Generalised fuzzy Petri nets for approximate reasoning in decision support systems

Zbigniew Suraj

Institute of Computer Science, University of Rzeszów, Poland
{zsuraj}@univ.rzeszow.pl

Abstract. The aim of this paper is to present a new class of Petri nets called generalised fuzzy Petri nets. The new class extends the existing fuzzy Petri nets by introducing two operators: triangular norms (t-norms) and t-conorms (s-norms), which are supposed to function as substitute for the *min* and *max* operators. To demonstrate the power and the usefulness of this model, an application of the generalised fuzzy Petri nets in the domain of train traffic control is provided. The new model is more flexible than the classical one as in the former class the user has the chance to define the input/output operators. The proposed approach can be used for knowledge representation and reasoning in decision support systems.

Key words: fuzzy Petri nets, knowledge representation, approximate reasoning, decision support systems

1 Introduction

Petri nets serve as a graphical and mathematical modelling tool applicable to many systems. Their graphical aspect allows representation of various interactions between discrete events more easily. However, the mathematical aspect allows formal modelling of these interactions and analysis of the modelled system properties [11],[17],[22].

The concept of a Petri net has its origin in C.A. Petri's dissertation [23]. In the last four decades, several extensions of Petri nets have been proposed improving such aspects as hierarchical nets, high level nets or temporal nets [11]. Currently, Petri nets are gaining a growing interest among people both in Artificial Intelligence due to its adequacy to represent the reasoning process as a dynamic discrete event system [1]-[9],[15]-[16],[18]-[21],[25]-[26] as well as in Molecular Biology with respect to their powerfulness and usefulness in modelling biological systems [10],[12]-[13],[24]. In 1988, C.G. Looney proposed in [16] so called "*F*uzzy *P*etri *N*ets (*FPNs*)". In his model logical propositions can be associated with Petri nets allowing for logical reasoning about the modelled system and its behaviour. The application of fuzzy Petri nets includes the design and implementation of decision support systems. In particular, they can be used for knowledge representation and modelling of reasoning processes in such systems. In this class of Petri net models not only well-known pieces of information

but also imprecise, vague and uncertain information is admissible and taken into account. Several authors have proposed different classes of fuzzy Petri nets. These models are based on different approaches combining Petri nets and fuzzy sets introduced by L. Zadeh in 1965 [27].

The aim of this paper is to further improve the fuzzy Petri net model for knowledge representation and reasoning and to overcome some deficiencies of former *FPN* approaches [2].

In the paper, we propose a new class of Petri nets called "*G*eneralised *F*uzzy *P*etri *N*ets (*GFPNs*)" for knowledge representation and reasoning in decision support systems. This model is a natural extension of fuzzy Petri nets. The t-norms and s-norms are introduced to the model as substitutes of *min* and *max* operators. The latter ones generalize naturally AND and OR logical operators with the Boolean values 0 and 1. The proposed model is not only more comfortable in terms of knowledge representation, but most of all it is more effective in the modelling process of approximate reasoning as in the new class of fuzzy Petri nets the user has the chance to define the input/output operators. The preliminary results of real-life data experiments using this model are promising. In order to demonstrate the modelling aspects of *GFPNs*, an application of generalised fuzzy Petri nets in the domain of train traffic control is provided.

The structure of this paper is as follows. Sect. 2 gives a brief introduction to fuzzy Petri nets. In Sect. 3 generalised fuzzy Petri nets formalism is presented. Sect. 4 describes an application of this class of *FPN* in the domain of train traffic control. In Sect. 5 conclusions are made.

2 Fuzzy Petri nets - basic definitions

One of the most known and applicable class of Petri nets in the domain of Artificial Intelligence are fuzzy Petri nets [3],[16]. They are a modification of classical Petri nets [17] relying on interpretation of net places as logical variables with values belonging to the closed interval [0,1] of all real numbers from 0 to 1 (0 and 1 are included). The concrete values of such variables represent a truth degree of statements assigned to the variables. Net transitions are interpreted as logical implications in which input places of a transition represent premises of a given implication corresponding to the transition whereas output places of the transition represent its conclusions. The definitions of input/output places of a transition in a net are given below. Similarly, we assume that the truth degree of a given implication belongs also to the closed interval of all real numbers from 0 to 1. Moreover, we assume that threshold values for all transitions in a given fuzzy Petri net are defined. The role of these values is to limit the possibility of transition firings. More precisely, if a logical premise value of a given transition is less than a threshold value of the transition then this transition is not possible to be fired.

