퍼지페트리네트 표현을 기반으로 하는 퍼지추론*

Fuzzy Reasonings based on Fuzzy Petri Net Representations

조 상 엽**
(Sang-Yeop Cho)

요 약 본 논문에서는 규칙기반 전문가시스템의 퍼지생성규칙을 표현할 수 있는 퍼지페트리네트 표현을 제안한다. 퍼지페트리네트 표현을 기반으로, 전진추론 알고리즘과 후진추론 알고리즘으로 구성된 퍼지추론 알고리즘을 제안한다.

본 논문이 제안한 알고리즘은 단순히 min과 max 계산만을 하는 기존의 알고리즘과는 달리 퍼지생성규칙의 전제부와 결론부에 퍼지개념의 유무에 따라 적절한 믿음값 평가함수을 사용하여 보다 더 인간적인 추론을 한다. 전진추론 알고리즘은 유한한 방향성 나무인 도달나무로 표현할 수 있다. 후진추론 알고리즘은 목표노드에서 시작노드까지의 후진추론 통로를 구한 후에 믿음값 평가함수를 이용하여 목표노드의 믿음값을 구한다.

키워드 전문가 시스템, 퍼지생성규칙, 전진추론 알고리즘, 후진추론 알고리즘, 퍼지페트리네트.

Abstract This paper proposes a fuzzy Petri net representation to represent the fuzzy production rules of a rule-based expert system. Based on the fuzzy Petri net representation, we present a fuzzy reasoning algorithms which consist of forward and backward reasoning algorithm.

The proposed algorithms, which use the proper belief evaluation functions according to fuzzy concepts in antecedent and consequent of a fuzzy production rule, are more closer to human intuition and reasoning than other methods. The forward reasoning algorithm can be represented by a reachability tree as a kind of finite directed tree. The backward reasoning algorithm generates the backward reasoning path from the goal to the initial nodes and then evaluates the belief value of the goal node using belief evaluation functions.

1. 서론

인공지능 분야에서는 사람들의 지식을 컴퓨터로 처리하기 위한 다양한 방법을 연구하고 있다. 이러한 방법 중에서 생성규칙을 이용하여 지식을 표현하는 규칙기반 전문가시스템을 많이 사용한다. 그러나 이러한 생성규칙은 사람의 지식에 포함되어 있는 애매함을 표현하지 못하고 있다. 이러한 애매함을 표현할 수 있는 이론적인 기반이 퍼지이론이다. 퍼지이론을 이용하여 애매함을 표현한 생성규칙을 퍼지생성규칙이라고 한다.

규칙기반 전문가시스템에서 사용하는 퍼지생성규칙과 퍼지 추론과정을 모형화할 수 있는 도구가 퍼지페트리네트이다. 퍼지페트리네트는 기존의 페트리네트[1][2]에 애매함을 표현할 수 있도록 퍼지이론을 결합하여 확장한 정보흐름분석 모형화 도구이다. 퍼지페트리네트를 지식표현과 추론에 이용한 연구로는 [3], [4], [5], [6], [7], [8], [9] 등이 있다. 이 연구들은 퍼지페트리네트의 구조상 본연적으로 갖는 하향식 정보흐름을 이용한 전진추론방법을 주로 사용하

* 본 논문의 연구는 청운대학교의 교내 연구비를 일부지원 받아 수행됨

** 청운대학교 인공지능컴퓨터학과
Dept. of Artificial Intelligence, Chungwoon University
연구분야 전문가시스템, 퍼지이론, 페트리네트 응용
주소 350-800
충남 홍성군 홍성읍 남장리 산 29
청운대학교 인공지능컴퓨터학과
전화 : 0451-630-3253
Fax : 0451-634-8700
E-Mail : sycho@cwunet.ac.kr

고, [8]과 [11]은 상향식 정보흐름을 이용한 후진추론방법을 제안하고 있다.

그러나 규칙기반 전문가시스템을 사용하는 응용분야의 문제에 따라 전진추론이나 또는 후진추론이 더 적합할 수가 있다. 본 논문에서는 문제를 해결하는 상황에서 적절한 추론방법을 선택적으로 사용할 수 있도록 전진추론 알고리즘과 후진 추론알고리즘으로 구성되는 퍼지추론 알고리즘을 제안한다.

규칙기반 전문가시스템에서 사용하는 대표적인 분야는 제어분야와 지식공학분야이다. 많은 연구들이 제어분야에 적합한 퍼지생성규칙의 분류를 이용하여 지식표현법과 추론알고리즘을 제안하고 있다 [3][5][9][10][12][13]. 이러한 분류를 지식공학분야에 바로 적용하는 것은 적절하지 못하다. 그러므로, 본 논문에서는 지식공학분야에 적절한 퍼지생성규칙의 분류를 사용하는 퍼지페트리네트와 퍼지추론을 제안한다.

기존의 추론알고리즘은 믿음값 계산과정에서 단순한 min과 max 계산만을 사용하나, 본 논문에서는 퍼지생성규칙내에 퍼지개념의 유무에 따라 사람의 사고과정과 유사한 계산방법을 이용하는 믿음값 평가함수를 사용한다.

본 논문의 구성은 다음과 같다. 2 장에서는 지식공학분야에서 사용하는 퍼지생성규칙의 분류를 설명한다. 3 장에서는 퍼지추론시 사용하는 합성추론규칙과 믿음값을 계산하기 위한 함수들을 기술한다. 4 장에서는 퍼지페트리네트의 구조와 그래프를 정의하고 퍼지페트리네트 표현을 제안한다. 5 장에서는 퍼지페트리네트 표현에서 사용할 전진 추론알고리즘과 후진 추론알고리즘으로 구성된 퍼지 추론알고리즘을 제안한다. 마지막으로 6 장에서는 본 논문의 연구결과를 기술한다.

2. 퍼지생성규칙의 분류

사람들이 사용하는 불확실하고 애매한 실세계의 지식을 적절하게 표현할 때 퍼지생성규칙을 이용하면 잘 기술할 수 있다. 퍼지생성규칙은 퍼지명제간의 퍼지관계를 기술하는 방법이라고 생각할 수 있다. 퍼지생성규칙의 형식은 다음과 같다.

$$\text{RuleName}_i\text{: IF x is A THEN y is B } (\beta_i) \tag{2.1}$$

(2.1)에서 RuleName_i는 퍼지생성규칙의 이름이고, $1 \le i \le n$이며, n은 퍼지생성규칙의 수이다. A와 B는 "크다", "많다" 등을 나타내는 퍼지변수이다. x is A와 y is B는 퍼지변수를 갖는 퍼지명제이다. IF x is A와 THEN y is B는 각각 퍼지생성규칙의 전제부와 결론부이다. $\beta_i \in [0,1]$인 퍼지생성규칙의 믿음값(belief)이다. (2.1)은 IF d_a THEN d_c (β_i)와 같이 간단히 표기하기도 한다. 여기에서 d_a와 d_c는 퍼지명제이다.

