \) and \( M \) is reachable from \( M' \) in \( \Sigma \) by a non-empty sequence of transitions;

(iii) \( \delta : Q \times (I \cup \{ \lambda \}) \rightarrow Q \) is given by: \( \delta(M', a_{M', M}) = \{ M \} \) if \( a_{M', M} \in I, \delta(M, \lambda) = \{ M' | (M, M') \in R \} \); it is undefined otherwise.

Let \( L' = L(A) \) and \( \varphi : \text{alph}(L') \rightarrow \mathcal{L}^f \) given by \( \varphi(a_{M', M}) = L(M', M) - \{ \lambda \} \), where \( L(M', M) \) is the language generated by the \( mPTNF \) \((\Sigma, M', \{ M \}).\)

We have \( L' \in \mathcal{L}_3 \). Let us prove that \( L = \varphi(L') \). First, \( \lambda \in L \) iff \( \lambda \in L' \) and hence \( \lambda \in L \) iff \( \lambda \in \varphi(L') \). Let now \( w \in L, \ w \neq \lambda \). There is a decomposition of \( w, w = w_1 \cdots w_{m+1}, m \geq 0 \), such that \( M_0 R^* M'_0 (w_1) \Sigma M_1 R^+ M'_1 \cdots R^+ M'_m (w_{m+1}) \Sigma M_{m+1} R^* M'_{m+1} \in \mathcal{M} \), where \( M_i \) and \( M'_i \) are markings of \( \gamma \) and \( w_i \neq \lambda \) for any \( 0 \leq i \leq m + 1 \).

The sequence \( u = a_{M'_0, M} a_{M'_1, M_2} \cdots a_{M'_{m+1}, M_{m+1}} \) determines a unique path, excepting \( \lambda \)-moves, from \( M_0 \) to \( M'_{m+1} \) in the automaton \( A \); hence \( u \in L' \). For any \( i, 1 \leq i \leq m + 1 \), we have \( w_i \in L(M_{i-1}, M_i) - \{ \lambda \} \) which shows that \( w \in \varphi(u) \), i.e. \( w \in \varphi(L') \). Thus the inclusion \( L \subseteq \varphi(L') \) is proved. The other inclusion can be analogously proved.

The case \( L \in RL_{\text{fin}}^\lambda \) can be simply settled starting from the remark that if \( L = L(\gamma), \ \gamma = (\Sigma, R, M_0, M, l) \), then \( L = l(L(\gamma')) \), where \( \gamma' = (\Sigma, R, M_0, M) \). Now, there exist a regular language \( L' \) and a substitution with \( \lambda \)-free languages \( \psi \) from \( \text{alph}(L') \) into \( L^f \) such that \( L(\gamma') = \psi(L') \). Define \( \varphi = l \circ \psi \) which is a substitution with \( \lambda \)-free languages, and obtain \( L = \varphi(L') \).

The previous idea does not work for the family \( RL_{\text{fin}}^\lambda \) because \( l \) is an arbitrary labelling function and, for some \( a, (l \circ \psi)(a) \) could contain \( \lambda \). But we will modify the construction in the case of \( RL_{\text{fin}}^f \) by setting \( \varphi(a_{M', M}) = l(L(M', M)) - \{ \lambda \} \), for any \( a_{M', M} \), and adding to the automaton \( A \) the arcs \((\overline{M}, M')\) labelled by \( \lambda \) whenever there exist \( A \) the arcs \((\overline{M}, M')\) and \((M, \overline{M}')\) labelled by \( \lambda \) and \((M', M)\) labelled by \( a_{M', M} \) and \( \lambda \in l(L(\Sigma, M', \{ M \})) \) (we mention that Petri net languages are recursive languages ([11]) and so we can effectively decide whether or not \( \lambda \) is a member of such language). It is easy to see that the theorem holds also true in this case. \( \square \)

The proof of Theorem 3.1 is effective. This fact permits us to show that terminal jumping Petri net languages are recursive.

