Computers & Industrial Engineering 60 (2011) 505–510

Contents lists available at ScienceDirect

Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

Reversed fuzzy Petri nets and their application for fault diagnosis ☆

Hesuan Hu [a,b,*], Zhiwu Li [a], Abdulrahman Al-Ahmari [b]

[a] School of Electro-Mechanical Engineering, Xidian University, Xi'an, Shaanxi 710071, PR China
[b] Industrial Engineering Department, College of Engineering, King Saud University, Riyadh 11421, Saudi Arabia

ARTICLE INFO

Article history:
Received 20 August 2008
Received in revised form 13 May 2010
Accepted 3 December 2010
Available online 5 January 2011

Keywords:
Manufacturing system
Fuzzy reasoning
Fault diagnosis

ABSTRACT

An alterative approach to the backward reasoning is presented. In classical reasoning, both users and developers of many expert systems are dedicated to the forward reasoning. However, in many newly arising expert systems such as various diagnosis systems, the backward reasoning is of special interest and often preferable. In this paper, the fuzzy Petri nets are used to analytically represent the knowledge of fault diagnosis in manufacturing systems and an iterative algorithm based on max-algebra is used to deduce the consequence–antecedent relationship between their manifestation and antecedent. Finally, the legitimacy and efficiency of the proposed approach are proved and validated by an illustrative example.

1. Introduction

As does human intelligence, artificial intelligence has many aspects. The most significant one is its ability to reason (Chen, 1988; Looney, 1988; Woods, 1986; Zadeh, 1988). The artificial intelligence quest for reasoning systems has matured to the point where computers can display expert behavior (Arnould & Tano, 1995; Chen, Ke, & Chang, 1990; Deng & Chang, 1990). To assist in the construction of these expert systems, an increasing number of researchers and practitioners realize that much knowledge in the real world is fuzzy rather than precise (Cardoso, Valette, & Dubois, 1999; Chen, 2002; Lee, Liu, & Chiang, 2003; Yao, 1994). However, many fuzzy reasoning cases that can be handled easily by humans appear to be too difficult for computers. Therefore, there is an increasing demand to improve the capabilities of computers to handle fuzzy reasoning problems in the knowledge representation field (Bugarin & Barro, 1994; Deng & Chang, 1990; Gao, Zhou, & Tang, 2004). In the context of fuzzy reasoning, the mechanism of both forward and backward ones are explored, which are two fundamentally different approaches to reasoning (Bugarin & Barro, 1994). With forward reasoning, propositions are combined with rules to deduce new propositions. Forward reasoning is of special interest in situations where no specific goals are obtainable, where most rules and the antecedent portion to be considered are well known. However, in many cases useless propositions are also deduced, leading such a method less prospective. As opposed to forward reasoning, backward reasoning works in a consequence-driven way (Chen, 2000; Hu, Li, & Wang, 2003; Yuan, Shi, Liu, & Shang, 2008; Zhang & Cui, 2008). Such an inference engine limits it only to taking the information that might be helpful to meet the specified consequence. Due to its specific approach, backward reasoning is preferable since it is able to efficiently determine the values of the unknown variables. To develop sufficiently precise notations so as to implement the fuzzy reasoning automatically, many mathematical knowledge representation tools have been devised, such as the fuzzy production rules, semantic networks, frames, and fuzzy Petri nets. Among them, the fuzzy Petri nets are prospective for their ability to describe clearly synchronization and concurrence. The advantages that the fuzzy Petri nets over other tools are as follows. First, fuzzy Petri nets are formal and general graphical model of information flow in systems. Second, fuzzy Petri nets can be modified to fit any particular process. Third, fuzzy Petri nets can naturally model logical and mathematical arrays such as addition, subtraction, multiplication and division (Murata, 1989; Ribaric, 1988). Many approaches were proposed to extend Petri nets to fuzzy Petri nets. In Chen et al. (1990), Petri nets are initially modified to be fuzzy Petri nets to model fuzzy reasoning with propositional logic. With a high level algorithm, the fuzzy Petri nets implement the automated reasoning in a parallel way. Based on the method in Chen et al. (1990), fuzzy Petri nets are improved in both representation and reasoning ability. They can determine the antecedent–consequence relationship between one proposition and another. Recently, the max-algebra is used to formally implement the fuzzy reasoning automatically (Gao, Wu, & Zhou, 2000, 2003, 2004). Since fuzzy Petri nets are forward-directed graphs with bars, arcs and arrows, all the aforementioned methods are focused on the forward reasoning. However, the backward reasoning remains unexplored

☆ This manuscript was processed by Area Editor Satish Bukkapatnam.

* Corresponding author at: School of Electro-Mechanical Engineering, Xidian University, Xi'an, Shaanxi 710071, PR China.
E-mail address: huhesuan@gmail.com (H. Hu).

