Intersection Search for a Fuzzy Petri Net-Based Knowledge Representation Scheme

Slobodan Ribarić[1], Nikola Pavešić[2], and Valentina Zadrija[1]

[1] Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia
[2] Faculty of Electrical Engineering, University of Ljubljana, Slovenia
{slobodan.ribaric,valentina.zadrija}@fer.hr
nikola.pavesic@fe.uni-lj.si

Abstract. This paper describes the intersection search as an inference procedure for a knowledge representation scheme based on the theory of Fuzzy Petri Nets. The procedure uses the dynamical properties of the scheme. The relationships between the concepts of interest, obtained by the intersection search algorithm, are accompanied by the value of the linguistic variable expressing the assurance for the relations. An illustrative example of the intersection search procedure is provided.

Keywords: Knowledge representation, Inference procedure, Fuzzy Petri Net, Intersection search.

1 Introduction

One of the central problems of the knowledge-based systems, especially for real-world tasks solving, where knowledge is based on vague, uncertain and fuzzy facts, is the development of a knowledge representation scheme. Among the knowledge representation schemes that support uncertain and fuzzy knowledge representation and reasoning there is a class of schemes based on the theory of Fuzzy Petri Nets (FPNs) [1]: Looney [2] and Chen et al. [3] proposed FPNs for rule-based decision making; Scarpelli et al. [4] described a reasoning algorithm for a high-level FPN; Chen [5] introduced a Weight FPN model for rule-based systems; Li and Lara-Rosano [6] proposed a model based on an Adaptive FPN, which is implemented for knowledge inference; Looney and Liang [7] proposed the fuzzy-belief Petri Nets (PN) as combination of the bi-directional fuzzy propagation of the fuzzy-belief network and the FPN; Lee et al. [8] introduced a reasoning algorithm based on possibilistic PN as a mechanism that mimics human inference; Canales et al. [9] described a method of fuzzy-knowledge learning based on an Adaptive FPN; Ha et al. [10] described knowledge representation by weighted fuzzy-production rules and inference with a generalized FPN; and Guo-Yan [11] proposed a hybrid of the PN and the Fuzzy PN to support an inference procedure. Shen [12] presented a knowledge-representation scheme based on a high-level FPN for modeling fuzzy IF-THEN-ELSE rules.

In this paper the intersection search procedure based on "spreading activation" for a Fuzzy Petri Net-based knowledge representation scheme is proposed.

J.D. Velásquez et al. (Eds.): KES 2009, Part I, LNAI 5711, pp. 1–10, 2009.

2 A Fuzzy Petri Net-Based Knowledge Representation Scheme

A network-based fuzzy knowledge representation scheme named KRFPN (**K**nowledge-**R**epresentation Scheme based on the **F**uzzy **P**etri-**N**ets theory) uses the concepts of the Fuzzy Petri Net theory to represent uncertain, vague and/or fuzzy information obtained from modeled, real-world situations. The knowledge representation scheme KRFPN is defined as the 13-tuple:

$$\text{KRFPN} = (P, T, I, O, M, \Omega, \mu, f, c, \lambda, \alpha, \beta, C), \tag{1}$$

where the first 10 components represent a Fuzzy Petri net FPN [18] defined as follows: $P = \{p_1, p_2, \ldots , p_n\}$ is a finite set of places; $T = \{t_1, t_2, \ldots , t_m\}$ is a finite set of transitions; $P \cap T = \emptyset$; $I: T \rightarrow P^{\infty}$ is an input function, a mapping from transitions to bags of places; $O: T \rightarrow P^{\infty}$ is an output function, a mapping from transitions to bags of places; $M = \{m_1, m_2, \ldots , m_r\}$, $1 \leq r < \infty$, is a set of tokens; $\Omega: P \rightarrow \wp(M)$ is a mapping, from P to $\wp(M)$, called a distribution of tokens, where $\wp(M)$ denotes the power set of M. Using Ω_0 we denote the initial distribution of tokens in the places of an FPN; $\mu: P \rightarrow N$ is a marking, a mapping from places to non-negative integers, N. A mapping μ can be represented as an *n*-component vector $\boldsymbol{\mu} = (\mu_1, \mu_2, \ldots , \mu_n)$, where n is a cardinality of the set P. An initial marking is denoted by the vector $\boldsymbol{\mu_0}$. A function $f: T \rightarrow [0, 1]$ is an association function, a mapping from transitions to real values between zero and one, and $c: M \rightarrow [0, 1]$ is an association function, a mapping from tokens to real values between zero and one, and $\lambda \in [0, 1]$ is a threshold value related to the firing of a FPN.

