Weighted Fuzzy Spiking Neural P Systems

Jun Wang, Peng Shi, *Senior Member, IEEE*, Hong Peng, Mario J. Pérez-Jiménez, and Tao Wang

Abstract

Spiking neural P systems (SN P systems) are a new class of computing models inspired by the neurophysiological be-havior of biological spiking neurons. In order to make SN P sys-tems capable of representing and processing fuzzy and uncertain knowledge, we propose a new class of spiking neural P systems in this paper called weighted fuzzy spiking neural P systems (WFSN P systems). New elements, including fuzzy truth value, certain factor, weighted fuzzy logic, output weight, threshold, new firing rule, and two types of neurons, are added to the original definition of SN P systems. This allows WFSN P systems to adequately characterize the features of weighted fuzzy production rules in a fuzzy rule-based system. Furthermore, a weighted fuzzy backward reasoning algorithm, based on WFSN P systems, is developed, which can ac-complish dynamic fuzzy reasoning of a rule-based system more flexibly and intelligently. In addition, we compare the proposed WFSN P systems with other knowledge representation methods, such as fuzzy production rule, conceptual graph, and Petri nets, to demonstrate the features and advantages of the proposed tech-niques.

Index Terms

Spiking neural P systems (SN P systems), weighted fuzzy production rules, weighted fuzzy reasoning, weighted fuzzy spiking neural P systems (WFSN P systems).

I. Introduction

NATURAL Computing is a novel field of computer science research that uses computational paradigms inspired

from various well-known natural phenomena in physics, chemistry, and biology. There are several fields in Natural Computing that are now well established, such as genetic algorithms [1], [2], artificial neural networks [3]–[6], particle swarm

optimization [7]–[11], DNA-based molecular computing [12]. Membrane computing, which is part of molecular computing, was introduced in [13] under the assumption that the processes taking place in the compartmental structure of a living cell can be interpreted as computations. Since then, a large number of variants have been considered, and the devices of the models are generally called P systems [14].

A new class of distributed and parallel computing devices, spiking neural P systems (SN P systems), presented in [15], was inspired by the neurophysiological behavior of neurons sending electrical impulses (spikes) along axons to other neurons. An SN P system can be viewed as a set of neurons placed in the nodes of a directed graph whose arcs represent the synaptic connections among the neurons, and each neuron contains a number of copies of single object type as well as a lot of firing/spiking and forgetting rules. The rules in each neuron are used in a sequential manner, but neurons function with each other in parallel. More recently, a large number of variants of SN P systems have been developed (see, [16]–[23] and the references therein). In addition to distributed and parallel computing abilities, SN P systems inherently feature: 1) high understandability (due to their directed graph structure); 2) dynamic behavior (it seems to be suitable to model dynamic behaviors of a system on the basis of neuron's firing/spiking mechanisms); 3) synchronization (it seems to be suitable to describe concurrent events or activities); 4) nonlinearity (it is capable of dealing with nonlinear problem); and 5) nondeterministic. No doubt, these features will be attractive to a lot of real-world problems, such as process control, expert systems, fault diagnosing, investment advising systems, and even the new intelligent wireless sensor networks.

However, many successful applications have determined that there is a great deal of fuzzy and uncertain information in the aforementioned real-world areas and our computing models are often required to be capable of dealing with fuzzy and uncertain knowledge. It is well known that fuzzy knowledge retrieved by human experts or extracted by fuzzy neural networks (FNNs) is usually represented by fuzzy production rules [24]–[27]. Nevertheless, fuzzy production rules are not straightforward and their fuzzy reasoning is usually a complicated process. For this reason, some knowledge representation methods were developed, such as conceptual graph [28], semantic networks [29], and fuzzy Petri nets [30]–[33]. Meanwhile, weighted fuzzy production rules and weighted fuzzy logics were developed in order to process fuzziness and uncertainty in the knowledge base [34]–[39].

As stated previously, some significant features possessed by SN P systems are attractive to real-world applications. Unfortunately, existing SN P systems and their variants lack the ability to process fuzzy and uncertain knowledge so far. The main motivation behind our study is to build a bridge between SN P systems and various real-world problems such that the SN P systems can serve as a new model in real-world problems. For this reason, we will extend SN P systems so that they are capable

of dealing with fuzzy and uncertain information and representing weighted fuzzy production rules. In this paper, we propose weighted fuzzy spiking neural P systems (WFSN P systems) by incorporating some new elements into the original definition of SN P systems, including a new type of neuron, fuzzy truth value, certain factor, weighted fuzzy logic, output weight, threshold, and new firing/spiking rules. WFSN P systems are especially suitable from expressing weighted fuzzy production rules in graphical form. In addition, a fuzzy backward reasoning algorithm, based on the WFSN P systems, is developed. The main advantages offered by the proposed WFSN P systems can be summarized as follows.

1) Because of the graphical nature of WFSN P systems, the structure of weighted fuzzy production rules in a fuzzy knowledge base can be easily modeled and visualized, and the model is relatively simple and legible.
2) The dynamic firing mechanism of neurons in WFSN P systems are capable of carrying out dynamic fuzzy reasoning process more intelligently.
3) Based on the parallel computing mechanism of WFSN P systems, the proposed reasoning algorithm is an efficient reasoning algorithm with parallel reasoning ability.

The rest of this paper is organized as follows. In Section II, we provide the original definition of SN P systems, and propose WFSN P systems and simplified versions. In Section III, we perform weighted fuzzy knowledge representation based on the WFSN P systems. A fuzzy backward reasoning algorithm based on WFSN P systems for a rule-based system is presented in Section IV. In Section V, we compare WFSN P systems with other knowledge representation methods to show the advantages of our results. Finally, Section VI gives the conclusions.

II. Weighted Fuzzy Spiking Neural P Systems

A. Spiking Neural P Systems

In this section, we briefly review SN P systems in standard form and in a computing version (i.e., able to take an input and provide an output). (A more detailed description of SN P systems can be found in [16]–[22]).

Definition 1: A computing *SN P system* of degree $m \geq 1$ is a construct of the form

$$\Pi = (O, \sigma_1, \ldots, \sigma_m, \text{syn}, \text{in}, \text{out})$$

where

1) $O = \{a\}$ singleton alphabet (the object a is called *spike*);
2) $\sigma_1, \ldots, \sigma_m$ neurons, of the form $\sigma_i = (n_i, \mathbf{r}_i)$ with $1 \leq i \leq m$, where:
   a) $n_i \geq 0$ initial number of spikes contained in neuron σ_i;
   b) $\mathbf{r}_i$ finite set of rules of the following two forms:
      1) $E/a^c \rightarrow a; d$, where E is a regular expression over a, and c, $d \geq 0$ are natural numbers;
      2) $a^s \rightarrow \lambda$, where $s \geq 1$ is a natural number, with restriction that for each rule $E/a^c \rightarrow a; d$ of type (i) from $\mathbf{r}_i$, we have $a^s \notin L(E)$;
3) $\text{syn} \subseteq \{\sigma_1, \sigma_2, \ldots, \sigma_m\} \times \{\sigma_1, \sigma_2, \ldots, \sigma_m\}$ with $i \neq j$ for all $(\sigma_i, \sigma_j) \in \text{syn}$, $1 \leq i, j \leq m$ (*synapses* between neurons);
4) $\text{in}, \text{out} \in \{\sigma_1, \sigma_2, \ldots, \sigma_m\}$ *input* and *output* neurons, respectively.

In the aforementioned definition, the rule of type (1) is called the $firing/spiking\ rule$, and that of type (2) is called the $forgetting\ rule$. The firing mechanism of neurons in SN P systems can be described as follows. If a neuron σ_i contains k spikes, $a^k \in L(E)$ and $k \geq c$, the firing/spiking rule $E/a^c \rightarrow a; d \in \mathbf{r}_i$ in neuron σ_i is enabled and can be applied. This means that c spikes are consumed, $k - c$ spikes remain in the neuron, the neuron fires, and then it produces a spike after d time units. If $d = 0$, the spike is emitted immediately. In the case $d \geq 1$, if the rule is used at step t, the neuron is "closed" and "blocked" at steps $t, t+1, \ldots, t+d-1$, and it cannot receive new spikes from other neurons. At step $(t+d)$, the neuron emits a spike and becomes again open; hence, other neurons can receive the spike. The spike emitted by neuron σ_i is replicated and it goes to all neurons σ_j such that $(\sigma_i, \sigma_j) \in \text{syn}$ (each such neuron σ_j of those receives a spike). A forgetting rule $a^c \rightarrow \lambda$ is applicable to a neuron whether the neuron contains exactly c spikes, and then, all c spikes are removed. Note that if all rules of a system have $d = 0$, i.e., no delay is involved, the parameter d is omitted.

SN P systems are synchronized because a global clock is assumed, marking the time for the whole system. Besides, SN P systems are nondeterministic because two rules $E_1/a^{c_1} \rightarrow a; d_1$ and $E_2/a^{c_2} \rightarrow a; d_2$ can have $L(E_1) \cap L(E_2) \neq \emptyset$. Therefore, it is possible that two or more rules of the system can be enabled in a neuron. In this case, one of them is nondeterministically chosen to be used. Moreover, in each time unit, if a neuron can use a rule, the rule must be used. Each neuron deals with its spikes in a sequential manner, only using one rule in each time unit, but the rules are used in parallel for all neurons of the system.

An instantaneous description or a configuration at any instant of an SN P system is described by both the number of spikes in each neuron and the state of the neuron, or more precisely, by the number of steps to count down until it becomes open (this number is zero if the neuron is already open). The *initial configuration* is described by the number of spikes initially placed in each neuron, $n_1, n_2, \ldots, n_m$, with all neurons being open. A configuration is a *halting configuration* if all neurons are open and no rule of the system is applicable to it. Using the rules described previously, one can define *transitions* among configurations. We say that configuration C_1 yields configuration C_2 in one *transition step*, which is denoted by $C_1 \Rightarrow_\Pi C_2$, if we can pass from C_1 to C_2 by applying the rules from the system following the previous remarks.

