Extended fuzzy Petri nets for decision support

Fedor Lehocki, Gabriel Juhas, Robert Lorenz, Martin Drozda

Angaben zur Veröffentlichung / Publication details:

Lehocki, Fedor, Gabriel Juhas, Robert Lorenz, and Martin Drozda. 2008. “Extended fuzzy Petri nets for decision support.” In *2008 First International Symposium on Applied Sciences on Biomedical and Communication Technologies, 25-28 Oct. 2008, Aalborg, Denmark*. Piscataway, NJ: IEEE. https://doi.org/10.1109/isabel.2008.4712586.

Extended Fuzzy Petri Nets for Decision Support

Fedor Lehocki, Gabriel Juhás
Slovak University of Technology
Faculty of Electrical Engineering and
Information Technology
Bratislava, Slovakia
{fedor.lehocki,gabriel.juhas}@stuba.sk

Robert Lorenz
University of Augsburg
Faculty of Applied Computer Science
Augsburg, Germany
robert.lorenz@informatik.uni-augsburg.de

Martin Drozda
Leibniz University of Hannover
Dept. of Computer Science
Hannover, Germany
drozda@sim.uni-hannover.de

Abstract—**We present results for the formalism of Extended Fuzzy Petri Nets by enriching the enabling and firing rule by introducing the so-called weights and thresholds in order to filter the propagated knowledge. It is shown that knowledge propagation described using such extended Fuzzy Petri nets still terminates in a unique stable state. Based on this result, the paper introduces an algorithm for knowledge propagation in medical decision support systems. We also discuss the general properties of knowledge propagation functions terminating in a unique stable state leading to the final recommendation of diagnose or treatment.**

I. INTRODUCTION

There are many different formalisms used for encoding the expert's knowledge, such as neural networks, semantic networks, machine learning or Petri nets; see [3] for a more complete list. They are used in decision support systems (DSS), designed to provide assessments or recommendations to aid in medical decision making.

Logical and Fuzzy Petri Nets are directed bipartite graphs with a significant degree of structural parallelism and pipelining; they seem to be a good choice for knowledge representation and reasoning in DSS. In this paper we provide results on why Petri nets represent strong and sound formalism for knowledge representation. For clarity we also briefly introduce the results from ref. [1] aimed at basic concepts of Logical and Fuzzy Petri Nets and continue with extended Fuzzy Petri Nets utilizing rule thresholds and weights. Several dialects of Logical and Fuzzy Petri Nets are used in the literature [2, 6, 9-11, 13, 15]. In this paper, we extend the approach presented in [1], which follows the approach given in [11]. Our approach assumes that if a transition produces a token to a place which is already marked by a token, the both tokens in that place are merged to one token. Transitions represent the production rules, and marked places represent valid propositions. Adding a token to an already marked place represents just a different proof of validity of an already valid proposition. Another assumption of [1] is that once a proposition is valid, it stays valid after application of a production rule, i.e. once a place of the net is marked by a token, firing of a transition does not consume the token from the place. A natural requirement on DSS is to provide a unique recommendation for any fixed values of inputs in finite time. Using Logical and Fuzzy Petri Nets, these requirements translate as the reachability of a unique stable marking from the given initial marking by application of a finite sequence of production rules. In [1] we have proved that this requirement is satisfied.

DSS do not always result in improved domain practice, for reasons that are not always clear. Based on experiences from [7] the most likely failures appear in a DSS knowledge base which is inaccurate or out of date. Another reason is the occurrence of problems with availability of input data from authentic sources or the incompleteness of data or knowledge. All this can result in appearance of inconsistent data elements in the reasoning space [8]. Mapping all those forms of inexactness onto a structured parallel distributed architecture such as Petri nets may result in increasing of the reasoning efficiency.

II. KNOWLEDGE REPRESENTATION AND PROPAGATION

Following the paper [1], in this section we informally introduce the idea of knowledge representation used through this paper. As a formal model of knowledge base and knowledge propagation we will use logical and fuzzy Petri nets defined in [1] and briefly described in Section 3, and their extended version defined in Section 4. Knowledge is represented by a set of propositions (which can have, in the simplest case, the values true or false). Knowledge propagation is described by a set of production rules. A production rule describes the relation between two sets of propositions. A set A of propositions represents the antecedent of the production rule and a set B of propositions represent the consequent of the production rule. The knowledge is propagated by firing of a production rule interpreted as follows: IF all propositions in the antecedent A have value true THEN the propositions in the consequent B are true.