**Corollary 3.1** \( RL_{\text{fin}}^\lambda \subseteq \mathcal{L}_{\text{rec}} \).

**Proof** We show that the membership problem for the family \( RL_{\text{fin}}^\lambda \) is decidable. Let \( \gamma = (\Sigma, R, M_0, M, l) \) be an \( l^\lambda mFJPNTf \), \( T \) the set of its transitions and \( V \) the range of \( l \). From Theorem 3.1 it follows that we can effectively compute a regular language \( L' \) (given by a finite automaton) and a substitution with \( \lambda \)-free languages \( \varphi : \text{alph}(L') \rightarrow \mathcal{L}^\lambda \) such that \( L(\gamma) = \varphi(L') \). Let \( w \in V^* \). Since \( \varphi \) is a substitution with \( \lambda \)-free languages we have:

- if \( w = \lambda \) then \( w \in L(\gamma) \) iff \( w \in L' \);
- if \( w \neq \lambda \), \( w = a_1 \cdots a_n (n \geq 1) \), then \( w \in L(\gamma) \) iff there exist \( w = b_1 \cdots b_m \in L' \) \((1 \leq m \leq n)\) and \( u_i \in \varphi(b_i), 1 \leq i \leq n\), such that \( |u_i| \leq n \) and \( w = u_1 \cdots u_m \).

Consequently, the membership problem for \( L(\gamma) \) can be reduced to the membership problem for a regular language and for some arbitrary Petri net languages. As the arbitrary Petri net languages are recursive ([11]) we conclude that the membership problem for \( RL_{\text{fin}}^\lambda \) is decidable. \( \square \)
Corollary 3.2 $\text{RL}_{\text{fin}} \subseteq \mathcal{L}_1$.

**Proof** For any language $L \in \text{RL}_{\text{fin}}$ there exist a regular language $L$ and a substitution with $\lambda$-free languages $\varphi$ from $\text{alph}(L')$ into $L$ such that $L = \varphi(L')$. But $L \subseteq \mathcal{L}_1$ ([11]) and $\mathcal{L}_1$ is closed under substitutions with $\lambda$-free languages, from which the theorem follows.

The converse of Theorem 3.1 holds true for labelled or $\lambda$-labelled jumping nets.

**Theorem 3.2** If $L \in \mathcal{L}_3$ and $\varphi$ is a substitution from $\text{alph}(L)$ into $L$ ($L^\lambda$, resp.) then $\varphi(L) \in \text{RL}_{\text{fin}}$ ($\text{RL}_{\text{fin}}^\lambda$, resp.).

**Sketch of Proof** Let $L \in \mathcal{L}_3$ and $\varphi : \text{alph}(L) \rightarrow L$. There is an $\text{lmP}TNf$ $\gamma = (\Sigma, M, \mathcal{M}, l)$ such that $L(\gamma) = L$. Moreover, $[M_0]$ is finite. Let $\text{alph}(L) = \{a_1, \ldots, a_n\}$, $n \geq 1$, and $L_i = \varphi(a_i), 1 \leq i \leq n$. There exist the $\text{lmP}TNf$ $\gamma_i = (\Sigma_i, M_i^0, \mathcal{M}_i, l_i), 1 \leq i \leq n$, such that $L_i = L(\gamma_i)$.

Construct an $\text{lmP}TNf$ $\gamma' = (\Sigma', R', M_0', \mathcal{M}', l')$ such that $\varphi(L) = L(\gamma')$, starting from the following idea. The nets $\Sigma, \Sigma_1, \ldots, \Sigma_n$ will be subnets of $\Sigma'$ and initially they will be "blocked". When a transition $t$ labelled by $l(t) = a_i, 1 \leq i \leq n$, occurs in $\gamma$ then in $\gamma'$ the subnet $\Sigma_i$ will be relieved (by means $R'$) and a transition sequence $w$ in $\gamma$ can now occur in $\gamma'$. When a final marking will be reached in $\Sigma_i$ this subnet will be blocked again by means of $R'$. □

**Corollary 3.3** $L \in \text{RL}_{\text{fin}}$ ($\text{RL}_{\text{fin}}^\lambda$, resp.) iff there exist $L' \in \mathcal{L}_3$ and a substitution with $\lambda$-free languages $\varphi : \text{alph}(L') \rightarrow L$ ($L^\lambda$, resp.) such that $L = \varphi(L')$.