A *computation* of Π is a (finite or infinite) sequence of configurations such that:

1) the first term of the sequence is the initial configuration of the system;
2) each noninitial configuration of the sequence is obtained from the previous configuration by a transition step;

3) if the sequence is finite (called *halting computation*), then the last term of the sequence is a halting configuration.

With any computation (halting or not), we can associate a *spike train*, which is a sequence of symbols 0, and 1, describing the behavior of the output neuron. If the output neuron spikes, then we write 1; otherwise, we write 0. In addition, we can also associate other forms of computation results according to different computing purposes, such as the distance between two consecutive steps when there are spikes that exit the system.

B. Weighted Fuzzy Spiking Neural P Systems

The introduction of fuzzy elements in P systems is an interesting and open issue. This paper will pay attention to this issue but be limited to the discussion of SN P systems for processing fuzzy and uncertain knowledge. Thus, we will extend the definition of SN P systems and propose a class of extended SN P system models, called WFSN P systems. The motivation for this is to model weighted fuzzy production rules in a fuzzy knowledge base and perform weighted fuzzy reasoning by using WFSN P systems in a more intelligent manner.

Definition 2: A computing *WFSN P system* of degree $m \geq 1$ is a construct of the form

$$\Pi = (O, N_p, N_r, \text{syn}, \text{IN}, \text{OUT})$$

where

1) $O = \{a\}$ singleton alphabet (the object a is called spike);
2) $N_p = \{\sigma_{p1}, \sigma_{p2}, \ldots, \sigma_{pm}\}$ *proposition neuron set*, where σ_{pi} is its ith *proposition neuron* associated with a fuzzy proposition in a fuzzy knowledge base, $1 \leq i \leq m$. Each proposition neuron σ_{pi} has the form $\sigma_{pi} = (\alpha_i, \vec{\omega}_i, \lambda_i, \mathbf{r}_i)$, where
   a) $\alpha_i \in [0, 1]$ *potential value* of pulse contained in proposition neuron σ_{pi}. α_i is used to express fuzzy truth value of a proposition associated with proposition neuron σ_{pi}.
   b) $\vec{\omega}_i = (\omega_{i1}, \omega_{i2}, \ldots, \omega_{is_i})$ output weight vector of the neuron σ_{pi}, where component $\omega_{ij} \in [0, 1]$ is the weight on jth output synapse (arc) of the neuron, $1 \leq j \leq s_i$, and s_i is the number of all output synapses (arc) of the neuron.
   c) $\mathbf{r}_i$ finite set of firing/spiking rules of the form $E/a^{\alpha} \rightarrow a^{\alpha}; d$, where $\alpha \in [0, 1]$, and $d \geq 0$ is a natural number. $E = \{\alpha \geq \lambda_i\}$ is called the firing condition, i.e., if $\alpha \geq \lambda_i$, then the firing rule will be enabled, where $\lambda_i \in [0, 1)$ is called the firing threshold.
3) $N_r = \{\sigma_{r1}, \sigma_{r2}, \ldots, \sigma_{rn}\}$ *rule neuron set*, where σ_{ri} is its ith *rule neuron* associated with a weighted fuzzy production rule in a fuzzy knowledge base, $1 \leq i \leq n$. Each rule neuron σ_{ri} has the form $\sigma_{ri} = (\alpha_i, \gamma_i, \vec{\nu}_i, \tau_i, \mathbf{r}_i)$, where
   a) $\alpha_i \in [0, 1]$ *potential value* of pulse contained in rule neuron σ_{ri}.
   b) $\gamma_i \in [0, 1]$ certain factor. It represents the strength of belief of a weighted fuzzy production rule associated with rule neuron σ_{ri}.
   c) $\vec{\nu}_i = (\nu_{i1}, \nu_{i2}, \ldots, \nu_{it_i})$ output weight vector of the neuron σ_{ri}, where component $\nu_{ij} \in [0, 1]$ is the weight on jth output synapse (arc) of the neuron, $1 \leq j \leq t_i$, and t_i is the number of all output synapses (arc) of the neuron.
   d) $\mathbf{r}_i$ finite set of firing/spiking rules of the form $E/a^{\alpha} \rightarrow a^{\beta}; d$, where $\alpha \in [0, 1]$, $\beta \in [0, 1]$, and $d \geq 0$ is a natural number. $E = \{\alpha \geq \tau_i\}$ is called the firing condition, i.e., if $\alpha \geq \tau_i$, then the firing rule will be enabled, where $\tau_i \in [0, 1)$ is called the firing threshold.
4) $\text{syn} \subseteq (N_p \times N_r) \bigcup (N_r \times N_p)$ *synapses* between both proposition neurons and rule neurons. Note that there are no synapse connections between any two proposition neurons or between any two rule neurons;
5) $\text{IN}, \text{OUT} \subseteq N_p$ *input neuron set* and *output neuron set*, respectively.

In the following, we illustrate how WFSN P systems are extended from the original definition of SN P systems. First, WFSN P systems consist of two types of neurons: proposition neurons and rule neurons. The intuitive purpose of introducing the two types of neurons is to express fuzzy propositions and weighted fuzzy production rules in a fuzzy knowledge base. Second, content of the neuron is now denoted by a fuzzy truth value instead of the number of spikes as in SN P systems. It can be interpreted as the (potential) value of spike from the viewpoint of biological neuron. For a proposition neuron, its content is used to express the fuzzy truth value of a fuzzy proposition associated with it. When a neuron fires and emits a spike, the (potential) value of the spike is transmitted into all successive neurons connected with the neuron. Third, each proposition neuron is assigned an output weight vector $\vec{\omega} = (\omega_1, \omega_2, \ldots, \omega_s)$. This indicates that the jth output synapse of the proposition neuron has the output weight ω_j. Therefore, when a proposition neuron fires and emits a spike with value α, its jth successive neuron will receive a spike with value $\alpha \otimes \omega_j$ from its output synapse, where "$\otimes$" is the multiplication operator of fuzzy truth values. Similarly, each rule neuron is also assigned an output weight vector $\vec{\nu} = (\nu_1, \nu_2, \ldots, \nu_t)$. When a rule neuron fires and emits a spike with value α, its jth successive neuron will receive a spike with value $(\alpha \oslash \nu_j) \otimes \gamma$, where "$\oslash$" is the division operator of fuzzy truth values. Fourth, because proposition neurons and rule neurons in WFSN P systems are used to characterize fuzzy propositions and weighted fuzzy production rules in a fuzzy knowledge base, respectively, both input neuron set and output neuron set only consist of proposition neurons, while rule neurons are interconnector of proposition neurons. Moreover, there are no direct connections between two proposition neurons or between two rule neurons. Fifth, WFSN P systems use the new firing condition $E = \{\alpha \geq \lambda_i\}$ or $E = \{\alpha \geq \tau_i\}$ rather than the original regular expression in SN P systems, and this controls whether the corresponding neuron fires or not. If a proposition neuron contains at least a spike and its value of spike is with $\alpha \geq \lambda_i$, then it fires. Likewise, if a rule neuron contains at least a spike and its value of spike is with $\alpha \geq \tau_i$, then it fires. Finally, when a neuron receives spikes from its several predecessor neurons, (potential) values of the received

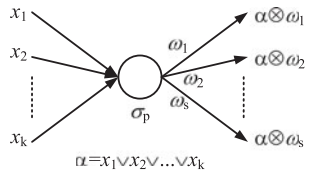

Fig. 1. Proposition neuron in S-WFSN P systems.

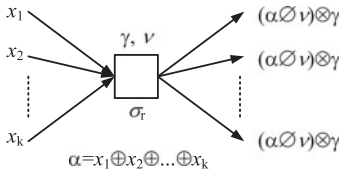

Fig. 2. Rule neuron in S-WFSN P systems.

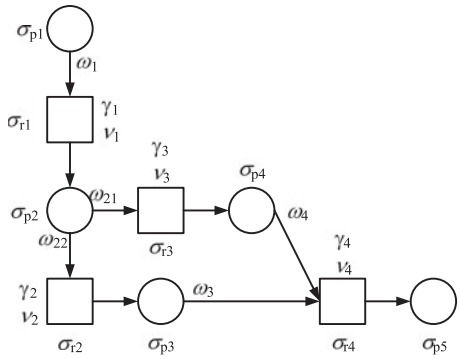

Fig. 3. Example of S-WFSN P systems: Π_0.

spikes will be calculated by using some logical operators unlike the neuron in SN P systems that simply accumulate the number of spikes received by it. The proposition neuron calculates (potential) values of spikes received by it from its predecessor neurons through logical "OR" operator "$\vee$," whereas rule neuron calculates potential values of spikes received by it through addition operator "$\oplus$" (see Figs. 1 and 2).

In addition to several aspects described previously, other original mechanisms in SN P systems are retained in WFSN P systems, for instance, time delay d, synchronization, nondeterminacy, and so forth.

As stated previously, the purpose of proposing the WFSN P systems is to model weighted fuzzy production rules in a knowledge base and develop a more intelligent weighted fuzzy reasoning algorithm. Some elements in WFSN P systems however, are redundant, such as time delay d, and firing thresholds λ_i and τ_i. Hence, we simplify the WFSN P systems by removing the redundant elements and denote the simplified version of WFSN P systems as S-WFSN P systems.

Compared with WFSN P systems, S-WFSN P systems have the following differences. First, time delay d is omitted; hence, all neurons are always open in S-WFSN P systems. Second, firing thresholds λ_i and τ_i in WFSN P systems are removed. Therefore, if a neuron contains at least a spike and its value of spike $\alpha_i > 0$, then it fires. Third, any neuron (proposition neuron or rule neuron), contains only a firing rule. Finally, each rule neuron has only an output weight factor ν_i, i.e., all its output synapses are assigned the same weight.