A marked FPN can be represented by a bipartite directed multi-graph containing two types of nodes: places (graphically represented by circles) and transitions (bars). The relationships that are based on input and output functions are represented by directed arcs. Each arc is directed from an element of one set (P or T) to the element of another set (T or P). The tokens in the marked FPN graphs are represented by labeled dots $c(m_i)$, where $c(m_i)$ denotes the value of the token.

Tokens give dynamical properties to an FPN, and they are used to define its *execution*, i.e., by firing an enabled transition t_j, tokens are removed from its input places (elements in $I(t_j)$). Simultaneously, new tokens are created and distributed to its output places (elements of $O(t_j)$). In an FPN, a transition t_j is *enabled* if each of its input places has at least as many tokens in it as there are arcs from the place to the transition and if the values of the tokens $c(m_l)$, $l = 1, 2, \ldots$ exceed a threshold value $\lambda \in [0, 1]$. The number of tokens at the input and output places of the fired transition is changed in accordance with the basic definition of the original PN [16]. The new token value in the output place is $c(m_l) \cdot f(t_i)$, where $c(m_l)$ is the value of the token at the input place $p_j \in I(t_i)$ and $f(t_i)$ is the degree of truth of the relation assigned to the transition $t_i \in T$.

The components α, β and C, introduce a semantic interpretation to the scheme: $\alpha: P \rightarrow D$ is a bijective function that maps a set of places onto a set of concepts D. The set of concepts D consists of the formal objects used for representing objects and facts from the agent's world. The elements from $D = D_1 \cup D_2 \cup D_3$ are as follows: elements that denote classes or categories of objects and represent higher levels of abstraction (D_1), elements corresponding to individual objects as the instances of the classes (D_2) and those elements representing the intrinsic properties of the concepts or values of these properties (D_3).

β: $T \rightarrow \Sigma$ is a surjective function that associates a description of the relationship among the facts and objects to every transition $t_i \in T$; i = 1, 2, … , *m*, where *m* is a cardinality of the set T. The set $\Sigma = \Sigma_1 \cup \Sigma_2 \cup \Sigma_3$ consists of elements corresponding to the relationships between the concepts used for the partial ordering of the set of concepts (Σ_1), the elements used to specify the types of properties to which the values from subset D_3 are assigned (Σ_2), and the elements corresponding to the relationships between the concepts, but not used for hierarchical structuring (Σ_3). For example, elements from Σ_3 may be used to specify the spatial relations among the objects.

The semantic interpretation requires the introduction of a set of contradictions C, which is a set of pairs of mutually contradictory relations (for example, *is_a* and *is_not_a*), as well as, pairs of mutually contradictory concepts if they are inherited for the same concept or object (for example, the object cannot simultaneously inherit properties such as "*Quadruped*" and "*Biped*").

The inverse function α^{-1}: $D \rightarrow P$, and the generalized inverse function β^{-1}: $\Sigma \rightarrow \tau$; $\tau \subseteq T$ are defined in the KRFPN.

Note that the KRFPN inherits dynamical properties from the FPN.

The uncertainty and confidence related to the facts, concepts and the relationships between them in the KRFPN are expressed by means of the values of the $f(t_i)$, $t_i \in T$, and $c(m_i)$, $m_i \in M$, association functions. The value of the function f, as well as the value of the function c, can be expressed by the truth scales and by their corresponding numerical intervals proposed in [3] - from "*always true*" [1.0, 1.0], "*extremely true*" [0.95, 0.99], "*very true*" [0.80, 0.94] to "*minimally true*" [0.01, 0.09], and "*not true*" [0.0, 0.0].

Example 1

In order to illustrate the basic components of the KRFPN, a simple example of the agent's knowledge base for a scene (adapted from [17]; Fig. 1) is introduced. Briefly, the scene may be described as follows: Shaggy, who is a human, and Scooby, the dog, are cartoon characters. Scooby, as a cartoon character, can talk and he is, like Shaggy, a mammal. Shaggy wears clothes and he is in front of Scooby. We suppose that both Shaggy and Scooby are hungry.

The knowledge base designed by the KRFPN has the following components (Fig. 2):

$P = \{p_1, p_2, \ldots, p_{12}\}$; $T = \{t_1, t_2, \ldots, t_{17}\}$;
$I(t_1) = \{p_1\}$; $I(t_2) = \{p_3\}, \ldots$; $I(t_{17}) = \{p_1\}$;
$O(t_1) = \{p_5\}$; $O(t_2) = \{p_4\}, \ldots$; $O(t_{17}) = \{p_3\}$.