0360-8352/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cie.2010.12.003

using Petri net theory. This paper, much inspired by the work in Gao et al. (2000, 2003, 2004), aims to, propose the concept of reversed Petri nets for fuzzy reasoning such that the backward reasoning can be implemented automatically; present a max-algebra based algorithm so that the backward reasoning can be efficiently implemented; and illustrate an example to demonstrate their effectiveness and efficiency. The organization of this paper is as follows. Section 2 introduces basic notations on both Petri nets and fuzzy Petri nets, and their similarity and difference. Section 3 presents a formal backward reasoning algorithm based on fuzzy Petri nets. Section 4 illustrates the proposed approach through a manufacturing fault diagnosis expert system. Concluding remarks are given in Section 5.

2. Petri nets and fuzzy petri nets

Petri nets are 3-tuple $N = (P,T,F)$, where P and T are finite, nonempty, and disjoint sets (Murata, 1989). P is the set of places and T is the set of transitions. The flow relation between P and T is denoted by $F \subseteq (P \times T) \cap (T \times P)$. The preset of a node $x \in P \cup T$ is defined by ${}^{\cdot}x = \{y \in P \cup T | (y,x) \in F\}$. The postset of a node $x \in P \cup T$ is defined by $x^{\cdot} = \{y \in P \cup T | (x,y) \in F\}$. The preset (postset) of a set is defined by the union of the presets (postsets) of their elements. A marking of N is a mapping $M : P \to \mathcal{INN}$, where $\mathcal{INN} = \{0, 1, 2, \ldots\}$. (N,M) is called a net system or a marked net. A transition t is said to be enabled if each of its input place p is marked with at least $w(p,t)$ tokens, where $w(p,t)$ is the weight of arc from p to t. $M[t\rangle$ means that transition t is enabled under M. After t fires at M, a new marking M' results. This is denoted as $M[t\rangle M'$. The set of all markings reachable from a marking M_0, in symbols $R(N,M_0)$, is the smallest set in which $M_0 \in R(N,M_0)$ and $M' \in R(N,M_0)$ if both $M \in R(N,M_0)$ and $M_0[t\rangle M'$ hold. For a Petri net with n places and m transitions, its incidence matrix A is an $n \times m$ matrix of integers and its typical entry is given by $a_{ij} = a_{ij}^{+} - a_{ij}^{-}$, where $a_{ij}^{+} = w(i,j)$ is the weight of arc to place p_i from its input transition t_j and $a_{ij}^{-} = w(i,j)$ is the weight of arc from place p_i to its output transition t_j. An example of Petri net is shown in Fig. 1a and b.

In Fig. 1, one can verify that $P = \{p_1,p_2\}$, $T = \{t_1\}$, and $F = \{(p_1,t_1), (t_1,p_1)\}$. Moreover, $p_1^{\cdot} = \{t_1\}$, $t_1^{\cdot} = \{p_2\}$, $p_2^{\cdot} = \emptyset$, ${}^{\cdot}p_1 = \emptyset$, ${}^{\cdot}t_1 = p_1$, ${}^{\cdot}p_2 = t_1$. The initial marking is $M_0 = [1\,0]^T$ under which t_1 is enabled since $M(p_1,t_1) = 1 \geqslant w(p_1,t_1)$. After t_1 fires, one token is removed from its preceding place, i.e., p_1, and deposited into its succeeding place, i.e., p_2.

The fuzzy Petri nets are derived from Petri nets Gao et al., 2003. Formally, a fuzzy Petri net can be defined as a 5-tuple $FN = (P,T,\Delta,\Gamma,\Theta,U)$, where P and T are finite, nonempty, and disjoint sets. $P = \{p_1,p_2,\ldots,p_n\}$ is the set of places or propositions. $T = \{t_1,t_2,\ldots,t_m\}$ is a set of transitions or rules. Δ is an $m \times n$ input matrix of rules, where $\Delta = \{\delta_{i\times j}\}$, and $\delta_{i\times j} \in \{0,1\}$, i={1,2,…,n}, j={1,2,…,m}. $\delta_{i\times j} = 0$ if p_i is not an input of t_j, $\delta_{i\times j}$=1 if p_i is an input of t_j. Γ is an $m \times n$ output matrix of rules, where Γ={$\gamma_{i\times j}$}, $\gamma_{i\times j} \in \{0,1\}$, i={1,2,…,n},j={1,2,…,m}. $\gamma_{i\times j} = 0$ if p_i is not an output of t_j, $\gamma_{i\times j} = 1$ if p_i is an output of t_j. $\Theta = [\theta_1,\theta_2,\ldots,\theta_n]^T$ is an n-dimensional vector corresponding to places in P, where $\theta_i \in [0,1]$ means the certainty factor (CF) of p_i,$i = \{1,2,\ldots,n\}$. By $\Theta^0 = [\theta_1^0, \theta_2^0, \ldots, \theta_n^0]^T$, we mean the initially obtainable CF corresponding to each place p_i. Take the fuzzy Petri net shown in Fig. 2a as instance, $\Theta^0 = [\theta_1,0]^T$. $U = diag[\mu_1,\mu_2,\ldots,\mu_m]^T$ is an m×m diagonal matrix

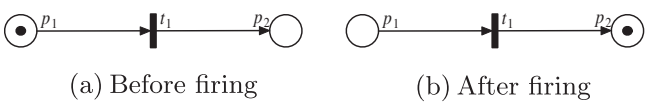


Fig. 1. A Petri net example.