퍼지생성규칙을 많이 사용하는 분야로는 제어분야와 지식공학분야가 있다. 각 분야에서 사용하는 퍼지생성규칙의 분류는 [3], [4], [5], [6], [9], [12], [13] 등에서 찾아 볼 수 있다. 본 논문에서는 지식공학분야에서 사용하는 퍼지생성규칙을 논리연결자의 유무에 따라 <표 1>과 같이 아홉 가지의 형으로 논리적인 구분을 하였다. 1형은 단순 퍼지생성규칙이라고 하고, 2형에서 9형은 합성 퍼지생성규칙이라고 한다.

본 연구에서는 아홉 가지의 퍼지생성규칙 중에서 1형에서 4형까지 만을 사용한다. 2형과 4형은 1형과 3형으로 축소할 수 있지만 퍼지생성규칙의 표현성을 위해 그대로 사용한다. 5형과 6형은 퍼지생성규칙을 다루기 쉽게하기 위해 1형과 2형으로 각각 축소하여 표현한다. 7형, 8형 그리고 9형은 의미있는 연역추론을 할 수 없으므로 사용하지 않는다.

<표 1> 퍼지생성규칙의 분류

형	퍼지생성규칙
1형	IF d_a THEN d_c (β_i)
2형	IF d_a THEN $d_{c1} \wedge d_{c2} \wedge \cdots \wedge d_{cn}$ (β_i)
3형	IF $d_{a1} \wedge d_{a2} \wedge \cdots \wedge d_{am}$ THEN d_c (β_i)
4형	IF $d_{a1} \wedge d_{a2} \wedge \cdots \wedge d_{am}$ THEN $d_{c1} \wedge d_{c2} \wedge \cdots \wedge d_{cn}$ (β_i)
5형	IF $d_{a1} \vee d_{a2} \vee \cdots \vee d_{am}$ THEN d_c (β_i)
6형	IF $d_{a1} \vee d_{a2} \vee \cdots \vee d_{am}$ THEN $d_{c1} \wedge d_{c2} \wedge \cdots \wedge d_{cn}$ (β_i)
7형	IF d_a THEN $d_{c1} \vee d_{c2} \vee \cdots \vee d_{cn}$ (β_i)
8형	IF $d_{a1} \wedge d_{a2} \wedge \cdots \wedge d_{am}$ THEN $d_{c1} \vee d_{c2} \vee \cdots \vee d_{cn}$ (β_i)
9형	IF $d_{a1} \vee d_{a2} \vee \cdots \vee d_{am}$ THEN $d_{c1} \vee d_{c2} \vee \cdots \vee d_{cn}$ (β_i)

3. 퍼지추론

퍼지생성규칙을 이용한 퍼지추론을 할 때에는 합성추론규칙이 사용된다. 합성추론규칙이 적용될 때 결론의 믿음값을 계산하기 위한 믿음값 평가함수, 합성퍼지생성규칙의 믿음값을 계산하는 믿음값 합성함수, 믿음값 결합시 사용하는 믿음값 결합함수를 설명한다 [4][5][14][15][16].

퍼지추론을 할 때 사용하는 추론규칙인 합성추론규칙은 다음과 같다.

$$\begin{array}{ll} \text{《규칙》}: \text{ IF x is A THEN y is B} & (\beta_r) \\ \text{《사실》}: \text{ x is A}' & (\beta_f) \\ \hline \text{《결론》}: \text{ y is B}' & (\beta_c) \end{array} \quad (3.1)$$

(3.1)에서 A, A', B 그리고 B'는 퍼지변수, 퍼지집합 또는 비퍼지집합이다. β_r, β_f 그리고 β_c는 각각 《규칙》, 《사실》 그리고 《결론》의 믿음값이다. β_c는 β_r과 β_f을 이용하여 구한다.

3.1 믿음값 평가함수

퍼지추론을 할 때 《결론》의 믿음값 β_c을 구하기 위해 사용하는 함수가 믿음값 평가함수 $\beta:(\beta_f, \beta_r) \rightarrow \beta_c$ 이다. $\beta:(\beta_f, \beta_r) \rightarrow \beta_c$는 퍼지 생성규칙의 전제부와 결론부에 퍼지변수의 유무에 따라 〈표 2〉와 같이 믿음값 β_c를 계산한다.

〈표 2〉 믿음값 평가함수

전제부 \ 결론부	비퍼지	퍼지
비퍼지	$\beta_c = \beta_f * \beta_r$	$\beta_c = \beta_f * \beta_r$
퍼지	$\beta_c = \beta_f * \beta_r * S$	$\beta_c = \beta_f * R_{sg}$

믿음값 평가함수는 퍼지개념의 유무에 따라 계산 방법이 세 가지로 구분된다. 첫째, 전제부가 퍼지한 경우이다. A에 퍼지개념이 없기 때문에 A'과 같다. 그래서 B는 B'과 같은 집합이 된다. 그러므로 $\beta_c = \beta_f * \beta_r$이다. 둘째, 전제부와 결론부가 모두 퍼지한 경우이다. 이 경우에는 전제부와 결론부에 대응하는 퍼지관계 R_{sg}를 구성한 후 《사실》의 믿음값과 곱하여 구한다. 그러므로 $\beta_c = \beta_f * R_{sg}$이다. R_{sg}는 인간의 추론과정을 가장 적절히 표현하는 퍼지관계이다[15][16]. 마지막으로 전제부가 퍼지하고 결론부는 퍼지하지 않은 경우이다. 《사실》과 《규칙》의 믿음값에 유사도(similarity) S를 곱하여 구한다. 그러므로 $\beta_c = \beta_f * \beta_r * S$이다. A와 A'은 모두 퍼지집합이지만, B와 B'는 비퍼지집합이다. 그러므로 A와 A'이 퍼지매치되는 정도인 유사도가 《결론》의 믿음값에 반영되어야 한다[15][16].

