범용 실시간 퍼지 제어를 위한 시간형 퍼지 패트리넷

이 강 수[†] · 김 소 연[†] · 윤 정 모[††]

요 약

본 논문에서는 실시간 퍼지 제어를 위한 모형으로서 '시간형 퍼지 패트리넷(TFPN) 모형'을 제시한다. TFPN 모형은 시간 패트리넷과 퍼지 패트리넷을 통합한 것으로서 퍼지 추론 뿐 아니라 퍼지 제어에 이용할 수 있다. 또한, 퍼지 제어 규칙의 구문적 명세 언어로서 '시간적 퍼지 제어 언어'를 정의하고, 이를 TFPN으로 모형화하는 방법을 제시한다. TFPN모형은 퍼지 제어에 대한 패트리넷 포멀리즘에 해당하며 그 수행 규칙은 마킹(퍼지화) 과정과 점화(추론 및 비퍼지화) 과정으로 구성된다. 제시된 모형의 사례 연구 결과, 기존의 퍼지 제어 모형보다 추론 및 제어 값의 계산 시간을 절약할 수 있으며, 제어 시스템의 불확실성을 자연스럽게 모형화하고 제어 규칙의 가시성을 높일 수 있다.

A Timed Fuzzy Petri Net Model for General Purpose Real-time Fuzzy Control

Gang-soo Lee[†] · So-yeon Kim[†] · Jung-mo Yun[††]

ABSTRACT

In this paper, we propose a Timed Fuzzy Petri Net (TFPN) model as a new model of real-time fuzzy control. The TFPN model, which is useful for fuzzy inference and fuzzy control, is an integrated model of Timed Petri Net and Fuzzy Petri Net. Additionally, a Timed Fuzzy Control Language is defined as a textual specification model of fuzzy control rules, and proposed a TFPN modeling method. The TFPN model is a Petri Net formalism of fuzzy control systems. Execution rule is consisted of marking(i.e., fuzzyfication) and firing(i.e., inference and defuzzyfication) procedures. A simple case work by using TFPN model shows us computing time of inference and defuzzyfication is low and uncertainty and visibility of fuzzy control rule are modeled effectively.

1. 서 론

패트리넷 모형[1](이하 '넷'으로 약칭함)과 퍼지 모형[2]을 통합한 '퍼지 넷'(fuzzy Petri net)[4]은 불확실한 지식 표현과 퍼지 추론[3, 6~10, 13~15, 21], 학습[5, 11, 19], 로보트 제어를 위한 시퀀스 생성[12, 18] 및 PLC (programmable logic controller)를 위한 전 처리 모형[20]으로 활용되고 있다. 그러나, 기존의 퍼지 넷들은 넷의 수행 개념 및 도형적 장점만을 활용할 뿐, 넷의 중요한 장점인 병행성(concurrency), 비동기성(asynchronous) 및 비결정성(nondeterminism)을 효과적으로 살리지는 못하고 있다. FPN은 불확실한 지식이나 생성 규칙(production rule)의 가시적 표현[5, 9, 21]과 추론에 적용하였고, 퍼지화, 비퍼지화 및 실

*이 논문은 1994년도 한국학술진흥재단의 공모과제 연구비에 의하여 연구되었음
† 정 회 원:한남대학교 전자계산공학과
†† 종신회원:서울산업대학교 전자계산학과
논문접수:1995년 10월 24일, 심사완료:1996년 12월 25일

제 제어 부분을 깊이 다루지 않았다. 즉, 퍼지 넷은 퍼지 제어 시스템의 모형화 및 분석 수준에만 적용하고 있다. 또한, AND 및 OR 형태의 생성 규칙에 대한 넷 모형에 일관성이 없으며[6, 17, 21], 이에 대한 넷 차원의 정립이 요구된다.

한편, 넷에서는 실 시스템의 불확실성을 확률론적으로 모형화하기 위해 스토케스틱 프로세스를 기반으로한 '스토케스틱 넷'(stochastic Petri net)[23]을 이용할 수 도 있으나, 퍼지 이론을 기반으로 하는 FPN 모형보다는 불확실성을 나타내기에 부족하다[2, 28]. 또한, 실 시스템의 시간 요소를 모형화하기 위해 '시간 넷' (time, timed 또는 temporal Petri net)[24, 25]이 사용되며, 시간 요소의 불확실성을 모형화하려면, 시간 넷과 퍼지 넷의 특성을 살린 통합적 모형을 이용하면 좋을 것이다.

넷 모형들은 공장 자동화 시스템의 최 상위 계층인 '엔터프라이즈'(예를 들어, CIM (computer integrated manufacturing) 및 유연 생산 시스템)부터 최하위 계층인 '현장 공정제어'(예를 들어, 로보트 제어 등)까지의 모든 계층에 성공적으로 이용되어 왔다[26, 27]. 특히, '시퀀스 제어기'는 CIM의 핵심 메커니즘이며, 사양서 작성, 제어 논리 설계, 구현, 시험 및 유지-보수 작업에 CIM의 대부분의 비용이 소요된다. 첫째, 'PLC 방식'의 시퀀스 제어기는 주로 relay ladder diagram이나 프로그램 언어로 개발되므로, 가시성, 융통성 및 유지 보수성이 결여되며[27, 29, 30], 시스템의 입·출력 및 상태가 불확실할 경우에도 PLC방식을 적용하기가 어렵다. 둘째, '퍼지 논리 제어기(fuzzy logic controller, FLC) 방식'은 불확실성 문제를 해결하고 있으며[2, 28, 29, 30], 분산형, 논리형 및 언어적 제어를 할 수 있고, 전용 칩까지 개발되어 있다는 장점이 있으나, 많은 량의 계산이 실시간적으로 요구되며, 제어 규칙의 병행성을 가시화하기가 어렵다는 단점이 있다. 또한, 범용 퍼지 제어기의 개발이 어렵고 제어 규칙들의 분석을 위한 모형이 부족하다.

이와 같은 배경에서, 본 연구에서는 퍼지 넷과 시간 넷 모형을 통합하여 제어 문제들의 병행성, 비결정성 및 불확실성을 쉽게 모형화하고, 기존의 PLC와 FLC 방식의 단점들을 해결할 수 있는 '시간적 퍼지 패트리넷(timed fuzzy Petri net, TFPN)'모형을 제시한다. TFPN모형은 불확실한 제어 규칙의 병행 명세, 병행 추론 뿐 아니라, 추론 결과를 구동기(actuator)의 제어에 적용하는 모형으로서, 'TFPN 제어방식'이 된다. 본 연구의 방향은 새로운 퍼지 제어 이론을 제시하는 것이 아니라, 넷의 응용 연구 차원에서 새로운 응용 가능성을 제시하는 것이다. 또한, 지금까지 주로 연구실에서만 이론적으로 연구 되어온 넷을 실제 산업 현장에서 이용할 수 있도록 하는 것이 연구의 목적이며, 이를 위해, 최근 많은 연구가 진행 되어온 CIM의 최하위 계층인 퍼지 제어기를 TFPN모형을 이용하여 개발하는 방법론을 제시한다.

본 논문의 2장에서는 일반적인 제어 시스템의 구성과 제어 시스템의 구문적 명세 모형인 시간적 퍼지 제어 언어를 정의한다. 3장에서는 TFPN을 정의하고 TFPN모형화 방법을 제시하며, 4장에서 그 특성 및 수행 규칙을 제시한다. 5장에서는 간단한 직렬 모터의 퍼지 제어 문제를 사례로 하여 제시된 방법의 적용 결과를 보인다. 끝으로, 6장에서는 기존의 퍼지 넷들과 TFPN의 비교 결과와 향후 연구 과제를 결론과 함께 제시한다.

2. 제어 시스템과 시간적 퍼지 제어 언어(TFCL)

전형적인 제어 시스템은 제어 대상 시스템내의 센서를 통해 제어용 자료를 획득하고, 내장된 제어규칙(FLC에서) 또는 프로그램(PLC에서)으로부터 제어값(예를 들어, 전류, 구동 시간 등)을 구하여 구동기를 작동하므로서 제어가 이루어진다. 본 연구에서는 타이머 등과 같은 시간적 구동기의 구동 시간을 퍼지 제어하는 것이다. 전형적인 제어 시스템은 [정의 1]과 같다.

[정의 1] 제어 시스템 = (SEN, ACT, lf_{SEN}, lf_{ACT}, Π, CONTROL)

- SEN = $\{sen_1,...,sen_i,...,sen_{|SEN|}\}$: 센서 집합(|SEN| > 0)
- lf_{SEN}: SEN → SF × α: 센서 레이블 함수. SF = $\{\langle SF_1,...,\langle SF_{|SF|}\}$: 센서 귀속 함수 집합
  SF_i: $[a_i, b_i] \rightarrow [0, 1]$: 센서 귀속 함수 ($a_i$와 b_i는 귀속 함수의 정의역내의 두 점).
  α는 [0, 1]사이의 임계치(threshold) 집합
- ACT: $\{act_1,...,act_j,...,act_{|ACT|}\}$: 구동기 집합(|ACT

|) 0)

- lf_{ACT}: ACT → AF: 구동기 레이블 함수
  여기서, AF = $\{AF_1,...,AF_{|AF|}\}$: 구동기 귀속 함수 집합
  AF_i: $[a_i, b_i] \rightarrow [0, 1]$: 구동기 귀속 함수 ($a_i$와 b_i는 귀속 함수의 시간 정의역내의 두 점)
- CONTROL ⊆ SEN × ACT: 제어 부분이며 TFCL (구문형태) 또는 TFPN(그래프형태)로 모형화됨
- Π: 귀속 함수의 분할(partition)
  Π_{SEN}: SF의 센서별 분할(예를 들어, $\Pi sen1$에 sen_1에 부여된 귀속 함수의 집합이다)
  Π_{ACT}: AF의 구동기별 분할(예를 들어, Π_{act1}은 act_1에 부여된 귀속 함수의 집합이다)

시간적 퍼지 제어 언어(timed fuzzy control language, TFCL)는 부록 A에서 정의하였으며, 제어 시스템의 CONTROL부분을 텍스트 형태로 표시한 새로운 퍼지 제어 언어이다. TFCL은 기존의 퍼지 제어 언어들과 유사하며 텍스트 형태이므로 가시성이 결여되고 병행적인 시스템의 제어 규칙을 모형화하기가 어렵다. 이 문제는 그래프 형태인 TFPN모형을 이용함으로서 다소 해결할 수 있다. TFCL은 TFPN과 논리적으로 동일한 의미와 특성을 가지므로, TFPN으로 변환할 수 있고 TFPN수행기에 의해 처리된다. 부록 A와같이 연산자들의 우선 순위는 NOT 〉 AND 〉 OR 이며, 〈단위제어규칙〉은 기존의 퍼지 생성 규칙[6, 9, 14, 21]처럼 IF-THEN형태로 기술되지만, 의미는 다르다. 즉, 기존의 퍼지 생성 규칙은 전제와 결론간의 함축(imply)을 나타내지만, 본 모형에서는 센서의 조건과 구동기의 제어 행위(또는, 사건)간의 관계(즉, '원인과 결과'[22, 28])를 나타낸다. 다음은 〈단위제어규칙〉의 예를 보인다.