In this paper we view a fuzzy Petri net as a decision support system based on specific rules of the form: IF *condition* THEN *action*, for which the condition is consumed and the action is produced each time the rule is used. In a decision

support system, the knowledge representation is based on a collection of rules. In a fuzzy Petri net, each transition may be seen as a rule which depicts a possible state change.

Definition 1. *A fuzzy Petri net (FP-net) is a tuple* $N = (P, T, S, I, O, \alpha, \beta, \gamma, M0)$*, where:*

- $P = \{p_1, p_2, \ldots, p_n\}$ *is a finite set of places,* $n > 0$*;*
- $T = \{t_1, t_2, \ldots, t_m\}$ *is a finite set of transitions,* $m > 0$*;*
- $S = \{s_1, s_2, \ldots, s_n\}$ *is a finite set of statements;*
- *the sets* P*,* T*,* S *are pairwise disjoint, i.e.,* $P \cap T = P \cap S = T \cap S = \emptyset$ *and card(P) = card(S);*
- $I : T \to 2^P$ *is the input function;*
- $O : T \to 2^P$ *is the output function;*
- $\alpha : P \to S$ *is the statement binding function;*
- $\beta : T \to [0, 1]$ *is the truth degree function;*
- $\gamma : T \to [0, 1]$ *is the threshold function;*
- $M0 : P \to [0, 1]$ *is the initial marking,*

and 2^P *denotes a family of all subsets of the set* P*.*

As for the graphical interpretation, places are denoted by circles and transitions by rectangles. The places are the nodes describing states (a place is a partial state) and the transitions depict the state changes. The function I describes the oriented arcs connecting places with transitions. It represents, for each transition t, the fragments of the state in which the system has to be, before the state change corresponding to t can occur. The function O describes the oriented arcs connecting transitions with places. It represents, for each transition t, fragments of the state in which the system will be after the occurrence of the state change corresponding to t. If $I(t) = \{p\}$ then a place p is called an input place of a transition t. Moreover, if $O(t) = \{p'\}$, then a place p' is called an output place of t. The initial marking $M0$ is an initial distribution of tokens in the places. It can be represented by a vector of dimension n of real numbers from the closed interval $[0, 1]$. For $p \in P$, $M0(p)$ is the token load of place p and represents a partial state of the system described by the fuzzy Petri net. This value can be interpreted as a truth value of a statement s bound with a given place p by means of the statement binding function α, i.e., $\alpha(p) = s$. Pictorially, the tokens are represented by means of grey "dots" together with the suitable real numbers placed inside the circles corresponding to appropriate places. We assume that if a truth value of a statement attached to a given place is equal to 0 then the token does not exist in the place. The number $\beta(t)$ is placed in a net picture under a transition t. Usually, this number is interpreted as a truth degree of an implication corresponding to a given transition t. The meaning of the threshold function γ is explained below.

Let N be a *FP*-net. A *marking* of N is a function $M : P \to [0, 1]$.

The fuzzy Petri net dynamics defines how new markings are computed from the current marking when transitions are fired (the corresponding state change

occurs). It describes the state changes of the decision support system modeled by the fuzzy Petri net.

Let N be a FP-net, $t \in T$, $I(t) = \{p_{i1}, p_{i2}, \ldots, p_{ik}\}$ be a set of input places for a transition t, and M - a marking of N.

A transition t is *enabled* for marking M if the minimum min for all input places of the transition t by M is positive and greater than or equal to the value of the threshold function γ to t, i.e.,

$$min(M(p_{i1}), M(p_{i2}), \ldots, M(p_{ik})) \geq \gamma(t) > 0 \text{ for each } p_{ij} \in I(t), j = 1, \ldots, k.$$

Only enabled transitions can be fired. Informally, firing the enabled transition t consists of removing (or not) dependently on the firing mode (it will be further discussed in detail) the token load of its input places by $I(t)$ (as the transition is enabled no negative token load will be obtained), and increasing the token load of all its output places by $O(t)$ without any alteration of the token loads of other places.