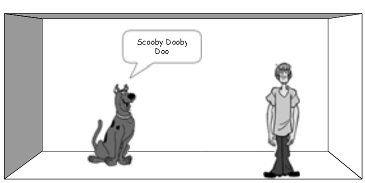

Fig. 1. A simple scene with *Scooby* and *Shaggy* [17]

The initial distribution of tokens is:
$\Omega_0 = \{\{m_1\}, \varnothing, ..., \varnothing\}$, where $c(m_1) = 1.0$, where $\varnothing$ denotes an empty set.

The vector $\boldsymbol{\mu}_0 = (1, 0, ... , 0)$ denotes that there is only one token in the place p_1. The function f is specified as follows:

$f(t_1) = f(t_2) = ... = f(t_9) = 1.0$: $f(t_{10}) = f(t_{12}) = 0.8$; and $f(t_{11}) = \ldots = f(t_{17}) = 0.9$; $f(t_i)$, $i = 1, 2, ... , m$ indicates the degree of our pursuance in the truth of the relation $\beta(t_i)$.

The set D is defined as follows:

D_1 = {*Cartoon_Character, Cartoon_Dog, Human, Mammal, Dog, Live*},D_2 = {*Scooby, Shaggy*} and D_3 = {*Talking, Brown, Hungry, Wears_Clothes*}.

The set Σ consists of:

Σ_1 = {*is_a, is_not_a*}, Σ_2 = {*has_characteristic, has_not_characteristic, has_color*} and Σ_3 = {*is_in_front_of, is_behind_of*}.

Functions α and β are:α: $p_1 \rightarrow$ *Shaggy*, β: $t_1 \rightarrow$ *is_a*, α: $p_2 \rightarrow$ *Cartoon_Character*, β: $t_2 \rightarrow$ *is_a*, …, α: $p_{12} \rightarrow$ *Wears_Clothes*, β: $t_{17} \rightarrow$ *is_in_front_of.*

A set of contradictions C is *{(has_characteristic, has_not_characteristic*), (*is_in_front_of, is_behind_of*), (*is_a, is_not_a*)}.

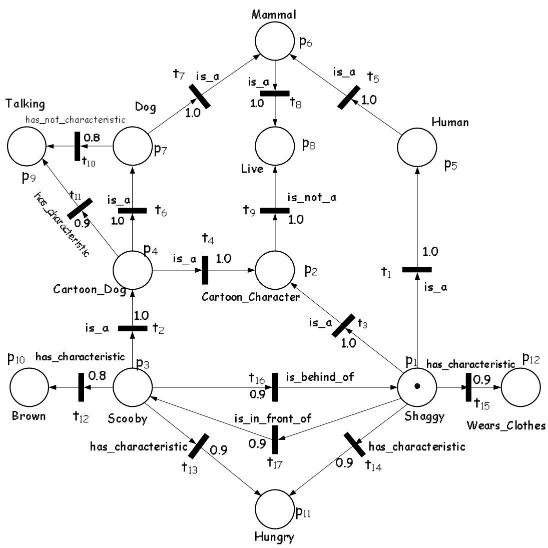

Fig. 2. The knowledge base designed by the KRFPN (Example 1)

For the initial distribution of tokens, the following transitions are enabled: t_1, t_3, t_{14}, t_{15} and t_{17}.

3 Intersection Search Algorithm

R. Quillian proposed a procedure that corresponds to the inference in semantic networks [14]. The procedure, called *intersection search* or *spreading activation,* makes it possible to find relationships between facts stored in a knowledge base by "spreading activation" out of two nodes (called *patriarch nodes*) and finding their intersection. The nodes where the activations meet are called *intersection nodes*. The paths from two patriarch nodes to the intersection nodes define the relationships between the facts.

Based on the above idea the intersection search algorithm for the fuzzy knowledge representation scheme KRFPN is here proposed. The intersection search inference procedure in the KRFPN is based on its dynamical properties, given by the firing enabled transitions, and the determination of the inheritance set of the KRFPN. The inheritance set for the KRFPN is based on concepts similar to the reachability set of the ordinary Petri nets (PNs), where the reachability relationship is the reflexive, transitive closure of the immediately reachable relationship [16]. The reachability set of the PN is graphically represented by a *reachability tree*.

The main differences between the inheritance set of the KRFPN and the reachability set of the PN [16] are as follows: (i) After firing an enabled transition, where the transition is related to the element in the subset Σ_1 (recall that the elements in Σ_1 are used for the hierarchical structuring) that specifies an exception or negation (for example, *is_not_a*), the created token(s) in the corresponding output place(s) has to be frozen. A frozen token in the output place is fixed and it cannot enable a transition. (ii) After firing all the enabled transitions for the distribution of tokens in the KRFPN, where the transitions are related to the elements in the subsets Σ_2 and Σ_3, the created tokens at the corresponding output places also have to be *frozen*. Recall that the elements in Σ_2 and Σ_3 are used to specify the properties and the non-hierarchical structuring, respectively. (iii) An inheritance tree, as a graphical representation of the inheritance set, is bounded by $k + 1$ levels, where k is a predefined number of levels. Such an inheritance tree is called a k-level inheritance tree. (iv) A k-level inheritance tree has the following additional types of nodes: a k-terminal node, a frozen node, and an identical node.