3.2 믿음값 합성함수

합성추론규칙이 합성 퍼지생성규칙에 적용될 때에는 믿음값 평가함수로 결론의 믿음값을 평가할 수 없다. 이때 믿음값을 계산하기 위해 다음과 같은 믿음값 합성함수를 사용한다.

$$\begin{aligned} \beta_c &= \beta_{comp}(\beta(\beta_f, \beta_r)) \\ &= \min_q(\max_p(\beta(\beta_{fp}, \beta_{rpq}))) \end{aligned}$$

여기에서 p = 1, 2, ..., m 이고, m은 전제부에 있는 퍼지명제의 수이다. q = 1, 2, ..., n 이고, n은 결론부에 있는 퍼지명제의 수이다.

3.3 믿음값 결합함수

퍼지생성규칙의 5형과 6형의 축소된 1형과 2형은 퍼지추론시 서로 다른 추론 통로를 통해 같은 노드, 즉 같은 결론에 도달할 수가 있다. 이러한 노드에서는 같은 결론이 두 개 이상의 서로 다른 믿음값을 갖게된다. 이러한 경우에 결론의 믿음값을 다시 계산하기 위해 사용하는 함수가 믿음값 결합함수이다.

$$\begin{aligned} \beta_c &= \beta_{comb}(\beta_c, \beta_c^{old}) \\ &= \max(\beta_c, \beta_c^{old}) \end{aligned}$$

여기에서 β_c^{old}는 이미 추론통로를 도달한 결론의 대한 믿음값이고, β_c는 다른 추론통로를 통해서 도달한 또 다른 결론의 믿음값이다.

4. 퍼지페트리네트 표현

페트리네트는 정보처리 시스템을 모형화할 때 사용하는 시각적인 도구이다[1][2]. 퍼지페트리네트는 퍼지이론을 이용하여 애매함을 표현할 수 있도록 페트리네트를 확장하여 얻을 수 있다[3][4][5][6][7][8][9][10][11]. 이 장에서는 본 논문에서 사용할 퍼지페트리네트 구조, 퍼지페트리네트 그래프 그리고 이를 이용한 퍼지페트리네트 표현을 기술한다.

4.1 퍼지페트리네트 구조

퍼지페트리네트 구조(Fuzzy Petri Net: FPN)의 정의는 다음과 같다.

정의 1: FPN = (P, T, D, I, O, τ, α, β, λ)

- P={p_1, p_2, ..., p_n}은 플레이스(place)의 유한집합
- T={t_1, t_2, ..., t_m}은 트랜지션(transition)의 유한집합
- D={d_1, d_2, ..., d_n}은 명제의 유한집합
- I: T→P^∞은 트랜지션을 입력 플레이스에 사상시키는 입력함수
- O: T→P^∞은 트랜지션을 출력 플레이스에 사상시키는 출력함수
- τ: P→{0,1}는 플레이스에 있는 토큰을 0과 1사이의 실수로 사상시키는 토큰가중함수
- α: P→D는 플레이스를 명제에 사상시키는 전단사함수
- β: T→{0,1}는 트랜지션을 0과 1사이의 실수로 사상시키는 트랜지션 가중함수
- λ: T→{0,1}는 트랜지션을 0과 1사이의 실수로 사상시키는 트랜지션 임계함수
- P∩T=∅

4.2 퍼지페트리네트 그래프

퍼지페트리네트를 시각적으로 표현하기 위한 방법이 퍼지 페트리네트 그래프이다. 퍼지페트리네트 그래프는 두 종류의 정점를 갖는 양분된 방향성 그래프(bipartite directed graph)이다. 퍼지페트리네트 그래프 G의 정의는 다음과 같다.

정의 2: G = (Vp, Vt, A, τ)

- V_p={V_{p1}, V_{p2}, ..., V_{pn}}는 원, ○으로 표현하는 플레이스 정점(vertex)의 집합
- V_t={V_{t1}, V_{t2}, ..., V_{tm}}는 수직선, | 으로 표현하는 트랜지션 정점의 집합
- A={a_1, a_2, ..., a_k}는 $a_i \in (V_p \times V_t) \cup (V_t \times V_p)$인 아크(arc)의 집합
- τ는 점, ●으로 표현하는 토큰
- $V_p \cap V_t = \varnothing$

$\tau_i = \tau(p_i)$는 플레이스 p_i에 있는 토큰의 가중값이고, $\beta_a = \beta(t_a)$는 트랜지션 t_a의 입력 플레이스와 출력 플레이스의 관계의 정도를 나타내는 가중값이며, $\lambda_a = \lambda(t_a)$는 트랜지션 t_a의 임계값이다. 만일 min{τ_i | $\tau_i = \tau(p_i)$, $p_i \in I(t_a)$} ≥ λ_a이면, t_a는 실행가능(enable)하다. 실행가능한 t_a는 입력 플레이스에서는 토큰을 제거하고 출력 플레이스에는 한 개씩의 토큰을 출력하면서 실행(fire)하게 된다. 출력 플레이스에 나타나는 토큰의 가중값은 트랜지션의 가중값과 실행하기 전에 입력 플레이스에 있는 토큰의 가중값을 이용하여 계산한다.

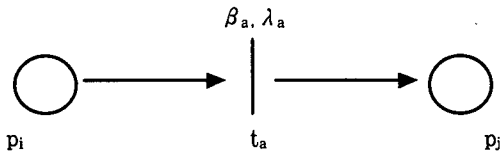
만일 $p_i \in I(t_a)$ 이고 $p_j \in O(t_a)$이라면, p_j는 p_i에서 직접 도달가능하다(direct reachable). p_i에서 직접 도달가능한 플레이스의 집합 DRS(p_i)을 직접도달집합(direct reachability set)이라고 한다. p_j가 p_i에서 직접도달가능하고, p_k는 p_j에서 직접도달가능하다면, p_k는 p_i에서 도달가능하다(reachable). 이러한 플레이스의 집합을 도달집합(reachability set) RS(p_i)라고 한다. 도달관계는 직접도달관계의 반사추이폐포(reflexive transitive closure)가 된다.

$p_i \in I(t_a)$이고 $p_j \in I(t_a)$이라면 p_j를 t_a에 대한 p_i의 이웃플레이스(neighbor place)라고 한다. 이러한 p_j의 집합을 이웃플레이스 집합(neighbor place set)이라고 하고 NPS(p_i)로 표시한다. 그리고 | NPS(p_i) | 는 원소의 수이다.