This informal definition points out the fact that firing the transition t is local in the sense that it only involves the tokens captured by $I(t)$ and $O(t)$ independently from the other remaining tokens of the current marking. Firing is based on two invisible primitives: removal of the tokens from input places and insertion of the tokens in output places. Transitions which do not share any places can be fired independently. This is why fuzzy Petri nets are a mathematical tool that can capture true concurrency.

There are two operating modes of the fuzzy Petri nets. In the first mode, each firing of an enabled transition t removes tokens from its input places and adds a token to each of its output place (if the token does not exist in the place yet). If in a given output place the token already exists then the fired transition does not place a new token. In the first case, a new value related to the generated token is computed as follows. After computing the minimum value from all values corresponding to input places of a fired transition, the computed value is timed by the value of function β corresponding to a fired transition. A new value is placed in all output places of a fired transition. In the second case, we assume that firing enabled transition does not remove tokens from its input places (copies of the tokens are only transmitted to output places of a fired transition). A final value of tokens is computed in an analogous way to the first case. In both cases, if in a given output place the token already exists then the fired transition computes a new value of the token as follows. The final value of a given output place of a fired transition is computed as maximum value from the one residing in the output place and the computed value of the fired transition.

Let $N = (P, T, S, I, O, \alpha, \beta, \gamma, M0)$ be a FP-net, $t \in T$, $I(t) = \{p_{i1}, p_{i2}, \ldots, p_{ik}\}$ be a set of input places for a transition t, $\beta(t)$ be a value of the truth degree function β corresponding to t and $\beta(t) \in (0, 1]$ (0 is not included), $\gamma(t)$ be a value of threshold function γ corresponding to t, and M be a marking of N. Moreover, let min, $*$, max denote the minimum, the algebraic product and the maximum, respectively.

Mode 1. If M is a marking of N enabling transition t and M' the marking derived from M by the firing transition t, then for each $p \in P$:

$$M'(p) = \begin{cases} 0 \text{ if } p \in I(t), \\ max(min(M(p_{i1}), M(p_{i2}), \ldots, M(p_{ik})) * \beta(t), M(p)) \text{ if } p \in O(t), \\ M(p) \text{ otherwise.} \end{cases}$$

In this mode, a procedure for computing the marking M' is as follows: (1) Tokens from all input places of the transition t are removed (the first condition from M' definition). (2) Tokens in all output places of t are modified in the following way: at first the value of minimum min for all input places of t is computed, then the product of the computed minimum for all input places of t and the value of truth degree function $\beta(t)$ is determined, and finally, a value corresponding to $M'(p)$ for each $p \in O(p)$ is obtained as a result of maximum for the computed product value and the current marking $M(p)$ (the second condition from M' definition). (3) Tokens in the remaining places of net N are not changed (the third condition from M' definition).

Mode 2. If M is a marking of N enabling transition t and M' the marking derived from M by the firing transition t, then for each $p \in P$:

$$M'(p) = \begin{cases} max(min(M(p_{i1}), M(p_{i2}), \ldots, M(p_{ik})) * \beta(t), M(p)) \text{ if } p \ \in O(t), \\ M(p) \text{ otherwise.} \end{cases}$$

The difference in the definition of a marking M' presented above (*Mode 2*) concerns input places of the fired transition t. In *Mode 1* tokens from all input places of the fired transition t are removed (*cf.* the first definition condition of *Mode 1*), whereas in *Mode 2* all tokens from input places of the fired transition t are copied (the second definition condition of *Mode 2*).

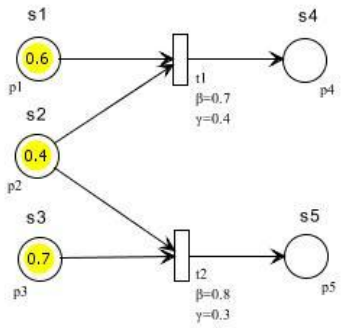


Fig. 1. A fuzzy Petri net with the initial marking before firing the enabled transitions