[예] Rule1: **IF** (온도 = 매우높다 (0.2) AND 습도 = 높다 (0.3))
센서1 귀속함수1 임계치1 센서2 귀속함수2 임계치2
조건퍼지명제1 조건퍼지명제2

THEN (창문개폐 = 오랫동안) (0.9).
구동기 귀속함수 규칙의 확신도
제어퍼지명제

3. 시간적 퍼지 패트리넷(TFPN) 모형

3.1 정의

TFPN모형은 제어 시스템의 CONTROL부분에 대한 수행가능(executable), 공식적(formal) 및 도형적(graphical) 명세 및 분석 모형으로서, 시간 넷, 퍼지 넷 및 퍼지 제어 개념을 통합한 모형이다.

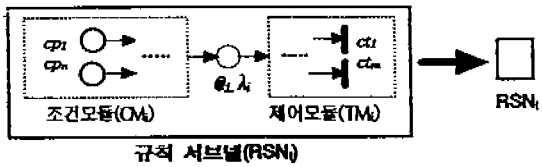
[정의 2] TFPN: (P, T, I, O, μ, χ, η)

- P = (CP, NP, @, lf_{CP}, λ): 플레이스 (원으로 표현)
  - CP = $\{cp_1,...,cp_n\}$: 조건(condition)플레이스 집합 (진한 원으로 표현)
    π_{seni}: CP의 sen_i별 분할
  - NP = $\{np_1,...,np_o$: 일반(normal)플레이스
  - @ = $\{@_1,...,@_m\}$: 중간(intermediate)플레이스
  - lf_{CP}: $CP \rightarrow$ SEN × SF × α: 조건 플레이스에 대한 레이블 함수 (레이블은 '〈센서변수,센서귀속함수,임계치〉'형태로 부여함). α는 [0, 1] 사이의 임계치 집합
  - λ: @ → [0, 1]: 규칙의 확신도(certainty factor) 가중함수
- T = (CT, MT, NT, lf_{CT}: 트랜지션 (막대로 표현)
  - CT = $\{ct_1,...,ct_m\}$: 제어(control)트랜지션 집합 (진한 막대로 표현)
    π_{acti}: CT의 act_i별 분할
  - MT = $\{mt_1,...,mt_n\}$: 최대(maximum)트랜지션 집합(이중 막대로 표시)
  - NT = $\{nt_1,...,nt_p\}$: 일반트랜지션 집합
  - lf_{CT}: $CT \rightarrow$ ACT × AF: 제어플레이스에 대한 레이블 함수
    (레이블은 '〈구동기변수, 구동기귀속함수〉' 형태로 부여함)
- I: T → P: 트랜지션에 대한 입력 함수(호(arc)로 표현)
- O: T → P: 트랜지션으로부터의 출력 함수(호로 표현)
- μ: P → [0, 1]: 퍼지 마킹 함수 ($\mu(p_i)$*(퍼지 값)로 표시)
- χ: P → {0, 1}: 토큰 마킹 함수 ($\chi(p_i)$를 $\#p_i$(토큰수)로 표시)

- η : CT → δ : 퍼지 점화 함수 (δ_i = [LB_i, UB_i], 점화 시간 구간이며 LB_i는 각각 하한 실수값 및 상한 실수 값임)

복잡한 TFPN은 모듈 및 서브넷 개념을 이용하여 추상화할 수 있다. (그림 1)과같이 한 개의 제어 규칙은 TFPN에서 한 개의 '규칙 서브넷'(rule subnet, RSN)을 형성하며, 각 RSN은 '조건 모듈'(condition module, CM)과 '제어모듈(control module, TM)이 중간 플레이스 @로 연결되어있다. 한 TFPN내에서 다음조건을 만족해야한다.

- |RSN| = |CM| = |TM| = |@| = 제어 규칙 수
- $RSN_i = CM_i \cup TM_i \cup @ \subseteq TFPN$, $RSN_i \cap RSN_j = \phi$,
- RSN은 '이분 가능(biseparable)' 그래프의 일종이다 (@를 제거하면 CM과 TM으로 양분됨).
- CM_i는 1개이상의 조건플레이스를 가지며 제어 트랜지션은 존재하지 않는다.
- TM_i는 1개이상의 제어트랜지션을 가지며 조건 플레이스는 존재하지 않는다.



<단위제어규칙> R_i: IF (<조건부>) THEN (<제어부>) (λ_i).

(그림 1) 〈단위제어규칙〉의 모형화 (여기서, cp_j : 〈sen_j, SF_j, α_j〉, ct_k : 〈act_k, AF_k〉)
(Fig. 1) Modeling of 〈Unit_Control_Rule〉(where cp_j : 〈sen_j, SF_j, α_j〉, ct_k : 〈act_k, AF_k〉)

3.2 TFPN 모형화 방법

제어 시스템을 위한 TFPN모형은 제어할 시스템으로부터 그 제어 규칙 및 귀속 함수로부터 구할 수 있다. 전문가가 직접 TFPN모형을 사용하여 제어 시스템을 개발 할 경우, TFPN모형은 최종적인 제어 규칙의 분석 및 제어 뿐아니라, 제어 규칙을 구하는 과정에서도 이용될 수 있다. 그러나, 본 절에서는 TFCL 또는 제어 규칙으로부터 TFPN모형을 구하는 방법만을 제시한다. (표 1)은 TFCL을 TFPN으로 모형화하는 방법을 보인다.

상향식 모형화의 경우, 우선 〈센서〉 및 〈구동기〉를 조건플레이스와 제어트랜지션으로 각각 모형화하고 〈단위제어규칙〉의 〈조건부〉와 〈제어부〉를 모형화하므로서, CM과 TM 및 @로 구성된 RSN을 얻는다. 각 RSN들은 〈관계집합〉 (즉, 단위 규칙간의 관계)을 고려하여 전체 규칙에 대한 TFPN을 구성한다. 하향식 모형화의 경우 상향식 방법을 역순으로 적용한다.

(1) 단위 제어 규칙의 모형화 : 〈단위제어규칙〉 Ri는 앞의 (그림 1)과 같이 〈조건부〉는 CM으로, 〈제어부〉는 TM으로, CM과 TM사이를 중간플레이스 $@_i$로 각각 모형화한다. $@_i$는 'THEN'을 의미한다. 규칙의 확신도 λ_i는 $@_i$에 부여하며 R_i는 하나의 RSN으로 추상화(간략화)할 수 있다. CM_i와 TM_i의 내부의 합성 논리 식은 (표 2)에서 제시된 각 논리들에 대한 모형화 규칙을 이용하여 전체 TFPN을 구성한다. (표 2)에서 OR논리에 대한 모형화 규칙은 기존의 연구 내용[6, 8, 9, 10, 14, 16, 20, 21]들과 다르다. 만일 OR논리를 최대트랜지션만으로 표현한다면, 입력 플레이스중 하나의 마킹이 0일 경우(즉, 퍼지값이 0) 점화 조건을 만족하지 않으므로, 넷의 점화 규칙을 수정해야 한다. NOT의 경우, '금지 호'(inhibitor arc)[1]을 사용하고 있다. 본 연구에서는 표준 넷의 규칙들을 최대로 유지하면서 TFPN을 정의하였다. (그림 2)는 단위 제어 규칙들의 모형화 예를 보인다.

(2) 전체 제어 규칙의 모형화 : R_i 및 R_j를 제어 규칙들이 라하고 &를 '가상 규칙'(일반플레이스나 일반트랜지션으로 모형화 한다)이라 할 때, 이들 간의 수행 시간적 부분 관계(partial ordering)는 [정의 3]과 같다.

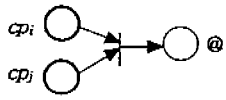
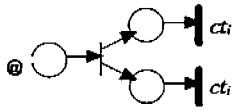
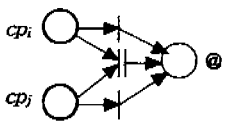
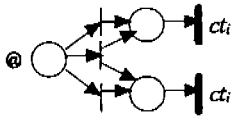
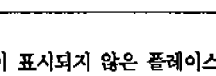
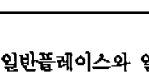
[정의 3] 단위 제어 규칙간의 부분 관계

- 순차(R_i *SEQ* R_j) : R_i는 R_j전에 발생(happened before)된다.
- 반복(R_i *ITR* R_i) : R_i *SEQ* R_i. R_i는 반복 적용된다.
- 병행(R_i *PAR* R_j) : $\neg(R_i\ SEQ\ R_j) \wedge \neg(R_j\ SEQ\ R_j)$. R_i와 R_j는 독립적이다.
- 분기(R_i *FORK* R_j) : (& *SEQ* R_i) ∧ (& *SEQ* R_j) ∧ (R_i *PAR* R_j). R_i와 R_j는 동시에 시작된다.
- 결합(R_i *JOIN* R_j) : (R_i *SEQ* &) ∧ (R_j *SEQ* &) ∧ (R_i *PAR* R_j). R_i와 R_j는 비동기적으로 종료된다.