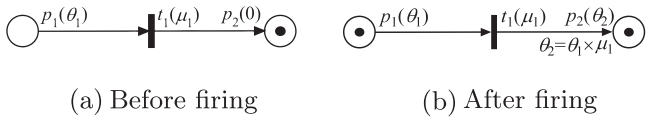


Fig. 2. A fuzzy Petri net example.

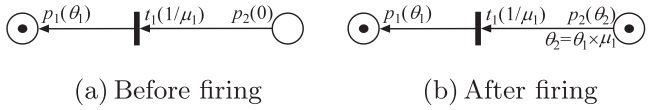


Fig. 3. The reversed fuzzy Petri nets.

such that $\mu_i = [0,\propto]$ means the CF of t_i,$i = \{1,2,\ldots,m\}$. Here, $\propto$ means an infinitely large number. Moreover, by $\rho = [\rho_1,\rho_2,\ldots,\rho_m]^T$, we mean the minimum CF corresponding to places preceding transition t_i, $i \in \{1,2,\ldots,m\}$. For example, in Fig. 2a, we have $\rho = [\theta_1]$ for transition t_1. In a fuzzy Petri net, a transition is said to be enabled if all of its input places are marked by a token with non-zero CF.

The major difference between Petri nets and fuzzy Petri nets are their firing rules. Petri nets address the properties of conflict because the activation of a transition will remove tokens from its input places while depositing one token into each of its output places, as shown in Fig. 1a and b. In a fuzzy Petri net, also a transition can be enabled to fire. However, in logical implication, the tokens would remain at their original positions and their copies would be sent to the output places. This logic is verified since in reasoning the antecedent portion remains verified although its consequence portion may already be proved. According to Chen et al. (1990), Gao et al. (2000), the CF of the resultant tokens can be calculated as Fig. 2b shows. The following definitions describe some fundamental characteristics of fuzzy Petri nets.

Definition 1. Let $N_1 = (P_1,T_1,F_1)$ and $N_2 = (P_2,T_2,F_2)$ be two net systems, respectively. N_2 is the reversed net of N_1 if $F_2 = \{(x,y) | (y,x) \in F\}$. Normally, this relationship is denoted by $N_1 = N_2^{-1}$ or $N_2 = N_1^{-1}$. Obviously, the incident matrix of N_1 is the negative version of incident matrix of N_2, i.e., $A_1 = -A_2$.

Definition 2. Let $FN_1 = (P_1,T_1,\Delta_1,\Gamma_1,\Theta_1,U_1)$ and $FN_2 = (P_2,T_2,\Delta_2, \Gamma_2,\Theta_2,U_2)$ be two net systems, respectively. FN_2 is the reversed net of FN_1 if $P_1 = P_2$, $T_1 = T_2$, $\Delta_1 = \Gamma_2$, $\Gamma_1 = \Delta_2$, $\Theta_1 = \Theta_2$, $U_1 = U_2^{-1}$. Normally, this relationship is denoted by $FN_2 = FN_1^{-1}$ or $FN_1 = FN_2^{-1}$.

The reversed nets corresponding to ones shown in Fig. 2a and b are illustrated in Fig. 3a and b, respectively.

Definition 3. Let p_{i_0} and p_{i_n} be two places. p_{i_n} is immediately reachable from p_{i_0} if ${}^{\cdot}p_{i_n} = p_{i_0}$. p_{i_n} is reachable from p_{i_0} if $\exists k \in \{0,1,2,\ldots,n-1\}$ such that any $p_{i_{k+1}}$ is immediately reachable from p_{i_k}. Given p, its immediate reachable set, denoted by $IRS(p)$, is all the places that are immediately reachable from p. Its reachable set, denoted by $RS(p)$, is all the places that are reachable from p.

Definition 4. p_j is an adjacent place of p_i if they are the two input places of a transition t. Moreover, the reachable set of p_i is denoted by $AP(p_i)$.