A similar result as that in Theorem 3.1 holds for $P$-type jumping languages.

**Theorem 3.3** For any $L \in \text{RP}_{\text{fin}}^P$ ($\text{RP}_{\text{fin}}^\lambda$, resp.) there exist a language $L' \in \mathcal{L}_{3, \text{pref}}$, a substitution with $\lambda$-free languages $\varphi$ from $\text{alph}(L')$ into $L^P$ ($L^\lambda$, resp.), and the languages $P_0$ and $P_a$, $a \in \text{alph}(L')$, are finite unions of free $P$-type languages ($P$-type languages, arbitrary $P$-type languages, resp.).

**Proof** Let $\gamma = (\Sigma, R, M_0)$ be an $mFJPTN$ such that $L = L(\gamma)$. We construct an finite automaton with $\lambda$-moves, $A = (Q, I, \delta, q_0, Q_f)$, similar to that described in the proof of Theorem 4.1, excepting only the sets of states and final states which will be $Q = \{M_0\} \cup \text{dom}(R) \cup \text{cod}(R)$ and $Q_f = Q$. Next we consider $L' = L(A)$ which is a prefix regular language, the substitution $\varphi$ as in the proof of Theorem 3.1, $P_0 = \bigcup_{(M_0, M)a} P(\Sigma, M)$, and $P_{a_{M', M}} = \bigcup_{(M, aM'' R^+ M') \in R^+} P(\Sigma, M'')$ for any $a_{M', M} \in \text{alph}(L')$. Now, let us prove the equality in theorem. Let $w \in L$.

If $w = \lambda$ or the computation induced by $w$ contains a group of jumps only at the beginning $(M_0 R^+ M'_0 [w] M)$ then $w \in P_0$. Otherwise there is a decomposition of $w$, $w = w_1 \cdots w_{m+1}$, $m \geq 1$, such that $M_0 R^+ M'_0 [w_1] M_1 R^+ M'_1 \cdots [w_m] M_m R^+ M'_m [w_{m+1}] M \in N^S$, where $w_i \neq \lambda$ for any $1 \leq i \leq m + 1$.

The sequence $u = a_{M'_m, M} a_{M'_m, M_2} \cdots a_{M'_1, M_1}$ determines a unique path (from $M_0$ to $M_m$) in the automaton $A$ and hence $u \in L'$. For any $i, 1 \leq i \leq m$, we have $w_i \in L(M'_i, M_i) = \varphi(a_{M'_i-1, M_i})$ which shows that $w_1 \cdots w_m \in \varphi(u)$, i.e. $w_1 \cdots w_m \in \varphi(u) \subseteq$
\( \varphi(\partial_{a}(M'_{m-1}, M_{m-1}, M_{m})(L')\{a_{M'_{m-1}, M_{m}}\}) \). But, it is clear that \( w_{m+1} \in P_{a_{M'_{m-1}, M_{m}}} \), and thus we obtain

\[
\omega \in \varphi(\partial_{a}(M'_{m-1}, M_{m})(L')\{a_{M'_{m-1}, M_{m}}\})P_{a_{M'_{m-1}, M_{m}}} \subseteq \bigcup_{a \in \text{alph}(L)} \varphi(\partial_{a}(L')\{a\})P_{a}.
\]

The other inclusion can be analogously proved.