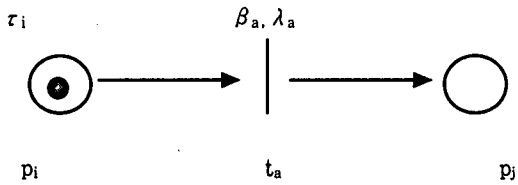
4.3 퍼지페트리네트 표현

이 절에서는 퍼지생성규칙의 1형, 2형, 3형 그리고 4형에 대한 퍼지페트리네트 표현을 기술한다. (그림 1)에서 (그림 3)는 1형 퍼지생성규칙, (그림 4)에서 (그림 6)는 2형 퍼지생성규칙, (그림 7)에서 (그림 9)는 3형 퍼지생성규칙 그리고 (그림 10)에서 (그림 12)는 4형 퍼지생성규칙에 대한 퍼지페트리네트 표현이다.

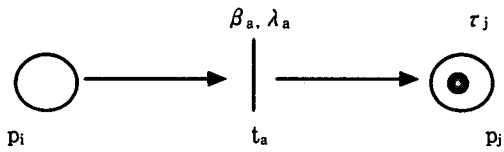
네 가지 형에 대한 퍼지페트리네트 표현에서 모든 입력 플레이스에 토큰이 존재하고, min{τ_i | $\tau_i = \tau(p_i)$, $p_i \in I(t_a)$} ≥ λ_a이라는 조건을 만족하면 실행가능해진다. 실행가능한 트랜지션 t_a가 실행되면 입력 플레이스에서는 토큰을 제거하고 각각의 출력 플레이스에 한개씩의 토큰이 나타나게 된다. 출력 플레이스에 나타난 토큰의 믿음값은 믿음값 합성함수 $\tau_j = \beta_{comp}(\beta(\tau_i, \beta_a))$ 또는 믿음값 결합함수 $\tau_j = \beta_{comb}(\beta(\tau_j, \tau_j^{old}))$를 이용하여 계산한다.