We consider a simple knowledge base given by a set of propositions and a set of production rules of the following form: logical product of the propositions in the antecedent A implies the logical product of the propositions in the consequent B. The knowledge is propagated by firing of a sequence of rules, where the consequent of one rule is used as the antecedent of the next rule.

For cases with uncertain validity of propositions, it is suitable to use fuzzy values, typically from the closed interval of real values $\langle 0.0,1.0\rangle$, where value 0 represents the case in which the proposition is false and the value 1 represents the case in which the proposition is true. Values between 0 and 1 represent the measure of validity for the proposition. For example consider the following proposition: "The temperature of a patient is high." Obviously the validity of this proposition is uncertain. We know that this proposition is more valid if a patient has temperature 40°C than if he has temperature 38°C. If the validity of propositions is expressed by fuzzy values then also the relation between propositions of the antecedent and propositions of the consequent is fuzzy. A production rule with fuzzy relation is called fuzzy production rule. The mechanism of firing the fuzzy production rule and the knowledge propagation in a fuzzy knowledge base will be explained in the Section 3.

III. Logical and Fuzzy Petri Nets

In this section we define logical Petri nets and their application in modelling of logical knowledge bases and knowledge propagation as given in [1].

Definition 1: A logical Petri Net (*LPN*) is a triple $N = (P,T,F)$, where $P = \{p_1,p_2, \ldots p_r\}$ is a finite, nonempty set of places, $T = \{t_1,t_2, \ldots t_n\}$ is a finite set of transitions and $F \subseteq (P\times T) \cup (T \times P)$ is a flow relation, i.e. a finite set of ordered pairs of the form (p_i,t_j) and (t_j,p_i).

Given an element $x\in P\cup T$ we denote by $\bullet x$ the set of all input elements of x, i.e. $\bullet x=\{y\in P\cup T \mid (y,x) \in F\}$, and by $x\bullet$ we denote the set of all output elements of x i.e. $x\bullet=\{y\in P\cup T \mid (x,y) \in F\}$. The state of an *LPN* $N = (P,T,F)$ is given by a mapping $m: P\rightarrow \{0, 1\}$ associating with each place the value 0 or the value 1, called marking.

Definition 2: A marked *LPN* is a pair (N,m_0), where $N = (P,T,F)$ is a Logical Petri Net and and $m_0 : P\rightarrow \{0, 1\}$ is a marking of LPN, called the initial marking.

In *LPNs* places represent propositions. The value 0 in the initial marking of a place means, that the initial value of the respective proposition is not true (logical false). The value 1 in the initial marking of a place represents, that the initial value of the respective proposition is true (logical truth).

Transitions together with their input and output places correspond to the production rules. Namely, given a transition $t\in T$, the set $\bullet t$ of input places represents the antecedent of the respective production rule and the set $t\bullet$ of output places represents the consequent of the respective production rule. Firing of a production rule is represented by firing of the respective transition.

Definition 3: Given an *LPN* $N=(P,T,F)$, a transition $t\in T$ and a marking m of N, transition t is enabled to fire in m if $m(p) = 1$ for each $p \in \bullet t$. The firing of such a transition t changes the marking m to the marking m' called the follower marking of m (w.r.t. the transition t) given as follows: $m'(p) = 1$ for each $p \in t\bullet$ and $m'(p) = m(p)$ else. We denote the fact that a transition t is enabled to fire in a marking m by $m[t>$ and that its firing leads to the follower m' by $m[t>m'$. Given a marking m, we denote by Xm the set of all transitions enabled to fire in m.