3. Backward reasoning using fuzzy petri nets

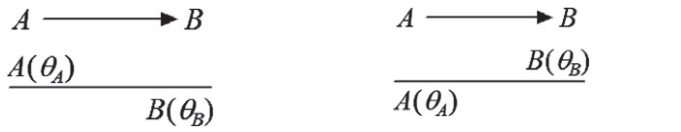
In Bugarin and Barro (1994), the fuzzy reasoning is distinguished into forward and the backward ones . The forward one is

defined as what shown in Fig. 4a. This means that if "*A* implies *B*" is verified, a sufficient condition for *B* to be true with a certainty factor θ_B is that *A* is true with a CF θ_A (Bugarin & Barro, 1994). Contrary to the forward reasoning, backward reasoning aims to prove that *B* is true with CF θ_B such that *A* is true with CF θ_A. The definition of back reasoning is shown in Fig. 4b.

In Looney (1988), fuzzy Petri nets are initially used to model knowledge representation in reasoning. Normally, propositions are denoted by places while rules are mapped into transitions. The token value in place $p_i \in P$ is denoted by θ_i, where $\theta_i \in [0,1]$. Place p_i means a proposition, and θ_i represents the CF. In Fig. 5a, a fuzzy reasoning with four antecedent propositions and two consequence propositions are shown.

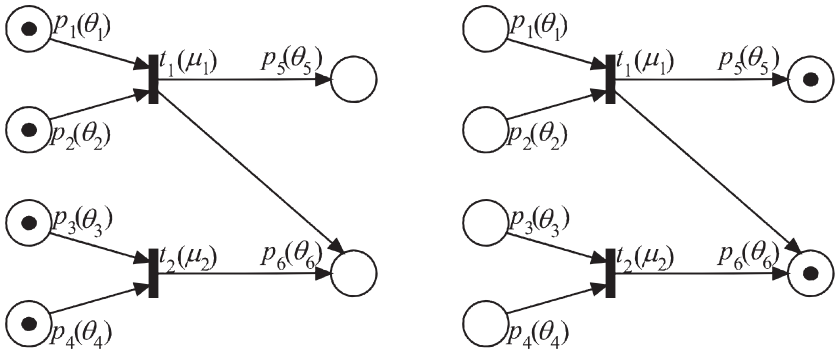
In order to formally describe the forward reasoning process, two operators derived from the max-algebra are introduced in Gao et al. (2000, 2003, 2004). The first one is $\oplus$. $A \oplus B = C$ where *A*, *B*, and *C* are all $n \times m$-dimensional matrices with a_{ij}, b_{ij}, and c_{ij} being their i_{th}-row and j_{th}-column entry, respectively, and $c_{ij} = \max\{a_{ij}, b_{ij}\}$. The second one is $\otimes$. $A \otimes B = C$ where *A*, *B* and *C* are all $n \times m$-dimensional matrices with a_{ij}, b_{ij}, and c_{ij} being their *i*th-row and *j*th-column entry, respectively, and $c_{ij} = \max\{a_{ik} \times b_{kj}\}$ where $k=\{1,2,\ldots,m\}$, $i=\{1,2,\ldots,n\}$, $j=\{1,2,\ldots,m\}$, respectively. The forward reasoning process is formally defined as follows. $\Theta(k+1) = \Theta(k) \bigoplus [(\Gamma \times U) \otimes \rho(k)]$, where $\rho(k) = neg\{\Delta^T neg[\Theta(k)]\}$. Notice that the operator *neg* is initially defined in Tzafestas and Caplovic (1997) whose representation can be described as follows. Let *x* be an *m*-dimensional vector, $neg(x) = 1_m - x$, where $1_m = [1_1, 1_2, \ldots, 1_m]^T$, and $x = [x_1, x_2, \ldots, x_m]^T$. Finally, an iterative algorithm is introduced to implement the forward reasoning automatically (Gao et al., 2000). This iterative algorithm is initially proposed in Gao et al. (2000) and further improved in Gao et al. (2003).

However, in many cases the fuzzy backward reasoning is preferable. For example, in a medical diagnosis system, normally the doctor knows the symptom first rather than the cause of disease. Expert systems with backward reasoning may help them implement such a diagnosis. Fig. 5b demonstrates the fuzzy Petri net model in which the CF of the consequence propositions are known whereas the antecedent propositions are not verified. Obviously,



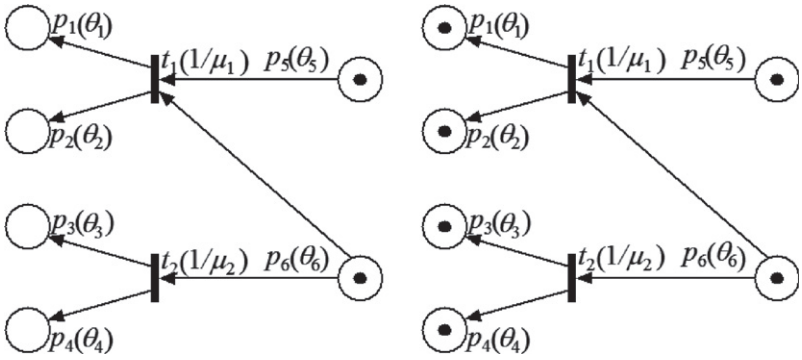
(a) Fuzzy forward reasoning (b) Fuzzy backward reasoning

Fig. 4. The fuzzy forward and backward reasoning.