The case \( L \in RL_{\text{fin}}^{\lambda} \) (\( L \in RL_{\text{fin}}^{\lambda}, \text{resp.} \)) can be settled as in the proof of Theorem 4.1. We only mention that the languages \( P_{0} \) and \( P_{a} \) are images by the labelling homomorphism \( l \) of finite unions of free P-type Petri net languages; that is, \( P_{0} \) and \( P_{a} \) are finite unions of P-type Petri net languages (arbitrary P-type Petri net languages, resp.). \( \square \)

**Corollary 3.4** For any \( L \in RP_{\text{fin}}^{\lambda} \) (\( RP_{\text{fin}}^{\lambda}, \text{resp.} \)) there exist a language \( L' \in L_{3, pef} \), a substitution with \( \lambda \)-free languages \( \varphi \) from \( \text{alph}(L') \) into \( L \) (\( L_{\lambda}, \text{resp.} \)), and the P-type languages (arbitrary P-type languages, resp.) \( P_{0} \) and \( P_{a} \), \( a \in \text{alph}(L') \), such that

\[
L = P_{0} \cup \bigcup_{a \in \text{alph}(L')} \varphi(\partial_{a}(L')\{a\})P_{a}.
\]

**Proof** The family of (arbitrary) P-type languages is closed under union ([11]). \( \square \)

The idea of proof of Theorem 4.2 cannot be used for the family \( RL_{\text{fin}}^{\lambda} \), because it is not generally true that \( T_{i} \cap T_{j} = \phi \) for any \( i \neq j \), and it cannot be used for P-type languages because the relation \( R^{2} \) is, in general, infinite.

Using similar constructions as for classical Petri net languages it is easy to prove that the families \( RL_{\text{fin}}^{\lambda} \) and \( RL_{\text{fin}}^{\lambda} \) are closed under finite union and catenation (one can use also the power of jumping relation in correlation with final markings). Then we have:

**Corollary 3.5** \( RP_{\text{fin}}^{\lambda} \subseteq RL_{\text{fin}}^{\lambda} \) and \( RP_{\text{fin}}^{\lambda} \subseteq RL_{\text{fin}}^{\lambda} \).

**Proof** We will prove only the inclusion \( RP_{\text{fin}}^{\lambda} \subseteq RL_{\text{fin}}^{\lambda} \), the other one being similar to this one. If \( L \in RP_{\text{fin}}^{\lambda} \) then \( L \) can be written as in Theorem 3.3:

\[
L = P_{0} \cup \bigcup_{a \in \text{alph}(L')} \varphi(\partial_{a}(L')\{a\})P_{a}.
\]

\( \partial_{a}(L')\{a\} \) is a regular language and so \( \varphi(\partial_{a}(L')\{a\}) \in RL_{\text{fin}}^{\lambda} \) for any \( a \in \text{alph}(L') \) (Theorem 3.2).

It is well-known that P-type Petri net languages are also L-type Petri net languages ([11]), that is \( P \subseteq L \), and so \( P_{0}, P_{a} \in L \subseteq RL_{\text{fin}}^{\lambda} \). Using the above remark, that is \( RL_{\text{fin}}^{\lambda} \) is closed under finite union and catenation, we obtain \( L \in RL_{\text{fin}}^{\lambda} \). \( \square \)

For P-type languages the next kind of pumping lemma holds true.

**Theorem 3.4** For any \( L \in RP_{\text{fin}}^{\lambda} \) there is a number \( k \in N \) such that for each word \( w \in L \), if \( |w| \geq k \) then there is a prefix \( w' \) of \( w \) which has a decomposition \( w' = xyz \) such that \( |y| \geq 1 \) and \( xy^{m+1}z \in L \) for all \( m \geq 0 \).