Example 1. Let us consider a fuzzy Petri net in Figure 1. For the net we have: the set of places $P = \{p1, p2, p3, p4, p5\}$, the set of transitions $T = \{t1, t2\}$,

the input function I and the output function O in the form: $I(t1) = \{p1, p2\}$, $I(t2) = \{p2, p3\}$, $O(t1) = \{p4\}$, $O(t2) = \{p5\}$. Moreover, there are: the set of statements $S = \{s1, s2, s3, s4, s5\}$, the statement binding function $\alpha : \alpha(p1) = s1$, $\alpha(p2) = s2$, $\alpha(p3) = s3$, $\alpha(p4) = s4$, $\alpha(p5) = s5$, the truth degree function β: $\beta(t1) = 0.7$, $\beta(t2) = 0.8$, the threshold function γ: $\gamma(t1) = 0.4$, $\gamma(t2) = 0.3$ and the initial marking $M0 = (0.6, 0.4, 0.7, 0, 0)$.

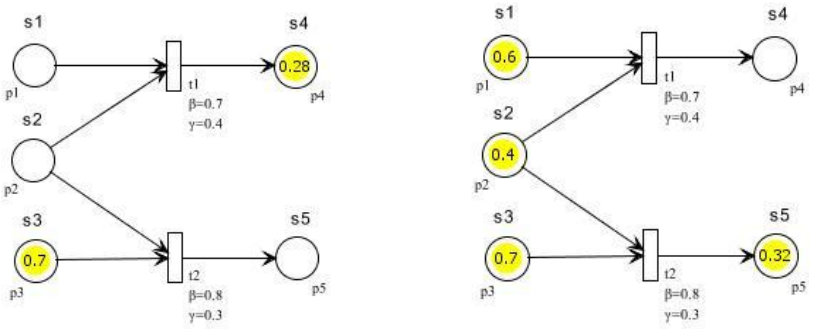


Fig. 2. An illustration of a firing rule: (a) the marking after firing $t1$, where $t2$ is disabled (*Mode 1*), (b) the marking after firing $t2$, where $t1$ and $t2$ are enabled (*Mode 2*)

Transitions $t1$ and $t2$ are enabled by the initial marking $M0$. Firing transition $t1$ by the marking $M0$ according to *Mode 1* transforms $M0$ to the marking $M1 = (0, 0, 0.7, 0.28, 0)$ (Figure 2(a)), and firing transition $t2$ by the initial marking $M0$ according to *Mode 2* results in the marking $M2 = (0.6, 0.4, 0.7, 0, 0.32)$ (Figure 2(b)).

3 Generalised fuzzy Petri nets

Now we are ready to define a new class of Petri net model called a generalised fuzzy Petri net. Before giving a formal definition of this model we remind some notions used later on. This section presents the main contribution to the paper.

A t-norm is defined as $t : [0, 1]\times[0, 1] \to [0, 1]$ such that, for each $a, b, c \in [0, 1]$: (1) it has 1 as the unit element, i.e., $t(a, 1) = a$; (2) it is monotone, i.e., if $a \leq b$ then $t(a, c) \leq t(b, c)$; (3) it is commutative, i.e., $t(a, b) = t(b, a)$; (4) it is associative, i.e., $t(t(a, b), c) = t(a, t(b, c))$.

More relevant examples of t-norms are: the minimum $t(a, b) = min(a, b)$ which is the most widely used, the algebraic product $t(a, b) = a*b$, the Łukasiewicz t-norm $t(a, b) = max(0, a + b - 1)$.

An s-norm (or a t-conorm) is defined as $s : [0, 1] \times [0, 1] \to [0, 1]$ such that, for each $a, b, c \in [0, 1]$: (1) it has 0 as the unit element, i.e., $s(a, 0) = a$, (2)

it is monotone, i.e., if $a \leq b$ then $s(a,c) \leq s(b,c)$, (3) it is commutative, i.e., $s(a,b) = s(b,a)$, and (4) it is associative, i.e., $s(s(a,b),c) = s(a,s(b,c))$.

More relevant examples of s-norms are: the maximum $s(a,b) = max(a,b)$ which is the most widely used, the probabilistic sum $s(a,b) = a + b - a * b$, the Łukasiewicz s-norm $s(a,b) = min(a+b,1)$.