We describe the operating principle of S-WFSN P systems as follows. Initially, the system provides a spike for each input neuron in IN (or each input neuron in IN receives a spike from the environment as its input), where the value of the spike equals the fuzzy truth value of the corresponding proposition. When the system halts, the contents contained in output neurons (or results exported by output neurons) are regarded as its computing results. In S-WFSN P systems, each neuron contains only a firing/spiking rule and its firing principle is explained as follows. First, if a proposition neuron has k predecessor rule neurons and it receives k spikes from them, the (potential) value of the received k spikes is calculated as its content α through logical "OR" operator "$\vee$." When $\alpha > 0$, the neuron fires and its firing/spiking rule $E/a^\alpha \rightarrow a^\alpha$ can be applied. Applying the firing/spiking rule $E/a^\alpha \rightarrow a^\alpha$ means that the spike contained in the neuron is consumed, and then, it produces a spike with value α, which will be weighted by the corresponding weight factor. In this paper, we denote a proposition neuron by a circle, as shown in Fig. 1. Here, $\alpha = x_1 \vee x_2 \vee \ldots \vee x_k$, and its outputs are $\alpha \otimes \omega_i (i = 1, 2, \ldots, s)$, respectively. Second, if a rule neuron has k predecessor proposition neurons, then it fires and its firing/spiking rule $E/a^\alpha \rightarrow a^\beta$ can be applied when it receives k spikes from its all predecessor proposition neurons. The value of the received k spikes is calculated as its content α through addition operator "$\oplus$." Applying the firing/spiking rule $E/a^\alpha \rightarrow a^\beta$ means that the spike contained in the neuron is consumed, and then, it produces a spike with value β where $\beta = (\alpha \oslash \nu) \otimes \gamma$. In this paper, we denote a rule neuron by a rectangle, as shown in Fig. 2. Here, $\alpha = x_1 \oplus x_2 \oplus \ldots \oplus x_k$, and all its outputs are $(\alpha \oslash \nu) \otimes \gamma$.

Example 1: Fig. 3 shows an example of S-WFSN P systems, which can be formally described as follows. $\Pi_0 = (\{a\}, \{\sigma_{p1}, \sigma_{p2}, \sigma_{p3}, \sigma_{p4}, \sigma_{p5}\}, \{\sigma_{r1}, \sigma_{r2}, \sigma_{r3}, \sigma_{r4}\}, \text{syn}, \text{IN}, \text{OUT})$, where

1) $\sigma_{pj} = (\alpha_j, \omega_j, r_j)(j = 1, \ldots, 5)$ proposition neurons. The weights of proposition neurons σ_{p1}, σ_{p3}, and σ_{p4} are $\omega_1 = 1.0, \omega_3 = 0.8$, and $\omega_4 = 0.7$, respectively, while proposition neuron σ_{p2} has two weights $\omega_{21} = 1.0$ and $\omega_{22} = 1.0$;
2) $\sigma_{ri} = (\alpha_i, \gamma_i, \nu_i, r_i)(j = 1, \ldots, 4)$ rule neurons. The certain factors of rule neurons $\sigma_{r1}, \sigma_{r2}, \sigma_{r3}$, and σ_{r4} are $\gamma_1 = 0.85, \gamma_2 = 0.90, \gamma_3 = 0.95$, and $\gamma_4 = 0.90$, respectively. The weights of rule neurons $\sigma_{r1}, \sigma_{r2}, \sigma_{r3}$, and σ_{r4} are $\nu_1 = \omega_1 = 1.0, \nu_2 = \omega_{22} = 1.0, \nu_3 = \omega_{21} = 1.0$, and $\nu_4 = \omega_3 \oplus \omega_4 = 1.5$, respectively;
3) $\text{syn} = \{(\sigma_{p1}, \sigma_{r1}), (\sigma_{p2}, \sigma_{r2}), (\sigma_{p2}, \sigma_{r3}), (\sigma_{p3}, \sigma_{r4}), (\sigma_{p4}, \sigma_{r4}), (\sigma_{r1}, \sigma_{p2}), (\sigma_{r2}, \sigma_{p3}), (\sigma_{r3}, \sigma_{p4}), (\sigma_{r4}, \sigma_{p54})\}$;
4) $\text{IN} = \{\sigma_{p1}\}, \text{OUT} = \{\sigma_{p5}\}$.

In addition to modeling weighted fuzzy production rules by using WFSN P systems, we will develop a fuzzy reasoning algorithm based on WFSN P systems in this paper. In order to conveniently describe our weighted fuzzy reasoning algorithm, we first, define several concepts (terminologies) here. Let σ_{rs}

TABLE I
IMMEDIATE RULE-INCIDENCE TABLE OF ALL PROPOSITION NEURONS IN EXAMPLE 1

Proposition Neuron σ_{pk}	IBRIS(σ_{pk})	Rule Neuron σ_{ri}
σ_{p1}	ϕ	
σ_{p2}	$\{\sigma_{p1}\}$	σ_{r1}
σ_{p3}	$\{\sigma_{p2}\}$	σ_{r2}
σ_{p4}	$\{\sigma_{p2}\}$	σ_{r3}
σ_{p5}	$\{\sigma_{p3}, \sigma_{p4}\}$	σ_{r4}

be a rule neuron and σ_{pi}, σ_{pj}, and σ_{pk} be three proposition neurons.

Definition 3: Immediately backward rule incidence. If $(\sigma_{pi}, \sigma_{rs}) \in$ syn and $(\sigma_{rs}, \sigma_{pk}) \in$ syn, i.e., σ_{rs} is the interconnector between σ_{pi} and σ_{pk}, then σ_{pi} is called an immediately backward rule incidence of σ_{pk}.

In *Example* 1, σ_{p1} is an immediately backward rule incidence of σ_{p2}, and σ_{p2} is an immediately backward rule incidence of σ_{p3} and σ_{p4}, while σ_{p3} and σ_{p4} are immediately backward rule incidences of σ_{p4}.

Definition 4: Backward rule incidence. If σ_{pi} is an immediately backward rule incidence of σ_{pj}, and σ_{pj} is an immediately backward rule incidence of σ_{pk}, then σ_{pi} is called a backward rule incidence of σ_{pk}.

From *Example* 1, we can observe that σ_{p1} is a backward rule incidence of σ_{p3} and σ_{p4}, respectively. Moreover, σ_{p2} is a backward rule incidence of σ_{p5}.

Definition 5: Immediately backward rule incidence set. For a proposition neuron σ_{pk}, its immediately backward rule incidence set is defined as follows:

IBRIS$(\sigma_{pk}) = \{\sigma_{pi} \in N_p \,|\, \sigma_{pi}$ is an immediately backward rule incidence of $\sigma_{pk}\}$.

Definition 6: Immediately backward rule incidence table. For WFSN P systems, its immediately backward rule incidence table is defined as follows:

IRIT $= \{(\sigma_{pk}, \text{IBRIS}(\sigma_{pk}), \sigma_{ri}) \,|$ for $\forall\sigma_{pk} \in N_p$ and $\forall\sigma_{pj} \in$ IBRIS(σ_{pk}), $\exists\sigma_{ri} \in N_r$ such that $(\sigma_{pj}, \sigma_{ri}) \in$ syn and $(\sigma_{ri}, \sigma_{pk}) \in$ syn$\}$.

For *Example* 1, Table I gives the immediately backward rule-incidence table of Π_0. From Table I, we can see that IBRIS$(\sigma_{p2}) = \{\sigma_{p1}\}$, IBRIS$(\sigma_{p3}) = \{\sigma_{p2}\}$, IBRIS$(\sigma_{p4}) = \{\sigma_{p2}\}$, while IBRIS$(\sigma_{p5}) = \{\sigma_{p3}, \sigma_{p4}\}$, where σ_{p3} and σ_{p4} are adjacent proposition neurons with respect to rule neuron σ_{r4}.

III. WEIGHTED FUZZY KNOWLEDGE REPRESENTATION

A. Weighted Fuzzy Production Rules

The weighted fuzzy production rules that are discussed here are similar to conventional fuzzy production rules. However, a weight factor (or vector) is assigned to each proposition in the antecedent part in a fuzzy production rule, and a certainty factor is also assigned to the rule. Weight factor of a proposition indicates the degree of its importance contributing to the consequent when comparing with other proposition in the antecedent part. Obviously, when there is only one proposition in the antecedent of a fuzzy production rule, weight is meaningless for the rule.

Generally, weighted fuzzy production rules can be categorized into four types as follows:

Type 1: R_i: IF p_j THEN p_k (CF $= \gamma_i$), ω.

This type rule is a simple fuzzy production rule. In the rule R_i, p_j and p_k are propositions, γ_i is certainty factor of the rule, and ω is the weight of proposition p_j. Since p_j is only one proposition in the antecedent part of the rule R_i, its weight ω is meaningless for the rule. Therefore, we can set $\omega = 1$.

Type 2: R_i: IF p_1 AND p_2 AND ... AND p_{k-1} THEN p_k (CF $= \gamma_i$), $\omega_1, \omega_2, \ldots, \omega_{k-1}$

where $\omega_1, \omega_2, \ldots, \omega_{k-1}$ are the weights of propositions $p_1, p_2, \ldots, p_{k-1}$ in the antecedent part of the rule R_i, respectively. This is a composite conjunctive fuzzy production rule.

Type 3: R_i: IF p_1 THEN p_2 AND p_3 AND ... AND p_k (CF $= \gamma_i$), ω

where ω is the weight of proposition p_1 in the antecedent part of the rule R_i. Similarly, we can set $\omega = 1$ since weight ω in the rule is meaningless.

