Parameterised 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 provide a new class of Petri nets called parameterised fuzzy Petri nets. The new class extends the generalised fuzzy Petri nets by introducing two parameterised families of sums and products, which are supposed to function as substitute for the t-norms and s-norms. The power and the usefulness of this model on the base of parameterised fuzzy Petri nets application in the domain of train traffic control are presented. The new model is more flexible than the generalised one as in the former class the user has the chance to define the parameterised input/output operators. The proposed model can be used for knowledge representation and approximate reasoning in decision support systems.

Keywords: parameterised 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. The concept of a Petri net has its origin in C.A. Petri's dissertation [14]. In literature several extensions of Petri nets have been proposed [6],[10]. Currently, Petri nets are gaining a growing interest among people both in Artificial Intelligence due to its adequacy to represent the approximate reasoning process as a dynamic discrete event system [1]-[4],[9],[11]-[13],[15]-[17] as well as in Molecular Biology as a modeling tool to describe complex processes in developmental biology [5],[7].

In the paper [15], "*G*eneralised *F*uzzy *P*etri *N*ets ($GFPNs$)" for knowledge representation and approximate reasoning have been proposed. This model is a natural extension of fuzzy Petri nets introduced by C.G. Looney [9]. What is the main modification of this approach is that t-norms and s-norms are introduced to the model as substitutes of min and max operators. The latter ones generalise naturally AND and OR logical operators with the Boolean values 0 and 1.

The aim of this paper is to further improve the generalised fuzzy Petri net model. We propose a new class of Petri nets called "*P*arameterised *F*uzzy *P*etri *N*ets ($PFPNs$)". The main difference between $GFPN$ model and the model proposed here is that $PFPN$ model accepts two parameterised families of sums and products, which are supposed to function as substitute for the t-norms and

s-norms. The new model is more flexible than the $GFPN$ one as in the former class the user has the chance to define the parameterised input/output operators. There has been intensive research in the field of logical operators carried out for the last three decades, which involves the development of parameterised families of sums and products [8]. The preliminary results of real-life data experiments using the proposed model are promising. In order to demonstrate the power and the usefulness of this model, an application of parameterised fuzzy Petri nets in the domain of train traffic control is presented.

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

2 Preliminaries

In this section, a definition of generalised fuzzy Petri nets [15] and basic notions related to them are recalled.

Let $[0,1]$ be the closed interval of all real numbers from 0 to 1 (0 and 1 are included).

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 $s(a,b) = min(a+b, 1)$.

Definition 1. *A generalised fuzzy Petri net (GFP-net) is a tuple $N = (P, T, S, I, O, \alpha, \beta, \gamma, Op, \delta, M0)$, where: (1) $P = \{p_1, p_2, \ldots, p_n\}$ is a finite set of places, $n > 0$; (2) $T = \{t_1, t_2, \ldots, t_m\}$ is a finite set of transitions, $m > 0$; (3) $S = \{s_1, s_2, \ldots, s_n\}$ 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); (4) $I : T \to 2^P$ is the input function; (5) $O : T \to 2^P$ is the output function; (6) $\alpha : P \to S$ is the statement binding function; (7) $\beta : T \to [0,1]$ is the truth degree function; (8) $\gamma : T \to [0,1]$ is the threshold function; (9) Op is a finite set of t-norms and s-norms called the set of operators; (10) $\delta : T \to Op \times Op \times Op$ is the operator binding function; (11) $M0 : P \to [0,1]$ 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, 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, the 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 $[0, 1]$. For $p \in P$, $M0(p)$ is the token load of place p and represents a partial state of the system described by a generalised 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 binding function α, i.e., $\alpha(p) = s$. Pictorially, the tokens are represented by means of 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 over a transition t. Usually, this number is interpreted as a truth degree of an implication corresponding to a given transition t [2],[3]. The meaning of the threshold function γ is explained below. The operator binding function δ connects transitions with triples of operators $(op_{In}, op_{Out1}, op_{Out2})$. The first operator appearing in this triple is called the input operator, and two remaining ones are called the output operators. The input operator op_{In} belongs to one of the classes: t-norms or s-norms. It concerns the way in which all input places are connected to a given transition t (more precisely, statements corresponding to those places). Moreover, the output operator: op_{Out1} belongs to the class of t-norms and op_{Out2} belongs to the class of s-norms. Both of them concern the way in which the marking is computed after firing the transition t. This issue is explained more precisely below.

The generalised 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 modelled by the generalised fuzzy Petri net.