Using the notions of t- and s-norms we formulate the definition a generalised fuzzy Petri nets as follows:

Definition 2. *A generalised fuzzy Petri net (GFP-net) is a tuple* $N' = (P, T, S, I, O, \alpha, \beta, \gamma, Op, \delta, M0)$*, where:*

- $P, T, S, I, O, \alpha, \beta, \gamma, M0$ *have the same meaning as in Definition 1;*
- Op *is a finite set of* t*-norms and* s*-norms called the set of operators;*
- $\delta : T \rightarrow Op \times Op \times Op$ *is the operator binding function.*

The operator binding function δ connects transitions with triples of operators $(op_{In}, op_{Out1}, op_{Out2})$. The first operator appearing in the triple is called the input operator, and two remaining ones are the output operators. The input operator op_{In} concerns the way in which all input places are connected with a given transition t (more precisely, statements corresponding to those places). However, the output operators op_{Out1} and op_{Out2} concern the way in which the next marking is computed after firing the transition t. In the case of input operator we assume that it can belong to one of two classes, i.e., t- or s-norm. This issue is more precisely discussed further on. In fuzzy Petri nets the operators minimum, algebraic product and maximum are usually used. As we know, the first two belong to the class of t-norms whereas the third belongs to the class of s-norms. In the new net model the input/output operators can be defined by a user of the model dependently on her/his needs. In general, selecting rule operators and parameters (if we consider so called parameterised families of t- and s-norms [14],[25]) depends on the rule type appearing in a given rule knowledge base as well as the quality of experimental data from which the rules are extracted [8],[21]. The behaviour of generalised fuzzy Petri nets is defined in an analogous way to the case of fuzzy Petri nets. It is worth emphasising that the definition of net marking is analogous to fuzzy Petri nets, although the definitions of transition rule and next marking are substantially modified.

Let N' be a *GFP*-net. A marking of N' is a function $M : P \rightarrow [0,1]$.

A transition $t \in T$ is *enabled* for marking M if the value of input operator op_{In} for all input places of the transition t by M is positive and greater than or equal to the value of threshold function γ corresponding to the transition t, i.e.,

$$op_{In}(M(p_{i1}), M(p_{i2}), \ldots, M(p_{ik})) \geq \gamma(t) > 0 \text{ for each } p_{ij} \in I(t), j = 1, \ldots, k.$$

Let $N' = (P, T, S, I, O, \alpha, \beta, \gamma, Op, \delta, M0)$ be a *GFP*-net, $t \in T$, $I(t) = \{p_{i1}, p_{i2}, \ldots, p_{ik}\}$ be a set of input places for a transition t and $\beta(t) \in (0,1]$. Moreover, let op_{In} be an input operator and op_{Out1}, op_{Out2} be output operators for the transition t.

Mode 1. If M is a marking of N' enabling transition t and M' the marking derived from M by firing transition t, then for each $p \in P$:

$$M'(p) = \begin{cases} 0 \text{ if } p \in I(t), \\ op_{Out2}(op_{Out1}(op_{In}(M(p_{i1}), M(p_{i2}), \ldots, M(p_{ik})), \beta(t)), M(p)) \\ \text{ if } p \in O(t), \\ M(p) \text{ otherwise.} \end{cases}$$

In this mode, a procedure for computing the marking $M^{'}$ is similar to appropriate procedure corresponding to fuzzy Petri nets and *Mode 1* presented above. The difference is that present procedure needs to set operators: op_{In}, op_{Out1}, op_{Out2} at first. Remaining stages of the procedure are analogous to the previous procedure concerning *Mode 1*.

Mode 2. If M is a marking of N' enabling transition t and M' the marking derived from M by firing transition t, then for each $p \in P$:

$$M'(p) = \begin{cases} op_{Out2}(op_{Out1}(op_{In}(M(p_{i1}), M(p_{i2}), \ldots, M(p_{ik})), \beta(t)), M(p)) \\ \text{ if } p \in O(t), \\ M(p) \text{ otherwise.} \end{cases}$$

The main difference in the definition of the marking $M^{'}$ presented above (*Mode 2*) and *Mode 1* is analogous to the fuzzy Petri nets.