Type 4: R_i: IF p_1 OR p_2 OR ... OR p_{k-1} THEN p_k (CF $= \gamma_i$), $\omega_1, \omega_2, \ldots, \omega_{k-1}$.

This is a composite disjunctive fuzzy production rule. In the rule R_i, $\omega_1, \omega_2, \ldots, \omega_{k-1}$ are the weights of propositions $p_1, p_2, \ldots, p_{k-1}$ in the antecedent part of the rule R_i, respectively.

B. Mapping Weighted Fuzzy Production Rules Into Weighted Fuzzy Spiking Neural P Systems

In order to model weighted fuzzy production rules in a fuzzy knowledge base by using S-WFSN P systems, we have to map the aforementioned weighted fuzzy production rules into S-WFSN P systems. The basic principle is to map each fuzzy proposition in fuzzy knowledge base into one proposition neuron of S-WFSN P systems and to map each fuzzy production rule into one rule neuron or several rule neurons. Thus, the weighted fuzzy production rules of four types described previously and their fuzzy reasoning processes can be modeled as follows.

For a rule of Type 1, assume that the fuzzy truth value of propositions p_j is α_j and certainty factor of the rule is γ_i. Hence, the rule of Type 1 can be modeled by the following S-WFSN P system Π_1, as shown in Fig. 4(a):

$\Pi_1 = (\{a\}, \{\sigma_{pj}, \sigma_{pk}\}, \{\sigma_{ri}\}, \text{syn}, \text{IN}, \text{OUT})$, where

1) σ_{pj} and σ_{pk} two proposition neurons associated with fuzzy propositions p_j and p_k, respectively. $\sigma_{pj} = (\alpha_j, \omega_j, r_j)$ and $\sigma_{pk} = (\alpha_k, \omega_k, r_k)$. Since the antecedent part of the rule has only one proposition, set $\omega_j = 1$;
2) σ_{ri} rule neuron associated with the fuzzy production rule R_i. $\sigma_{ri} = (\alpha_i, \gamma_i, \nu_i, r_i)$, where $\nu_i = 1$;
3) syn $= \{(\sigma_{pj}, \sigma_{ri}), (\sigma_{ri}, \sigma_{pk})\}$, IN $= \{\sigma_{pj}\}$, OUT $= \{\sigma_{pk}\}$.

Furthermore, Fig. 4(a) shows the dynamic fuzzy reasoning process modeled by Π_1. The fuzzy reasoning process is

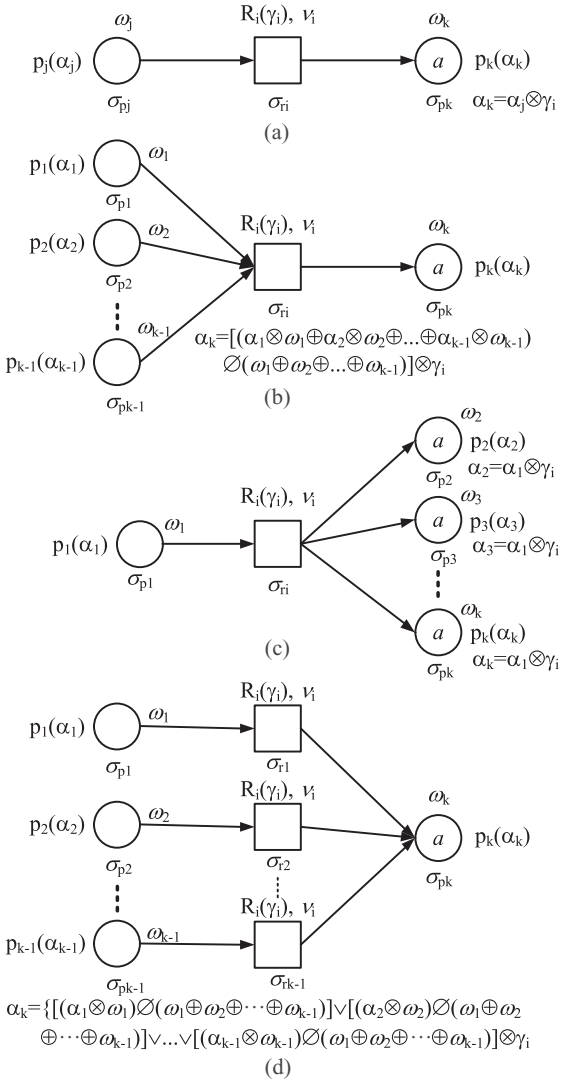

Fig. 4. Four weighted fuzzy rule presentations with S-WFSN P systems and their rule reasoning. (a) Type 1. (b) Type 2. (c) Type 3. (d) Type 4.

automatically carried out via three time units. Initially, a spike with value α_j is assigned into proposition neuron σ_{pj}. Thus, σ_{pj} fires and emits a spike with value α_j. Next, rule neuron σ_{ri} receives the spike and fires, and then, it sends a spike with value $\alpha_j \otimes \gamma_i$ to proposition neuron σ_{pk}. Finally, proposition neuron σ_{pk} receives the spike and its value of spike $\alpha_j \otimes \gamma_i$ is regarded as the result computed by the S-WFSN P system Π_1, as shown in Fig. 4(a).

For a rule of Type 2, assume that the fuzzy truth values of propositions $p_1, p_2, \ldots, p_{k-1}$ are $\alpha_1, \alpha_2, \ldots, \alpha_{k-1}$, respectively, and certainty factor of the rule is γ_i. Hence, the rule of Type 2 can be modeled by the following S-WFSN P system Π_2, as shown in Fig. 4(b):

$\Pi_2 = (\{a\}, \{\sigma_{p1}, \ldots, \sigma_{pk-1}, \sigma_{pk}\}, \{\sigma_{ri}\}, \text{syn}, \text{IN}, \text{OUT})$,

where

1) σ_{p1}, σ_{p2},..., σ_{pk-1}, and σ_{pk} proposition neurons associated with fuzzy propositions p_1, p_2,..., p_{k-1}, and p_k, respectively. $\sigma_{pj} = (\alpha_j, \omega_j, r_j)$, $j = 1, 2, \ldots, k-1, k$. $\omega_1, \omega_2, \ldots, \omega_{k-1}$ are weights of the propositions $p_1, p_2, \ldots, p_{k-1}$ in the antecedent part of the rule, respectively;
2) σ_{ri} rule neuron associated with the fuzzy production rule R_i. $\sigma_{ri} = (\alpha_{ri}, \gamma_i, \nu_i, r_i)$, where $\nu_i = \omega_1 \oplus \omega_2 \oplus \ldots \oplus \omega_{k-1}$;
3) $\text{syn} = \{(\sigma_{p1}, \sigma_{ri}), \ldots, (\sigma_{pk-1}, \sigma_{ri}), (\sigma_{ri}, \sigma_{pk})\}$, $\text{IN} = \{\sigma_{p1}, \sigma_{p2}, \ldots, \sigma_{pk-1}\}$, $\text{OUT} = \{\sigma_{pk}\}$.

Fig. 4(b) shows the dynamic fuzzy reasoning process modeled by Π_2. The fuzzy reasoning process is automatically performed as follows. Initially, a spike is provided for each proposition neuron σ_{pj} in IN and their values are $\alpha_1, \alpha_2, \ldots, \alpha_{k-1}$, respectively. These proposition neurons concurrently fire, and then, each of them emits a spike with value $\alpha_j \otimes \omega_j$. Next, rule neuron σ_{ri} receives spikes from these proposition neurons and values of the spikes are computed by addition operator "$\oplus$" as its content, i.e., $\alpha_{ri} = (\alpha_1 \otimes \omega_1) \oplus (\alpha_2 \otimes \omega_2) \oplus \ldots \oplus (\alpha_{k-1} \otimes \omega_{k-1})$. Thus, the rule neuron fires, and then, it sends a spike with value $[\alpha_{ri} \oslash \nu_i] \otimes \gamma_i$ to proposition neuron σ_{pk}. Finally, proposition neuron σ_{pk} will receive the spike as its content, as shown in Fig. 4(b). Therefore, the result computed by Π_2 is $\alpha_k = \{[(\alpha_1 \otimes \omega_1) \oplus (\alpha_2 \otimes \omega_2) \oplus \ldots \oplus (\alpha_{k-1} \otimes \omega_{k-1})] \oslash (\omega_1 \oplus \omega_2 \oplus \ldots \oplus \omega_{k-1})\} \otimes \gamma_i$.

For a rule of Type 3, assume that the fuzzy truth value of propositions p_1 is α_1 and certainty factor of the rule is γ_i. Hence, the rule of Type 3 can be modeled by the following S-WFSN P system Π_3, as shown in Fig. 4(c):

$\Pi_3 = (\{a\}, \{\sigma_{p1}, \ldots, \sigma_{pk-1}, \sigma_{pk}\}, \{\sigma_{ri}\}, \text{syn}, \text{IN}, \text{OUT})$,

where

1) σ_{p1}, σ_{p2},..., σ_{pk-1}, and σ_{pk} proposition neurons associated with fuzzy propositions p_1, p_2,..., p_{k-1}, and p_k, respectively. $\sigma_{p1} = (\alpha_1, \omega_1, r_1)$. Since p_1 is only one proposition in the antecedent part of the rule, set the weight $\omega_1 = 1$;
2) σ_{ri} rule neuron associated with the fuzzy production rule R_i. $\sigma_{ri} = (\alpha_{ri}, \gamma_i, \nu_i, r_i)$, where $\nu_i = 1$;
3) $\text{syn} = \{(\sigma_{p1}, \sigma_{ri}), (\sigma_{ri}, \sigma_{p2}), (\sigma_{ri}, \sigma_{p3}), \ldots, (\sigma_{ri}, \sigma_{pk})\}$, $\text{IN} = \{\sigma_{p1}\}$, $\text{OUT} = \{\sigma_{p2}, \sigma_{p3}, \ldots, \sigma_{pk}\}$.