**Proof** Let \( \gamma = (\Sigma, R, M_{0}, l) \) be an \( l^{m}FJPTN \) such that \( L = P(\gamma) \). Consider the automaton \( A \), the substitution \( \varphi \) and the languages \( L', P_{0} \) and \( P_{a} \) (\( a \in I \)) as in the proof of Theorem 4.3 (the languages \( P_{0} \) and \( P_{a} \), \( a \in I \), are arbitrary P-type Petri net languages). We have:

\[
L = P_{0} \cup \bigcup_{a \in \text{alph}(L')} \varphi(\partial_{a}(L')\{a\})P_{a}.
\]

Let \( k_{1}, k_{0}, k_{a} \) (\( a \in I \)) be the constants from the pumping lemmata for the regular language \( L' \) ([10]) and for the arbitrary P-type Petri net languages \( P_{0} \) and \( P_{a} \), \( a \in I \) ([4]). Consider \( k_{2} = \max\{k_{0}, k_{a} | a \in I \} \) and \( k = k_{1}k_{2} \). We shall prove that the number \( k \) satisfies the theorem.

Theorem 3.4...
Let \( w \in L \) such that \(|w| \geq k \). If \( w \in P_0 \) then we apply the pumping lemma for \( w \) with respect to \( P_0 \) and we obtain the theorem, with \( w' = w \). Otherwise, there is a word \( u = a_1 \cdots a_s \in L' \) such that \( w \in \varphi(u)P_a \). We have to consider two cases.

**Case 1** \( s \geq k_1 \). From the pumping lemma for regular languages, \( u \) has a decomposition \( u = u_1u_2u_3 \) such that \(|u_2| \geq 1 \) and \( u_1u_2u_3 \in L' \) for any \( i \geq 0 \). Since \( w \in \varphi(u)P_a \), there exist \( x \in \varphi(u_1) \), \( y \in \varphi(u_2) \), \( z \in \varphi(u_3) \) and \( v \in P_a \) such that \( w = xyzv \). \( \varphi \) being a substitution with \( \lambda \)-free languages it follows that \(|y| \geq 1 \).

From \( u_1u_2u_3 \in L' \) it follows that \( \varphi(u_1)[\varphi(u_2)]^i\varphi(u_3)P_a \subseteq L \) for any \( i \geq 0 \). Hence, \( xy^izv \in L \) for any \( i \geq 0 \), and the theorem is satisfied with \( w' = w \).

**Case 2** \( s < k_1 \). From \( w \in \varphi(a_1 \cdots a_s)P_a \), it follows that there exist \( w_j \in \varphi(a_j) \), \( 1 \leq j \leq s \), and \( w_{s+1} \in P_a \), such that \( w = w_1 \cdots w_s w_{s+1} \). Since \(|w| \geq k = k_k 1_k \) and \(|w| = |w_1| + \cdots + |w_s| + |w_{s+1}| \) and \( s < k_1 \), there is \( j \in \{1, \ldots, s+1\} \) such that \(|w_j| > k_2 \geq k_j \).

If \( j = s + 1 \) then we apply the pumping lemma for the language \( P_a \), and we obtain the theorem with \( w' = w \).

If \( j = 1 \) then it is easy to remark that \( L(M_0, M_1) \subseteq P_0 \), where \( a_1 = a_{M_0}M_1 \) and \( M_0R^*M_0' \mid w_1 > M_1 \). Thus \( w_1 \in P_0 \), and now we have to apply the pumping lemma for the word \( w_1 \) with respect to \( P_0 \). Then \( w_1 = x_1y_1z_1 \) with \(|y_1| \geq 1 \) and \( x_1y_1z_1 \in P_0 \) for any \( i \geq 1 \). Consider \( w' = w_1 \), \( x = x_1 \), \( y = y_1 \) and \( z = z_1 \) and the theorem is satisfied.