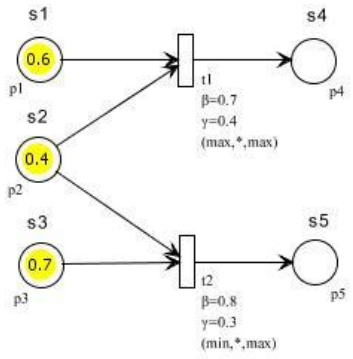
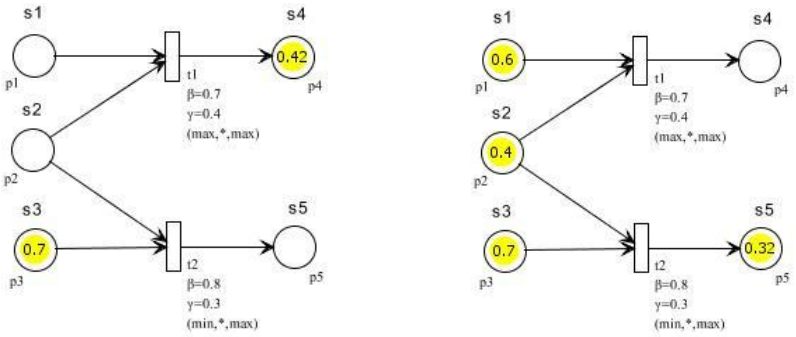


Fig. 3. A generalised fuzzy Petri net with the initial marking before firing the enabled transitions $t1$ and $t2$

Example 2. Let us consider a generalised fuzzy Petri net in Figure 3. For the net we have: the set of places P, the set of transitions T, the set of statements S, the input function I, the output function O, the statement binding function α, the truth degree function β, the threshold function γ, and and the initial marking $M0$ are described analogously to Example 1. Moreover, there are: the set of operators $Op = \{max, min, *\}$ and the operator binding function δ defined as follows: $\delta(t1) = (max, *, max)$, $\delta(t2) = (min, *, max)$.



(a) (b)

Fig. 4. An illustration of a firing rule: (a) the marking after firing $t1$, where $t2$ is disabled (*Mode 1*), (b) the marking after firing $t2$, where $t1$ and $t2$ are enabled (*Mode 2*)

Transitions $t1$ and $t2$ are enabled by the initial marking $M0$. Firing transition $t1$ by the marking $M0$ according to *Mode 1* transforms $M0$ to the marking $M1 = (0, 0, 0.7, 0.42, 0)$ (Figure 4(a)), and firing transition $t2$ by the initial marking $M0$ according to *Mode 2* results in the marking $M2 = (0.6, 0.4, 0.7, 0, 0.32)$ (Figure 4(b)).

4 Illustrating example

This section presents an application of *GFPN* in the domain of train traffic control [5]. The example is based on a simplified version of the real-life problem. We assume the following situation: a train B waits at a certain station for a train A to arrive in order to allow some passengers to change train A to train B. Now a conflict arises when the train A is late.

In this situation, the following alternatives can be taken into consideration:

- Train B waits for train A to arrive. In this case, train B will depart with delay.
- Train B departs in time. In this case, passengers disembarking train A have to wait for a later train.
- Train B departs in time, and an additional train is employed for late train $A's$ passengers.

To make a decision, several inner conditions have to be taken into account such as the delay period, the number of passengers changing trains, etc. The discussion regarding an optimal solution to the problem of divergent aims such as: minimization of delays throughout the traffic network, warranty of connections for the customer satisfaction, efficient use of expensive resources, etc. is disregarded at this point.

In order to describe the traffic conflict, we propose to consider the following three rules:

- IF $s2$ OR $s3$ THEN $s6$;
- IF $s1$ AND $s4$ AND $s6$ THEN $s7$;
- IF $s4$ AND $s5$ THEN $s8$,

where: $s1$ = "Train B is the last train in this direction today", $s2$ = "The delay of train A is huge", $s3$ = "There is an urgent need for the track of train B", $s4$ = "Many passengers would like to change for train B", $s5$ = "The delay of train A is short", $s6$ = "(Let) train B depart according to schedule", $s7$ = "Employ an additional train C (in the same direction as train B)", and $s8$ = "Let train B wait for train A".