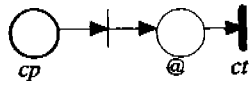
〈표 1〉 TFCL과 TFPN과의 대응 관계(모형화 방법)
〈Table 1〉 Correspondence between TFCL and TFPN

TFCL	TFPN 제어 모형
<센서> 및 <조건퍼지명제>	조건플레이스 및 <센서, 센서귀속함수, 임계치>형태로 부여된 레이블
한 센서 sen_i	조건플레이스 집합의 sen_i별 분할 π_{sen_i}
sen_i의 퍼지값	π_{sen_i}내의 조건플레이스 cp_j들의 퍼지값 (즉, 퍼지 마킹 과정), $cp_j^*=\mu(cp_j)$
<구동기> 및 <제어퍼지명제>	제어트랜지션 및 <구동기, 구동기귀속함수> 형태로 부여된 레이블
한 구동기 act_i	제어플레이스 집합의 act_i별 분할 π_{act_i}
act_i의 실제값	π_{act_i}내의 제어트랜지션 ct_j들의 퍼지값, $\delta_j=\eta(ip_j^*)$, (즉, 비퍼지화 과정), (여기서, ip_j는 ct_j의 유일한 입력플레이스)
<단위제어규칙> - <조건부>의 퍼지 논리식 - <제어부>의 퍼지 논리식 - <확신도>	규칙 서브넷 (RSN) - 조건 모듈(CM) - 제어 모듈(TM) - @의 레이블, λ
<제어규칙집합>	TFPN 전체 구조
<귀속함수집합>	조건플레이스 및 제어트랜지션에 부여된 레이블의 데이터 베이스
<관계집합>	단위 규칙 서브넷간에 연결된 일반트랜지션, 일반플레이스 및 호선
추론 과정	일반 및 최대트랜지션의 수행 과정
상태 및 상태 전이도	마킹 및 도달성 트리
제어 과정	제어트랜지션의 점화

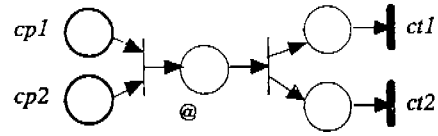
〈표 2〉 〈조건부〉 및 〈제어부〉의 모형화 규칙
〈Table 2〉 Modeling rules of condition and event part in a unit control rule

논리	조건부(CM)	제어부(TM)
단일		
AND		
OR		
NOT		

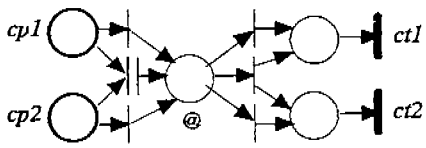
(주) 명칭이 표시되지 않은 플레이스와 트랜지션은 각각 일반플레이스와 일반트랜지션임



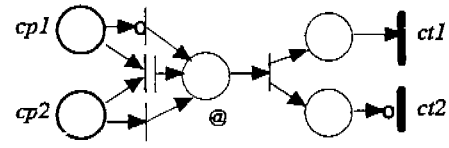
(a) IF (cp=SF (α)) THEN (ct=AF) (λ).



(b) IF (cp_1=SF_1 (α_1) AND cp_2=SF_2 (α_2)) THEN (ct_1=AF_1 AND ct_2=AF_2) (λ).



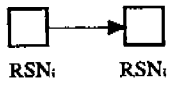
(c) IF (cp_1=SF_1 (α_1) OR cp_2=SF_2 (α_2)) THEN (ct_1=AF_1 OR ct_2=AF_2) (λ).



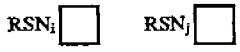
(d) IF (NOT cp_1=MF_1 (α_1) OR cp_2=MF_2 (α_2)) THEN (ct_1=AF_1 AND NOT ct_2=AF_2) (λ).

(조건플레이스의 레이블을 cp_i: <sen_i, SF_i, α_i> 이라 가정)

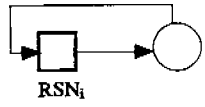
(그림 2) 제어 규칙들의 TFPN 모형화 예
(Fig. 2) Examples of TFPN modeling of a control rules



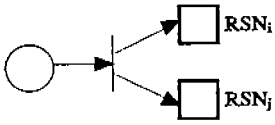
(a) 순차



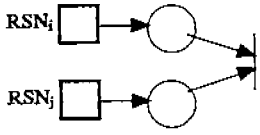
(b) 병행



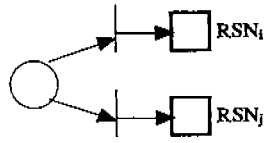
(c) 반복



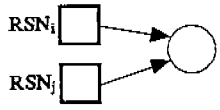
(d) 분기



(e) 결합



(f) 상호 배타



(g) 상호보완

(주) 명칭이 표기되지 않은 플레이스와 트랜지션은 각각 일반플레이스와 일반트랜지션임.

(그림 3) 제어 규칙들간의 부분 관계에 따른 〈제어규칙집합〉의 모형화 방법
(Fig. 3) Modeling method of 〈control_rule_set〉 according to relationships between control rules

- 상호 배타(mutual exclusion) (R_i *ME* R_j):[¬(& *SEQ* R_i) ∧(& *SEQ* R_j)] ∨[(& *SEQ* R_i) ∧ ¬(& *SEQ* R_j)]. R_i와 R_j중 1개만 선택되어 시작된다.
- 상호 보완(mutual inclusion) (R_i *MI* R_j):[(R_i *SEQ* &) ∨(R_j *SEQ* Q)] ∧(R_i *PAR* R_j). R_i와 R_j중 적어도 1개가 시작된다.

〈단위제어규칙〉은 규칙 서브넷으로 이미 모형화했음으로 [정의 3]의 관계와 (그림 3)의 규칙에 따라 연결하여 전체 TFPN을 점진적으로 모형화한다. 어떤 시스템을 넷으로 모형화하는 방법은 알고리즘적으로는 기술하기 어렵고 경험에 의존해야 하므로[1], TFPN모형을 구할 때도 제시된 방법을 참고하여 모형화한다.

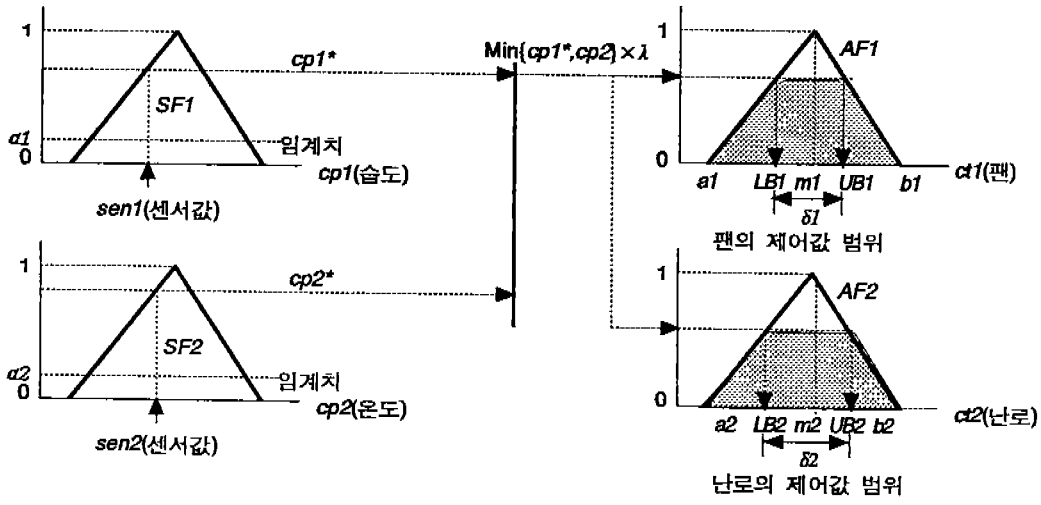
TFPN은 일종의 '레이블형' 넷이며, 호선의 다중도가 1이므로 'ordinary' 넷이다. 또한, '퍼지 마킹 함수' μ는 플레이스의 퍼지값을 결정하며 '토큰 마킹 함수' χ는 토큰 수(0 또는 1개)를 결정한다. 따라서, 플레이스내의 토큰 수(토큰 마킹)는 외형적으로 0 또는 1개가 존재하므로, 토큰 마킹만 고려한다면 'C/E net (condition/event net)'의 일종이다. 특히, P/T net (place/transition) net)[1]에서 마킹 함수 μ'는 "μ': $P\rightarrow$ N(자연수)"로 정의되므로, 플레이스에는 단위 토큰이 무한 정수개가 존재할 수 있다. TFPN의 퍼지 마킹 함수 μ에서 플레이스의 퍼지값(즉, 0과 1사이의 실수값)을 토큰 수로 간주하면, TFPN은 무한 정수개의 토큰이 존재하며 모형의 수준은 P/T net과 같아진다. 왜냐하면, 무한수의 개념에서, 자연수의 N의 갯수는 0과 1사이의 실수의 갯수와 같고 무한수이기 때문이다. 따라서, P/T net의 정의 [1]에서 "μ': $P\rightarrow$ N"을 "μ : $P\rightarrow$ 1/N"로 대치한다면, TFPN에서 "μ: $P\rightarrow$ [0, 1]"와 같아지므로 TFPN은 P/T-net과 같은 수준의 모형임을 알 수 있다.

[속성 1] TFPN은 'ordinary' 넷이며, 토큰 마킹만 고려한다면 'C/E net'의 일종이다. 또한, P/T-net과 같은 수준의 모형이다.

4. 수행 규칙 및 분석 방법

4.1 마킹 과정

센서값으로부터 그 센서를 모형화한 조건플레이스의 퍼지 마킹값을 얻거나(즉, 퍼지화(fuzzyfication) 과정), 점화에 의해 새로운 퍼지값을 구하는 과정(일반 및 중간플레이스에서)이다.



(그림 4) 퍼지 마킹과 퍼지 점화 개념 ("IF($cp1=SF1$ ($\alpha 1$) AND $cp2=SF2$ ($\alpha 2$)) THEN ($ct1=AF1$ AND $ct2=AF2$) (λ)" 규칙의 처리 예)

(Fig. 4) concepts of fuzzy marking and fuzzy firing (modeling of a rule, "IF($cp1=SF1$ ($\alpha 1$) AND $cp2=SF2$ ($\alpha 2$)) THEN ($ct1=AF1$ AND $ct2=AF2$) (λ)")

(1) 퍼지 마킹 : 조건플레이스의 경우, sen_i로부터 측정값이 입력되면, π_{seni}내의 모든 조건플레이스 cp_j에 대해 (여기서, cp_j의 레이블이 $\langle sen_i, SF_i, \alpha_i\rangle$이라 하자), 다음 과정이 동시에 처리된다(즉, 병행 마킹).

if $(SF_j(sen_i) \geq \alpha_j)$ then $cp_j^* = \mu(cp_j) = SF_j(sen_i)$
else $cp_j^* = \mu(cp_j) = 0$. (여기서, μ는 퍼지 마킹 함수이며 cp_j^*는 cp_j의 퍼지값임)

일반플레이스 np_i의 경우, $np_i^* = \mu(np_i) = [0, 1]$이며, 중간플레이스 $@_i$의 경우, $@_i$의 레이블이 λ_i일 때, $@_i^* = \mu(@_i) = [0, 1] \times \lambda_i$

(2) 토큰 마킹 : if $(p_j^* > 0)$ then $\#p_j = \chi(p_j) = 1$
else $\#p_j = \chi(p_j) = 0$. (여기서, χ는 토큰 마킹 함수이며 $\#p_j$는 p_j내의 토큰 수임)

한 센서 sen_i는 귀속 함수(언어 변수)만 다른 다수의 조건플레이스 cp_j들(즉, 조건플레이스 집합의 sen_i에 대한 분할 π_{seni})로 모형화되므로, sen_i 값이 입력되면 sen_i내의 모든 조건플레이스들이 병행적으로 퍼지 마킹되고 0이상의 퍼지값을 갖는 플레이스들은 토큰 마킹된다(즉, 토큰이 부여된다). 중간플레이스에서는 확신도를 고려하여 퍼지 마킹 값을 구한다. 예를 들어, (그림 4)는 가장 많이 사용되는 규칙[2~8]인 (그림 2-(b))에 대한 수행 개념을 보인다.