Taking into account the above particularities, a k-level inheritance tree can be constructed by applying the slightly modified algorithm for the reachability tree given in [16]. The algorithm for construction of a k-level inheritance tree is given in [18].

A k-level inheritance tree consists of the nodes $\boldsymbol{\pi} = (\pi_1, \pi_2, \ldots, \pi_n)$, where n is the cardinality of the set of places P, and the directed, labeled arcs. In order to simplify and make the notation uniform, the nodes in the tree are denoted by n-component vectors in the form $\boldsymbol{\pi} = (\pi_1, \pi_2, \ldots, \pi_n)$. Each component π_i; $i = 1, 2, ..., n$ of $\boldsymbol{\pi}$ is represented by an empty set $\varnothing$ for $\mu(p_i) = 0$, i.e., there is no token(s) at the place p_i, or by a set $\{c(m_k), \ldots, c(m_l), \ldots, c(m_s)\}$, where $c(m_l)$ is the second component of the pair $(p_i, c(m_l))$ and represents the value of the token m_l at the place p_i.

For example, the 3-level inheritance tree for the knowledge base designed by the KRFPN (*Example 1*), where a token m_1 is initially at a place p_1, is shown in Fig. 3.

Note that in order to simplify and make the notation shorter the nodes in the tree $\boldsymbol{\pi}_{pr}$, p, r = 0, 1, 2, ... contain only a component that is different to $\varnothing$ (Fig. 3). This component contains information about the place where the token is and the token's value.

By using the components of the k-level inheritance tree, a node at the level i - 1, a labeled arc, a node at the level i (successor of the node at level i - 1), the functions α

and β, and a triplet named an *inheritance assertion* is formed. The strength of the assertion is defined as the value of the token at the successor node, i.e., as a product of the token value at the node at level i - 1 and the value of the association function of the corresponding transition.

The inheritance paths, starting from the root node of the inheritance tree and finishing at the leaves of the tree, represent sequences of the inheritance assertions. An inheritance path is interpreted as the conjunction of the inheritance assertions in which the redundant concepts connected by AND are omitted. The strength of an inheritance path is defined by the value of the token at the node that is a leaf of the inheritance tree.

In network-based knowledge representation schemes there is the well-known problem of the conflicting multiple inheritance [13], which in the KRFPN is expressed as follows: two inheritance paths are in conflict if the same concept inherits the mutually contradictory elements from D. Two inheritance paths are, also, in conflict if the same concept inherits the concept or property from D, but over contradictory relations from Σ. To resolve the situations involving conflicting multiple inheritance in the KRFPN we used Touretzky's principle of inferential distance ordering (PIDO) [13].

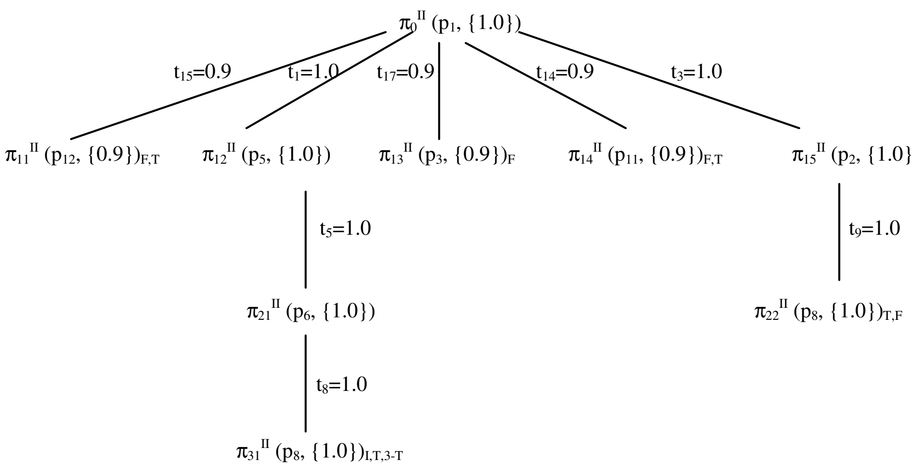

Fig. 3. The 3-level inheritance tree for the knowledge base (Example 1) (T denotes the terminal node, F denotes the frozen node and 3-T denotes the terminal node at level k = 3)

In situations when PIDO fails, procedure described in the ***Step 9*** below is applied. The intersection search algorithm for the KRFPN is presented as follows:

Input: Two concepts of interest, d_1 and d_2, for which we want to determine possible relationships; the depth of the inheritance k; $0 \leq k < \infty$, and $\lambda \in [0, 1]$.