(a) Forward reasoning fuzzy Petri net (b) Backward reasoning fuzzy Petri net

Fig. 5. Fuzzy Petri nets for forward and backward reasoning.



(a) The reversed fuzzy Petri net (b) The resultant fuzzy Petri net

Fig. 6. The reversed fuzzy Petri net and the resultant fuzzy Petri net.

Fig. 5b is only for forward reasoning. Reversing all arrows, which in the theory of Petri nets, is to obtain the reversed fuzzy Petri net, can do backward reasoning. The reversed net of Fig. 5b is shown in Fig. 6a. Suppose that p_y is an immediate reachable place of p_x and μ_i is the transition between them. So, the CF of p_y is equal to the one of p_x timed by the CF of t_i, i.e., μ_i. Correspondingly, in case the CF of p_y is determined, the one of p_x is equal to the CF of p_y divided by μ_i. This is equivalent to say that the CF of p_x is equal to the CF of p_y timed by $1/\mu_i$. Analogously, after firing a set of fuzzy rules in the reversed fuzzy Petri net, the iteration formula becomes $\Theta(k+1) = \Theta(k) \bigoplus [(\Delta \times U^{-1}) \otimes \rho(k)]$, where $\rho(k) = \mathrm{neg}\{\Gamma^{T} \cdot neg[\Theta(k)]\}$. Notice that Δ, Γ and U are matrices corresponding to Fig. 5b, which are called the original net of Fig. 6a. In Fig. 5b, let $\Theta^0 = [0,0,0,0,0.54,0.56]^T$, $U = diag[0.6,0.7]^T$, then

$$\Theta^1 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0.54 \\ 0.56 \end{pmatrix} \bigoplus \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} \times \begin{pmatrix} 0.6 & 0 \\ 0 & 0.7 \end{pmatrix}^{-1} \otimes \begin{pmatrix} 0.54 \\ 0.56 \end{pmatrix} = \begin{pmatrix} 0.9 \\ 0.9 \\ 0.8 \\ 0.8 \\ 0.56 \\ 0.54 \end{pmatrix}$$

The resultant fuzzy Petri net is obtained and shown in Fig. 6b, with Θ^1=[0.9,0.9,0.8,0.8,0.54,0.56]^T. Based on the structure of reversed fuzzy Petri nets, the algorithm to implement the backward reasoning is as follows. Step one, let $k = 0$; Step two, calculate the new CF by $\Theta(k+1) = \Theta(k) \bigoplus [(\Delta \times U^{-1}) \otimes neg\{\Gamma^T neg[\Theta(k)]\}]$. Notice that all the notations such as Θ, U, Δ, Γ are referred to the original fuzzy Petri net rather than its reversed version; Step three, $k = k + 1$, iterate Step two until $\Theta(k+1) = \Theta(k)$.

4. An illustrative example

The below example is taken from a practical manufacturing fault diagnosis system. In fact, it is an electric equipment manufacturing system. Since we are only concerned about its fault diagnosis aspect, the detailed description of the entire system is omitted. The production rules for such a system can be described as follows:

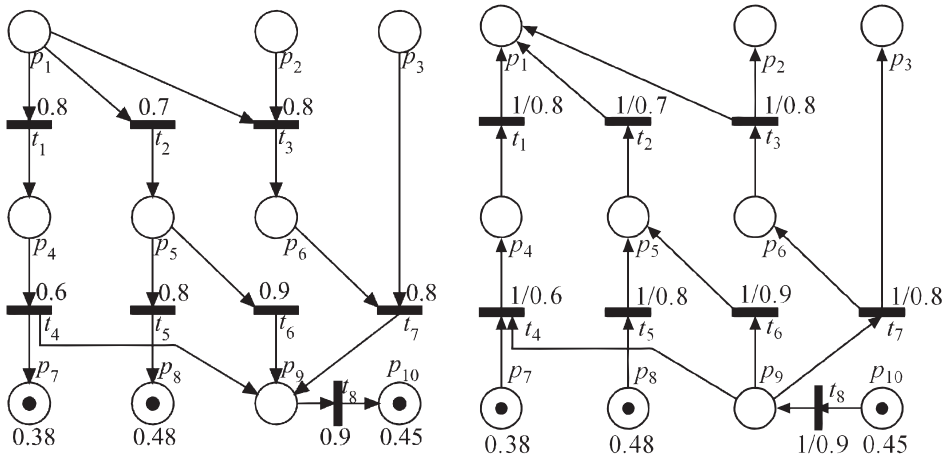
R_1: IF hardware fails THEN measuring resolution decreases (CF = 0.8).
R_2: IF hardware fails THEN control unit fails (CF = 0.7).
R_3: IF hardware fails AND software fails THEN servo system fails (CF = 0.8).
R_4: IF measuring resolution decreases THEN machining resolution decreases AND system is open-loop controlled (CF = 0.6).
R_5: IF control unit fails THEN tools are broken (CF = 0.8).
R_6: IF control unit fails THEN system is open-loop controlled (CF = 0.9).
R_7: IF servo system fails AND sensors are broken THEN system is open-loop controlled (CF = 0.8).
R_8: IF system is open-loop controlled THEN machining interrupts (CF = 0.9).