We extend the enabledness and firing rule to sequences of transitions as follows:

Definition 4: Let $N=(P,T,F)$ be an *LPN*, m be a marking m of N and $s = t_1 \ldots t_n$ be a finite sequence of transitions of N. Then s is said to be enabled to fire in m if there exists a finite sequence of markings $m_1 \ldots m_n$ such that $m[t_1>m_1 \ldots. m_{n-1}[t_n>m_n$. The firing of such a sequence s changes the marking m to the marking m_n called the follower marking of m (w.r.t. the sequence s). We denote the fact that a firing sequence s is enabled to fire in m and its firing leads to the follower m' by $m[s>m'$.

Given a marking m of an *LPN*, we can define reachability of markings as follows.

Definition 5: Let m be a marking of an *LPN* $N=(P,T,F)$. Then a marking m' of N is said to be reachable from the marking m if there exists a firing sequence s such that $m[s>m'$.

Since firing a transition does not change the marking of input places and does not decrease the marking of out places, as a fundamental consequence of the firing rule of *LPN*s we have that all markings which are reachable from a given marking m have the value 1 in all the places p with $m(p)$=1.

In [1] we have proved that firing transitions in a marked LPN always leads to a stable marking, i.e. a marking which stays unchanged by firing any further transitions.

Theorem 1: Let (N,m_0), $N=(P,T,F)$, be a MLPN. Then there exists a firing sequence s and a marking m such that $m_0[s>m$ and $m[t>m$ for each transition t enabled in m.

In [1] we show that the firing rule has even a stronger consequence, namely that the marking m from Theorem 1 is unique, i.e. its value does not depend on the sequence leading to it.

Theorem 2: Let (N,m_0), $N=(P,T,F)$, be a MLPN. Let s_1 and s_2 be two firing sequences with $m_0[s_1>m_1$ and $m_0[s_2>m_2$ such that $m_1[t>m_1$ and $m_2[t>m_2$ for each transition t enabled in m_1 resp. m_2. Then $m_1 = m_2$.

According to the previous result, we can uniquely define the stable marking of a MLPN.

Definition 6: Let (N,m_0), $N=(P,T,F)$, be a MLPN. Le s be a firing sequence s with $m_0[s>m$ and $m[t>m$ for each transition t enabled in m. Then marking m is said to be the stable marking of (N,m_0) and s is said to be a stabilizing firing sequence of (N,m_0).

Because LPNs cannot deal with vague or fuzzy information such as „very good“ and "healthy" several kinds of Fuzzy Petri Nets (FPN) have been introduced [11]. They are used for fuzzy knowledge representation and reasoning. A FPN differs from a LPN only in markings and the firing rule. A marking of a FPN is a mapping $m: P\rightarrow \langle 0, 1\rangle$, which assigns a real value between zero and one to each place. In this paper we use the

Fuzzy Petri Nets defined in [1].

Definition 7: A marked FPN is a tuple (N,m_0), where $N=(P,T,F)$ is a LPN and m_0 is a marking of a FPN, called the initial marking.

Definition 8: Given a FPN $N=(P,T,F)$, a transition $t\in T$ and a marking m of N, transition t is enabled to fire in m if $m(p) > 0$ for each $p \in \bullet t$. The firing of such a transition t changes the marking m to the marking m' called the follower marking of m (w.r.t. the transition t) given by $m'(p) = max(m(p),min\{m(q) \mid q \in \bullet t\})$ for each $p \in t\bullet$ and $m'(p) = m(p)$ else.

We denote the fact that a transition t is enabled to fire in a marking m by $m[t>$ and that its firing leads to the follower m' by $m[t>m'$. Given a marking m, we denote by Xm the set of all transitions enabled to fire in m. Given the firing rule for transitions, the firing rule for finite sequences of transitions and the notion of reachable marking is given analogously as for LPNs. Note that this firing rule can be expressed as a fuzzy norm T_n [12]; in our case we use T_3. In [1] we have proved:

Theorem 3: Let (N,m_0), $N=(P,T,F)$, be a marked FPN. Then there exists a firing sequence s and a marking m such that $m_0[s>m$ and $m[t>m$ for each transition t enabled in m.

If s_1 and s_2 are two firing sequences with $m_0[s_1>m_1$, $m_0[s_2>m_2$ and $m_1[t>m_1$ and $m_2[t>m_2$ for each transition t enabled in m_1 resp. m_2, then $m_1 = m_2$.