Output: The relationships between the concepts d_1 and d_2 expressed by assertions (by means of the inheritance paths) from patriarch nodes to the intersection nodes.

In order to make the algorithm clearer, each of its steps will be illustrated for the following task: For the knowledge base (Fig. 2) find relationships between the concepts d_1 = *Scooby* and d_2 = *Shaggy*. The depth of the inheritance is k = 3 and λ = 0.1.

Step 1. For the given concepts of interest, d_1 and d_2, by using the inverse function α^{-1}, find the corresponding places p_i and p_j: α^{-1}: $d_1 \rightarrow p_i$, α^{-1}: $d_2 \rightarrow p_j$. If $d_1 \notin D$ or $d_2 \notin D$ stop the algorithm and send the message: "d_u is an unknown concept, the relationships are unknown"; u=1, 2.

For our example: α^{-1}: *Scooby* $\rightarrow p_3$, α^{-1}: *Shaggy* $\rightarrow p_1$.

Step 2. Define the initial markings $\boldsymbol{\mu}^{\mathbf{I}}_0 = (\mu^{I}_1, \mu^{I}_2, \dots, \mu^{I}_n)$ and $\boldsymbol{\mu}^{\mathbf{II}}_0 = (\mu^{II}_1, \mu^{II}_2, \dots, \mu^{II}_n)$, where *n* is a cardinality of the set of places P:

$$\mu^{I}_{k} = \begin{cases} 1 \; for & k = i \\ 0 \; for \;\; all & k \neq i \end{cases} \quad \text{and} \quad \mu^{II}_{k} = \begin{cases} 1 \; for & k = j \\ 0 \; for \;\; all & k \neq j \end{cases}, \qquad k = 1, 2, \dots, n$$

In our example: $\boldsymbol{\mu}^{\mathbf{I}}_0 = (0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0)$ and
$\boldsymbol{\mu}^{\mathbf{II}}_0 = (1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$.

Step 3. Define the initial distribution of tokens $\Omega^{I}_0 = \boldsymbol{\pi}^{\mathbf{I}}_{\mathbf{0}} = (\emptyset, \emptyset, \emptyset, \dots \{(p_i, c(m^{I}_1))\}, \dots, \emptyset, \emptyset)$ and $\Omega^{II}_0 = \boldsymbol{\pi}^{\mathbf{II}}_{\mathbf{0}} = (\emptyset, \emptyset, \emptyset, \dots \{(p_j, c(m^{II}_1))\}, \dots, \emptyset, \emptyset)$, and set $c(m^{I}_1) = c(m^{II}_1) = 1.0$;
$\Omega^{I}_0 = \boldsymbol{\pi}^{\mathbf{I}}_{\mathbf{0}} = (\emptyset, \emptyset, \{(p_3, c(m^{I}_1) = 1.0)\}, \dots, \emptyset, \emptyset)$ and
$\Omega^{II}_0 = \boldsymbol{\pi}^{\mathbf{II}}_{\mathbf{0}} = (\{(p_1, c(m^{II}_1) = 1.0)\}, \emptyset, \emptyset, \dots, \emptyset, \emptyset)$

Step 4. For the initial distribution of tokens $\Omega^{I}_0 = \boldsymbol{\pi}^{\mathbf{I}}_{\mathbf{0}}$ construct *k* levels of the inheritance tree *InhTree^I*. Fig. 4. depicts the *InhTree^I*.

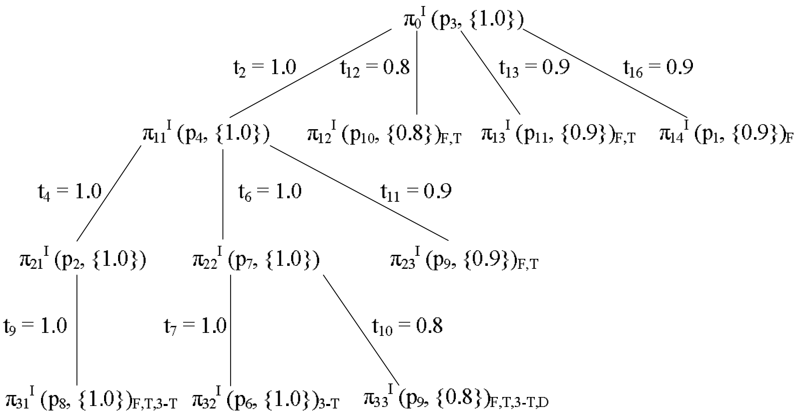

Fig. 4. The inheritance tree *InhTree^I*

Step 5. For the initial distribution of tokens $\Omega^{II}_0 = \boldsymbol{\pi}^{\mathbf{II}}_{\mathbf{0}}$ construct *k* levels of the inheritance tree *InhTree^{II}*. The inheritance tree *InhTree^{II}* is shown in Fig. 3.