If \( 1 < j < s + 1 \) then let us suppose that \( a_{j-1} = a_{M_{j-1}M_{j-1}} \) and \( a_j = a_{M_{j-1}'M_j} \). Then, \( \varphi(a_j) = L(M_{j-1}'M_j) = L(\Sigma, M_{j-1}, \{M_j\}) \) and \( P_{a_{j-1}} = \bigcup_{\langle,M_{j-1},M \rangle \in R^+ X} P(\Sigma, M) \). Since \( M_{j-1}R^*M_{j-1}' \) it follows that \( \varphi(a_j) \subseteq P_{a_{j-1}} \) and \( \varphi(a_j) \cdots \varphi(a_{j-1})P_{a_{j-1}} \subseteq L \). Thus \( w \cdots w_j \in \varphi(a_j) \cdots \varphi(a_{j-1})P_{a_{j-1}} \), and now we have to apply the pumping lemma for the word \( w \) with respect to \( P_{a_{j-1}} \). Then, \( w_j = x_jy_jz_j \) with \(|y_j| \geq 1 \) and \( x_jy_jz_j \in P_{a_{j-1}} \) for any \( i \geq 1 \). Consider \( w' = w_1 \cdots w_j, x = w_1 \cdots w_{j-1}x_j, y = y_j \) and \( z = z_j \) and the theorem is satisfied in this case too. □

### 4 Comparisons Between Languages of Families

The first remark of this section is that any family of \( L \)-type jumping Petri net languages is closed under "*" (the net jumps from any final marking to the initial marking). This fact also holds for the family \( \mathrm{RP}^f, \mathrm{RP}, \mathrm{RP}^\lambda \), but not for \( \mathrm{RP}^f, \mathrm{RP}^\lambda \). The closure under "*" of the family \( \mathrm{RP}^\lambda \) can be proved using the following idea. At any reachable marking of the net some \( \lambda \)-transitions are enabled. These transitions will reset the current marking to the zero-marking \( 0 \) (all the components are 0) and then, a jump from \( 0 \) to the initial marking will restart the net. The non-closure under Kleene star of the families of Petri net languages leads us to the following results:

**Theorem 4.1** (1) \( P^\lambda \subseteq \mathrm{RP}^\lambda \);  
(2) \( L^f \subseteq \mathrm{RL}^f \), \( L \subseteq \mathrm{RL} \), \( L^\lambda \subseteq \mathrm{RL}^\lambda \).

**Theorem 4.2** \( \mathcal{L}_3 \subseteq \mathrm{RL}^f \).

**Proof** The inclusion follows from the fact that \( \mathcal{L}_3 = \mathrm{RL}(\mathrm{fss})^f \subseteq \mathrm{RL}^f \): it is strict because \( \mathcal{L}_3 \cup L^f \subseteq \mathrm{RL}^f \) and the families \( \mathcal{L}_3 \) and \( L^f \) are incomparable (\([11]\)). □

**Theorem 4.3** \( L = \{a^n b^n \mid n \geq 0 \} \notin \mathrm{RL}^f \).
Proof Suppose by contradiction that \( L \in \text{RL}_{\text{fn}}^{f} \). Then there exist a regular language \( L' \) and a substitution with \( \lambda \)-free languages \( \varphi : \text{alph}(L') \rightarrow \text{L}^{f} \) such that \( L = \varphi(L') \).

Case 1 \( L' \) is infinite. There exist \( u \in L' \) and a decomposition of \( u \), \( u = u_{1}u_{2}u_{3} \) such that \( |u_{2}| \geq 1 \) and \( u_{1}u_{2}u_{3} \in L' \) for any \( i \geq 0 \) (the pumping lemma for regular languages).

Since \( \varphi(u_{1}u_{2}u_{3}) = \varphi(u_{1})[\varphi(u_{2})]^{i}\varphi(u_{3}) \subseteq L \) and \( u_{2} \neq \lambda \), it follows that there exist \( w_{1} \in \varphi(u_{1}) \), \( w_{2} \in \varphi(u_{2}) \) and \( w_{3} \in \varphi(u_{3}) \) such that \( w_{1}w_{2}w_{3} \in L \) for any \( i \geq 0 \). It is easy to see that no matter how \( w_{1}, w_{2}, w_{3} (w_{2} \neq \lambda) \) are chosen we cannot have \( w_{1}w_{2}w_{3} \in L \) for any \( i \geq 0 \).