That means in particular that the stable marking is unique.

Definition 9: Let (N,m_0), $N=(P,T,F)$, be a marked FPN. Let s be a firing sequence with $m_0[s>m$ and $m[t>m$ for each transition t enabled in m. Then marking m is said to be the stable marking of (N,m_0) and s is said to be a stabilizing firing sequence of (N,m_0).

As we have shown in [1], it is possible to compute the stable marking of a marked FPN through calculating maximal concurrent steps by linear algebra.

Definition 10: Let m be a marking of a FPN $N=(P,T,F)$. Then Xm is called maximal concurrent step w.r.t. m and firing of Xm in the marking m leads to the follower marking m' given by $m'(p) = max(m(p) , max\{min\{m(q) \mid q \in \bullet t\} \mid t \in \bullet p\cap Xm\})$ for each $p \in \cup_{t\in Xm} t\bullet$ and $m'(p) = m(p)$ else. Firing of Xm in m is denoted by $m[Xm>m'$.

There is a firing sequence of maximal concurrent steps leading to the stable marking [1].

Theorem 4: Let (N,m_0), $N=(P,T,F)$, be a marked FPN. Then there is a sequence of markings $m_1,...,m_k$ with $k \le (|P| \cdot |P-1|)/2$, such that $m_0[Xm_0>m_1[Xm_1>\ldots>m_k$ and m_k is the stable marking of (N,m_0).

For the purposes of the computation of the stable marking we introduced in [1] the following matrix operations. For vectors $\boldsymbol{a}=(a_1 , a_2, ..., a_n)$ and $\boldsymbol{b}=(b_1 ,b_2, ...,b_n)$ and an $(r \times n)$-dimensional matrix $\boldsymbol{Y}$ having rows $\boldsymbol{y}_1,...,\boldsymbol{y}_k$ with elements from the closed interval $\langle 0,1\rangle$ we denote:

$\boldsymbol{a}\ \underline{or}\ \boldsymbol{b} = (max(a_1, b_1),max(a_2,b_2),...,max(a_n,b_n))$.

$\boldsymbol{a}\ \underline{and}\ \boldsymbol{b} = max(min(a_1, b_1),min\ (a_2,b_2),..., min\ (a_n,b_n))$

$\underline{neg}\ \boldsymbol{a} = (1\text{-}a_1,\ ...,\ 1\text{-}a_n)$

$\boldsymbol{Y}\ \underline{and}\ \boldsymbol{a} = (\boldsymbol{y}_1\ \underline{and}\ \boldsymbol{a},..., \boldsymbol{y}_r\ \underline{and}\ \boldsymbol{a})$.

Given an LPN, we define $(r \times n)$-dimensional matrices $\boldsymbol{I}$ and $\boldsymbol{O}$. $\boldsymbol{I}$ is called input matrix and describes the flow relation between input places and transitions: For each row $a \in \{1,..,r\}$ and each column $b \in \{1,..,n\}$, if $p_a \in \bullet t_b$ then $i_{ab} = 1$, else $i_{ab} = 0$. $\boldsymbol{O}$ is called output matrix and describes the flow relation between output places and transitions: For each row $a \in \{1,..,r\}$ and each column $b \in \{1,..,n\}$, if $p_a \in t_b\bullet$ then $o_{ab} = 1$, else $o_{ab} = 0$. We also define for a marking m of an LPN r-dimensional vector $\boldsymbol{M}=(m(p_1),...,m(p_r))$. In the case of $m(p)=0$ we say that the place p is not marked. We define an n-dimensional vector $\boldsymbol{V}$ representing the transitions not enabled in a marking m by the value 1, an n-dimensional vector $\boldsymbol{X}$ representing the transitions enabled in a marking m, i.e. the set Xm, by non-zero values and a vector $\boldsymbol{M'}$ representing the follower marking when firing m, using the following definitions of the operators $\underline{or}$, $\underline{and}$ and $\underline{neg}$.

Definition 11: Let $N = (P,T,F)$ be a FPN and m be a marking. We denote:

$\boldsymbol{V} = \boldsymbol{I}^T\ \underline{and}\ (\underline{neg}\ \boldsymbol{M})$.