Fig. 4(c) shows dynamic fuzzy reasoning process modeled by Π_3. The fuzzy reasoning process is automatically performed as follows. Initially, a spike is provided for proposition neuron σ_{p1} and its values is α_1. Thus, the proposition neuron fires, and then, it emits a spike with value α_1. Next, rule neuron σ_{ri} receives the spikes and fires, and then, it sends a spike with value $\alpha_1 \otimes \gamma_i$ to its all successive proposition neurons. Finally, proposition neurons $\sigma_{p2}, \sigma_{p3}, \ldots, \sigma_{pk}$ will receive the spike as their contents, as shown in Fig. 4(c). Therefore, the results computed by Π_3 are $\alpha_2 = \alpha_1 \otimes \gamma_i$, $\alpha_3 = \alpha_1 \otimes \gamma_i$, ..., $\alpha_k = \alpha_1 \otimes \gamma_i$.

For a rule of Type 4, assume that fuzzy truth values of propositions p_1, p_2,..., p_{k-1} are α_1, α_2,..., α_{k-1}, respectively, and the certainty factor of the rule is γ_i. Hence, the rule of Type 4 can be modeled by the following S-WFSN P system Π_4, as shown in Fig. 4(d):

$\Pi_4 = (\{a\}, \{\sigma_{p1}, \ldots, \sigma_{pk-1}, \sigma_{pk}\}, \{\sigma_{ri}\}, \text{syn}, \text{IN}, \text{OUT})$,

where

1) σ_{p1}, σ_{p2},..., σ_{pk-1}, and σ_{pk} proposition neurons associated with fuzzy propositions p_1, p_2,..., p_{k-1}, and p_k

respectively. $\sigma_{pj} = (\alpha_j, \omega_j, r_j)$, $j = 1, 2, \ldots, k-1, k$. $\omega_1, \omega_2, \ldots, \omega_{k-1}$ are the weights of the propositions $p_1, p_2, \ldots, p_{k-1}$ in the antecedent part of the rule, respectively;

2) $\sigma_{r1}, \sigma_{r2}, \ldots, \sigma_{rk-1}$ rule neurons associated with the fuzzy production rule R_i. $\sigma_{rj} = (\alpha_{rj}, \gamma_i, \nu_i, r_j), j = 1, 2, \ldots, k-1$, where $\nu_i = \omega_1 \oplus \omega_2 \oplus \ldots \oplus \omega_{k-1}$;
3) $\text{syn} = \{(\sigma_{p1}, \sigma_{r1}), (\sigma_{p2}, \sigma_{r2}), \ldots, (\sigma_{pk-1}, \sigma_{rk-1}), (\sigma_{r1}, \sigma_{pk}), \ldots, (\sigma_{rk-1}, \sigma_{pk})\}$, $\text{IN} = \{\sigma_{p1}, \sigma_{p2}, \ldots, \sigma_{pk-1}\}$, $\text{OUT} = \{\sigma_{pk}\}$.

Fig. 4(d) shows the dynamic fuzzy reasoning process modeled by Π_4. The fuzzy reasoning process is automatically performed as follows. Initially, a spike is provided for each proposition neuron σ_{pj} in IN and their values are $\alpha_1, \alpha_2, \ldots, \alpha_{k-1}$, respectively. These proposition neurons σ_{pj} concurrently fire, and then, each of them emits a spike with value $\alpha_j \otimes \omega_j$ into the corresponding rule neuron σ_{rj}, $j = 1, 2, \ldots, k-1$. Next, each rule neuron σ_{rj} receives the corresponding spike and fires, and then, it sends a spike with value $[(\alpha_j \otimes \omega_j) \oslash \nu_i] \otimes \gamma_i$ to proposition neuron σ_{pk}. Finally, proposition neuron σ_{pk} receives the spikes from the rule neurons, and the value of the spikes is computed by logical "OR" operator "$\vee$" as its content, i.e., $\alpha_k = \{[(\alpha_1 \otimes \omega_1) \oslash \nu_i] \otimes \gamma_i\} \vee \{[(\alpha_2 \otimes \omega_2) \oslash \nu_i] \otimes \gamma_i\} \vee \ldots \vee \{[(\alpha_{k-1} \otimes \omega_{k-1}) \oslash \nu_i] \otimes \gamma_i\}$. Therefore, the result computed by Π_4 is $\alpha_k = \{[(\alpha_1 \otimes \omega_1) \oslash (\omega_1 \oplus \omega_2 \oplus \ldots \oplus \omega_{k-1})] \vee [(\alpha_2 \otimes \omega_2) \oslash (\omega_1 \oplus \omega_2 \oplus \ldots \oplus \omega_{k-1})] \vee \ldots \vee [(\alpha_{k-1} \otimes \omega_{k-1}) \oslash (\omega_1 \oplus \omega_2 \oplus \ldots \oplus \omega_{k-1})]\} \otimes \gamma_i$.

As is well-known, fuzzy production rules are not straightforward and their fuzzy reasoning is usually a complicated process. However, from the aforementioned discussions, we can see that the structure of weighted fuzzy production rules modeled by the proposed WFSN P systems is visual and is easily comprehended due to its graphical nature. Moreover, owing to the parallel computing ability of WFSN P systems and the neuron's firing mechanism, the proposed WFSN P systems are able to complete fuzzy reasoning process concurrently and automatically, and the computing process only takes three time units.

IV. Weighted Fuzzy Reasoning Algorithm

In this section, we will present a weighted fuzzy reasoning algorithm based on S-WFSN P systems. From the previous discussion, we know that for a fuzzy knowledge base, proposition neurons express its all fuzzy propositions, while rule neurons model its weighted fuzzy production rules. Generally, we should provide the fuzzy truth values for a part of fuzzy propositions before reasoning, and the proposition neurons associated with the part of fuzzy propositions are in fact input neurons of the S-WFSN P system model. The goal of fuzzy reasoning method is to reason out the fuzzy truth values of other unknown fuzzy propositions (proposition neurons) from known fuzzy propositions (input neurons). These unknown fuzzy propositions are associated with output neurons of the S-WFSN P system model. Suppose we modeled the weighted fuzzy production rules of a fuzzy knowledge base by an S-WFSN P system model Π.

Based on S-WFSN P systems, we developed a weighted fuzzy reasoning algorithm, which is called the weighted fuzzy backward reasoning algorithm. The basis of the weighted fuzzy backward reasoning algorithm is the construction of a fuzzy "$\oplus$-OR" (Addition-OR) tree, which is similar to the algorithm in [40]. The tree uses a special data structure, i.e., a triple $(\sigma_{pk}, \text{IBRIS}(\sigma_{pk}), \alpha(\sigma_{pk}))$ is used to express a node in the tree, where σ_{pk} is the kth proposition neuron, $\text{IBRIS}(\sigma_{pk})$ is its immediately backward rule incidence set of σ_{pk}, and $\alpha(\sigma_{pk})$ is the fuzzy truth value of σ_{pk}. The weighted fuzzy backward reasoning algorithm (Algorithm 1) consists of three components: 1) building the immediately backward rule-incidence table IRIT (Algorithm 2); 2) generating fuzzy "$\oplus$-OR" tree (Algorithm 3); and 3) computing fuzzy truth values (Algorithm 4). Initially, each input neuron is assigned an initial fuzzy truth value. When the system halts, the system's outputs are fuzzy truth values in output neurons.

In the following, we explain the basic ideas behind the three components. First Algorithm 2 builds the immediately backward rule-incidence table IRIT according to Π. For each proposition neuron σ_{pk} in Π, the algorithm will generate its immediately backward rule-incidence set $\text{IBRIS}(\sigma_{pk})$ and determine the corresponding rule neuron σ_{ri} according to syn. Thus, $(\sigma_{pk}, \text{IBRIS}(\sigma_{pk}), \sigma_{ri})$ is composed of a tuple of IRIT. Second, Algorithm 3 is used to generate a fuzzy "$\oplus$-OR" tree of Π based on the built IRIT. This algorithm starts from output neurons and then, creates each node of fuzzy "$\oplus$-OR" tree according to backward connection relationship of proposition neurons. Each created node has the structure $(\sigma_{pk}, \text{IBRIS}(\sigma_{pk}), -)$, where the mark "$-$" denotes that the fuzzy truth value of the proposition neuron is unknown. In addition to the sides of first level in the tree, other sides are labeled by the certainty factors associated with fuzzy production rules. Finally, Algorithm 4 computes the fuzzy truth value of each nonterminal node of the fuzzy "$\oplus$-OR" tree. Here, fuzzy truth values of terminal nodes, which are associated with input neurons of Π, are known. Starting from terminal nodes, fuzzy truth values of all nonterminal nodes are backward computed on the basis of step 10 or step 12 of the algorithm.

The weighted fuzzy backward reasoning algorithm and its three component algorithms are listed in Tables II–V (Algorithms 1–4), respectively. In the following, we briefly discuss the computational complexities and convergence of the aforementioned algorithms. First, Algorithm 2 contains triple loop (m, n, and m times, respectively); therefore, its time complexity is $O(m^2 n)$, and space complexity is $O(m^2)$. By analyzing Algorithm 3, we can conclude that its time complexity is $O(m^2)$, and space complexity is $O(m^2)$. Similarly, time complexity and space complexity of Algorithm 4 are $O(m^2)$ and $O(m^2)$, respectively. Therefore, the time complexity of Algorithm 1 is $O(m^2 n)$, while its space complexity is $O(m^2)$. Finally, we analyze the convergence of these algorithms. From the descriptions of these algorithms, we know that the roles of Algorithms 2 and 3 are to construct a table $IRIT$ and a tree T, respectively. Therefore, the convergence of the proposed weighted fuzzy reasoning algorithm mainly depends on the convergence of Algorithm 4. Let l be the largest of the numbers of proposition neurons in all paths of Π from input neurons to output neurons. We easily conclude from Algorithm 4 that the algorithm can deduce the

TABLE II
ALGORITHM 1: WEIGHTED FUZZY BACKWARD REASONING ALGORITHM

Input: $\{\alpha(\sigma_{pi}) \mid \sigma_{pi} \in IN\}$
Output: $\{\alpha(\sigma_{pk}) \mid \sigma_{pk} \in OUT\}$
Begin (1) Call IRIT building algorithm; (2) Call fuzzy "$\oplus$-OR" tree generating algorithm; (3) Call fuzzy truth value computing algorithm; End.