4.2 점화 과정

$I(t_i)$내의 모든 입력 플레이스 ip_j와 $O(t_i)$내의 모든 출력 플레이스 op_k에 대해, $\#ip_j = 1$이고 $\#op_k = 0$이면, t_i는 '점화가능(enabling)'이라 한다. 병행적으로 점화가능된 t_i와 t_j에 대해, $I(t_i) \cap I(t_j) = \phi$, $O(I(t_i)) = \{t_i\}$ 및 $O(I(t_j)) = \{t_j\}$이면 t_i와 t_j는 '병행 점화가능'이라 하자.

(1) 퍼지 추론(fuzzy inference) : 일반 및 최대트랜지션의 점화 과정을 의미한다. 최대 트랜지션의 점화후 퍼지 마킹 μ의 변화는 다음과 같으며, 병행 점화가능인 트랜지션들은 동시에 점화한다. t_i가 점화가능이고, $I(t_i)$와 $O(t_i)$내의 모든 플레이스 ip_j와 op_k에 대하여,

- t_i가 일반트랜지션인 경우(AND의 의미를 갖음):
  $op_k^* = \mathrm{Min}\{ip_j^*\}\}$
  $\forall ip_j \in I(t_i)$
- t_i가 최대트랜지션인 경우(OR의 의미를 갖음):
  $op_k^* = \mathrm{Max}\{\{ip_j^*\}\}$
  $\forall ip_j \in I(t_i)$
- 점화후 마킹의 변화는 다음과 같다 : $ip_j^* = 0$, $\#ip_j = 0$, $\#ip_k = 1$

이 규칙들은 맘다니의 min-max 연산 규칙[2, 28]을 적용한 것이며, 기존의 퍼지 넷들[6, 9, 14, 16, 20]도 이 규칙을 주로 이용한다.

(2) 비퍼지화(defuzzyfication) : 제어트랜지션 ct_i의 점화 과정을 의미하며, 구동기 act_i의 제어 시간(즉, 트랜지션의 점화 지연 시간)을 구하는 과정이다. 여기서는 개량된 '무게 중심법'을 이용하여 비퍼지화를 수행한다.

[단계 1] 점화 시간구간 δ_i 계산 및 점화 : $I(ct_i)$와 $O(ct_i)$내의 모든 플레이스 ip_i 및 op_j에 대하여,

$\delta_i = [LB_i, UB_i] = \eta(ct_i) = AF_i^T(ip_i^*)$, $ip_i^* = 0$, $op_j^* = \delta_i$, $\#ip_i = 0$, $\#op_j = 1$.

(표 2)에 의하면, 모든 제어트랜지션의 입력 플레이스는 1개 이며(즉, ip_i는 ct_i의 유일한 입력 플레이스임), AF_i^T는 ct_i에 부여된 구동기 귀속 함수 AF_i의 '역관계'(inverse relation)이다. AF는 단사(one-to-one)함수가 아니므로, 역함수(inverse function)는 존재하지 않으므로, 역관계 개념을 고려하였다. (그림 4)의 오른쪽 부분은 δ_i의 개념을 보인다. ct_i에 삼각형 또는 가우스형 구동기 귀속 함수 AF_i가 부여되었을때,

- $ip_i^* =$ 일 경우 : ct_i의 제어 시간 범위(즉, 점화 지연 시간)는 $\delta_i = AF_i^T(1)$로 계산되며, 귀속 함수가 대칭이라면, $\delta_i = [(a_i + b_i)/2, (a_i + b_i)/2]$이며 상수형이다. 이러한 제어트랜지션들을 갖는 TFPN은 '상수형' 시간 넷[24]이 된다.
- $ip_i^* \geq \alpha_i \rangle 0$일 경우 : ct_i의 제어 시간 범위(즉, 점화 지연 시간)는 $\delta_i = AF_i^T(ip_i^*)$로 계산되며, 구간형태가 된다. 즉, $ip_i^* = \alpha_i$이면 $[a_i, b_i]$가 되고 일반적으로 $[LB_i, UB_i]$형태가 된다. 따라서, 이러한 제

어트랜지션들을 갖는 TFPN은 '구간형' 시간 넷[25]이 된다. 그러므로, TFPN은 TPN의 일반형이라 할 수 있다.

예를 들어, (그림 2)에서 각 규칙의 제어트랜지션들의 제어 시간 구간 δ는 다음과 같이 계산된다.

(a) $\delta = \eta(ct) = AF^T(\mu(cp) \times \lambda)$

(b) $\delta_1 = \eta(ct_1) = AF_1^T(Min(\mu(sen_1), \mu(sen_2)) \times \lambda)$ AND
$\delta_2 = \eta(ct_2) = AF_2^T(Min(\mu(sen_1), \mu(sen_2)) \times \lambda)$

(c) $\delta_1 = \eta(ct_1) = AF_1^T(Max(\mu(sen_1), \mu(sen_2)) \times \lambda)$ OR
$\delta_2 = \eta(ct_2) = AF_2^T(Max(\mu(sen_1), \mu(sen_2)) \times \lambda)$

(d) $\delta_1 = \eta(ct_1) = AF_1^T(Max(1-\mu(sen_1), \mu_{SF2}(sen_2)) \times \lambda)$ AND
$\delta_2 = 1 - \eta(ct_2) = 1AF_2^T(Max(\mu(sen_1), \mu(sen_2)) \times \lambda)$

병행 점화:2개 이상의 제어트랜지션이 병행 점화가 능하면 '병행 점화(concurrent firing)'하고, 점화 직후 점화된 트랜지션의 출력 플레이스 이외의 토큰들을 삭제한다(이를 '토큰 리셀'이라 한다). 제어트랜지션 집합의 act_i별 분할 π_{acti}내의 제어트랜지션들이 동시에 점화되면, 점화와 동시에 '무게 중심법'에 의한 '비퍼지 과정'을 적용하여, act_i에 대한 합성 제어 시간 범위 Δ_i 및 상수값 τ_i를 구한다. π_{acti}중에서 t시점에서 동시에 점화되는 제어 트랜지션의 집합을 CFact_i라 하고(즉, CF$act_i \subseteq \pi act_i$), CT$act_i$내의 모든 ct_i에 대해, ct_i의 귀속함수 AF_i의 범위를 $[a_i, b_i]$라 하고 δ_i의 범위를 $[LB_i, UB_i]$라 하자. 즉, $\alpha_i = AF_i(a_i) = AF_i(b_i)$, $ip_i^* = AF_i(LB_i) = AF_i(UB_i)$.

[단계 2] δ_i의 가중치(면적) ω_i 및 무게 중심 m_i계산: ip_i^*에 의해 상부가 절단된 귀속 함수 AF_i는 밑변 $= (b_i, a_i)$, 윗변 $= \delta_i = UB_i - LB_i$ 및 높이 $= ip_i^*$인 가우스형 또는 삼각형($ip_i^* = 1$일 때) 또는 사다리꼴 또는 종형($ip_i^* < 1$일 때) 도형에 대해, 면적과 무게 중심은 다음과 같이 계산한다.

- 면적 : $\omega_i = [(b_i - a_i) + (UB_i - LB_i)] \times (ip_i^*)/2$
  (ip_i^*는 ct_i의 유일한 입력 플레이스임) (1)
- 무게 중심 : $m_i = (a_i + top_i + b_i)/3$ (2)

여기서, top_i은 $\delta_i = [LB_i, UB_i]$의 중앙 지점이다. 즉, $top_i = (UB_i + LB_i)/2$

면적의 경우, (식 1)은 삼각형 또는 사다리꼴의 면적과 같으며, 가우스형 또는 종형 도형의 면적의 추정치로 활용한다. 가우스 또는 종형 도형의 경우 정확한 면적을 구하려면 많은 시간이 소요되고 귀속 함수 자체도 추정된 것이므로, (식 1)을 사용해도 무리가 없다고 판단된다. 무게 중심의 경우, 밑변이 $[a, b]$이고 top지점에서 높이가 1인 삼각형의 좌측 직각 삼각형의 넓이와 무게 중심을 각각 LS 및 LM이라 하고, 우측 직각 삼각형의 넓이와 무게 중심을 각각 RS 및 RM이라 하면,

$$LS = (top - a)/2,\ RS = (b - top)/2,\ LM = (2 \times top + a)/3,\ RM = (2 \times top + b)/3 \quad (3)$$

따라서, 두 삼각형간의 무게 중심은 $m = (LS \times LM + RS \times RM)/(LS + \mathrm{RS}) = (a + top + b)/3$이 된다. (식 2)에서, 우측 직각 삼각형인 경우 $top = b$이며, 좌측 직각 삼각형의 경우 $top = a$가 된다. 사다리꼴의 경우, (식 2)는 윗변의 중간 위치를 최대 높이로하는 삼각형으로 간주하여 무게 중심을 구한 것이며, 실제의 무게 중심과는 오차가 발생한다. 사다리꼴의 정확한 무게 중심은 (식 4)로 계산할 수 있다(전개 과정 생략).

$$m' = (-LB^2 - a \times LB - a^2 + UB^2 + b \times UB + b^2)/[3 \times (UB - LB - a + b)] \quad (4)$$

여기서, LB와 UB는 각각 사다리꼴의 윗변의 하한 및 상한 값이고, a와 b는 아랫변의 하한 및 상한 값이다. (식 2)는 덧셈 2회, 나눗셈 1회가 소요되지만, (식 4)는 덧셈 4회, 뺄셈 5회, 곱셈 7회 및 나눗셈 1회가 소요된다. (식 2)와 (식 4)간의 오차를 비교하기 위해, 가장 오차가 큰 도형인 우측(또는 좌측) 직각 사다리꼴의 경우, m과 m'간의 오차는 (식 5)와 같이 (식 2)에 $top = (LB + UB)/2$ 및 $UB = b$을 대입한값과 (식 4)에 $b = UB$를 대입한 값과의 차이와 같다.

$$\text{최대오차} = |m - m'| = |(-LB \times a - LB^2 - a \times b + LB \times b)/(12 \times b - 6 \times a - 6 \times LB)| \quad (5)$$

예를 들어, a, b LB, UB가 각각 0, 3, 1, 3인 우측 직각 사다리꼴의 경우, (식 2)에 의한 추정 값은 1.