$\boldsymbol{X} = \underline{neg}\ \boldsymbol{V}$.

$\boldsymbol{M'} = \boldsymbol{M}\ \underline{or}\ (\boldsymbol{O}\ \underline{and}\ \boldsymbol{X})$.

The central result of [1] is the following theorem:

Theorem 5: Let $N = (P,T,F)$ be a FPN with $P = \{p_1, \ldots p_r\}$ and $T = \{t_1, .., t_n\}$ and m be a marking of N. Let $\boldsymbol{V}$, $\boldsymbol{X}$ and $\boldsymbol{M'}$ be as in Definition 11. Then $\boldsymbol{V} = (v_1,\ldots,v_n)$ represents which transitions in the marking m are not enabled, i.e. $v_b = 1$ if and only if t_b is not enabled to fire in m. Moreover if $v_b < 1$, then $v_b = 1 - min\{m(q) \mid q \in \bullet t_b\}$. $\boldsymbol{X} = (x_1,\ldots,x_n)$ represents which transitions are enabled in the marking m, i.e. $x_b > 0$ if and only if $t_b \in Xm$. In this case $x_b = \min\{m(q) \mid q \in \bullet tb\}$. $\boldsymbol{M'}$ represents the marking reached after firing maximal concurrent step Xm in m, i.e. $\boldsymbol{M'} = (m'(p_1),...,m'(p_r))$, where $m[Xm>m'$.

Based on these results, in [1] we presented the following algorithm for computation of the stable marking of a marked FPN, which terminates by Theorems 4 and 5:

Algorithm 1

```
Step 1:  Arrange matrices $\boldsymbol{I}$ ; $\boldsymbol{O}$
Step 2:  Set k=0 and arrange vector $\boldsymbol{M}_0$ representing the
         initial marking
Step 3:  Calculate vector $\boldsymbol{V}_k = \boldsymbol{I}^T\ \underline{and}\ (\underline{neg}\ \boldsymbol{M}_k)$
Step 4:  Calculate vector $\boldsymbol{X}_k = \underline{neg}\ \boldsymbol{V}_k$
Step 5:  Calculate vector $\boldsymbol{M}_{k+1} = \boldsymbol{M}_k\ \underline{or}\ (\boldsymbol{O}\ \underline{and}\ \boldsymbol{X}_k)$
Step 6:  Go to Step 3 until $\boldsymbol{M}_{k+1} = \boldsymbol{M}_k$
```

IV. Extended Fuzzy Petri Nets

In this section we discuss a possible extension of FPNs by rule weights (also called certainty factors) and threshold values [11] together with a simple example.

Threshold values represent a lower bound for the propagation of values. If a propagation value of a rule (minimum of markings of input places) is smaller than the threshold value of the rule, then the information represented by the propagation value is not certain enough and the rule is not enabled. Threshold values allow for requiring a minimum quality of information to be propagated. Threshold values are given by a mapping $tr{:}T \to \langle 0, 1\rangle$.

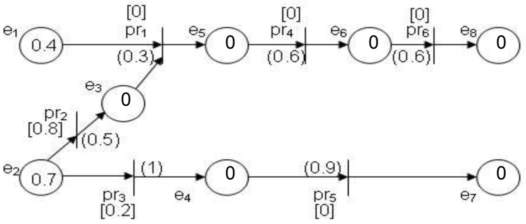

Fig. 1. Initial marking $\boldsymbol{M_0}$ of an extended FPN.

Rule weights represent an upper bound for the propagation of values. If a propagation value of a rule is bigger than the rule weight, then only the rule weight is propagated. Rule weights allow for restricting the influence of rules. Rule weights are given by a mapping $w{:}T \to \langle 0, 1\rangle$. In figures, threshold values are given in brackets of the form [*tr*] and rule weights are given in brackets of the form (*w*) (see Fig. 1).

A transition *t* is enabled to fire in a marking *m* if $m(p) \geq tr(t)$ for each $p \in \bullet t$. For a transition *t* we denote $val(t)=\min\{m(q) \mid q \in \bullet t\}$. The firing of such a transition *t* changes the marking *m* to the marking *m'* called the follower marking of m (w.r.t. the transition *t*) given by $m'(p) = max(m(p),min(val(t),w(t)))$ for each $p \in t\bullet$ and $m'(p) = m(p)$ else.