TABLE III
ALGORITHM 2: IRIT BUILDING ALGORITHM

Input: Π
Output: IRIT
Begin (1) IRIT $\leftarrow \phi$; (2) for $k \leftarrow 1$ to m do (3) for $i \leftarrow 1$ to n do (4) if $(\sigma_{ri}, \sigma_{pk}) \in syn$ then (5) IBRIS$(\sigma_{pk}) \leftarrow \phi$; (6) for $j \leftarrow 1$ to m do (7) if $(\sigma_{pj}, \sigma_{ri}) \in syn$ then (8) IBRIS$(\sigma_{pk})) \leftarrow$ IBRIS$(\sigma_{pk})) \bigcup \{\sigma_{pj}\}$; (9) endif; (10) endfor; (11) IRIT $\leftarrow$ IRIT $\bigcup \{(\sigma_{pk}, \text{IBRIS}(\sigma_{pk}), \sigma_{ri})\}$; (12) endif; (13) endfor; (14) endfor; End.

values of all unknown proposition neurons at step l, i.e., the algorithm will be converged at step l.

In order to clearly understand the algorithms described previously, we use two examples to illustrate the weighted fuzzy backward reasoning process. For Example 1 (Π_0), Table I gives the immediate rule-incidence table of its all proposition neurons. By the fuzzy "$\oplus$-OR" tree generating algorithm, a fuzzy "$\oplus$-OR" tree of Π_0 is generated, as shown in Fig. 5. Assume that the truth value of proposition p_1 associated with input proposition neuron σ_{p1} is 0.8. By performing the fuzzy truth value evaluating algorithm, the truth values of output proposition neurons in Π_0 are obtained, as shown in Fig. 5, from which it can be clearly seen that the truth value of output proposition neuron σ_{p5} is 0.63.

Example 2: Let p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, and p_9 be nine propositions. Assume the knowledge base of a rule-based system contains the following weighted fuzzy production rules.

R_1: IF p_1 THEN p_5 (CF $= \gamma_1$), ω_1.

R_2: IF p_2 AND p_3 THEN p_6 (CF $= \gamma_2$), ω_2, ω_{31}.

R_3: IF p_3 AND p_4 THEN p_7 (CF $= \gamma_3$), ω_{32}, ω_4.

R_4: IF p_5 AND p_6 THEN p_8 (CF $= \gamma_4$), ω_5, ω_6.

R_5: IF p_7 THEN p_9 (CF $= \gamma_5$), ω_7.

Here, true values, certainty factors, and weights are extended to use triangular fuzzy numbers. Assume the certainty factors $\gamma_1, \gamma_2, \gamma_3, \gamma_4$, and γ_5 are $(0.80, 0.90, 1.0)$, $(0.70, 0.80, 0.90)$, $(0.75, 0.85, 0.95)$, $(0.85, 0.95, 1.0)$, and $(0.80, 0.90, 1.0)$, respectively.

TABLE IV
ALGORITHM 3: FUZZY "$\oplus$-OR" TREE GENERATING ALGORITHM

Input: Π and IRIT
Output: The fuzzy "$\oplus$-OR" tree T of Π
Begin (1) $Q \leftarrow \phi$; /*Q is a queue of proposition neurons*/ (2) Create the (virtual) root node $(\sigma_0, OUT, -)$ of tree T ; /*"$-$" denotes that the fuzzy truth value of the proposition is unknown*/ (3) for each $\sigma_{pk} \in OUT$ do (4) Create a new node $(\sigma_{pk}, \text{IBRIS}(\sigma_{pk}), -)$ of tree T; (5) Mark the node as a nonterminal node; (6) endfor; (7) $Q \leftarrow Q \bigcup OUT$; (8) while $Q \neq \phi$ do (9) Take out a neuron σ from queue Q in FIFO manner; (10) if $\sigma \in IN$ then (11) Mark node $(\sigma, \text{IBRIS}(\sigma), -)$ as a terminal node; (12) loop; (13) endif; (14) for each $(\sigma, \text{IBRIS}(\sigma)) \in$ IRIT do (15) if IBRIS$(\sigma) = \phi$ then (16) Mark $(\sigma, \text{IBRIS}(\sigma), -)$ as a terminal node; (17) else if IBRIS$(\sigma) = \{\sigma'\}$ then (18) Create a new node $(\sigma', \text{IBRIS}(\sigma'), -)$ of T; (19) Label its arc using certainty factor CF$_{jk}$ of the rule neuron between σ and σ'; (20) $Q \leftarrow Q \bigcup \{\sigma'\}$; (21) else if IBRIS$(\sigma) = \{\sigma_1, \sigma_2, \ldots, \sigma_s\}$ then (22) for $k \leftarrow 1$ to s do (23) Create a new node $(\sigma_k, \text{IBRIS}(\sigma_k), -)$ of T; (24) Label its arc using certainty factor CF$_{jk}$ of the rule neuron between σ and σ_k; (25) $Q \leftarrow Q \bigcup \{\sigma_k\}$; (26) endfor; (27) endif; (28) endfor; (29) $Q \leftarrow Q - \{\sigma\}$ (30) endwhile; End.

Let $\omega_1 = (1.0, 1.0, 1.0), \omega_2 = (0.85, 0.95, 1.0), \omega_{31} = (0.70, 0.80, 0.90), \omega_{32} = (0.85, 0.95, 1.0), \omega_4 = (0.75, 0.85, 0.95), \omega_5 = 0.85, 0.95, 1.0), \omega_6 = (0.75, 0.85, 0.95)$, and $\omega_7 = (1.0, 1.0, 1.0)$.

The weighted fuzzy production rules can be modeled by using the following WFSN P systems Π_5, as shown in Fig. 6:

$\Pi_5 = (\{a\}, \{\sigma_{p1}, \sigma_{p2}, \sigma_{p3}, \sigma_{p4}, \sigma_{p5}, \sigma_{p6}, \sigma_{p7}, \sigma_{p8}, \sigma_{p9}\}, \{\sigma_{r1}, \sigma_{r2}, \sigma_{r3}, \sigma_{r4}, \sigma_{r5}\}, \text{syn}, \text{IN}, \text{OUT})$,

where

1) $\sigma_{p1}, \sigma_{p2}, \sigma_{p3}, \sigma_{p4}, \sigma_{p5}, \sigma_{p6}, \sigma_{p7}, \sigma_{p8}$, and σ_{p9} proposition neurons;
2) $\sigma_{r1}, \sigma_{r2}, \sigma_{r3}, \sigma_{r4}$, and σ_{r5} rule neurons. Here, $\nu_1 = \omega_1 = (1.0, 1.0, 1.0)$, $\nu_2 = \omega_2 \oplus \omega_{31} = (1.55, 1.75, 1.90)$,

TABLE V
ALGORITHM 4: FUZZY TRUTH VALUE COMPUTING ALGORITHM

Input: $\{\alpha(\sigma_{pi}) \mid \sigma_{pi} \in IN\}$, and the fuzzy "$\oplus$-OR" tree T

Output: $\{\alpha(\sigma_{pk}) \mid \sigma_{pk} \in OUT\}$

Begin

/*Q_1 and Q_2 are two queues of proposition neurons*/

/*Q_1 indicates the IBRIS of some proposition neurons*/

(1) $Q_1 \leftarrow \phi$, $Q_2 \leftarrow \phi$;

(2) Push all terminal nodes in tree T into queue Q_2 in sequential way from left to right;

(3) while $Q_2 \neq \{\sigma_0\}$ do

(4) while (Q_1 is not in IRIT) do

(5) Take out a neuron σ form Q_2 in FIFO manner;

(6) $Q_1 \leftarrow Q_1 \bigcup \{\sigma\}$;

(7) endwhile;

(8) Assume σ_{pk} is the neuron such that Q_1=IBRIS(σ_{pk}), where the corresponding rule neuron is σ_{ri};

(9) if $Q_1 = \{\sigma_{pj}\}$ then

(10) $\alpha(\sigma_{pk}) = [(\alpha(\sigma_{pj}) \otimes \omega_j) \oslash \nu_i] \otimes \gamma_i$;

(11) else if $Q_1 = \{\sigma_{pj_1}, \sigma_{pj_2}, \ldots, \sigma_{pj_s}\}$ then

(12) $\alpha(\sigma_{pk}) = [(\alpha(\sigma_{pj_1}) \otimes \omega_{j_1} \oplus \ldots \oplus \alpha(\sigma_{pj_s}) \otimes \omega_{j_s}) \oslash \nu_i] \otimes \gamma_i$;

(13) endif;

(14) $Q_1 \leftarrow \phi$;

(15) Push the σ_{pk} into queue Q_2;

(16) endwhile;

End.