In Fig. 7a, a fuzzy Petri net is used to describe such a set of rules. Transitions t_{1-8} correspond to rules R_{1-8}, respectively. Their CF are {0.8,0.7,0.8,0.6,0.8,0.9,0.8,0.9}. The meanings of places are shown in Table 1. We know only the CF of the fault phenomena that are "machining resolution decreases", "tools are broken", and "machining interrupts", which correspond to p_7, p_9 and p_{10}, respectively. Numerically, we have $\theta_7 = 0.38$, $\theta_9 = 0.48$, and $\theta_{10} = 0.45$. Furthermore, the unknown CF are denoted as zeros at the initial stage. Obviously, the backward reasoning is necessary since the consequence portion rather than the antecedent portion is known. The both fuzzy Petri net and its reversed one are developed as shown in Fig. 7.

In Section 1, we describe fuzzy Petri nets in a quite detailed way. Through our description, one can easily understand the establishment process of the illustrative Petri net. For clarification, we present the following sentences to facilitate the understanding.

Table 1
Places and their meanings.

Place	Meaning	CF
p_1	Hardware fails	0
p_2	Software fails	0
p_3	Sensors are broken	0
p_4	Measuring resolution decreases	0
p_5	Control unit fails	0
p_6	Servo system fails	0
p_7	Machining resolution decreases	0.38
p_8	Tools are broken	0.48
p_9	System is open-loop controlled	0
p_{10}	Machining interrupts	0.45



(a) The fuzzy Petri net model (b) The reversed fuzzy Petri net model

Fig. 7. An illustrative example.

From rules R_1 and R_2, we know that p_4 and p_5 are the immediate reachable places of p_1. As shown in Fig. 7a, p_1 is introduced along with p_4 and p_5. Two transitions, i.e., t_1 and t_2, are introduced between p_1 and the latter two. From rule R_3, we know that p_6 is the immediate reachable place of both p_1 and p_2. We can introduce p_2 and p_6. t_3 is introduced between p_6 and p_1 as well as p_2. In the same way, we can establish the other part of the entire Petri net. The working mechanism of the established Petri net follows the one we described in Section 3.

Once the fuzzy Petri net is established, one can easily obtain the corresponding matrices. For clarity, we take the development of the output matrix, i.e., Γ, as an example. It should be a 10×8 matrix. Each element is denoted by γ_{ij}, where $i \in \{1,2,\ldots,10\}$ and $j \in \{1,2,\ldots,8\}$. Since p_1 is the output of t_1, t_2, and t_3, one can verify that $\gamma_{11} = 1$, $\gamma_{12} = 1$, and $\gamma_{13} = 1$. Since p_2, p_3, and p_4 are the output of t_3, t_7, and t_4, respectively, one can verify that $\gamma_{23} = 1$, $\gamma_{37} = 1$, and $\gamma_{44} = 1$. Similarly, we know that $\gamma_{55} = 1$, $\gamma_{56} = 1$, $\gamma_{67} = 1$, and $\gamma_{98} = 1$. Any other γ_{ij} is equal to zero since p_i is not the output of t_j.

From the definition of fuzzy Petri net and the structure of the reversed fuzzy Petri net shown in Fig. 7b, we have $\Theta^0 = [0000000.380.4800.45]^T$,

$$\Delta = \begin{pmatrix} 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0\\ 1&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0\\ 0&0&0&1&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&0&0&1&0&1&1&0\\ 0&0&0&0&0&0&0&1 \end{pmatrix} \Gamma = \begin{pmatrix} 1&1&1&0&0&0&0&0\\ 0&0&1&0&0&0&0&0\\ 0&0&0&0&0&0&1&0\\ 0&0&0&1&0&0&0&0\\ 0&0&0&0&1&1&0&0\\ 0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&0&0 \end{pmatrix}$$