733333이며, (식 4)에 의한 실제 값은 1.666667이며 오차는 0.0666667이 된다. (식 3)은 밑변 1, 높이 1인 우측 직각 삼각형의 무게중심은 밑변의 2/3 지점이라는 사실로부터 유도된 것이며, 다음 식에 의해 유도된다 [2].

$$\int x f(x)dx / \int f(x)dx = \int_0^1 x^2\,dx / \int_0^1 x\,dx = 2/3$$

[단계 3] 합성 제어 시간 상수값(무게 중심) τ의 계산: 구동기 act_i는 최종적으로 상수 형태의 지연 시간값에 의해 작동되므로, π_{acti}중에서 t시점에서 동시에 점화되는 제어 트랜지션 집합 CFact_i내의 ct_k들의 δ_k, m_k 및 ω_k로부터 (즉, 식 1과 2) 다음과 같이 τ_i를 구하여 구동기로 보낸다.

$$\tau_i = \sum_k (m_k \times \omega_k) / \sum_k (\omega_k) \qquad (7)$$

예를 들어, "R1: IF ($cp1 = SF1$) ($\alpha 1$) THEN ($ct1 = AF1$) ($\lambda 1$)."와 "R2: IF ($cp2 = SF2$) ($\alpha 2$) THEN ($ct2 = AF2$) ($\lambda 2$)."를 하나의 센서 sen과 구동기 act에 대한 제어 규칙들(여기서, $cp1$과 $cp2$는 sen을 제어하는 조건 플레이스이며, $ct1$과 $ct2$는 act를 제어하는 제어트랜지션이다.)이라 하고, $\mu(cp1) \geq \alpha 1$, $\mu(cp2) \geq \alpha 2$라 하자. 구동기 귀속 함수들은 (그림 5)와 같이, $cp1^* = 0.3$, $cp2^* = 0.5$, $AF1 = [a1, b1] = [2.5, 9]$, $AF2 = [a2, b2] = [5, 10]$, $\delta 1 = [LB1, UB1] = [3, 8]$, $\delta 2 = [LB2, UB2] = [7, 10]$, $top1 = 5.5$, $top2 = 9.5$라 가정하면,

$m1 = (a1 + top1 + b1)/3 = 5.667$, $m2 = (a2 + op2 + b2)/3 = (5 + 8.5 + 10)/3 = 7.833$,

$\omega 1 = [(b1 - a1) + (UB1 - LB1] \times (ip1^*)/2 = 1.275$,

$\omega 2 = [(b2 - a2) + (UB2 - LB2] \times (ip2^*)/2 = 2.125$

$\tau = (m1\omega 1 + m2\omega 2)/(\omega 1 + \omega 2) = 7.021$

따라서, 구동기로 보내지는 합성 제어 시간 상수값은 7.021 (초)가 된다.

4.3 도달성 그래프

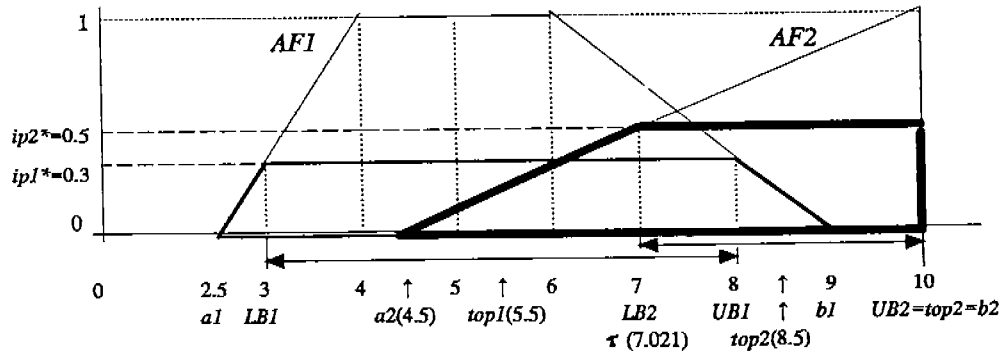
(1) 정의 및 속성: TFPN에 초기 토큰 마킹을 부여하고 각 센서값이 해당 조건 플레이스로 입력되면, TFPN은 수행이 시작되며, 수행 결과 '도달성 그래프' (또는, 트리)를 형성한다. 넷은 비해석적(uninterpreted) 모형이므로[1], 수행 결과의 의미는 다양하게 해석된다. TFPN의 도달성 그래프는 '토큰 마킹의 도달성 그래프(TMRG)'와 '퍼지 마킹의 도달성 그래프(FMRG)'로 구분되며, TM과 FM을 각각 토큰 마킹넷과 퍼지 마킹넷이라 할 때 다음과 같이 정의된다.

[정의 4] 도달성 그래프: (NODE, ARC, lf_N, lf_A)

- NODE: 노드 집합, ARC: 호선 집합
- TMRG에서: lf_N: NODE→TM: 노드에 부여된 토큰 마킹

  $TM_k = \{p_i \mid \#p_i = \chi(p_i) = 1,\ p_i \in P\}$,

  (여기서, p^*는 TMRG의 구조적 특성에 영향을 주지 않으며 참고 자료일 뿐이다.)



(그림 5) 합성 제어 시간 상수값(무게 중심) τ의 계산 예
(Fig. 5) An example of calculation of a composit control time constant τ

- FMRG에서 : lf_NNODE→FM : 노드에 부여된 퍼지 마킹
  $FM_k = \{\langle p_i, p_i^* \rangle p_i^* = \mu(p_i) \rangle 0, p_i \in P\}$
- 일반 및 최대트랜지션일 경우 : $lf_A \rightarrow T$: 호선에 점화(또는, 병행 점화)된 트랜지션(들)을 부여함
- 제어트랜지션일 경우 : lf_A : ARC→CT×τ : 호선에 점화(또는, 병행 점화)된 트랜지션(들)과 합성 퍼지 제어 상수 값을 부여함

[속성 2] 유한 TFPN에 대한 TMRG는 유한하며 FMRG는 무한하다.

〈증명〉 각 마킹넷 M_i에는 최대 |P| 개(유한 수)의 요소가 존재(모든 플레이스에 토큰 마킹시)하며, 플레이스에는 1개 또는 0개의 토큰(즉, 토큰 마킹)이 존재하므로, 서로 다른 마킹넷의 총 수는 $2^{|P|}$개로서 유한하다. NODE수는 마킹 크라스내의 마킹넷 수와 같으며(즉, |NODE| = |M|), M은 유한개의 마킹넷으로 구성된 집합의 집합이므로, 노드의 수는 유한하다. 따라서, 유한 TFPN에 대한 도달성 그래프는 유한하다. 그러나, TMRG와는 달리, FMRG는 각 플레이스에 0과 1사이의 실수값(무한개)이 존재하므로, 기존의 P/T net[1]과 같이 무한개의 노드를 갖는다. 따라서, FMRG는 무한하다.

임의의 토큰 마킹 TM_i에서 트랜지션들이 점화되어 새로운 토큰 마킹 TM_j이 될 수 있다면, 즉, $TM_i \xrightarrow{t_i} TM_{i+1} \xrightarrow{t_{i+1}} \cdots \xrightarrow{t_j} TM_j$, TM_j는 TM_i로부터 트랜지션 점화 시퀀스 "$t_i\ t_{i+1}..t_j$"에 의해 '토큰 도달가능(token reachable)'하다고 한다. 수행중 토큰 수가 일정하게 유지되면 '보존적(conservative)'이며, 각 플레이스에 토큰 수가 1개 이하일 때 '안전적(safeness)' 이라 한다. 또한, 토큰 수가 K개 이하일 경우를 'K-유한성(K-bound)'이라 한다. '데드록'은 TFPN에서 점화 불가능한 트랜지션이 존재할 경우를 나타내며, 데드록이 없을 때, TFPN은 '생존적(live)' 이라 한다. '개방형(open) 넷'이란, 넷이 모형화한 부분이외의 영역으로부터 영향을 받는 넷을 의미한다. 예를 들어, 한 객체를 넷으로 모형화 했다면, 객체는 적어도 한번은 외부로부터의 메시지 또는 인터럽트에 의해 상태가 변화되므로, 이 경우 넷은 개방형이 된다. 개방형이 아닌 넷은 '폐쇄형(closed)'이다.

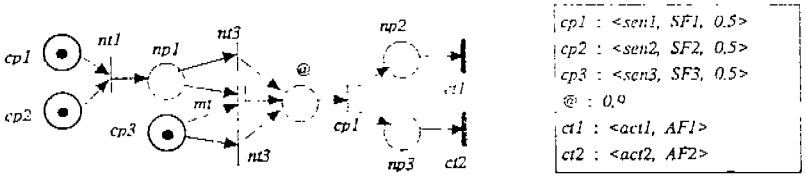
[속성 2] 토큰 마킹은 '안전적'이지만 '비 보존적'이다. 토큰 도달성은 센서의 입력 값에 의해서 결정되며, TFPN은 '개방형' 넷 모형이다.

[정의 2]로부터 TFPN의 각 플레이스에는 1개 또는 0개의 토큰만 존재하므로, 토큰 마킹은 기존의 C/E net에서의 마킹과 동일하며 안전적이다. TFPN은 (표 2)와 (그림 3)에 따라 모형화되므로, 수행중 토큰은 감소하거나 증가된다. 따라서, 비 보존적이다. 또한, 조건플레이스의 토큰 마킹은 조건플레이스의 퍼지 마킹(즉, 센서값, 센서 귀속 함수 및 임계치로부터 계산된 퍼지값)에 의해 결정되며, 센서값은 TFPN의 외부로부터 주어지므로, TFPN은 개방형 넷이 된다.

(2) 도달성 그래프의 예 : (그림 6-(a))에서, 초기 퍼지 마킹이 $\{cp_1^* = \mu(sen_1) = 0.9,\ cp_2^* = \mu(sen_2) = 0.8,\ cp_3^* = \mu(sen_3) = 0.7\}$ 일 때, (즉, 토큰 마킹은 $\{cp_1, cp_2, cp_3\}$) nt_1과 nt_3이 점화가능하며, nt_1의 점화후 $np_1^* = Min\{0.9, 0.8\} = 0.8$이고, 새로운 마킹은 $\{np_1(0.8),\ cp_3(0.7)\}$ (괄호 내의 값은 퍼지값임)이 된다. 이때, mt, nt_2 및 nt_3이 점화가능하며, mt의 점화후 $@^* = Min\{0.8, 0.7\} \times \lambda = 0.7 * 0.9 = 0.63$이며 새로운 마킹은 $\{@(0.63)\}$이 된다. nt_4가 점화되면 새로운 마킹은 $\{np_1(0.63),\ np_3(0.63)\}$가 되고, ct_1과 ct_2가 동시에 점화되면 하나의 제어 루프가 종료된다. 이때, 각 구동기에 전해지는 제어 시간 구간 값은 각각 $\delta_1 = \eta(ct_1) = AF_1^T(0.63)$이 된다. (그림 6-(b))는 토큰 마킹 도달성 그래프를 보이며 정의된 점화 규칙(병행 점화 및 토큰 리셋)을 사용하므로서 마킹 수가 감소되었다(26개에서 11개로 감소). 초기 마킹에서 제어트랜지션이 점화될 때 까지의 모든 스퀀스들을 (그림 6-(c))에서 보였다.