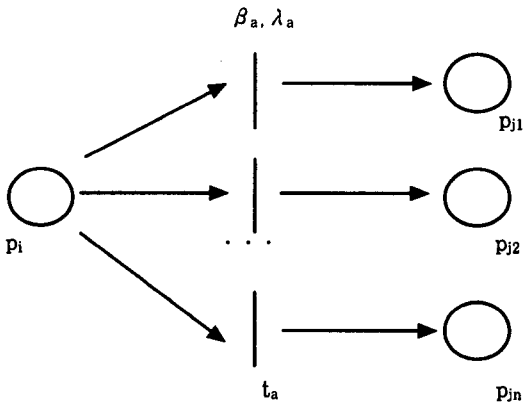
(그림 1) 1형 퍼지페트리네트 표현



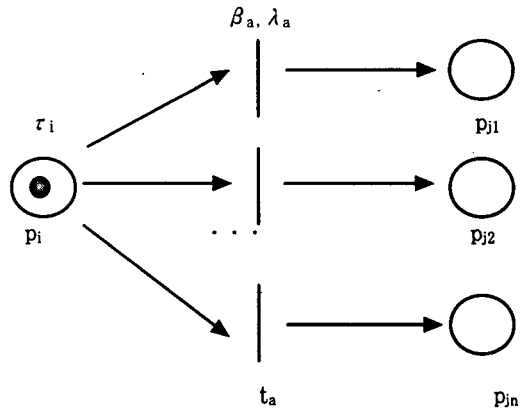
(그림 2) 실행가능한 1형 퍼지페트리네트 표현



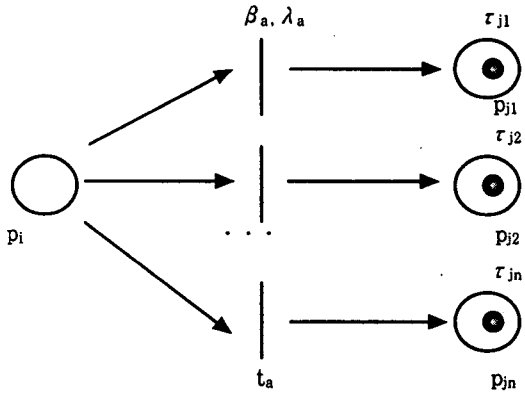
(그림 3) 실행된 1형 퍼지페트리네트 표현



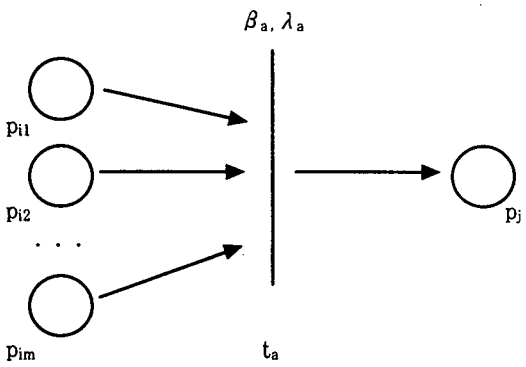
(그림 4) 2형 퍼지페트리네트 표현



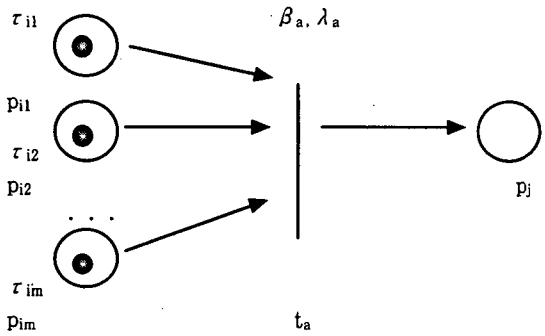
(그림 5) 실행가능한 2형 퍼지페트리네트 표현



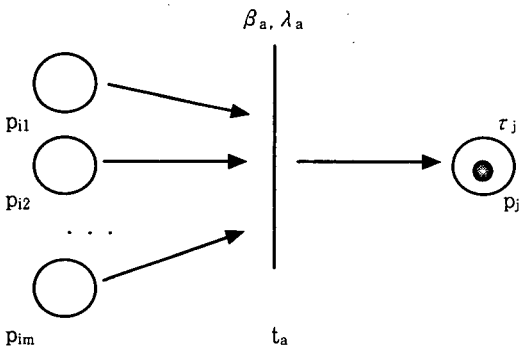
(그림 6) 실행된 2형 퍼지페트리네트 표현



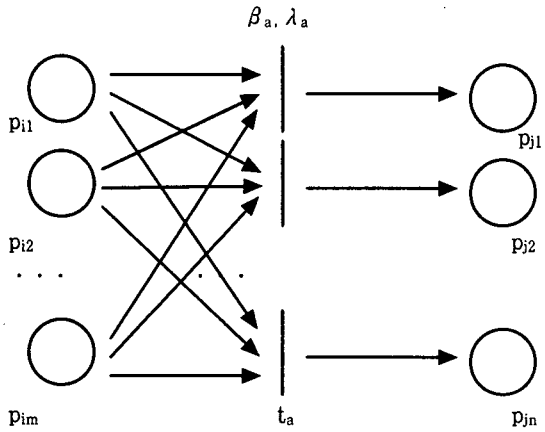
(그림 7) 3형 퍼지페트리네트 표현



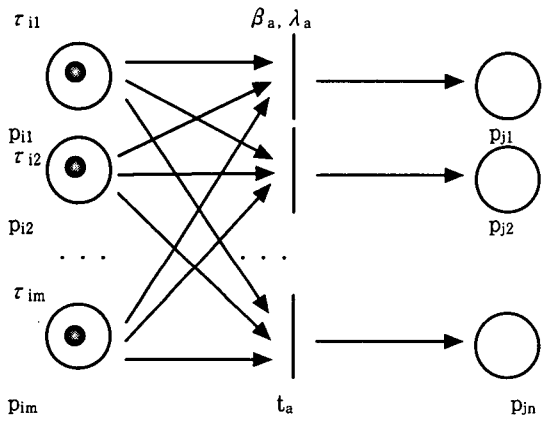
(그림 8) 실행가능한 3형 퍼지페트리네트 표현



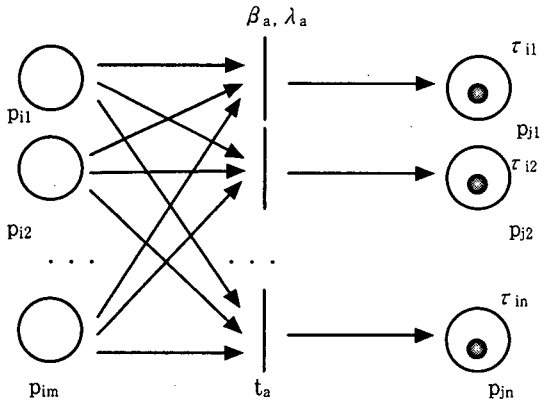
(그림 9) 실행된 3형 퍼지페트리네트 표현



(그림 10) 4형 퍼지페트리네트 표현



(그림 11) 실행가능한 4형 퍼지페트리네트 표현



(그림 12) 실행된 4형 퍼지페트리네트 표현

5. 퍼지추론 알고리즘

퍼지추론 알고리즘은 퍼지추론 시스템을 모형화하거나 또는 퍼지추론엔진을 구현하기 위한 퍼지페트리네트 표현에서 퍼지추론 알고리즘으로 사용하기 위한 것이다. 여기에서 제안하는 퍼지추론 알고리즘은 전진추론 알고리즘과 후진추론 알고리즘으로 구성된다.

5.1 전진추론알고리즘

전진추론 알고리즘은 NPS 중 믿음값이 없는 경우 사용자가 믿음값을 입력해야 하는 대화식 알고리즘이다. 시작 플레이스 p_s의 믿음값 $\tau_s = \tau(p_s)$를 알면 목표 플레이스 p_g에 있는 토큰의 믿음값 $\tau_g = \tau(p_g)$을 평가할 수 있다. 전진추론 알고리즘은 도달나무로 표현할 수 있다. 도달나무는 유한한 방향성 나무이고 나무의 각 노드는 (p_i)로 표현한다. (p_s)와 (p_g)는 각각 시작과 목표노드가 된다.

OPEN은 알고리즘이 생성한 노드 중에서 아직 조사하지 않은 노드, 즉 자식 노드를 생성하지 않은 노드를 관리하고, CLOSED는 이미 조사한 노드, 즉 자식 노드를 이미 생성한 노드를 관리하고, GOALS는 목표노드를 저장하는 리스트이다. λ는 임계값이고, β_{ik}는 p_i와 p_k사이에 있는 트랜지션의 믿음값이다. $|(p_i)|$는 플레이스 p_i에 있는 토큰의 수를 나타낸다.

Forward_Reasoning 알고리즘

입력: 시작노드 p_s; 출력: 목표노드의 믿음값 τ_g

```
I. OPEN ← (ps)
II. repeat until (OPEN == ∅)
    do for each pi in OPEN
    1 OPEN -= (pi)
    2 if (pi == pg) then GOALS ← (pg)
      else {
        if ( | DRS(pi) | ==0) then CLOSED ← (pi)
        else {
        2.1 if ( | DRS(pi) | ≥1 && NPS(pi) == ∅)
            then if ( τi ≥ λ && pg∈DRS(pi))
                 then { τg = βcomp( τi, βig);
                        GOALS ← (pg)}
        2.2 if ( | DRS(pi) | ≥1 && NPS(pi) != ∅)
            then {
              do for each pk in NPS(pi)
                 if ( | (pk) | == 0)
                    then { input τk; CLOSED ← (pk)}
                 if ( τi ≥ λ && ∀τk ≥ λ && pg∈DRS(pi))
                 then { τg = βcomp( τi, βig); GOALS ← (pg)}
            }
        2.3 do for each pj in DRS(pi)
              if (pj∉CLOSED)
              then {
                τj = βcomp( τi, βij);
                if (pj∈OPEN)
                then τj = βcomb( τj, τj^old)
                else OPEN ← (pj)
            }
        }
    3 CLOSED ← (pi)
III. return τg = max{ τgi | τgi∈GOALS}
```

5.2 예 1

d_1, d_2, d_3, d_4, d_5, d_6, d_7, d_8, d_9 그리고 d_{10}은 퍼지명제이고, 임계값 λ = 0.20이고, d_1과 d_2의 믿음값은 각각 0.90이며 목표노드는 d_9라고 가정하자. 지식베이스 내의 퍼지생성규칙이 다음과 같다고 가정하자.