Obviously, as for FPNs, the firing of transitions increases markings and sets of enabled transitions. The only difference is that threshold values additionally prevent some rules to be enabled and rule weights restrict the possibility to strictly increase markings by firing of transitions. Therefore also an analogous result to Theorem 3 holds, i.e. there is a unique stable marking which can be reached by firing sequences which are long enough. The set of possible markings of places is now given by $\{m \mid \exists p{:}\ m_0(p)=m>0\}\cup\{n \mid \exists t{:}\ w(t)=n>0\}$, i.e. by the initial marking and the rule weights. The cardinality of this set determines an upper bound for the length of firing sequences leading to stable markings. Maximal concurrent steps and their firing rule can be defined as for FPNs, taking additionally the rule weights into account, i.e. *Xm* is the set of transitions enabled in *m* (w.r.t. threshold values) and the firing of *Xm* in a marking *m* leads to the follower marking *m'* given by $m'(p) = max(m(p), max\{min(val(t),w(t)) \mid t \in \bullet p \cap Xm\})$ for each $p \in \cup_{t\in Xm}\, t\bullet$ and $m'(p) = m(p)$ else. In an analogous way as for FPNs a firing sequence of maximal concurrent steps leading to the stable marking can be constructed. For the computation of the stable marking by a firing sequence of maximal concurrent steps, it is necessary to make some changes to the algorithm introduced to FPNs. The definitions of the operations and, or and neg are the same. Also the matrix $\boldsymbol{I}$ and the vector $\boldsymbol{M}$ do not change. To account for the rule weights, the matrix $\boldsymbol{O}$ is changed as follows: If $p_a \in t_b\bullet$, then $o_{ab} = w(t_b)$ and $o_{ab} = 0$ else. This restricts the propagated value to the rule weight. Negation of $\boldsymbol{V}$ results in information about rules which are enabled without considering thresholds (here: pr_2, pr_3). Considering the rule thresholds we set those values in $\boldsymbol{V}$ to 0 which are smaller than the threshold in order to represent enabledness for FPNs. We introduce a new operation *top* defined for $\boldsymbol{a}=(a_1,a_2, \ldots,a_n)$ and $\boldsymbol{b}=(b_1,b_2, \ldots,b_n)$ by $\boldsymbol{a}$ *top* $\boldsymbol{b} = (c_1,\ldots,c_n)$, $c_i = a_i$ if $a_i \geq bi$ and $c_i = 0$ else. If $\boldsymbol{W} = (w(t_1),\ldots,w(t_n))$, then $\boldsymbol{X} = ($*neg* $\boldsymbol{V})$ top $\boldsymbol{W}$ defines the transitions which are enabled by non-zero values. In our example $\boldsymbol{W} = (0,0.8,0.2,0,0,0)$ *and* $\boldsymbol{X} = (0,0,0.7,0,0,0)$. The non-zero values represent the propagation values of enabled transitions as for FPNs, i.e. $x_b = val(t_b)$ for t_b enabled.

Now $\boldsymbol{M'}$ is computed as for FPNs, i.e. $\boldsymbol{M'} = \boldsymbol{M}$ *or* $(\boldsymbol{O}$ *and* $\boldsymbol{X})$ using the new matrix $\boldsymbol{O}$. Indeed $\boldsymbol{M'}$ defines the follower marking. Let $\boldsymbol{M'} = (m_1',\ldots,m_r')$. By definition $m_a' = max(m_a,max(min(o_{a1}, x_1),\ldots,min\ (o_{an}\ ,x_n)))$. Since $x_b = val(t_b)$ for t_b enabled, $x_b = 0$ else and $o_{ab} = w(t_b)$ for each place $p_a \in t_b\bullet$, we get $m'(p_a) = (max(m(p_a)\ , max\{min(val(t_b),w(t_b)) \mid t_b \in \bullet p_a \cap Xm\})$ as required. We get the following algorithm for computation of the stable marking of a marked extended FPN:

Algorithm 2

```
Step 1:  Arrange matrices $I$ ; $O$ and vector $W$
Step 2:  Set k=0 and arrange vector $M_0$ representing the
         initial marking
Step 3:  Calculate vector $V_k = I^T$ and (neg $M_k$)
Step 4:  Calculate vector $X_k$ = (neg $V_k$ ) top $W$
Step 5:  Calculate vector $M_{k+1} = M_k$ or ($O$ and $X_k$)
Step 6:  Go to Step 3 until $M_{k+1} = M_k$
```

In our example, after three iterations ($k = 3$) we reach the stable state $\boldsymbol{M_3} = (0.4,0.7,0,0.7,0,0,0.7,0)$ (Fig. 2).

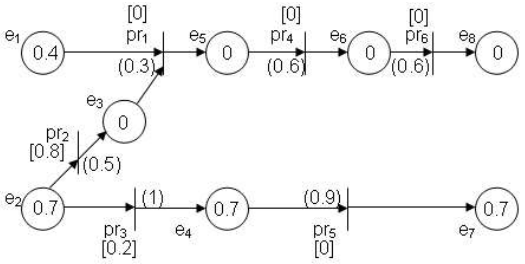

Fig. 2. Stable marking $\boldsymbol{M_3}$ of an extended FPN.

V. Other Knowledge Propagation Functions

Up to now, in this paper we considered three particular methods of knowledge propagation encoded in the firing rule of Petri nets, each extending the previous one. Of course, the presented methods are only examples and they are not the only possible choices. The crucial result was, that for each method the behavior of the net always reaches a unique stable state which can be used as information for decision support, and that there is an efficient computation (having linear runtime in the size of the net) of this stable state. In this section we intend to define a Petri net defining knowledge propagation in a very general way using quite arbitrary knowledge propagation functions determining the firing rule of transitions. We then focus on the identification of properties of such knowledge propagation functions which may support that a unique stable state is always reached.

Definition 15: Let *(P,T,F)* be defined as for LPNs and FPNs and let K be an arbitrary set. For each transition t, let there be a fixed ordering among the input places of t, i.e. $\bullet t = \{pt_1,...,pt_{nt}\}$. For each place p in $t\bullet$ let $f_{t,p} : K^{nt} \rightarrow K$ be a partial mapping. Let $g_p : 2^K \rightarrow K$ be a total mapping (2^K denotes the set of all subsets of K) for each place p. Let *Map* be the family of all those mappings. Then *(P,T,F,Map)* is called a Knowledge Propagation Petri Net (KPPN). A marking of a KPPN is a mapping m: $P \rightarrow K$. A transition t is enabled to fire in a marking m, if for each p in $t\bullet$, $f_{t,p}(m(pt_1),...,m(pt_{nt}))$ is defined. Its firing leads to the follower marking m' given by $m'(p) = g_p(\{f_{t,p}(m(pt_1),...,m(pt_{nt})),m(p)\})$ for output places p of t. Markings of other places remain unchanged. The set of transitions enabled in a marking m is denoted by Xm. Then the firing of the concurrent step Xm leads to the follower marking $m'(p)=g_p(\{f_{t,p}(m(pt_1),...,m(pt_{nt})) \mid t \in Xm\} \cup \{m(p)\})$.

It is easy to see how to instantiate LPNs and FPNs as KPPNs. For example, for FPNs, $K=\langle 0,1\rangle$, $f_{t,p}$ is the minimum mapping and g_p the maximum mapping. For extended FPNs, the functions $f_{t,p}$ are only defined partially according to the thresholds and the rule weights are encoded in the functions g_p.

These mappings for FPNs satisfy the following general properties:

- The firing of transitions increases the marking.
- The set of reachable markings is finite for arbitrary initial marking.

It is easy to see, considering the presented proofs in [1], that these two properties are actually enough to establish the existence of a unique stable state. Instead of considering functions which always increase markings, of course also functions which always decrease markings can be considered with the same result. On the other side, none of these properties alone is enough. If the firing of transitions increases the marking but the set of reachable markings is infinite for a certain initial marking then, in general, not each infinite firing sequence leads to a stable state. Consider for example mappings which always propagate the double marking of some input place to output places (for $K = \langle 0,\infty\rangle$). If the firing of transitions neither always increases nor always decreases the marking but the set of reachable markings is finite for all initial markings then also in general not each infinite firing sequence leads to a stable state. Consider for example two transitions which both have exactly one input and one output place and can be fired alternatively, where one transition just propagates the marking of its input place and the other transition propagates the difference between 1 and the marking of its input place (for $K=\langle 0,1\rangle$).