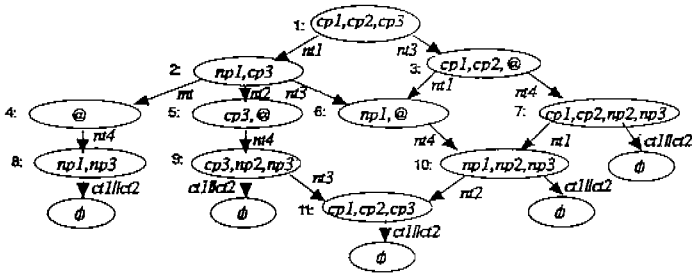
4.4 범용 퍼지 제어 시스템의 구현

(그림 7)은 본 연구에서 제시한 TFPN을 이용한 범용 퍼지 제어 시스템의 구조를 보인다. 제어기(수행기) 부분은 부록 B의 알고리즘으로 처리된다. 본 시스템은 MS-Windows 환경에서 Visual Basic으로 GUI를 완성하였으며, 수행기 등의 초기 버전을 Turbo C로 개발하였고 계속 기능을 확장중이다.



IF (*cp1=SF1* (0.5) AND *cp2=SF2* (0.5) OR *cp3=SF3* (0.5)) THEN (*ct1=AF1*) AND (*ct2=AF2*) (0.9)

(a) 제어 규칙의 TFCL과 TFPN의 예 (귀속 함수들의 데이터베이스는 생략함)

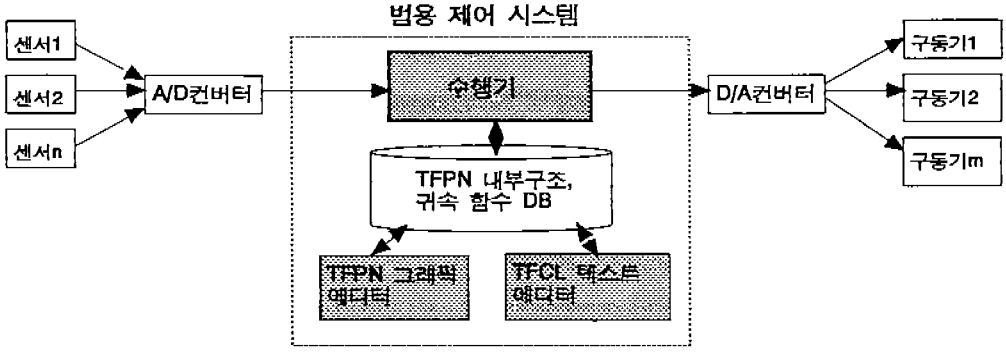


(b) 토큰 마킹 도달성 그래프 (ct_i // ct_j는 '병행 점화'를 의미한다)

(1): 1→2→4→8→ϕ	(0.63,0.63)	(2): 1→2→5→9→ϕ	(0.72,0.72)
(3): 1→2→5→9→11→ϕ	(0.72,0.72)	(4): 1→2→6→10→11→ϕ	(0.63,0.63)
(5): 1→2→6→10→ϕ	(0.63,0.63)	(6): 1→3→6→10→11→ϕ	(0.63,0.63)
(7): 1→3→6→10→ϕ	(0.63,0.63)	(8): 1→3→7→10→11→ϕ	(0.63,0.63)
(9): 1→3→7→10→ϕ	(0.63,0.63)	(10): 1→3→7→ϕ	(0.63,0.63)

(c) 추론 시퀀스 및 ($np2^*$, $np3^*$) 값

(그림 6) 도달성 그래프와 추론 시퀀스의 예
(Fig. 6) An example of reachability graph inference sequence



(그림 7) TFPN을 이용한 범용 퍼지 제어 시스템의 구조
(Fig. 7) The organization of general purpose fuzzy control system using TFPN model

```
BEGIN TFCL
MEMBERSHIP
   Sensor A
     A_Null: ((0,1), (0.5,0.75), (1.0,0.5), (1.5,0.25), (2.0,0)).
     A_Zero: ((0,0), (0.5,0.25), (1,0.5), (1.5,0.75), (2,1), (2.5,0.75), (3,0.5), (3.5,0.25), (4,0)).
     A_Small:  ((2,0), (2.5,0.25), (3,0.5), (3.5,0.75), (4,1), (4.5,0.75), (5,0.5), (5.5,0.25), (6,0)).
     A_Medium:  ((4,0), (4.5,0.25), (5,0.5), (5.5,0.75), (6,1), (6.5,0.75), (7,0.5), (7.5,0.25), (8,0)).
     A_Large:   ((6,0), (6.5,0.25), (7,0.5), (7.5,0.75), (8,1), (8.5,0.75), (9,0.5), (9.5,0.25), (10,0)).
     A_Very_Large: ((8,0), (8.5,0.25), (9,0.5), (9.5,0.75), (10,1)).
   Actuator N
     N_Zero: ((400,1), (500,0.75), (600,0.5), (700,0.25), (800,0)).
     N_Small: ((400,0), (500,0.25), (600,0.5), (700,0.75), (800,1), (900,0.75), (1000,0.5), (1100,0.25), (1200,0)).
     N_Medium: ((800,0), (900,0.25), (1000,0.5), (1100,0.75), (1200,1), (1300,0.75), (1400,0.5), (1500,0.25), (1600,0)).
     N_Large: ((1200,0)(1300,0.25), (1400,0.5), (1500,0.75), (1600,1), (1700,0.75), (1800,0.5), (1900,0.25), (2000,0)).
     N_Very_Large: ((1600,0), (1700,0.25), (1800,0.5), (1900,0.75), (2000,1)].
END
RULE-SET
  R1: IF (A=A_Null) THEN (N=N_Very_Large).      R2: IF (A=A_Zero) THEN (N=N_Large).
  R3: IF (A=A_Small) THEN (N=N_Medium).         R4: IF (A=A_Medium) THEN (N=N_Small).
  R5: IF (A=A_Large) THEN (N=N_Zero).           R6: IF (A=A_Very_large) THEN (N=N_Zero).
END
RULE-RELATION
  Rel1: R1 FORK R2 FORK R3 FORK R4 FORK R5 FORK R6.
  Rel2: R1 MI R2 MI R3 MI R4 MI R5 MI R6.
  Rel3: R1 ITR R1. Rel4: R2 ITR R2. Rel5: R3 ITR R3. Rel6: R4 ITR R4. Rel7: R5 ITR R5. Rel8: R6 ITR R6.
END
END TFCL
```

(그림 8) 직렬 모터 제어의 TFCL
(Fig. 8) TFCL of a motor controller

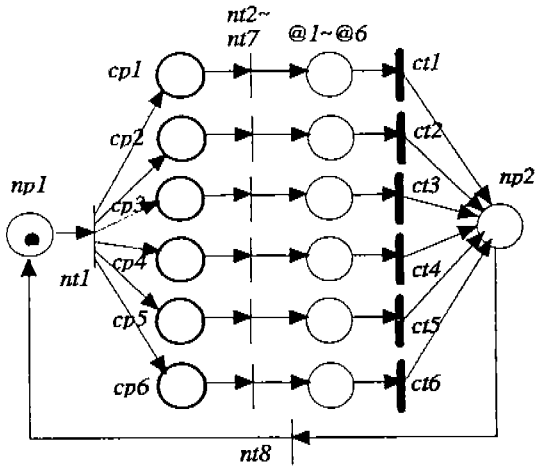
5. 사례 연구: 직렬 모터 제어 시스템

직렬 모터 제어 시스템[31]은 전류 센서 A로부터 구동 전류를 측정하고 퍼지 제어 규칙을 이용하여 모터(구동기)의 회전수 N을 제어하는 6개의 규칙을 가진 간단한 시스템이다. 문헌 [31]의 방법에서는 6개의 퍼지 함축을 만들고(각 함축당 |N| × |I| 테이블 필요, |N|은 N영역내의 지점수), 퍼지 관계 합(maximum)을 취하여 최종적으로 1개의 퍼지 함축을 만든다. 최종적으로, 비퍼지 과정(무게 중심법)을 통해 최종 값을 얻는다. 따라서, 많은 메모리와 계산 량이 요구된다. 시스템의 센서, 구동기 및 귀속 함수 집합들은 다음과 같다.

- 센서 집합(분할): $SEN = \pi A = \{A\}$
- 구동기 집합(분할): $ACT = \pi N = \{N\}$
- 전류 센서 귀속 함수 집합(분할): SF = A = {A_Null, A_Zero, A_Small, A_Medium, A_Large, A_Very_Large}
- 모터(구동기) 귀속 함수 집합(분할): $AF = \prod N =$ {N_Zero, N_Small, N_Medium, N_Large, N_Very_Large}

(1) TFCL 모형: (그림 8)은 직렬 모터 제어 시스템에 대한 TFCL을 보인다. 각 조건들의 임계치는 0이며 제어 규칙들의 확신도는 1이라 가정한다.

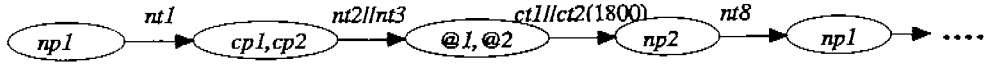
(2) TFPN모형: TFCL로부터 3장에서 제시한 방법을 적용하여 (그림 9)와 같은 TFPN모형을 얻을 수 있다. 각 규칙들의 조건부와 제어부가 단일 형태이므로, TFPN구조도 매우 간단하다. 여기서, 모터의 RPM값



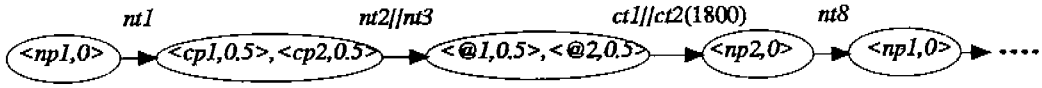
CP={*cp1*:<A,A_Null,1>, *cp2*:<A,A_Zero,1>,
cp3:<A,A_Small,1>, *cp4*:<A,A_Medium,1>,
cp5:<A,A_Large,1>, *cp6*:<A,A_Very_Large,1>}
CT={*ct1*:<N,N_Very_Large>, *ct2*: <N,N_Large>,
ct3:<N,N_Medium>, *ct4*:<N,N_Small>,
ct5:<N,N_Zero>, *ct6*:<N,N_Zero>}

(a) TFPN모형
(a) A TFPN model

(b) CP와 CT의 레이블(각 귀속 함수들의 DB는 생략함)
(b) Labels of CP and CT



(c) 토큰 마킹 도달성 그래프 (I=1경우)
(c) A token marking reachability graph when I=1



(d) 퍼지 마킹 도달성 그래프 (I=1경우)
(d) A fuzzy marking reachability graph when I=1

(그림 9) 직렬 모터 제어의 TFPN모형
(Fig. 9) TFPN model of a motor controller

(즉, N값)은 모터를 모형화한 제어트랜지션의 지연시간 구간 δ로 간주하고 있다. 전류 센서값에따라 조건플레이스들이 마킹되므로, 전체적인 도달성 그래프를 그리기는 불가능하며, (그림 9-(c))와 (d)는 I=1일 경우의 도달성 그래프들을 보인다. (그림 10)과 부록 C는 I값의 변화에 따른 합성 제어시간 상수 값들을 문헌 [31]의 결과와 비교하여 보인다. I가 7.5 이후 문헌 [31]의 결과에 비해 매우 빠르고 간단한 방법으로 좀더 안정된 결과가 나타남을 알 수 있다. 그러나, 본 사례 연구는 TFPN의 특성들중 병행 마킹, 병행 점화 및 합성 제어시간 상수 값의 활용 예를 보였지만, 제어 규칙이 단순하므로 TFPN의 모든 개념들을 이용하지는 않고 있다.