RuleName$_1$: IF d_1 THEN $d_3 \wedge d_4$ (β=0.90)
RuleName$_2$: IF $d_2 \wedge d_3$ THEN $d_5 \wedge d_6$ (β=0.80)
RuleName$_3$: IF d_4 THEN d_7 (β=0.85)
RuleName$_4$: IF $d_5 \wedge d_6$ THEN d_8 (β=0.95)
RuleName$_5$: IF $d_7 \wedge d_8$ THEN d_9 (β=0.95)

이 지식베이스의 퍼지페트리네트 표현은 (그림 13)과 같고, 직접도달집합과 도달집합 그리고 이웃플레이스 집합은 각각 〈표 3〉과 〈표 4〉와 같다. (그림 13)에서 p_1은 시작노드이고, p_9은 목표노드이다. $NPS(p_3) = \{p_2\}$이므로 t_2가 실행되기 위해서는 (p_2)의 믿음값을 사용자가 입력해야한다. 전진추론 알고리즘을 실행한 후 생성되는 도달나무는 (그림 14)와 같다. 믿음값 계산의 편의상 퍼지 생성규칙의 전제부와 결론부가 비퍼지하다고 가정한다. GOALS = $\{\tau(p_{9a}), \tau(p_{9b}), \tau(p_{9c})\}$이므로 τ_9의 믿음값은 믿음값 결합함수에 의해 다음과 같이 계산하게된다.

$$\tau_9 = \max(\tau(p_{9a}), \tau(p_{9b}), \tau(p_{9c}))$$

= max(0.63, 0.63, 0.69)
= 0.69

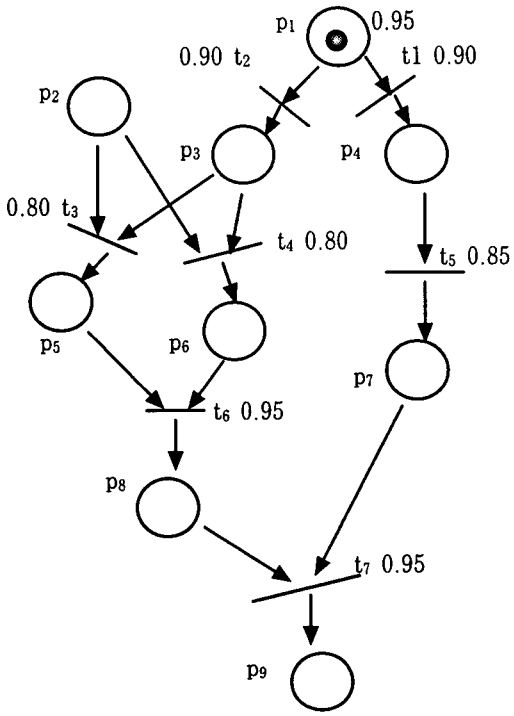
전진추론 알고리즘의 중요한 속성 중에 하나가 알고리즘의 유한성이다. 알고리즘의 유한성은 도달나무가 유한하다는 성질을 이용하여 증명할 수 있다. 전진추론 알고리즘의 유한성 증명은 [2], [5]에서 볼 수 있다.

〈표 3〉 직접도달집합과 도달집합

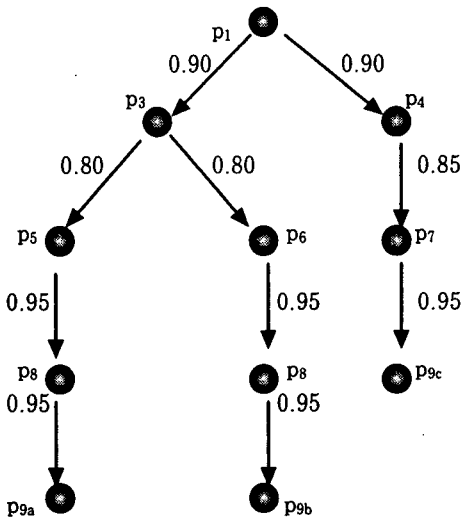
플레이스 p_i	DRS(p_i)	RS(p_i)
p_1	p_3, p_4	p_3, p_4, p_5, p_6, p_7, p_8, p_9
p_2	p_5, p_6	p_5, p_6, p_8, p_9
p_3	p_5, p_6	p_5, p_6, p_8, p_9
p_4	p_7	p_7, p_9
p_5	p_8	p_8, p_9
p_6	p_8	p_8, p_9
p_7	p_9	p_9
p_8	p_9	p_9
p_9	∅	∅

〈표 4〉 이웃플레이스 집합

플레이스 p_i	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9
NPS(p_i)	∅	p_3	p_2	∅	p_6	p_5	p_8	p_7	∅



(그림 13) 퍼지페트리네트 표현



(그림 14) 도달나무

5.3 후진추론 알고리즘

후진추론 알고리즘은 목표노드에서 시작노드까지의 추론통로를 찾아주는 Find_Backward_Path와 목표노드의 믿음값을 평가하는 Evaluate_Belief인 두 가지 알고리즘으로 구성된다.

5.3.1 Find_Backward_Path 알고리즘

Find_Backward_Path 알고리즘은 주어진 퍼지페트리네트 FPN을 이용하여 목표노드에서 시작노드까지 후진추론통로를 찾아준다. 즉, 주어진 FPN에서 후진추론통로만을 갖는 축소된 FPN을 출력한다. 이 알고리즘에서 FPN은 퍼지생성규칙을 표현한 퍼지페트리네트이고, GOAL은 목표노드이다. DRS는 직접도달집합이며, N은 FPN내의 플레이스의 수이다. OPEN은 부모노드를 찾기 위해 현재 처리중인 플레이스, 즉 노드의 리스트이다. 현재 처리중인 노드의 부모노드를 찾아서 저장하기 위한 리스트가 PARENTS와 PARENTSHIP이다. PLACES와 TRANSITIONS는 FPN을 이용하여 추론 알고리즘이 찾아낸 플레이스와 트랜지션들의 리스트이다. Find_Backward_Path 알고리즘이 종료된 후에 PLACES와 TRANSITIONS에는 후진통로 상에 있는 플레이스와 트랜지션들만 남는다. p_i와 p_j는 플레이스이고, t_{ij}는 플레이스 p_i와 p_j 사이에 있는 트랜지션을 나타낸다.