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

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, $\beta(t)$ 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 op_{In} be an input operator and op_{Out1}, op_{Out2} be output operators for the transition t.

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

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

(*Mode 1*) If M is a marking of N enabling the transition t and M' the marking derived from M by firing 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 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 input operator op_{In} for all input places of t is computed, then the value of output operator op_{Out1} for the value of input operator op_{In} 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 output operator op_{Out2} for the value of output operator op_{Out1} 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 the transition t and M' the marking derived from M by firing 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*) 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*).

Example 1. Consider a generalised fuzzy Petri net in Figure 1. For the net we have: the set of places $P = \{p_1, p_2, p_3, p_4, p_5\}$, the set of transitions $T = \{t_1, t_2\}$, the set of statements $S = \{s_1, s_2, s_3, s_4, s_5\}$, the input function I and the output function O in the form: $I(t_1) = \{p_1, p_2\}$, $I(t_2) = \{p_2, p_3\}$, $O(t_1) = \{p_4\}$, $O(t_2) = \{p_5\}$. Moreover, there are: the statement binding function $\alpha : \alpha(p_1) = s_1$, $\alpha(p_2) = s_2$, $\alpha(p_3) = s_3$, $\alpha(p_4) = s_4$, $\alpha(p_5) = s_5$, the truth degree function β: $\beta(t_1) = 0.7$, $\beta(t_2) = 0.8$, the threshold function γ: $\gamma(t_1) = 0.4$, $\gamma(t_2) = 0.3$, the set of operators $Op = \{max, min, *\}$, the operator binding function δ: $\delta(t_1) = (max, *, max)$, $\delta(t_2) = (min, *, max)$, and the initial marking $M0 = (0.6, 0.4, 0.7, 0, 0)$.

Transitions t_1 and t_2 are enabled by the initial marking $M0$. Firing transition t_1 by the marking $M0$ according to *Mode 1* transforms $M0$ to the marking $M^{'} = (0, 0, 0.7, 0.42, 0)$ (Figure 2(a)), and firing transition t_2 by the initial marking $M0$ according to *Mode 2* results in the marking $M^{"} = (0.6, 0.4, 0.7, 0, 0.32)$ (Figure 2(b)).

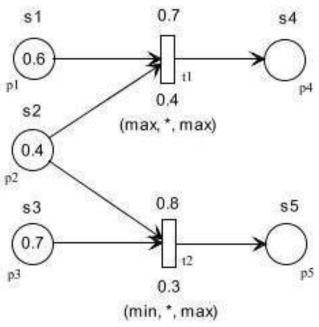


Fig. 1. A generalised fuzzy Petri net with the initial marking before firing the enabled transitions t_1 and t_2

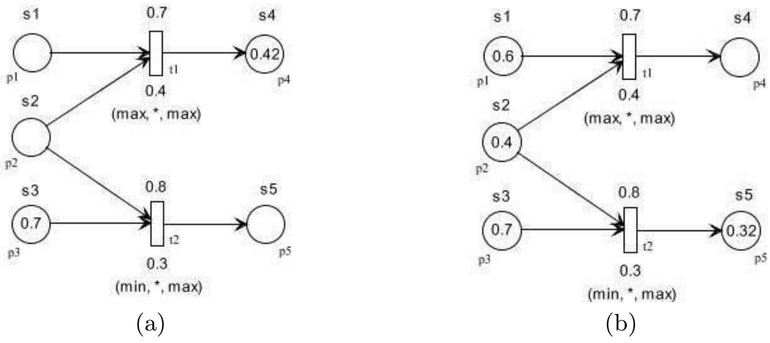


Fig. 2. An illustration of a firing rule: (a) the marking after firing t_1, where t_2 is disabled (*Mode 1*), (b) the marking after firing t_2, where t_1 and t_2 are enabled (*Mode 2*)

For more detailed information about *GFPNs* the reader is referred to [15] and [16].

3 Parameterised Fuzzy Petri Nets

Now we are ready to define a new class of Petri net model called parameterised fuzzy Petri nets. This model combines positive features of generalised fuzzy Petri nets and additional possibilities of parameterised families of sums and products [8]. This section presents the main contribution to the paper.

In Table 1 an exemplary list of parameterised families of sums and products is presented. For more details one shall refer to [8].

It is easy to observe that the first pair of parameterised families of sums $S(a, b, v)$ and products $T(a, b, v)$ from Table 1 for the parameter $v = 1$ correspond to the probabilistic sum $s(a, b) = a + b - a * b$ and the algebraic product $t(a, b) = a * b$, respectively.