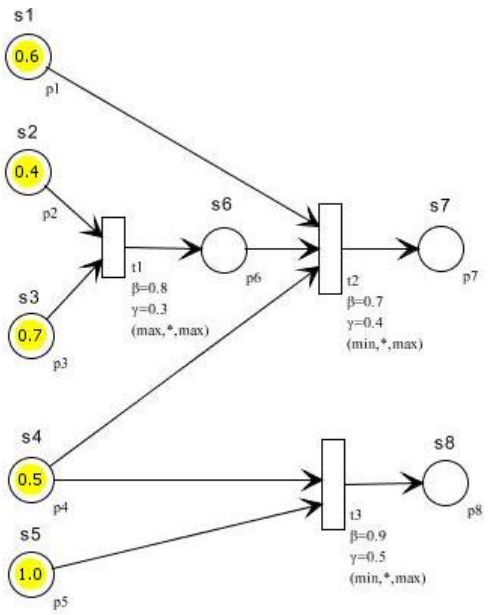


Fig. 5. An example of *GFPN* model of train traffic control

In Figure 5 the *GFPN* model corresponding to these rules, where the logical operators OR, AND are interpreted as *max* and *min*, respectively, is shown. Note that the places $p1, p2, p3$ and $p4$ include the fuzzy values 0.6, 0.4, 0.7 and 0.5 corresponding to the statements $s1, s2, s3$ and $s4$, respectively. In this example, the statement $s5$ attached to the place $p5$ is the only crisp and its value is equal to 1. By means of evaluation of the statements attached to the places from $p1$ up to $p5$, we observe that the transitions $t1$ and $t3$ can be fired. Firing these transitions according to the firing rules for the *GFPN* model allows the computation of the

support for the alternatives in question. In this way, the possible alternatives are ordered with regard to the preference they achieve from the knowledge base. This order forms the basis for further examinations and simulations and, in the end, for the dispatching proposal. If one chooses a sequence of transitions $t1t2$ then they obtain the final value, corresponding to the statement $s7$, equal to 0.35. In the second possible case (i.e., for the transition $t3$ only), the final value, corresponding now to the statement $s8$, equals 0.45.

5 Conclusions

The *GFPN* model combining the graphical power of Petri nets, possibilities of fuzzy sets and the theory of t-norms to model rule-based expert knowledge in a decision support system have been described in the paper. Having discussed basic notions from the fuzzy Petri net theory, *GFPN* formalism has been presented. Using a simple real-life example the suitability and the usefulness of the proposed approach for the design and implementation of decision support systems have been proved. Success of the elaborated approach looks promising with regard to alike application problems that could be solved similarly.

An experimental application called *PNES* on *IBM PC*, in *Java*, consisting of an editor and a simulator have been developed. The editor allows the inputting and editing of generalised fuzzy Petri nets, while the simulator that starts with a given initial marking and executes enabled transitions visualising reached markings. All figures and simulation results presented in the paper were produced by the application.

Acknowledgment. The author is grateful to anonymous referees for their helpful comments.

References

1. Cardoso, J., Valette, R., and Dubois, D.: "Fuzzy Petri nets: An overview". In: Proc. 13th IFAC World Congress, San Francisco, CA, U.S.A., 30 June-5 July 1996, pp. 443-448
2. Cardoso, J. and Camargo, H. (eds.): "Fuzziness in Petri Nets", Springer, Berlin 1999
3. Chen, S.M., Ke, J.S., and Chang, J.F.: "Knowledge representation using fuzzy Petri nets". IEEE Trans. on Knowledge and Data Engineering **2**(3), Sept. 1990, 311-319
4. Dubois, D. and Prade, H.: "Fuzzy Sets in Approximate Reasoning, Part 1: Inference with Possibility Distributions, Fuzzy Sets and Systems". Supplement to Vol. **100** (A selection of the most cited papers in Fuzzy Sets and Systems), 1999, 73-132
5. Fay, A. and Schnieder, E.: "Fuzzy Petri Nets for Knowledge Modelling in Expert Systems". In: Cardoso, J. and Camargo, H. (eds.), "Fuzziness in Petri Nets", Springer, Berlin 1999, pp. 300-318
6. Fryc, B., Pancerz, K., and Suraj, Z.: "On Approximate Petri nets with fuzzy operators in data classification process". In: Wakulicz-Deja, A. (ed.), Decision Support Systems, Inst. of Comp. Sci., Silesia University, Katowice 2009, pp. 53-61