Case 2 \( L' \) is finite. If so, let \( L' = \{u_{1}, \ldots, u_{k}\} \), \( k \geq 1 \). Since \( L \) is infinite, there exists \( j \in \{1, \ldots, k\} \) such that \( \varphi(u_{j}) \) is infinite. Let \( u_{j} = a_{1} \ldots a_{m_{j}} \), \( m_{j} \geq 1 \), and \( \varphi(u_{j}) = \{a_{1}^{i}b^{j}, a_{2}^{i}b^{k}, \ldots\} \), where \( 0 \leq i_{1} < i_{2} < \ldots \). Then there is \( i \in \{1, \ldots, m_{j}\} \) such that \( \varphi(a_{i}) \) is infinite. We have to consider now the next cases.

If \( \varphi(a_{i}) = \{a^{\alpha_{1}}, a^{\alpha_{2}}, \ldots\} \), where \( 0 \leq \alpha_{1} < \alpha_{1} < \ldots \), then it is easy to see that no matter how the words in \( \varphi(a_{i}) \) are catenated to the left or to the right we cannot obtain only words in \( \varphi(u_{j}) \). Similar reason tells that \( \varphi(a_{i}) \) cannot be \( \{b^{\beta_{1}}, b^{\beta_{2}}, \ldots\} \), where \( 0 \leq \beta_{1} < \beta_{2} < \ldots \).

As any subset of \( \varphi(u_{j}) \) of cardinality at least two is not a member of \( \text{L}^{f} \), the only case which remains to be considered is \( \varphi(a_{i}) = \{a^{\alpha_{1}}b^{\beta_{1}}, a^{\alpha_{2}}b^{\beta_{2}}, \ldots\} \), where \( \alpha \)'s and \( \beta \)'s are natural numbers and there is \( n \) such that \( \alpha_{n} \neq \beta_{n} \). There is also an \( p, p \neq n \), such that either \( \alpha_{n} \neq \alpha_{p} \) or \( \beta_{n} \neq \beta_{p} \). A straightforward analysis shows us that no matter how the language \( \varphi(a_{i}) \) is catenated to the left or to the right we cannot obtain only words in \( \varphi(u_{j}) \).

In both cases we have derived a contradiction and hence \( L \notin \text{RL}_{\text{fn}}^{f} \). \( \square \)

Corollary 4.1 \( \text{RL}_{\text{fn}}^{f} \subseteq \text{RL}_{\text{fn}} \).

Proof \( \{a^{n}b^{n} | n \geq 0\} \in \text{L} \subseteq \text{RL}_{\text{fn}}^{f} \) and \( \{a^{n}b^{n} | n \leq 0\} \notin \text{RL}_{\text{fn}}^{f} \). \( \square \)

Corollary 4.2 The families \( \text{RL}_{\text{fn}}^{f} \) and \( \text{L}_{2} \) are incomparable.

Proof \( \{a^{n}b^{n} | n \geq 0\} \in \text{L}_{2} - \text{RL}_{\text{fn}}^{f} \) and \( \{a^{n}d^{n}e^{n} | n \geq 1\} \in \text{RL}_{\text{fn}}^{f} - \text{L}_{2} \). \( \square \)

5 Finite Jumping Nets and Global Inhibitor Nets

We will make a connection between finite jumping nets and a subclass of inhibitor nets, global inhibitor nets. We recall that ([9]) an inhibitor net is a pair \( \gamma = (\Sigma, I) \), where \( \Sigma \) is a Petri net and \( I \subseteq S \times T \) such that \( I \cap F = \emptyset \).

In an inhibitor net \( \gamma \) the transition \( t \) is \( i\)-enabled at a marking \( M \), abbreviated \( M[t]_{\gamma,i} \), iff \( t^{-} \leq M \) and \( M(s) = 0 \) for any \( s \in \{s \in S | (s, t) \in I\} \). If \( M[t]_{\gamma,\gamma} \) then \( t \) may occur yielding a new marking \( M' \), abbreviated \( M[t]_{\gamma,i}M' \), given by \( M' = M + \Delta t \). As we can see, an inhibitor net has the capability to perform zero-tests on some places. A global inhibitor net is defined as an inhibitor net performing zero-tests on all places, that is \( (s, t) \in I \Rightarrow (s', t) \in I, \forall s' \in S \).