Table 1. An exemplary list of parameterised families of sums and products

Sum S(a,b,v)	*Product T(a,b,v)*	*Range v*
$\frac{a+b-(2-v)*a*b}{1-(1-v)*a*b}$	$\frac{a*b}{v+(1-v)*(a+b-a*b)}$	$(0,\infty)$
$1-[max(0,(1-a)^{-v}+(1-b)^{-v}-1)]^{\frac{-1}{v}}$	$[max(0,a^{-v}+b^{-v}-1)]^{\frac{-1}{v}}$	$(-\infty,\infty)$
$\frac{a+b-a*b-min(a,b,1-v)}{max(1-a,1-b,v)}$	$\frac{a*b}{max(a,b,v)}$	$(0,1)$
$1-\log_v[1+\frac{(v^{1-a}-1)(v^{1-b}-1)}{v-1}$	$\log_v[1+\frac{(v^a-1)(v^b-1)}{v-1}$	$(0,\infty)$
$min[1,(a^v+b^v)^{\frac{1}{v}}$	$1-min[1,((1-a)^v+(1-b)^v)^{\frac{1}{v}}]$	$(0,\infty)$
$\frac{1}{1+[(\frac{1}{a}-1)^{-v}+(\frac{1}{b}-1)^{-v}]^{\frac{-1}{v}}}$	$\frac{1}{1+[(\frac{1}{a}-1)^{v}+(\frac{1}{b}-1)^{v}]^{\frac{1}{v}}}$	$(0,\infty)$

Definition 2. *A parameterised fuzzy Petri net (PFP-net) is a tuple $N' = (P, T, S, I, O, \alpha, \beta, \gamma, Op, \delta, M0)$, where (1) $P, T, S, I, O, \alpha, \beta, \gamma, \delta, M0$ have the same meaning as in Definition 1, and (2) Op is a finite set of parameterised families of sums and products called the set of parameterised operators.*

The behaviour of parameterised fuzzy Petri nets is defined in an analogous way as for the case of generalised fuzzy Petri nets. The main difference between these two models is that for a parameterised fuzzy Petri net instead of a concrete t-norm and s-norm we take a suitable pair of parameterised families of sums and products.

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

Let $N' = (P, T, S, I, O, \alpha, \beta, \gamma, Op, \delta, M0)$ be a *PFP*-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 $\beta(t) \in (0,1]$, $\gamma(t)$ be a value of threshold function γ corresponding to t, M be a marking of N', and v be a parameter value for a parameterised family of sums and products. Moreover, let op^v_{In} be an input parameterised operator and op^v_{Out1}, op^v_{Out2} be output parameterised operators with a parameter value v corresponding to t.

A transition $t \in T$ is *enabled* for marking M and a parameter value v, if the value of input parameterised operator op^v_{In} for the transition t is positive and greater than or equal to the value of threshold function γ corresponding to t and the parameter value v, i.e.,

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

(*Mode 1*) If M with a parameter value v is a marking of N' enabling a transition t and $M^{'}_v$ the marking derived from M with v by firing t, then for each $p \in P$:

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

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

(*Mode 2*) If M with a parameter value v is a marking of N' enabling transition t and $M_v^{'}$ the marking derived from M with v by firing t, then for each $p \in P$:

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

The difference in the definitions of the marking $M_v^{'}$ presented above (*Mode 2*) and *Mode 1* is analogous to the case of generalised fuzzy Petri nets.

Example 2. Consider a parameterised fuzzy Petri net in Figure 3(a). For the net we have: the set of places P, the set of transitions T, the input function I, the output function O, the set of statements S, the binding function α, the truth degree function β, the threshold function γ and the initial marking $M0$ described analogously to Example 1. Moreover, there are: the set of operators $Op = \{S(.), T(.)\}$ and the operator binding function δ defined as follows: $\delta(t_1) = (S(.), T(.), S(.))$, $\delta(t_2) = (T(.), T(.), S(.))$ with a relevant parameter value v.

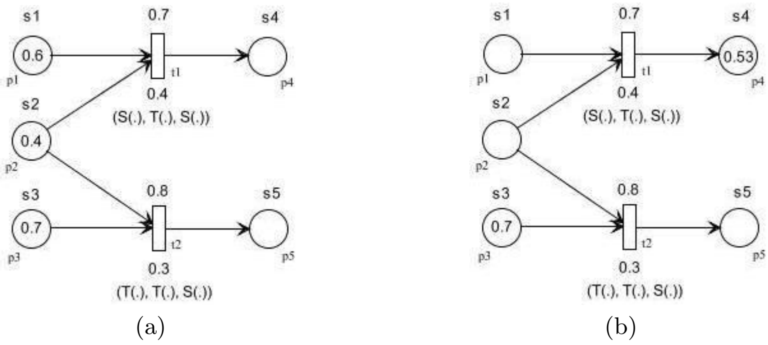


Fig. 3. (a) A parameterised fuzzy Petri net with the initial marking before firing the enabled transitions t_1 and t_2. (b) An illustration of a firing rule: the marking after firing t_1, where t_2 is disabled (*Mode 1*).