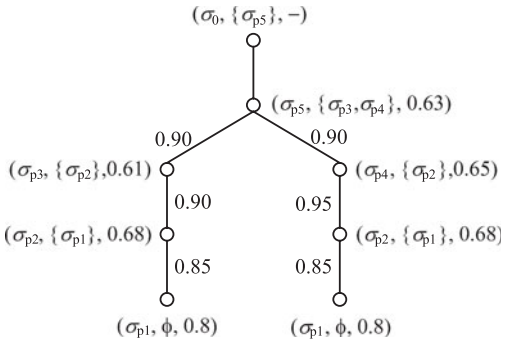

Fig. 5. Generated fuzzy "$\oplus$-OR" tree of Π_0 and computation of the fuzzy truth values of the fuzzy "$\oplus$-OR" tree of Π_0.

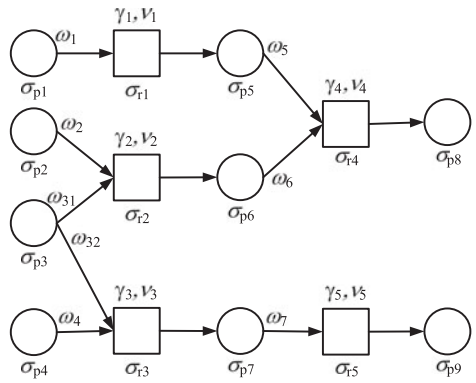

Fig. 6. Example 2 modeled by WFSN P systems Π_5.

TABLE VI
IMMEDIATE RULE-INCIDENCE TABLE OF ALL PROPOSITION NEURONS IN EXAMPLE 2

Proposition Neuron σ_{pk}	IBRIS(σ_{pk})	Rule Neuron σ_{ri}
σ_{p1}	ϕ	
σ_{p2}	ϕ	
σ_{p3}	ϕ	
σ_{p4}	ϕ	
σ_{p5}	$\{\sigma_{p1}\}$	σ_{r1}
σ_{p6}	$\{\sigma_{p2}, \sigma_{p3}\}$	σ_{r2}
σ_{p7}	$\{\sigma_{p3}, \sigma_{p4}\}$	σ_{r3}
σ_{p8}	$\{\sigma_{p5}, \sigma_{p6}\}$	σ_{r4}
σ_{p9}	$\{\sigma_{p7}\}$	σ_{r5}

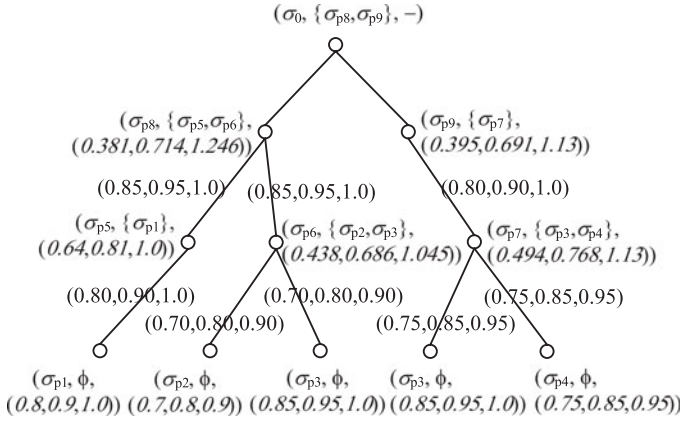

Fig. 7. Generated fuzzy "$\oplus$-OR" tree of Π_5 and computation of the fuzzy truth values of the fuzzy "$\oplus$-OR" tree of Π_5.

$\nu_3 = \omega_{32} \oplus \omega_4 = (1.6, 1.8, 1.95)$, $\nu_4 = \omega_5 \oplus \omega_6 = (1.6, 1.8, 1.95)$, and $\nu_5 = \omega_7 = (1.0, 1.0, 1.0)$;

3) syn$= \{(\sigma_{p1}, \sigma_{r1}), (\sigma_{p2}, \sigma_{r2}), (\sigma_{p3}, \sigma_{r2}), (\sigma_{p3}, \sigma_{r3}), (\sigma_{p4}, \sigma_{r3}), (\sigma_{p5}, \sigma_{r4}), (\sigma_{p6}, \sigma_{r4}), (\sigma_{p7}, \sigma_{r5}), (\sigma_{r1}, \sigma_{p5}), (\sigma_{r2}, \sigma_{p6}), (\sigma_{r3}, \sigma_{p7}), (\sigma_{r4}, \sigma_{p8}), (\sigma_{r5}, \sigma_{p9})\}$;

4) IN $= \{\sigma_{p1}, \sigma_{p2}, \sigma_{p3}, \sigma_{p4}\}$, OUT $= \{\sigma_{p8}, \sigma_{p9}\}$.

For Example 2 (Π_5), Table VI gives the immediate rule-incidence table of all its proposition neurons. By the fuzzy "$\oplus$-OR" tree generating algorithm, a fuzzy "$\oplus$-OR" tree of Π_5 is generated, as shown in Fig. 7. Assume that the truth values of propositions p_1, p_2, p_3, and p_4 associated with input proposition neurons $\sigma_{p1}, \sigma_{p2}, \sigma_{p3}$, and σ_{p4} are $(0.80, 0.90, 1.0), (0.70, 0.80, 0.90), (0.85, 0.95, 1.0)$, and $(0.75, 0.85, 0.95)$, respectively. By performing the fuzzy truth value computing algorithm, the truth values of output proposition neurons in Π_5 are obtained, as shown in Fig. 7, from which it can be easily seen that the fuzzy truth values of output proposition neurons σ_{p8} and σ_{p9} are fuzzy numbers $(0.381, 0.714, 1.246)$ and $(0.395, 0.691, 1.130)$.

V. Comparisons With Other Methods

As described previously, the proposed WFSN P systems are an extended version of SN P systems. The main motivation behind developing the WFSN P systems is to build a bridge by which SN P systems can be used to deal with real-world problems, such as process control, expert system, and fault diagnosing. Compared with the existing SN P systems and their variants, the proposed WFSN P systems feature the following differences: 1) there are two types of neurons, proposition neurons and rule neurons; 2) fuzzy truth value is used to express the contents of neurons; 3) each neuron is assigned an output weight vector; 4) weighted fuzzy logic is used to process operation of the pulse value; and 5) a new firing mechanism is applied, including new firing/spiking rules and the use of firing condition and threshold. These differences can ensure that the proposed WFSN P systems are able to express fuzzy and uncertain knowledge and process weighted fuzzy reasoning.

The weighted fuzzy production rule is one of most popular methods that represent fuzzy and uncertain knowledge. For instance, when we use fuzzy neural network (FNN) to extract knowledge from the measured data, fuzzy production rules are usually used to express the extracted results. However, the knowledge expressed by fuzzy production rules is usually difficult to understand because it lacks straightforward structure and representing form, and its fuzzy reasoning process is very complicated. In addition, some knowledge representation methods have been developed, such as concept graph, semantic networks, etc. One of the advantages is their graphical structure. However, their reasoning process lacks the capability of parallel reasoning.

As a graphical modeling method, Petri nets have a powerful ability that describes and studies information processing system, and possess several perfect characterizations, such as being concurrent, asynchronous, distributed, parallel, nondeterministic, etc. Thus, Petri nets are capable of describing fuzzy rule-based system. SN P systems are a novel distributed and parallel computing model. From the previous discussion, we know that the proposed WFSN P systems are very suitable for expressing a fuzzy rule-based system and for modeling its dynamic reasoning process. As two models that describe fuzzy rule-based system, SN P systems and Petri nets share some common features.

1) They are distributed and parallel computing models or systems.
2) Their graphical nature can visualize the structure of a fuzzy rule-based system, and the models represented by them are relatively simple and easily understandable.
3) It is easy to model dynamic reasoning process of a fuzzy rule-based system by them because of their firing mechanism.

Compared with Petri nets, however, the proposed WFSN P systems have the following different features and advantages

1) For WFSN P systems (SN P systems), most of their mechanisms are originated from living cells or neurophysiological behavior of neurons, such as firing and emitting a spike in neuron. The different types of cells or neurons may provide new inspirations to extend SN P systems. In fact, different types of variants of SN P systems have been addressed so far. This opens the door to dealing with fuzzy and uncertain knowledge (acquisition, representation, and reasoning) and learning problems by integrating fuzzy logic, evolution mechanism, and learning mechanism of neural network, as in, for example, the WFSN P systems discussed in this paper and the membrane algorithm developed in [41].
2) The time-delay mechanism of SN P systems is inborn and inherent. Although S-WFSN P systems, which are used to express fuzzy knowledge, omit their time-delay mechanism in this paper, the time-delay mechanism is often very useful for knowledge acquisition and representation situations. For instance, in some industry control processes, due to the use of sensors with different sample frequencies, the data obtained may include some time delays. As a result, the extracted knowledge should also include the delay factor. The proposed WFSN P systems can potentially satisfy the need to express such fuzzy knowledge and perform fuzzy reasoning.
3) Although both SN P systems and Petri nets have nondeterministic features, their principles are different. For SN P systems, when multiple firing rules in a neuron are enabled, they will nondeterministically choose a firing rule so that the configuration (or state) of whole system is changed. Of course, due to flexible definition mechanism of the neuron in SN P systems, we can easily develop nondeterministic feature like in Petri nets.
4) Since SN P systems are essentially a nonlinear computing model, the feature is advantageous for extracting fuzzy and uncertain knowledge from the measured data. In many real-world industry situations, the modeled relationships between input and output are usually nonlinear. Therefore, the proposed WFSN P systems seem to be suitable for such nonlinear situations.
5) SN P systems are synchronized, while Petri nets are asynchronous. However, due to use of some different mechanisms, some extended SN P systems are also asynchronous [21].