6. 평가 및 결론

6.1 기존 퍼지 넷과 비교

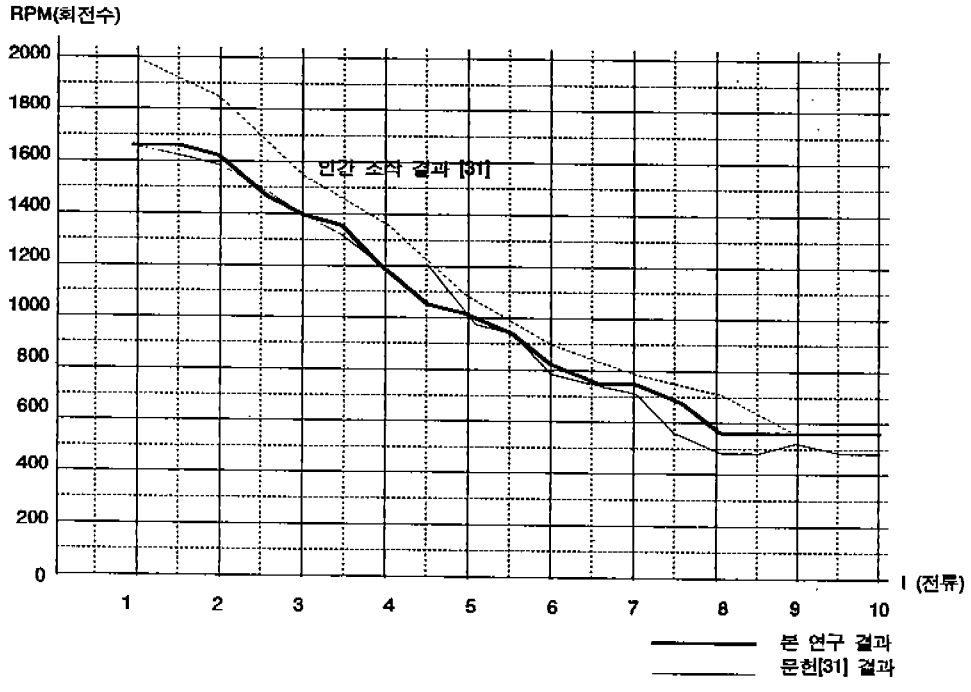
(1)플레이스 및 토큰:퍼지 생성 규칙을 모형화할 때, 플레이스와 토큰은 퍼지 명제[5, 6, 8, 9, 13, 14, 15, 16, 19, 20] 또는 술어[7, 10, 21]를 나타내지만, 제

어 시스템을 모형화할때는 제어 대상 객체[12, 18, 13]나 지역 퍼지 변수[12, 18]를 나타낸다. 신경망 회로를 모형화한 경우[11], 감각 세포 및 시냅스를 나타낸다. 문헌 [9, 8, 15, 20, 22]을 제외한 퍼지 넷들에서는 플레이스에 귀속 함수(또는, 언어 변수)를 직접 레이블화하지 않고, 0과 1사이의 진리 스케일(또는, 진리값)만을 고려하고 있다. 토큰은 대부분 퍼지 명제(또는, 술어, 객체 및 시냅스 등)의 퍼지값(또는, 만족도[15], 확신도[13], 일치도[8] 등)로 해석되며, 임계치 이상의 값만을 토큰으로 간주한다. 표준 넷에서는 토큰의 수만을 고려하지만, 퍼지 넷에서는 토큰의 퍼지값을 고려한다. 퍼지 넷에서 토큰은 추상적[7], 상태[12, 18] 및 제어/상태[10]로 해석되기도 한다.

TFPN에서 플레이스와 토큰은, 제어의 조건 명제, 규칙간의 관계 설정용 또는 단위 제어규칙 내의조건부의 최종 조건으로 해석한다. 플레이스내에 토큰은 1개 이하만 존재한다. 토큰의 존재 유무만 고려한다면, '콘트롤 토큰'[10]과 유사하다. 조건의 퍼지값은 토큰의 속성으로 나타내며, 0과 1사이의 실수 값을 갖는다. 문헌 [22]에서는 센서와 구동기를 5종류의 플레이스 객체로 모형화하고 있지만, TFPN에서는 센서는 플레이스로, 구동기는 트랜지션으로 각각 모형화하므로서, 좀더 자연스러운 모형화가 가능하다.

(2)트랜지션:트랜지션은 기존의 퍼지넷에서 퍼지 명제간의 함축을 의미하지만, TFPN에서는 '원인-결과'(즉, 제어 기능)을 나타낸다. 예를 들어, 생성규칙 또는 제어규칙 " if p then q"의 경우, 기존의 연구에서 p와 q는 각각 전제 및 결론 명제(또는, 술어)를 나타내고, p ○→ | → ○ q로 모형화 하지만, TFPN에서는 각각 조건(원인)과 제어(결과)를 나타내며, p ○→ | q로 모형화한다. 기존 방법의 경우, p와 q가 모두 플레이스로 모형화되므로, 센서(p)와 구동기(q)에 관련된 명제가 포함된 제어 규칙을 모형화때는 가시성이 저하된다. 표준 넷에서도 플레이스와 트랜지션은 각각 조건과 사건(또는, 기능)을 나타내므로, TFPN의 접근 방식이 좀더 자연스럽다.

기존의 퍼지 넷에서는 퍼지 생성 규칙 자체의 확신도(또는, 가중치[12, 18], 진리값[7], 퍼지값[8], fuzzy



date[13] 및 임계치[16]) 또는 진리치 함수[15]를 트랜지션에 부여했다. 그러나, TFPN에서는 트랜지션뿐 아니라 플레이스에도 퍼지 개념을 부여하고 있으며, 제어 규칙 자체의 확신도는 중간플레이스에 부여한다. 논리 연산자가 포함된 퍼지 생성 규칙을 모형화하기 위해, 문헌 [22, 27]에서는 많은 종류의 트랜지션들을 사용하므로, 넷의 구조는 간단해지지만, 가시성과 분석성이 저하된다. TFPN에서는 표준 넷의 개념 및 표기법의 확장을 극소화하였고 OR논리를 모형화하기위해 '최대트랜지션'을 추가 정의하였다.

(3) 호(arc) : TFPN에서는 부정의 논리를 나타내기 위해 '금지 호'를 사용한다. 기존 퍼지 넷들 중에서는 [7]과 [16]만 부정 호와 금지 호를 각각 허용하고 있을 뿐이다. 문헌 [10, 22]에서는 호에도 임계치 또는 확신도를 부여하고 있으며, 술어 넷(Pr./T net)에 퍼지 개념을 추가한 연구[21]에서는 술어의 변수를 호에 레이블로 부여하고 있다. 호에 이와 같은 레이블이 부여되면 가시성이 저하되므로, TFPN에서는 호에 레이블을 부여하지 않는다.

(4) 점화 규칙 : 기존 퍼지 넷에서 트랜지션의 점화가능 조건은 입력플레이스의 퍼지 마킹값(또는, 자극값[8]))이 주어진 임계치 이상이거나, 자극 값이 0 이상[8] 또는 0.5 이상[19]일 때 점화된다. 임계치 개념이나 점화 개념을 적용하지 않는 경우도 있다[7, 15]. 점화가능된 트랜지션의 점화후 입력플레이스내의 토큰은 삭제되거나[10, 12, 15, 21] 잔류한다[5, 9, 11, 13, 19]. 특히, 문헌 [6]의 경우 점화후 입력 토큰이 삭제되는 것으로 정의하고 있지만, 그의 논문내의 예 에서는 토큰의 잔류를 가정하는 등, 점화후 입력 토큰의 처리 문제를 불확실하게 다루고 있다. 따라서, 넷의 모형화 및 해석에 혼란을 야기한다. 본 TFPN에서는 표준 넷과 같은 점화규칙(즉, 점화후 입력토큰은 삭제)을 사용하고 있다.

퍼지 지식 표현 및 추론을 위한 퍼지 넷들의 경우 [5~10, 14, 15, 19, 21], 트랜지션의 점화와 점화 시퀀스는 생성 규칙의 적용(즉, 추론) 및 추론 시퀀스로 간주하지만, TFPN과 [20]에서는 한 제어 규칙의 적용(즉, 제어 행위) 및 제어 시퀀스를 각각 나타낸다. 문헌 [20]에서는 제어 시퀀스를 이용하여 PLC프로그램을 생성하는 방법을 택하고 있지만, TFPN에서는 제어 시퀀스와 비퍼지 결과를 이용하여 구동기를 직접 제어하고 있다. 또한, TFPN에서는 센서별 조건플레이스 분할 및 구동기별 제어트랜지션 분할내의 플레이스나 트랜지션의 병행 마킹, 병행 점화 및 토큰 넷 개념을 이용하므로서, 넷의 장점을 살리고 있다. 이는 '병렬 퍼지 추론' 방법과 유사하다.

(5) 시간 개념 : 문헌 [20]에서는 시간 정보를 컬러드 토큰에 부여하여 참고 자료로 사용하고 있지만, 점화 규칙에는 시간 개념을 부여하지 않았으므로, 시간 넷은 아니다. 문헌 [22]에서도 각 플레이스와 트랜지션 객체들에 Age, 시간 구간 및 지연 시간을 객체의 속성으로서 부여하고 있지만, 시간 넷은 아니다. TFPN에서 제어트랜지션은 상수형(즉, 제어트랜지션의 입력 퍼지값이 1일때) 또는 구간형 퍼지 지연 시간을 가지므로, 일반화된 (퍼지) 시간 넷이라 할 수 있다. TFPN모형에서 합성 제어 시간 상수 값은 구동기의 수행 지연 시간을 퍼지 제어하는 경우, 구동기의 제어 시간으로 간주하고 있지만, 다른 제어 수치(예를 들어, 회전수, 압력 등)로 간주할 수 도 있다.