Step 6. Find the nodes $\boldsymbol{\pi}^{\mathbf{I}}_{\mathbf{pr}}$, *p, r* = 0, 1,2,... in *InhTree^I* and $\boldsymbol{\pi}^{\mathbf{II}}_{\mathbf{st}}$, *s,t* = 0, 1, 2, ... in *InhTree^{II}* that match one another. Two nodes, one from *InhTree^I* and another from *InhTree^{II}*, are the *matched nodes* if they have tokens in the same places, regardless of the number and the values of the tokens. These nodes are defined as the intersection nodes. For example, the nodes $\boldsymbol{\pi}^{\mathbf{I}}_{\mathbf{pr}} = (\varnothing, \varnothing, \varnothing, \{(p_4, c(m^I_1) = 0.8), (p_4, c(m^I_2) = 1.0)\}, \varnothing, \ldots, \varnothing)$ and $\boldsymbol{\pi}^{\mathbf{II}}_{\mathbf{st}} = (\varnothing, \varnothing, \varnothing, \{(p_4, c(m^{II}_1) = 0.6)\}, \varnothing, \ldots, \varnothing\}$are matched nodes.

If there are no such nodes the algorithm stops and sends the message: "There are no relationships between the concepts (facts) d_1 and d_2 ".

For our example (see Fig. 4 and Fig. 3):

$$(\boldsymbol{\pi}^{\mathbf{I}}_{\mathbf{0}}, \boldsymbol{\pi}^{\mathbf{II}}_{\mathbf{13}}): \boldsymbol{\pi}^{\mathbf{I}}_{\mathbf{0}} = (p_3, \{1.0\}) \text{ and } \boldsymbol{\pi}^{\mathbf{II}}_{\mathbf{13}} = (p_3, \{0.9\});$$
$$(\boldsymbol{\pi}^{\mathbf{I}}_{\mathbf{14}}, \boldsymbol{\pi}^{\mathbf{II}}_{\mathbf{0}}): \boldsymbol{\pi}^{\mathbf{I}}_{\mathbf{14}} = (p_1, \{0.9\}) \text{ and } \boldsymbol{\pi}^{\mathbf{II}}_{\mathbf{0}} = (p_1, \{1.0\});$$
$$(\boldsymbol{\pi}^{\mathbf{I}}_{\mathbf{13}}, \boldsymbol{\pi}^{\mathbf{II}}_{\mathbf{14}}): \boldsymbol{\pi}^{\mathbf{I}}_{\mathbf{13}} = (p_{11}, \{0.9\}) \text{ and } \boldsymbol{\pi}^{\mathbf{II}}_{\mathbf{14}} = (p_{11}, \{0.9\});$$
$$(\boldsymbol{\pi}^{\mathbf{I}}_{\mathbf{21}}, \boldsymbol{\pi}^{\mathbf{II}}_{\mathbf{15}}): \boldsymbol{\pi}^{\mathbf{I}}_{\mathbf{21}} = (p_2, \{1.0\}) \text{ and } \boldsymbol{\pi}^{\mathbf{II}}_{\mathbf{15}} = (p_2, \{1.0\});$$
$$(\boldsymbol{\pi}^{\mathbf{I}}_{\mathbf{31}}, \boldsymbol{\pi}^{\mathbf{II}}_{\mathbf{22}}): \boldsymbol{\pi}^{\mathbf{I}}_{\mathbf{31}} = (p_8, \{1.0\}) \text{ and } \boldsymbol{\pi}^{\mathbf{II}}_{\mathbf{22}} = (p_8, \{1.0\});$$
$$(\boldsymbol{\pi}^{\mathbf{I}}_{\mathbf{31}}, \boldsymbol{\pi}^{\mathbf{II}}_{\mathbf{31}}): \boldsymbol{\pi}^{\mathbf{I}}_{\mathbf{31}} = (p_8, \{1.0\}) \text{ and } \boldsymbol{\pi}^{\mathbf{II}}_{\mathbf{31}} = (p_8, \{1.0\});$$
$$(\boldsymbol{\pi}^{\mathbf{I}}_{\mathbf{32}}, \boldsymbol{\pi}^{\mathbf{II}}_{\mathbf{21}}): \boldsymbol{\pi}^{\mathbf{I}}_{\mathbf{32}} = (p_6, \{1.0\}) \text{ and } \boldsymbol{\pi}^{\mathbf{II}}_{\mathbf{21}} = (p_6, \{1.0\}).$$

Step 7. For all the matched nodes in $InhTree^I$ and $InhTree^{II}$ apply the semantic function α for the corresponding places to obtain a set of intersection concepts.