$U^{-1} = diag[0.8,0.7,0.8,0.6,0.8,0.9,0.8,0.9]$, and $\rho^0 = [0000.380.48\ 000.45]^T$. According to the backward reasoning algorithm, we have

$$\Theta^1 = \begin{pmatrix} 0\\0\\0\\0\\0\\0\\0.38\\0.48\\0\\0.45 \end{pmatrix} \oplus \begin{pmatrix} 1&1&1&0&0&0&0&0\\ 0&0&1&0&0&0&0&0\\ 0&0&0&0&0&0&1&0\\ 0&0&0&1&0&0&0&0\\ 0&0&0&0&1&1&0&0\\ 0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&0&0 \end{pmatrix} \times \begin{pmatrix} 0.8&0&0&0&0&0&0&0\\ 0&0.7&0&0&0&0&0&0\\ 0&0&0.8&0&0&0&0&0\\ 0&0&0&0.6&0&0&0&0\\ 0&0&0&0&0.8&0&0&0\\ 0&0&0&0&0&0.9&0&0\\ 0&0&0&0&0&0&0.8&0\\ 0&0&0&0&0&0&0&0.9 \end{pmatrix}^{-1} \otimes \begin{pmatrix} 0\\0\\0\\0.38\\0.48\\0\\0\\0.45 \end{pmatrix}^T = \begin{pmatrix} 0\\0\\0\\0.6\\0\\0.38\\0.48\\0.5\\0.45 \end{pmatrix}$$

Similarly, we have $\Theta^2 = [000.6250.630.60.6250.380.480.50.45]^T$, $\Theta^3 = [0.860.780.6250.630.60.6250.380.480.50.45]^T$, and $\Theta^4 = [0.860.780.6250.630.60.6250.380.480.50.45]^T$. Since $\Theta^4 = \Theta^3$, the iteration terminates. Obviously, we can conclude that the hardware, software and sensor errors cause the fault with their CF being 0.86, 0.78, and 0.625, respectively. This should help engineers examine the fault causes in a correct order.

To the best of our knowledge, many existing approaches are on the basis of reachable set of Fuzzy reasoning Petri nets. Such methods require the enumeration of all possible paths so that the certainty factors can be properly evaluated. For this problem, we need to establish the one table for the immediate reachable set and reachable set as well as one table for adjacent places.

From Tables 2 and 3, there are four paths from the four goal places, i.e., $p_7 - p_{10}$, to the start place, i.e., p_1. They are $p_7 \to p_4 \to p_1$, $p_8 \to p_5 \to p_1$, $p_{10} \to p_9 \to p_5 \to p_1$, and $p_1 0 \to p_9 \to p_6 \to p_1$. We need to enumerate all these paths one by one to obtain different values of CF of p_1. From the first path, we know that the CF of p_4 is 0.38/0.6, i.e., 0.63. As consequence, the CF of p_1 is 0.63/0.8, i.e., 0.79. In the similar way, we can check the other three paths and obtain three different CF which are 0.86, 0.79, and 0.78. Among them, the maximum one is 0.86 which corresponds to the CF of p_1. Similarly, we can obtain the CF of p_2 and p_3. They are 0.78 and 0.625, respectively.

Compared to our proposed method, the conventional method can produce the same results; however, it requires the enumeration of all potential paths from the goal places to the start ones. In the worst case, their number can increase drastically with the scale of the problems. Our method does not the enumeration of all these paths, thus leading to a more efficient method.

Table 2
Immediate reachable and reachable sets for the net in Fig. 7.

p_i	$IRS(p_i)$	$RS(p_i)$
p_1	$\{p_4,p_5,p_6\}$	$\{p_4$–$p_{10}\}$
p_2	$\{p_6\}$	$\{p_6,p_9,p_{10}\}$
p_3	$\{p_9\}$	$\{p_9,p_{10}\}$
p_4	$\{p_7,p_9\}$	$\{p_7,p_9\}$
p_5	$\{p_8,p_9\}$	$\{p_8$–$p_{10}\}$
p_6	$\{p_9\}$	$\{p_9,p_{10}\}$
p_7	$\emptyset$	$\emptyset$
p_8	$\emptyset$	$\emptyset$
p_9	$\{p_{10}\}$	$\{p_{10}\}$
p_{10}	$\emptyset$	$\emptyset$

Table 3
Adjacent places for the net in Fig. 7.

p_i	p_k	AP_{ik}
p_1	p_4	$\emptyset$
p_1	p_5	$\emptyset$
p_1	p_6	$\{p_2\}$
p_2	p_6	$\{p_1\}$
p_3	p_9	$\{p_6\}$
p_4	p_7	$\emptyset$
p_4	p_9	$\emptyset$
p_5	p_8	$\emptyset$
p_5	p_9	$\emptyset$
p_6	p_9	$\{p_3\}$
p_9	p_{10}	$\emptyset$

5. Concluding remarks

In this paper, a particular kind of Petri nets, namely fuzzy Petri nets, is used to solve the backward reasoning problems arising in many areas. A max-algebra and reversed Petri nets based algorithm is proposed such that the reasoning process can be implemented formally and automatically. The proposed algorithm exploit the matrix similar to that used in the forward reasoning since they are mutually inversed. Finally, a practical rule based fault diagnosis in a manufacturing system is illustrated to demonstrate the effectiveness and efficiency of our method. Further research will focus on the following approaches. First, the weighted fuzzy reasoning should be investigated. Second, an accurate and effective index system is needed to evaluate and analysis the quality of resultant backward reasoning values. Third, combination with neural nets and probabilistic Petri nets should be investigated such that a more practical and sophisticated diagnostic expert system can be developed.