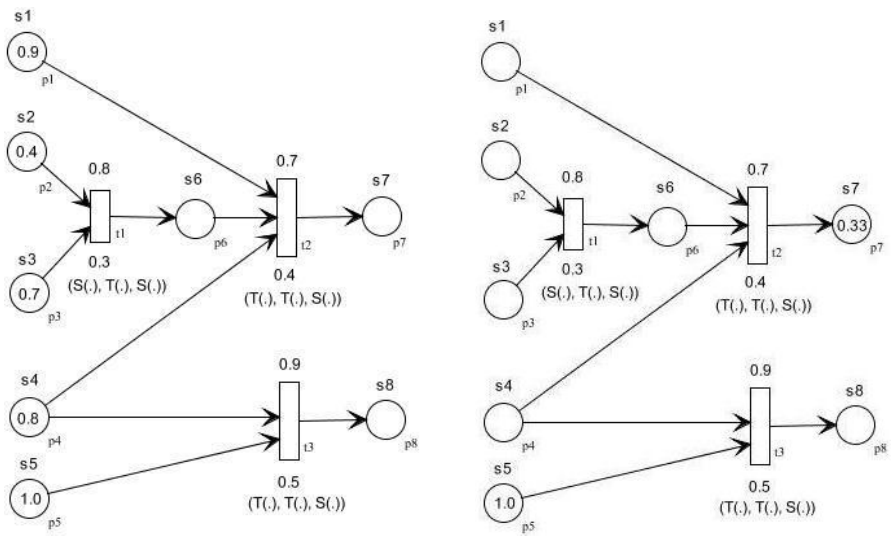
If we take, for instance, the first pair of parameterised families of sums $S(a, b, v)$ and products $T(a, b, v)$ from Table 1 and a parameter value $v = 1$, then $S(a, b, 1) = a+b-a*b$ and $T(a, b, 1) = a*b$. For the transition t_1 we have $S(0.6, 0.4, 1) = 0.6+0.4-0.24 = 0.76$ and $T(0.76, 0.7, 1) = 0.76*0.7 = 0.532$ by the initial marking $M0$ and $v = 1$. Because the global value for all input places of t_1 and $v = 1$ equals 0.76, hence it is positive and greater than $\gamma(t_1) = 0.4$. Thus, the transition t_1 is enabled by $M0$ and $v = 1$. Firing transition t_1 by the marking $M0$ and $v = 1$ according to *Mode 1* transforms $M0$ to the marking $M_1^{'} = (0, 0, 0.7, 0.53, 0)$ (Figure 3(b)). It is easy to check that by the initial marking $M0$ and $v = 1$ the transition t_2 is not enabled.

4 Illustrating Example

In this section we present an application of *PFPN* in the domain of train traffic control [15].

The considered 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.



(a) (b)

Fig. 4. (a) An example of *PFPN* model of train traffic control. (b) An illustration of a firing rule: the marking after firing a sequence of transitions $t_1 t_2$ (*Mode 1*)

To make a decision, several inner conditions have to be taken into account, for example: 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: (1) IF s_2 OR s_3 THEN s_6; (2) IF s_1 AND s_4 AND s_6 THEN s_7; (3) IF s_4 AND s_5 THEN s_8, where s_1: 'Train B is the last train in this direction today'; s_2: 'The delay of train A is huge'; s_3: 'There is an urgent need for the track of train B'; s_4: 'Many passengers would like to change for train B'; s_5: 'The delay of train A is short'; s_6: '(Let) train B depart according to schedule'; s_7: 'Employ an additional train C (in the same direction as train B)'; s_8: 'Let train B wait for train A'.