Find_Backward_Path 알고리즘
입력: 퍼지페트리네트 FPN, 목표노드 GOAL,
직접도달집합 DRS, 플레이스의 수 N
출력: 목표노드에서 시작노드까지의 통로

```
I. OPEN = GOAL
II. repeat until (OPEN == ∅)
  1. PARENTS = ∅
  2. do for each pj in OPEN
    2.1 OPEN -= (pj)
    2.2 PLACES ← (pj)
    2.3 PARENTSHIP = ∅
    2.4 do for i from 1 to N
        if (pj ∈ DRS(pi))
        then {PARENTSHIP ← (pi);
              TRANSITIONS ← (tij)}
    2.5 PARENTS ← PARENTSHIP
  3. OPEN = PARENTS
III. return PLACES, TRANSITIONS
```

5.3.2 Evaluate_Belief 알고리즘

Evaluate_Belief 알고리즘은 Find_Backward_Path 알고리즘이 찾아낸 축소된 퍼지페트리네트를 이용하여 목표노드의 믿음값 τ_g를 계산하여 출력한다. 알고리즘에서 STARTS는 축소된 퍼지페트리네트 내에 있는 시작노드의 리스트이고, TRANSITIONS는 축소된 퍼지페트리네트 내에 있는 트랜지션의 리스트이다. TOKENPLACES는 축소된 퍼지 페트리네트 내에서 토큰을 가지고 있는 플레이스들의 리스트이다. ENABLES는 현재 실행가능한 트랜지션의 리스트이다. p_i와 p_j는 플레이스이고, t_a는 트랜지션이며 $\tau_j = \tau(p_j)$이다. τ_j^{old}는 다른 통로를 통해서 먼저 계산되어 TOKENPLACES내에 이미 존재하는 플레이스 p_j의 믿음값이다.

Evaluate_Belief 알고리즘
입력: 시작노드 STARTS,
트랜지션집합 TRANSITIONS
출력: 목표노드의 믿음값 τ_g

```
I. TOKENPLACES = STARTS
II. if (pg) ∈ TOKENPLACES
    then GOALS ← (pj)
    else {
      repeat until (TRANSITIONS == ∅)
      1. do for each ta in TRANSITIONS
          if (I(ta) ⊆ TOKENPLACES)
          then { ENABLES ← (ta);
                 TRANSITIONS -= (ta)}
      2: do for each ta in ENABLES
        2.1 do for each pi∈I(ta)
            if ( | (pi) | == 0) then input τi
        2.2 if (min{τi | τi=τ(pi), pi∈I(ta)} ≥ λa)
            then do for each pj in O(ta)
            { τj = βcomp(τi, βa)
              if (pj==pg) then GOALS ← pj
              if (pj ∈ TOKENPLACES)
              then τj = βcomb(τj, τj^old)
              else TOKENPLACES ← (pj)
            }
```

2.3 ENABLES -= (t_a)

}

Ⅲ. return τ_g=max{τ_{gi} | τ_{gi}∈GOALS}

5.4 예 2

d1, d2, d3, d4, d5, d6, d7, d8, d9 그리고 d10은 명제이고, 임계값 λ = 0.20이고, d1과 d2의 믿음값은 각각 0.90이며 목표노드는 d9라고 가정하자. 그리고 퍼지생성규칙은 다음과 같다고 가정한다.

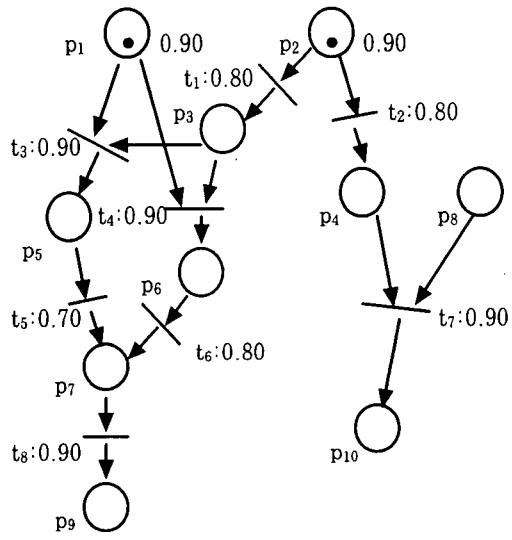
RuleName$_1$: IF $d_1 \wedge d_3$ THEN $d_5 \wedge d_6$ (β =0.90)
RuleName$_2$: IF d_2 THEN $d_3 \wedge d_4$ (β=0.80)
RuleName$_3$: IF $d_4 \wedge d_8$ THEN d_{10} (β=0.90)
RuleName$_4$: IF d_5 THEN d_7 (β=0.70)
RuleName$_5$: IF d_6 THEN d_7 (β=0.80)
RuleName$_6$: IF d_7 THEN d_9 (β=0.90)

퍼지생성규칙의 퍼지페트리네트로 표현과 직접도달집합은 각각 (그림 15)와 〈표 5〉와 같다. Find_Backward_Path는 퍼지페트리네트와 목표노드를 입력으로 받아 직접도달집합을 이용하여 목표노드에서 시작노드까지 후진방향으로 부모관계를 갖는 플레이스를 반복적으로 찾아 후진추론통로를 찾아낸다. 그러므로 Find_Backward_Path는 퍼지페트리네트에서 후진추론통로만을 갖는 더 작은 차원의 축소된 퍼지 페트리네트를 찾아준다.

(그림 15)에서 목표노드는 p_9이다. p_9의 부모노드는 p_7이고, 이들간의 트랜지션은 t_8이 된다. 그리고 p_7의 부모노드는 p_5와 p_6이 되고, 트랜지션은 t_5와 t_6이 된다. 이와 같은 방법을 반복 적용하면, 플레이스 p_1, p_2, p_3, p_5, p_6, p_7 과 p_9 그리고 트랜지션 t_1, t_3, t_4, t_5, t_6 과 t_8이 Find_Backward_Path의 실행결과가 된다. 축소된 퍼지 페트리네트에서의 입력과 출력 플레이스는 〈표 6〉와 같다.

Evaluate_Belief는 축소된 퍼지페트리네트와 시작노드를 입력으로 받는다. Evaluate_Belief는 축소된 퍼지페트리네트 상에서 실행가능한 트랜지션을 모두 찾아 순차적으로 실행시킨다. 그리고 추론통로 상에 나타나는 모든 플레이스의 믿음값을 믿음값 합성함수와 믿음값 결합함수를 이용하여 계산한다. 이렇게 반복 실행하여 얻은 목표노드의 믿음값을 이 알고리즘의 실행결과로서 출력한다.

시작노드에서 믿음값의 계산이 시작된다. 믿음값 계산은 전제부와 결론부가 모두 비퍼지하다고 가정한다. 플레이스 p_2에 토큰이 존재하므로 트랜지션 t_1이 실행가능해진다. t_1이 실행되면 p_3의 믿음값은 0.72가 된다. 다음으로 t_3와 t_4 실행되어 p_5와 p_6의 믿음값은 모두 0.65가 된다. 다음으로 t_5와 t_6이 실행되어, t_5를 통한 p_7의 믿음값은 0.46이고 t_6을 통한 p_7의 믿음값은 0.52가 된다. 믿음값 합성함수에 의해 p_7의 믿음값은 0.52가 된다. 그리고 목표노드 p_9의 믿음값은 0.47이 된다.