7. Fryc, B., Pancerz, K., and Suraj, Z.: "Approximate Petri Nets for Rule-Based Decision Making". In: Proc. of 4th Int. Conf. on Rough Sets and Current Trends in Computing, Uppsala, Sweden, June 1-5, 2004, LNAI **3066**, Springer, Berlin 2004, pp. 733-742
8. Fryc, B., Pancerz, K., Peters, J.F., and Suraj, Z.: "On Fuzzy Reasoning Using Matrix Representation of Extended Fuzzy Petri Nets". Fundamenta Informaticae **60**(1-4), 2004, 143-157
9. Garg, M. L., Ahson, S. I., and Gupta, P.V.: "A fuzzy Petri net for knowledge representation and reasoning". Information Processing Letters, 1991, 165-171
10. Heiner, M., Donaldson, R., and Gilbert, D.: "Petri Nets for Systems Biology". Chapter 3, Jones & Bartlett Learning, LCC, 2010, pp. 61-97
11. Jensen, K. and Rozenberg, G.: "High-level Petri Nets". Springer, Berlin 1991
12. Kielbassa, J., Bortfeldt, R., Schuster, S., and Koch, I.: "Modelling of the U1 snRNP assembly pathway in alternative splicing in human cells using Petri nets". Computational Biology and Chemistry, 2008, 1-16
13. Kleijn, J., Koutny, M., and Rozenberg, G.: "Petri Nets for Biologically Motivated Computing". Scientific Annals of Computer Science **21**, 2011, 199-225
14. Klir, G.J. and Folger, T.A.: "Fuzzy Sets, Uncertainty and Information". Prentice-Hall: Englewood Cliffs, NJ, 1988
15. Li, X. and Lara-Rosano, F.: "Adaptive fuzzy Petri nets for dynamic knowledge representation and inference". Expert Systems with Applications. **19**, 2000, 235-241
16. Looney, C.G.: "Fuzzy Petri Nets for Rule-Based Decision-making. IEEE Trans. Syst., Man, Cybern. **18**-1, 1988, 178-183
17. Murata, T.: "Petri Nets: Properties, Analysis and Applications". Proc. of the IEEE, Vol.**77**, April 1989, pp. 541-580
18. Pedrycz, W. and Gomide, F.: "A generalized fuzzy Petri net model". IEEE Trans. on Fuzzy Systems. **2**-4, 1994, 295-301
19. Pedrycz, W.: "Generalized fuzzy Petri nets as pattern classifiers". Pattern Recognition Letters. **20**-14, December 1999, 1489-1498
20. Pedrycz, W. and Peters, J.F.: "Learning in fuzzy Petri nets". In: Fuzzy Petri Nets, Cardoso, J., and Sandri, S. (eds.), Physica-Verlag, Berlin, 1998
21. Peters, J.F., Skowron, A., Suraj, Z., Ramanna, S., and Paryzek, A.: "Modelling Real-Time Decision-Making Systems with Roughly Fuzzy Petri Nets". In: Proc. of the 6th European Congress on Intelligent Techniques and Soft Computing (EUFIT'98), Aachen, Germany, Sept. 7-10, 1998, pp. 985-989
22. Peterson, J.L.: "Petri net theory and the modeling of systems". Prentice-Hall, Inc., Englewood Cliffs, N.J., 1981
23. Petri, C.A.: "Kommunikation mit Automaten". Schriften des IIM Nr. **2**, Institut für Instrumentelle Mathematik, Bonn, 1962. English translation: Technical Report RADC-TR-65-377, Griffiths Air Force Base, New York, Vol.1, Suppl. 1, 1966
24. Pinney, J.W., Westhead, D.R., and McConkey, G.A.: "Petri net representations in systems biology". Biochemical Society Transactions **31**(6), 2003, 1513-1515
25. Suraj, Z.: "Parameterised Fuzzy Petri Nets for Approximate Reasoning in Decision Support Systems". The First International Conference on Advanced Machine Learning Technology and Applications (AMLTA 2012), December 8-10, 2012, Cairo, Egypt (submitted)
26. Suraj, Z. and Fryc, B.: "Timed Approximate Petri Nets". Fundamenta Informaticae **71**, 2006, 83-99
27. Zadeh, L.A.: "Fuzzy sets". Information and Control **8**, 1965, 338-353