α: $p_3 \rightarrow$ *Scooby*, α: $p_1 \rightarrow$ *Shaggy*, α: $p_{11} \rightarrow$ Hungry, α: $p_2 \rightarrow$ *Cartoon_Character*, α: $p_8 \rightarrow$ *Live*, α: $p_6 \rightarrow$ *Mammal*;

The set of intersection concepts is: {*Scooby, Shaggy, Hungry, Cartoon_Character, Live, Mammal*}.

Step 8. Find the inheritance paths, starting from the root node (the patriarch node) of the inheritance tree $InhTree^I$ and finishing at the nodes (the leaves) of the *k*-level inheritance tree. Do the same for the inheritance tree $InhTree^{II}$.

The strength of an inheritance path is defined as the minimal value of the tokens in the leaf-node.

$InhTree^I$:

(i) *Scooby is_a Cartoon_Dog* AND *is_a Cartoon_Character* AND *is_not_a Live*; strength = 1.0,

(ii) *Scooby is_a Cartoon_Dog* AND *is_a Dog* AND *is_a Mammal*;strength = 1.0,

(iii) *Scooby is_a Cartoon_Dog* AND *has_characteristic Talking*; strength = 0.9,

(iv) *Scooby is_a Cartoon_Dog* AND *is_a Dog* AND *has_not_characteristic Talking*; strength = 0.8,

(v) *Scooby has_color Brown*; strength = 0.8
Scooby has_characteristic Hungry; strength = 0.9,

(vi) *Scooby is_behind_of Shaggy*; strength = 0.9.

$InhTree^{II}$:

(i)' *Shaggy is_front_of Scooby*; strength = 0.9,

(ii)' *Shaggy has_characteristic Hungry*; strength = 0.9,

(iii)' *Shaggy is_a Cartoon_Character* AND *is_not_a Live*; strength =1.0,

(iv)' *Shaggy is_a Human* AND *is_a Mammal* AND *is_a Live*; strength = 1.0

(v)' *Shaggy has_characteristic Wears_Clothes*; strength = 0.9

Step 9. If there are assertions, in one or both sets of the inheritance assertions, involving conflict due to multiple inheritance use the PIDO or, if that fails, make a decision on the basis of the more direct inheritance path. If two or more inheritance paths have the same length the concept inherits the property that corresponds to the stronger path. On the basis of the above criteria, remove the inheritance assertion that is the source of the conflict and all the inheritance assertions that follow it.

InhTree^{I}:
The inheritance paths (iii) and (iv) are in conflict: Can Scooby talk or not? Using the PIDO concept results in rejecting the inheritance path (iv) because the concept Cartoon_Dog is “nearer” to the concept Scooby than to the concept Dog.

InhTree^{II}:
The inheritance paths (iii)’ and (iv)’ are in conflict: Is Shaggy live or not? To resolve the conflict, the decision is made on the basis of the more direct inheritance path, because the PIDO concept failed (there is no hierarchical relationship between the concepts Cartoon_Character and Human). The more direct inheritance path is (iii)’.

Step 10. After the elimination of the conflicting assertions, the elements of the set of intersection concepts determined in Step 7 are identified in the inheritance paths. If a certain inheritance path does not contain any of the intersection concepts, the given path does not describe the relationship between the concepts of interest.

For our example:

(i) *Scooby is_a Cartoon_Dog* AND *is_a* ***Cartoon_Character***; *Always true,*
(ii) *Shaggy is_a* ***Cartoon_Character***; *Always true,*
(iii) *Scooby is_a Cartoon_Dog* AND *is_a Cartoon_Character* AND *is_not_a* ***Live*;** *Always true,*
(iv) *Shaggy is_a Cartoon_Character* AND *is_not_a* ***Live***; *Always true,*
(v) *Scooby is_a Cartoon_Dog* AND *is_a Dog* AND *is_a* ***Mammal***; *Always true,*
(vi) *Shaggy is_a Human* AND *is_a* ***Mammal***; *Always true,*
(vii) *Scooby has_characteristic* ***Hungry***; *Very true,*
(viii) *Shaggy has_characteristic* ***Hungry***; *Very true,*
(ix) *Scooby is_behind_of* ***Shaggy***; *Very true,*
(x) ***Shaggy***; *Always true,*
(xi) ***Scooby***; *Always true,*
(xii) *Shaggy is_front_of* ***Scooby***; *Very true.*

(Note that the concepts corresponding to the intersection nodes are denoted bold).