In Figure 4(a) the *PFPN* model corresponding to these rules, where the logical operators OR, AND are interpreted as the probabilistic sum $S(\cdot)$ and the algebraic product $T(\cdot)$, respectively, is shown. Note that the places p_1, p_2, p_3 and p_4 include the fuzzy values 0.9, 0.4, 0.7 and 0.8 corresponding to the statements s_1, s_2, s_3 and s_4, respectively. In this example, the statement s_5 attached to the place p_5 is the only crisp and its value is equal to 1. By means of evaluation of the statements attached to the places from p_1 up to p_5, we observe that the transitions t_1 and t_3 can be fired. Firing these transitions according to the firing rules for the *PFPN* 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 $t_1 t_2$ then they obtain the final value, corresponding to the statement s_7, equal to 0.33 (Figure 4(b)). In the second possible case (i.e., for the transition t_3 only), the final value, corresponding now to the statement s_8, equals 0.72.

5 Conclusions

In the paper a *PFPN* model has been proposed. This model combines the positive features of generalised fuzzy Petri nets and the additional possibilities of parameterised families of sums and products. Moreover, a *PFPN* formalism allows to handle complex and parameterised fuzzy rule bases. Using a simple real-life example suitability and usefulness of the proposed approach for the design and implementation of decision support systems have been shown. Success of the elaborated approach looks promising with regard to alike application problems that could be solved similarly.

One of the most important considerations in designing practical systems is the time representation, particularly in time-critical systems. Various approaches related to time within the context of Petri nets have been developed [1],[6],[10],[17]. In our further research we would like to consider a time factor together with the proposed net model and elaborate timed Petri net analysis methods.

Acknowledgment. The author is grateful to the anonymous referees for their critical comments and suggestions on the paper.

References

1. Cardoso, J., Camargo, H. (eds.): Fuzziness in Petri Nets. Springer, Berlin (1999)
2. Chen, S.M., Ke, J.S., Chang, J.F.: Knowledge representation using fuzzy Petri nets. IEEE Trans. on Knowledge and Data Engineering 2(3), 311–319 (1990)
3. Fryc, B., Pancerz, K., Suraj, Z.: Approximate Petri Nets for Rule-Based Decision Making. In: Tsumoto, S., Słowiński, R., Komorowski, J., Grzymała-Busse, J.W. (eds.) RSCTC 2004. LNCS (LNAI), vol. 3066, pp. 733–742. Springer, Heidelberg (2004)
4. Fryc, B., Pancerz, K., Peters, J.F., Suraj, Z.: On Fuzzy Reasoning Using Matrix Representation of Extended Fuzzy Petri Nets. Fundamenta Informaticae 60(1-4), 143–157 (2004)
5. Heiner, M., Donaldson, R., Gilbert, D.: Petri Nets for Systems Biology, ch. 3, pp. 61–97. Jones & Bartlett Learning, LCC (2010)
6. Jensen, K., Rozenberg, G.: High-level Petri Nets. Springer, Berlin (1991)
7. Kleijn, J., Koutny, M., Rozenberg, G.: Petri Nets for Biologically Motivated Computing. Scientific Annals of Computer Science 21, 199–225 (2011)
8. Klir, G.J., Folger, T.A.: Fuzzy Sets, Uncertainty and Information. Prentice-Hall, Englewood Cliffs (1988)
9. Looney, C.G.: Fuzzy Petri Nets for Rule-Based Decision-making. IEEE Trans. Syst., Man, Cybern. 18(1), 178–183 (1988)
10. Murata, T.: Petri Nets: Properties, Analysis and Applications. Proc. of the IEEE 77, 541–580 (1989)
11. Pedrycz, W.: Generalized fuzzy Petri nets as pattern classifiers. Pattern Recognition Letters 20(14), 1489–1498 (1999)
12. Pedrycz, W., Peters, J.F.: Learning in fuzzy Petri nets. In: Cardoso, J., Sandri, S. (eds.) Fuzzy Petri Nets. Physica-Verlag, Berlin (1998)
13. Peters, J.F., Skowron, A., Suraj, Z., Ramanna, S., 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 1998), Aachen, Germany, September 7-10, pp. 985–989 (1998)
14. 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)
15. Suraj, Z.: Generalised fuzzy Petri nets for approximate reasoning in decision support systems. In: Proc. of Int. Workshop on Concurrency, Specification, and Programming (CS&P 2012), September 28-30. LNCS (LNAI). Springer, Berlin (to appear, 2012)
16. Suraj, Z.: Knowledge Representation and Reasoning Based on Generalised Fuzzy Petri Nets. In: Proc. of the 12th Int. Conf. on Intelligent Systems Design and Applications (ISDA 2012), Kochi, India, November 27-29 (submitted, 2012)
17. Suraj, Z., Fryc, B.: Timed Approximate Petri Nets. Fundamenta Informaticae 71, 83–99 (2006)