VI. Conclusions

It is a topic of a further research to identify other suitable knowledge propagation functions obtaining other Petri net dialects for DSS as well as other necessary and/or sufficient conditions for the existence of a unique stable state in the used dialects. The perspective of future research can be targeted to adaptive estimations and change of rule weights and thresholds resulting in more dynamic reasoning in DSS compared to the static estimation of those values by knowledge engineer as presented in the paper.

Acknowledgments

This work was supported in part by the Slovak Ministry of Education under Grants AV 1019, AV 103, DAAD SAMANET 07/2006, VEGA 1/0872/08, APVV-0618-07 TAWOP and by the German Academic Exchange Service within the SAMANET project. Martin Drozda was also supported by the German Research Foundation (SZ 51/24-2).

References

[1] F. Lehocki, G. Juhás, R. Lorenz, H. Szczerbicka, M. Drozda. Decision Support with Logical and Fuzzy Petri Nets. *Cybernetics and Systems: An International Journal*, 39: 587–610, 2008.

[2] T. Murata, V.S. Subrahmanian, T. Wakayana. A Petri Net Model for Reasoning in the Presence of Inconsistency. *IEEE Transactions on Knowledge and Data Engineering*, vol. 3, no. 3, pp.281-292, 1991.

[3] E. Coiera. Guide to health informatics. Hodder Arnold, London, 2003.

[4] K. Gurney. *Introduction to Neural Networks*. CRC PRESS, London, 1997.

[5] A. Keles. ESTDD: Expert System for Thyroid Disease Diagnostics. *Expert Systems with Applications*, vol. 34, pp. 242-246, 2008.

[6] X. Li, F. Lara-Rosano. Adaptive Fuzzy Petri Nets for Dynamic Knowledge Representation and Inference. *Expert Systems with Applications*, vol. 19, pp. 235-241, 2000.

[7] E. Coiera, J.I. Westbrook, J.C. Wyatt. The Safety and Quality of Decision Support Systems, In Haux R, Kulikowski C (eds): *IMIA Yearbook of Medical Informatics*, pp. 20-25, Schattauer Verlagsgesellschaft GmbH, Stuttgart, 2006.

[8] N.A. Shebl, B.D. Franklin, N. Barber. Clinical decision support systems and antibiotic use. *Pharmacy World & Science*, vol. 29, pp. 342-349, 2007.

[9] F. Girault, B. Pradin-Chézalviel, R. Valette. A logic for Petri Nets. *Journal européen des systèmes automatisés*, vol. 31, n. 3, pp. 525-542, 1997.

[10] J. Cardoso, R. Valette, B. Pradin-Chezalviel. Fuzzy Petri nets and linear logic. In *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, vol. 2, pp. 258-263, 1993.

[11] S.M. Chen, J. Ke, J. Chang. Knowledge Representation Using Fuzzy Petri Nets. *IEEE Transactions on Knowledge and Data Engineering*, vol. 2, no. 3, pp. 311-319, 1990.

[12] H.J. Zimmermann. *Fuzzy Set Theory and its Applications*. Kluwer Academic Publishers, Dordrecht, pp. 23-39, 1991.

[13] A. Konar, A.K. Mandal. Uncertainty Management in Expert Systems Using Fuzzy Petri Nets. *IEEE Transactions on Knowledge and Data Engineering*, vol. 8, no. 1, pp. 96-105, 1996.

[14] G.P. Purcell. What makes a good clinical decision support system. *British Medical Journal*, vol. 330, no. 7494, pp.740-741, 2005.

[15] C.G. Looney. Fuzzy Petri Nets for rule-based decision making. *IEEE Transactions on Systems, Man and Cybernetics*, vol. 18, no. 1, pp. 178-183., 1998.