4 Conclusion

An original intersection search procedure for the knowledge representation scheme based on the Fuzzy Petri Net theory, named KRFPN, is proposed. The procedure uses *k*-level inheritance trees that are generated on the basis of the dynamical properties of the scheme. The relationships between the concepts of interest, obtained by the proposed intersection search algorithm, are accompanied by the value of a linguistic variable expressing the degree of assurance for the relationship assigned to the transitions.

The very important properties of the proposed algorithm are that the *k*-level inheritance trees are finite and that the upper bound of the time complexity of the algorithm is $O(nm)$, where n is the number of the concepts (places) and m is the number of relations (transitions) in the knowledge base.

The program simulator for the KRFPN was developed and the fuzzy inference procedures, including the proposed intersection search algorithm, were tested on numerous examples [19].

References

1. Cardoso, J., Camargo, H. (eds.): Fuzziness in Petri Nets. Physica-Verlag, Heidelberg (1999)
2. Looney, C.G.: Fuzzy Petri Nets for Rule-based Decisionmaking. IEEE Trans. on the System, Man and Cybernetics 18(1), 178–183 (1988)
3. Chen, S.-M., Ke, J.-S., Chang, J.-F.: Knowledge Representation Using Fuzzy Petri Nets. IEEE Trans. on Knowledge and Data Engineering 2(3), 311–319 (1990)
4. Scarpelli, H., Gomide, F., Yager, R.R.: A Reasoning Algorithm for High Level Fuzzy Petri Nets. IEEE Trans. on Fuzzy Systems 4(3), 282–294 (1996)
5. Chen, S.-M.: Weighted Fuzzy Reasoning Using Weighted Fuzzy Petri Nets. IEEE Trans. on Knowledge and Data Engineering 14(2), 386–397 (2002)
6. Li, X., Lara-Rosano, F.: Adaptive Fuzzy Petri Nets for Dynamic Knowledge Representation and Inference. Expert Systems with Applications 19, 235–241 (2000)
7. Looney, C.G., Liang, L.R.: Inference via Fuzzy Belief Petri Nets. In: Proceed. of the 15th Int. Conf. on Tools with Artificial Intelligence (ICTAI 2003), pp. 510–514 (2003)
8. Lee, J., Liu, K.F.R., Chaing, W.: Model Uncertainty Reasoning With Possibilistic Petri Nets. IEEE Trans. on Systems, Man and Cybernetics–Part B: Cybernetics 33(2), 214–224 (2003)
9. Canales, J.C., Li, X., Yu, W.: Fuzzy Knowledge Learning via Adaptive Fuzzy Petri Net with Triangular Function Model, Intelligent Control and Automation. In: The Sixth World Congress on WCICA 2006, vol. 1, pp. 4249–4253 (2006)
10. Ha, M.-H., Li, J., Li, H.-J., Wang, P.: A New Form of Knowledge Representation and Reasoning. In: Proc. of the Fourth Int. Conf. on Machine Learning and Cybernetics, Guangzhou, pp. 2577–2582 (2005)
11. Guo-Yan, H.: Analysis of Artificial Intelligence Based Petri Net Approach to Intelligent Integration of Design. In: Proc. of the International Conference on Machine Learning and Cybernetics, pp. 1691–1695 (2006)
12. Shen, V.R.L.: Knowledge Representation Using High-Level Fuzzy Petri Nets. IEEE Trans. on Systems, Man, and Cybernetics-Part A 36(6), 1220–1227 (2006)
13. Touretzky, D.S.: The Mathematics of Inheritance Systems. Pitman, London (1986)
14. Quillian, M.R.: Word Concepts: A Theory and Simulation of Some Basic Semantic Capabilities. Behavioral Science 12(5), 410–430 (1967)
15. Shastri, L.: Semantic Networks: An Evidential Formalization and Its Connectionist Realization. Pitman, London (1988)
16. Peterson, J.L.: Petri Net Theory and Modeling of Systems. Prentice-Hall, Englewood Cliffs (1981)
17. `http://en.wikipedia.org/wiki/Scooby-Doo`
18. Ribarić, S., Pavešić, N.: Inference Procedures for Fuzzy Knowledge Representation Scheme. Applied Artificial Intelligence 23, 1, 16–43 (2009)
19. Zadrija, V.: Diploma Thesis, Faculty of EE and Computing. University of Zagreb (June 2008)