References

Arnould, T., & Tano, S. (1995). Interval-valued fuzzy backward reasoning. *IEEE Transactions on Fuzzy Systems, 3*(4), 425–437.

Bugarin, A. J., & Barro, S. (1994). Fuzzy reasoning supported by Petri nets. *IEEE Transactions on Fuzzy Systems, 2*(2), 135–149.

Chen, S. M. (1988). A new approach to handling fuzzy decision making problem. *IEEE Transactions on System, Man, and Cybernetics, 18*(6), 1012–1016.

Chen, S. M., Ke, J. S., & Chang, J. F. (1990). Knowledge representation using fuzzy Petri nets. *IEEE Transactions on Knowledge and Data Engineering, 2*(3), 311–319.

Chen, S. M. (2002). Weighted fuzzy reasoning using weighted fuzzy Petri nets. *IEEE Transactions on Knowledge and Data Engineering, 14*(2), 386–397.

Chen, S. M. (2000). Fuzzy backward reasoning using fuzzy Petri nets. *IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics* 30(6), 846–856.

Cardoso, J., Valette, R., & Dubois, D. (1999). Possibilistic Petri nets. *IEEE Transactions on Systems, Man, and Cybernetics – Part B, 29*(5), 573–582.

Deng, Y., & Chang, S. K. (1990). A G-net model for knowledge representation and reasoning. *IEEE Transactions on Knowledge and Data Engineering, 2*(3), 295–310.

Gao, M. M., Zhou, M. C., Huang, X. G., & Wu, Z. M. (2003). Fuzzy reasoning Petri nets. *IEEE Transactions on Systems, Man, and Cybernetics – Part A, 33*(3), 314–324.

Gao, M. M., Zhou, M. C., & Tang, Y. (2004). Intelligent decision making in disassembly process based on fuzzy reasoning Petri nets. *IEEE Transactions on Systems, Man, and Cybernetics – Part B, 34*(5), 2029–2034.

Gao, M. M., Wu, Z. M., & Zhou, M. C. (2000). A Petri net-based formal reasoning algorithm for fuzzy production rule-based systems. In *Proceedings of IEEE international conference on systems, man, and cybernetics* (Vol. 4, pp. 3093–3097).

Hu, C., Li, P., & Wang, H. (2003). Improved modeling algorithm of fuzzy petri net for fuzzy reasoning. In *Proceedings of the IEEE international conference on systems, man and cybernetics* (pp. 4992–4997).

Looney, C. G. (1988). Fuzzy Petri nets for rule based decision making. *IEEE Transactions on Systems, Man, and Cybernetics, 18*(1), 178–183.

Lee, J., Liu, K. F. R., & Chiang, W. L. (2003). Modeling uncertainty reasoning with possibilistic Petri nets. *IEEE Transactions on Systems, Man, and Cybernetics – Part B, 33*(2), 214–224.

Murata, T. (1989). Petri nets: Properties, analysis and applications. *Proceedings of IEEE, 77*(4), 480–541.

Ribaric, S. (1988). Knowledge representation scheme based on Petri net theory. *International Journal of Pattern Recognition and Artificial Intelligence, 2*(4), 691–700.

Tzafestas, S. G., & Caplovic, F. (1997). Petri net-based approach to synthesis of intelligent control systems for DEDS. In *Computer-assisted management and control of manufacturing systems* (pp. 323–351). New York: Springer-Verlag.

Woods, W. A. (1986). Important issues in knowledge representation. *Proceedings of IEEE, 74*(10), 1322–1334.

Yao, Y. (1994). A Petri net model for temporal knowledge representation and reasoning. *IEEE Transactions on Systems, Man, Cybernetics, 24*(9), 1374–1382.

Yuan, J., Shi, H. B., Liu, C. & Shang, W. L. (2008). Backward concurrent reasoning based on fuzzy petri nets. In *Proceedings of IEEE international conference on fuzzy systems* (pp. 832–837).

Zadeh, L. A. (1988). Fuzzy logic. *IEEE Computer Magazine, 21*(4), 83–93.

Zhang, B. & Cui, S. (2008). A parallel backward reasoning study using fuzzy petri net. In *Proceedings of international conference on computer science and software engineering* (pp. 315–319).