(그림 15) 퍼지페트리네트 표현

〈표 5〉 직접도달집합

플레이스 p_i	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}
DRS(p_i)	p_5 p_6	p_3 p_4	p_5 p_6	p_{10}	p_7	p_7	p_9	p_{10}	∅	∅

〈표 6〉 축소된 퍼지페트리네트의 입출력 플레이스의 집합

트랜지션 t_i	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8
I(t_i)	p_2		p_1 p_3	p_1 p_3	p_5	p_6		p_7
O(t_i)	p_3		p_5	p_6	p_7	p_7		p_9

6. 결론

본 논문에서는 지식공학분야에서 사용하는 부정확한 지식표현을 위해 사용하는 퍼지생성규칙의 분류를 기반으로 하는 퍼지페트리네트 표현과 여기에서 사용할 수 있는 추론 알고리즘인 전진추론 알고리즘과 후진추론 알고리즘을 제안하였다. 퍼지페트리네트 표현은 응용영역의 애매한 지식을 체계적이고 구조적으로 표현할 수 있고, 응용분야의 문제에 따라 이 문제를 해결하는 데 적합한 추론 알고리즘을 선택하여 추론의 효율을 높일 수 있다.

퍼지페트리네트 표현은 다른 그래프표현에 비해 퍼지추론 과정이나 퍼지시스템을 용이하게 모형화할 수 있을 뿐만 아니라 퍼지생성규칙들간의 관계를 쉽게 표현할 수 있다. 그리고 체계적인 퍼지페트리네트 표현은 전문가시스템의 규칙베이스의 검증에도 사용할 수 있다.

알고리즘에서 사용된 믿음값 평가함수들은 기존의 연구처럼 단순히 min과 max 연산만을 사용하지 않고, 규칙의 전제부나 결론부에 퍼지개념의 유무에 따른 적절한 평가함수를 사용하므로 사람들이 문제를 해결하기 위해 사용하는 추론과 직관에 보다 더 유사하다.

전진추론 알고리즘은 추론통로를 도달나무로 표현할 수 있으며, 추론시간은 추론시 만들어지는 도달나무에서 생성되는 노드의 수에 비례한다. 후진추론 알고리즘의 추론시간은 축소된 퍼지페트리네트에 나타나는 플레이스와 트랜지션의 수에 비례한다.

본 논문이 제안한 추론 알고리즘은 단일 프레세서상에서 개발되었으므로 퍼지페트리네트의 장점 중의 하나인 병렬성을 이용할 수 있는 병렬 추론 알고리즘의 개발이 필요하고, 지식표현에 사용한 퍼지명제의 제한된 표현성을 극복하기 위해 술어논리 등과 같은 보다 유연한 지식표현에 대한 퍼지페트리네트 표현과 추론 메카니즘의 연구가 필요하다. 그리고 퍼지페트리네트 표현이 복잡해질 경우에 사용할 수 있는 계층적인 표현방법과 이때 사용할 수 있는 추론 알고리즘의 개발이 필요하다.

참고문헌

[1] Murata, T., "Petri Nets: Properties, Analysis and Applications," Proceedings of the IEEE, Vol. 77, No. 4, April, pp541-580, 1989.

[2] Peterson, J. L., Petri Net Theory and the Modeling of Systems, Prentice-hall, 1981.

[3] 전명근, 변증남, "Fuzzy Petri Nets를 이용한 퍼지추론 시스템의 모델링 및 추론기관의 구현," 전자공학회논문지, 제 29 권, 제 7 호, pp508-519, 1992, 7.

[4] 조상엽, 김기태, "퍼지페트리네트를 이용한 퍼지생성규칙의 표현," 한국정보과학회논문지, 제 21 권, 제 2 호, pp298-306, 1994, 2.

[5] Chen, S., Ke, J., and Chang, J., "Knowledge Representation Using Fuzzy Petri-nets," IEEE Trans. on KDE, Vol. 2, No. 3, Sep., pp311-319, 1990.

[6] Liu, N. K., and Dillon, T. S., "Modeling Uncertainty in Expert Sytems," PRICAI-92, pp610-616, 1992.

[7] Garg, M. L., Ahson, S. I., and Gupta, D. V., "A Fuzzy Petri-nets for Knowledge Representation and Reasoning," Information Processing Letters, 39, pp165-171, 1992.

[8] Konar, A., and Mandal, A. K., "Uncertainty Management in Expert Systems Using Fuzzy Petri Nets," IEEE Trans. on KDE, Vol. 8, No. 1, pp96-105, 1996.

[9] Sheng-Ke Yu, "Comments on Knowledge Representation Using Fuzzy Petri Nets," IEEE Trans. on KDE, Vol. 7, No. 1, Feb., pp190-192, 1995.

[10] Manoj, T. V., Leena J., and Soney, R. B., "Knowledge Representation Using Fuzzy Petri Nets-Revisited," IEEE Trans. on KDE, Vol. 10, No. 4, Jul./Aug., pp666-667, 1998.

[11] 조상엽, "퍼지생성규칙을 위한 퍼지페트리네트 표현에서 후진추론," 한국정보처리학회논문지, 제 5 권, 제 4 호, pp951-958, 1998, 4.

[12] Looney, G. C., and Alfize, A. A., "Logical Controls via Boolean Rule Matrix Transformation," IEEE Trans. on SMC, Vol. 17, No. 6, Nov./Dec., pp1077-1082, 1987.

[13] Looney, G. C., "Fuzzy Petri Nets for Rule-based Decision Making," IEEE Trans. on SMC, Vol. 18, No. 1, Jen./Feb., 1988.

[14] Buchanan, B. G., and Shortliffe, E. H., Rule-based Expert Systems: the MYCIN Experiments of the Stanford Heuristic

Programming Project, Readings, MA: Addison-Wesely, 1984.

[15] Leung, K. S., and Lam, W., "Fuzzy Concepts in Expert Systems," IEEE Computer, Sep., pp43-56, 1988.

[16] Fukami, S., Mizumoto, M., and Tanaka, K., "Some Considerations on Fuzzy Conditional Inference," Fuzzy Sets and Systems, Vol. 4, pp243-273, 1980.