VI. Conclusion

Most of the research related to SN P Systems has focused on theoretical computing issues, such as computing ability and computing effectiveness, and only a small number of real-world applications have been addressed. The current definitions of SN P systems and their variants are not suitable for a large number of real-world applications. Thus, how to extend SN P systems so that they are suitable for one or several real-world applications becomes an interesting open issue. This paper focused on the representation of fuzzy and uncertain knowledge and weighted fuzzy reasoning. In this paper, we extended SN P systems to process fuzzy and uncertain knowledge, and proposed WFSN P systems. The presented WFSN P systems can model and visualize weighted fuzzy production rules in a rule-based system. Moreover, based on the inherent parallel computing features of WFSN P systems and the neuron's firing mechanism, a parallel

and fast weighted fuzzy reasoning algorithm has been developed. Several real-world applications relate to the aforementioned problem, such as process control, expert system, fault diagnosing, and investment advising system. Therefore, the results obtained in this paper can be applied to earlier real-world application areas. In summary, the significance of proposing the WFSN P systems lies in the following: 1) from the viewpoint of SN P systems, WFSN P systems are a new type of SN P systems and extend the scope of application of SN P systems, and 2) from the viewpoint of expressing fuzzy knowledge, WFSN P systems provide a modeling tool for it and the developed reasoning algorithm is a parallel and fast fuzzy reasoning algorithm.

Future research work will seek to extend the learning ability of WFSN P systems and apply WFSN P systems to solve real-world problems associated with the learning tasks.

ACKNOWLEDGMENT

The authors would like to thank Associate Editor J.-Y. Chang and the reviewers for their very insightful and constructive suggestions, which have helped greatly improve the presentation of this paper.

REFERENCES

[1] L. Davis, *Handbook of Genetic Algorithms*. New York: Van Nostrand, 1991.

[2] J. Wang, H. Peng, and P. Shi, "An optimal image watermarking approach based on a multi-objective genetic algorithm," *J. Inform. Sci*., vol. 181, no. 24, pp. 5501–5514, 2011.

[3] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd ed. Upper Saddle River, NJ: Pearson Education, 1999.

[4] D. J. Livingstone, *Artificial Neural Networks: Methods and Applications*. New York: Humana Press, 2011.

[5] C.-F. Juang, T.-C. Chen, and W.-Y. Cheng, "Speedup of implementing fuzzy neural networks with high-dimensional inputs through parallel processing on graphic processing units," *IEEE Trans. Fuzzy Syst*., vol. 19, no. 4, pp. 717–728, Aug. 2011.

[6] C.-J. Lin, C.-F. Wu, and C.-Y. Lee, "Design of a recurrent functional neural fuzzy network using modified differential evolution," *Int. J. Innovative Comput., Inform. Control*, vol. 7, no. 2, pp. 669–684, 2011.

[7] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, Perth, W.A., Australia, 1995, pp. 1942–1948.

[8] P. Puranik, P. Bajaj, A. Abraham, P. Palsodkar, and A. Deshmukh, "Human perception-based color image segmentation using comprehensive learning particle swarm optimization," *J. Inform. Hiding Multimedia Signal Process*., vol. 2, no. 3, pp. 227–235, 2011.

[9] W. Pedrycz and M. Song, "Analytic hierarchy process (AHP) in group decision making and its optimization with an allocation of information granularity," *IEEE Trans. Fuzzy Syst*., vol. 19, no. 3, pp. 527–539, Jun. 2011.

[10] C.-F. Juang and Y.-C. Chang, "Evolutionary-group-based particle-swarm-optimized fuzzy controller with application to mobile-robot navigation in unknown environments," *IEEE Trans. Fuzzy Syst*., vol. 19, no. 2, pp. 379–392, Apr. 2011.

[11] T. Niknam, H. D. Mojarrad, and M. Nayeripour, "A new hybrid fuzzy adaptive particle swarm optimization for non-convex economic dispatch," *Int. J. Innovative Comput., Inform. Control*, vol. 7, no. 1, pp. 189–202, 2011.

[12] G. Păun, G. Rozenberg, and A. Salomaa, *DNA Computing: New Computing Paradigms*. New York: Springer, 1998.

[13] G. Păun, "Computing with membranes," *J. Comput. Syst. Sci*., vol. 61, no. 1, pp. 108–143, 2000.

[14] G. Păun, G. Rozenberg, and A. Salomaa, *The Oxford Handbook of Membrane Computing*. New York: Oxford Univ. Press, 2010.

[15] M. Ionescu, G. Păun, and T. Yokomori, "Spiking neural P systems," *Fundameta Informaticae*, vol. 71, no. 2–3, pp. 279–308, 2006.

[16] G. Păun and M. J. Pérez-Jiménez, G. Rozenberg, "Spike train in spiking neural P systems," *Int. J. Found. Comput. Sci*., vol. 17, no. 4, pp. 975–1002, 2006.

[17] H. Chen, T.-O. Ishdorj, G. Păun, and M. J. Perez-Jimenez, "Handling languages with spiking neural P systems with extended rules," *Romanian J. Inform. Sci. Technol*., vol. 9, no. 3, pp. 151–162, 2006.

[18] O. H. Ibarra, A. Păun, G. Păun, A. Rodríguez-Patón, P. Sosík, and S. Woodworth, "Normal forms for spiking neural P systems," *Theoretical Comput. Sci*., vol. 372, no. 2–3, pp. 196–217, 2007.

[19] M. Ionescu, G. Păun, and T. Yokomori, "Spiking neural P systems with an exhaustive use of rules," *Int. J. Unconvent. Comput*., vol. 3, no. 2, pp. 135–154, 2007.

[20] R. Freund, M. Ionescu, and M. Oswald, "Extended spiking neural P systems with decaying spikes and/or total spiking," *Int. J. Found. Comput. Sci*., vol. 19, no. 5, pp. 1223–1234, 2008.

[21] M. Cavalierea, O. H. Ibarrab, G. Păun, O. Egecioglub, M. Ionescuc, and S. Woodworth, "Asynchronous spiking neural P systems," *Theoretical Comput. Sci*., vol. 410, no. 24–25, pp. 2352–2364, 2009.

[22] J. Wang, H. J. Hoogeboom, L. Pan, and G. Păun, "Spiking neural P systems with weights and thresholds," in *Proc. 10th Workshop Membrane Comput*., Aug. 2009, pp. 514–533.

[23] J. Wang, L. Zhou, H. Peng, and G. Zhang, "An extended spiking neural P system for fuzzy knowledge representation," *Int. J. Innovative Comput., Inform. Control*, vol. 7, no. 7A, pp. 3709–3724, 2011.

[24] C. V. Negoita, *Expert Systems and Fuzzy Systems*. Redwood City, CA: Benjamin Cummings, 1985.

[25] S. M. Chen, "A new approach to handling fuzzy decision-making problems," *IEEE Trans. Syst., Man, Cybern*., vol. 18, no. 6, pp. 1012–1016, 1988.

[26] J. S. R. Jang, C. T. Sun, and E. Mizutani, *Neuro-Fuzzy and Soft Computing*. Englewood Cliffs, NJ: Prentice–Hall, 1997.

[27] C. Li, J. Yi, and T. Wang, "Encoding prior knowledge into data driven design of interval type-2 fuzzy logic systems," *Int. J. Innovative Comput., Inform. Control*, vol. 7, no. 3, pp. 1133–1145, 2011.

[28] J. F. Sowa, *Conceptual Structures: Information Processing in Mind and Machines*. Reading, MA: Addison-Wesley, 1984.

[29] C. L. Chang, *Introduction to Artificial Intelligence Techniques*. Austin, TX: JMA Press, 1985.

[30] S. M. Chen, "Fuzzy backward reasoning using fuzzy Petri nets," *IEEE Trans. Syst., Man, Cybern. B, Cybern*., vol. 30, no. 6, pp. 846–856, Dec. 2000.

[31] D. S. Yeung and E. C. C. Tsang, "A multilevel weighted fuzzy reasoning algorithm for expert systems," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 28, no. 2, pp. 149–158, Mar. 1998.

[32] M. Gao, M. Zhou, X. Huang, and Z. Wu, "Fuzzy reasoning Petri nets," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 33, no. 3, pp. 314–324, May 2003.

[33] X. Li, W. Yu, and F. Lara-Rosano, "Dynamic knowledge inference and learning under adaptive fuzzy Petri net framework," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 30, no. 4, pp. 442–450, Nov. 2000.

[34] X.-G. He, "Weighted fuzzy logics and its application," in *Proc. 12th Annu. Int. Comput. Softw. Appl. Conf*., 1988, pp. 485–489.

[35] S. M. Chen, "A weighted fuzzy reasoning algorithm for medical diagnosis," *Decision Support Syst*., vol. 11, no. 1, pp. 37–43, 1991.

[36] D. S. Yeung and E. C. C. Tsang, "Weighted fuzzy production rules," *Fuzzy Sets Syst*., vol. 88, no. 3, pp. 299–313, 1997.

[37] D. S. Yeung and E. C. C. Tsang, "A multilevel weighted fuzzy reasoning algorithm for expert systems," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 28, no. 2, pp. 149–158, Mar. 1998.

[38] S. M. Chen, "Weighted fuzzy reasoning using weighted fuzzy Petri nets," *IEEE Trans. Knowl. Data Eng*., vol. 14, no. 2, pp. 386–397, Mar.–Apr. 2002.

[39] S.-M. Chen, Y.-K. Ko, Y.-C. Chang, and J.-S. Pan, "Weighted fuzzy interpolative reasoning based on weighted increment transformations and weighted ratio transformation techniques," *IEEE Trans. Fuzzy Syst*., vol. 17, no. 6, pp. 1412–1427, Dec. 2009.

[40] S. M. Chen, "Fuzzy backward reasoning using fuzzy Petri nets," *IEEE Trans. Syst., Man, Cybern. B, Cybern*., vol. 30, no. 6, pp. 846–856, Dec. 2000.

[41] L. Zhang and Y. Huang, "A variant of P systems for optimization," *Neurocomputing*, vol. 72, no. 4–6, pp. 1355–1360, 2009.