Now we show that \( \text{FJPTN} \)'s can be simulated by global inhibitor nets. Let \( \gamma = (\Sigma, R, M_{0}, l) \) be an \( \text{FJPTN} \) with only a jump, \( R = \{(M, M')\} \). Construct the following inhibitor net (the net is pictorially represented in Figure 6.1 and the relation \( I \) is given by \( I = \{(s, t') | s \text{ is a place}\})$.


It is clear that $t'$ performs a zero-test on all places and so this net is a global inhibitor net. Its activity can be described as follows:
- the transition $t$ blocks $\Sigma$ and then the transitions $t_1', \ldots, t_m'$ check whether or not the current marking covers $M$ (if all $t_i'$ can occur than the current marking covers $M$).

The zero-test performed by $t'$ checks when the current marking is exactly $M$ ($t'$ can occur iff no token is in the net). If this is the case the marking $M'$ is setted for $\Sigma$.

The above construction can be easily generalized to an $FJPTN$ with arbitrarily many jumps.

Now we show that any arbitrarily labelled global inhibitor net can be simulated by an $FJPTN$. Indeed, let $\gamma = (\Sigma, I, M_0, l)$ be such an inhibitor net. Assume $I = \{(s, t)|s \in S\}$, where $t$ is a fixed transition. If $l(t) = \lambda$ then we can simulate the extent of change caused by the occurrence of $t$ using the jump $(0, M)$, where $M(s) = W(t, s)$ for all $s \in S$ (we recall that $I \cap F = \emptyset$, that is $W(s, t) = 0$ for all $s \in S$). If $l(t) = a \neq \lambda$ then we simulate the extent of change caused by $t$ using the net Figure 6.2 and the jump $\{((0,1,0), (M,0,1))\}$ ($M$ is as above). By this jump the net $\Sigma$ will be blocked; it is relieved after occurring the transition $t$ labelled by $a$ ($t$ being a new transition).

The generalization to an arbitrary global inhibitor net is straightforward.
Final Remarks

We consider that the extension of Petri nets allowing finite jumps is quite reasonable: on the one hand such nets have the basic decision problems decidable and, on the other hand the finite jumps strictly increase the power of the nets.

We close with some important open problems, in our estimation.

P1. Are the inclusions $RL_{\text{fin}} \subseteq L_1$, $R_{\text{fin}} \subseteq L_{\text{rec}}$, $RL_{\text{fin}} \subseteq RL_{\text{fin}}$, $RP_{\text{fin}} \subseteq RL_{\text{fin}}$, $RL_{\text{fin}} \subseteq RL_{\text{fin}}$ proper or not?

P2. Are the families $RP_{\text{fin}}$ and $RP_{\text{fin}}$ closed under ""?

P3. Define $RX^X_k$ ($RX_k$, $RX^X_k$, resp.) as being the family of $X$-type languages generated by jumping nets having at most $k$ jumps ($k \geq 0$), that is $|R| \leq k$. We have

\[
X^f \subseteq RX^f_k \subseteq RX^f_{k+1} \subseteq RX^f_{\text{fin}}
\]

\[
X \subseteq RX_k \subseteq RX_{k+1} \subseteq RX_{\text{fin}}
\]

\[
X^\lambda \subseteq RX^\lambda_k \subseteq RX^\lambda_{k+1} \subseteq RX^\lambda_{\text{fin}}
\]

for all $k \geq 1$ and $X \in \{P, L\}$.

Does this restriction define proper hierarchies of jumping Petri net languages?

P4. What about the connection between $FJPTN$'s and global inhibitor nets in the case we do not allow $\lambda$-transitions?

References