(6) 퍼지화 및 비퍼지화 : 기존의 퍼지 넷에서는 플레이스에 귀속 함수(또는, 언어 변수)를 부여하고 퍼지 추론 과정을 제시하고 있지만, 귀속 함수로부터 퍼지 값을 얻는 방법과 비퍼지화 과정에 대한 방법은 구체적으로 제시하지 않고 있다[20, 22]. 문헌 [22]의 경우 퍼지화 및 비퍼지화 트랜지션을 별도로 사용하여 모형화하지만, TFPN처럼 퍼지화 및 비퍼지화 개념을 넷의 특성과 연결시키지는 못했다. 특히, 퍼지 제어에서 비퍼지화 과정은 추론 과정 못지 않게 중요하며 많은 시간이 소요된다[2]. 본 연구에서 제시된 개량된 무게 중심법에의한 비퍼지화 방법의 사례 연구 결과, 기존의 방법[31]보다 적은 계산 량으로 인간 조작에 좀더 가까운 결과를 얻었다.

6.2 결 론

본 논문에서는 기존의 퍼지 넷들의 문제점을 보완하여 퍼지 제어에 이용할 수 있는 TFCL과 TFPN모형을 정의하고, TFPN의 모형화 방법, 특성 및 수행 방법을 제시하였다. TFPN모형은 조건-사건(원인과 결과)형 퍼지 제어 규칙을 갖는 제어 시스템의 명세, 설계 및 구현에 적합한 모형이며, 퍼지 넷 및 시간 넷의 통합 모형이므로, 이들 넷의 장점을 가지고 있다. 즉, 제어 규칙의 가시성, 병행성 및 비결정성을 공식

적으로(formal) 모형화 및 분석할 수 있고 제어에 적용 할 수 있다. 특히, 제어 규칙이 갖는 불확실성 문제는 퍼지 넬 개념을 통해 해결한다. TFPN모형은 퍼지 제어 뿐아니라, 퍼지 추론에도 적용 할 수 있다. 몇가지 TFPN모형의 장점들은 사례 연구로부터 확인되었다.

본 연구에서는 간단하게 다루었지만, 넬과 비슷한 시기에 발표되었으며 가장 활발히 이용되고 있는 모형인 퍼지 이론과 넬간의 비교 연구, 불확실성의 모형화 방법의 관점에서 확률 넬과 퍼지 넬간의 관계의 연구 및 기존 퍼지 넬들의 비교 연구는 향후 계속되어야 한다. 또한, 제시된 모형 및 방법론을 지원하는 범용 퍼지 제어기의 확장하고 및 실제의 여러 문제에 적용하는 연구를 향후 연구 과제로 남기고 있다.

참 고 문 헌

[1] J. L. Peterson, "*Petri nets theory and the modeling of systems*," Prentice-Hall, 1981.

[2] 이광형, 오길록, "퍼지 이론 및 응용," 1권, 2권, 홍릉과학 출판사, 1991.

[3] 이강수, "패트리넬을 이용한 지식표현에 방법에 관한 연구," 대전산업대학논문집, pp. 271~292, 1986.

[4] H. P. Lipp, "The application of a Fuzzy Petri Net for controlling complex industrial process," *Proc. IFAC Conf. on Fuzzy Information Control*, pp. 459~469, 1983.

[5] C. Loony, "Fuzzy Petri nets for rule-based decisionmaking," *IEEE tran. on Systems, Man and Cybernetics (S.M.C.)*, Vol. 18, No. 1, pp. 178~183, Jan/Feb. 1988.

[6] S. M. Chen, J. S. Ke, and J. F. Chang, "Knowledge representation using Fuzzy Petri nets," *IEEE tran. Knowledge and Data Engineering*, Vol. 2, No. 3, pp. 311~319, Sep. 1990.

[7] M. Garg, S. Ahson, and P. Gubta, "A Fuzzy Petri net for knowledge representation and reasoning," *Information Processing Letters*, Vol. 39, pp. 165~171, Aug. 1991.

[8] E. Tazaki and K. Yoshida, "A Fuzzy Petri net model for approximate resoning and its application to medical diagnosis," *Proc. IEEE Inter. Conf. on S.M.C.*, pp. 627~631, 1992.

[9] 전명근, 변증남, "Fuzzy Petri Nets를 이용한 퍼지 추론 시스템의 모델링 및 추론기관의 구현," 전자공학회논문지, 제 29권 7호, pp. 508~519, 1992년 7월.

[10] N. K. Liu and T. Dillon, "Modeling uncertainty in expert systems," *PRICAI'92*, pp. 610~616, 1992.

[11] 김성렬, 고원국, 이상호, 이철희, "퍼지 페트리네트를 이용한 신경계 행위 표현," 정보과학회논문지, 제 20권 5호, pp. 644~655, 1993년 5월.

[12] T. Cao and A. Sanderson, "Variable reasoning and analysis about uncertainty with Fuzzy Petri nets," *Proc. of Application and Theory of Petri Nets, Lecture Notes in Computer Science*, Springer-Verlag, Vol. 691, pp. 126~145, June 1993.

[13] J. Cardoso, R. Valette, and B. P. Chezalviel, "Fuzzy Petri nets and linear logic," *Proc. IEEE Int. Conf. on S.M.C.*, pp. 258~263, 1993.

[14] 조상엽, 김기태, "퍼지 페트리네트를 이용한 퍼지 생성규칙의 표현," 정보과학회논문지, 제 21권 2호, pp. 298~306, 1994년 2월.

[15] A. J. Bugarin and S. Barrorm "Fuzzy reasoning supported by Petri nets," *IEEE tran. on Fuzzy Systems*, Vol. 2, No. 2, pp. 135~150, May 1994.

[16] W. Pedrycz and F. Momide, "A generalized Fuzzy Petri net model," *IEEE tran. on Fuzzy Systems*, Vol. 2, No. 4, pp. 295~301, Nov. 1994.

[17] S. K. Yu, "Comments on Knowledge representation using Fuzzy Petri nets," *IEEE tran. Knowledge and Data Engineering*, Vol. 7, No. 1, pp. 190~192, Feb. 1995.

[18] T. Cao and A. Sanderson, "Task sequence planning using Fuzzy Petri nets," *IEEE tran. on S.M.C.*, Vol. 25, No. 5, pp. 755~769, May 1995.

[19] S. Ahson, "Petri net models of fuzzy neural network," *IEEE tran. on S.M.C.*, Vol. 25, No. 6, pp. 926~932, June 1995.

[20] L. Gomes and A. S. Garcao, "Programmable controller design based on a synchronized colored Petri net model and implementing fuzzy reasoning," *Proc. of Application and Theory of Petri*

Nets, Lecture Notes in Computer Science. Springer-Verlag, Vol. 935, pp. 213~1237, June 1995.

[21] S. K. Yu, "Knowledge representation and reasoning using fuzzy Pr/T net-systems," *Fuzzy Sets and Systems*, Vol. 75, pp. 33~45, 1995.

[22] R. Tang, et al, "A continuous Fuzzy Petri net tool for intelligent process monitoring and control," *IEEE tran. on Control Systems Technology*, Vol. 3, No. 3, pp. 318~329, Sep. 1995.

[23] G. Florin, C. Fraize, S. Natkin, "Stochastic petri nets: properties, applications and tools," Microelectronics and Reliability, Vol. 31, No. 4, pp. 669-697, 1991.

[24] W. M. Zuberek, "Timed Petri nets: definitions, properties, and applications," *Microelectronics and Reliability*, Vol. 31, No. 4, pp. 627-644, 1991.

[25] B. Berthomieu, M. Daiz, "Modeling and verification of time dependent systems using timed Petri nets," *IEEE tran. on Software Engineering*, Vol. 17, No. 3, pp. 259-273, March 1991.

[26] M. Silva and R. Valette, "Petri nets and flexible manufacturing," *Advances in Petri Nets, Lecture Notes in Computer Science*, Vol. 424, pp. 374~417, 1990

[27] R. David, "Grafcet: A powerful tool for specification of logic controllers," *IEEE tran. on Control, Systems Technology*, Vol. 3, No. 3, pp. 253~268, Sep. 1995.

[28] Selected Papers, Special Issue on Fuzzy Logic with Engineering Application, *Proceedings of the IEEE*, Vol. 83, No. 3, pp. 343~466, March 1995.

[29] A. V. Yakovlev, et al, "Highlevel Modeling and Design of Asynchronous Interface Logic," *IEEE Design & Test of Computer*, pp. 32~40, Spring 1995.

[30] S. Hauck, "Asynchronous Design Methodologies: An Overview," *Proceeding of IEEE*, Vol. 83, No. 1, 69~93, Jan. 1995.

[31] 엄정국, 원성현, "기초 응용 퍼지이론과 시스템," 정보시대 출판부, pp. 280~285, 1992.

부록 A. 시간적 퍼지 제어 언어(timed fuzzy control language, TFCL)의 정의 :

<TFCL> ::= BEGIN TFCL
MEMBERSHIP
<귀속함수집합>
END
RULE-SET
<제어규칙집합>
END
RULE-RELATION
<관계집합>
END
END TFCR

<귀속함수집합> ::= Sensor [<센서명>
<귀속함수요소>]+
Actuator [<구동기명>
<귀속함수요소>]+

<센서명>::= 센서 명칭

<귀속함수요소> ::= <언어변수> : (<귀속함수>) | <귀속함수요소>

<귀속함수> ::= (<원바>, <값>) | , <귀속함수>

<원바> ::= 실수

<값> ::= 0 | _ | 0.5 | _ | 1

<제어규칙집합> ::= <단위제어규칙> | <제어규칙집합> <단위제어규칙>

<단위제어규칙> ::= <제어규칙번호> : IF (<조건부>) THEN (<제어부>) [(<확신도>)].

<조건부> ::= <not조건> | <조건부> <not조건>

<not조건> ::= NOT <조건퍼지명제> | <and조건>

<and조건> ::= <or조건> | <and조건> AND <or조건>

<or조건> ::= <조건퍼지명제> | <or조건> OR <조건퍼지명제>

<조건퍼지명제> ::= <센서> = <언어변수> ((<임계치>))

<센서> ::= sen1 | sen2 | _ | sen_n

<임계치> ::= 0 | _ | 0.5 | _ | 1

<제어부> ::= <not제어> | <제어부> <not제어>

<not제어> ::= NOT <제어퍼지명제> | <and제어>

<and제어> ::= <or제어> | <and제어> AND <or제어>

<or제어> ::= <제어명제> | <or제어> OR <제어퍼지명제>

<제어퍼지술어> ::= <구동기> = <언어변수>

<구동기> ::= act1 | act2 | _ | act_n

<언어변수> ::= Very-large | Small | Medium | Large | _

<확신도> ::= 0 | _ | 0.5 | _ | 1

<규칙번호> ::= R1 | R2 | _ | R_n

<관계집합> ::= <단위관계> | <관계> <단위관계>

<단위관계> ::= <관계번호> : <관계>.

<관계> ::= <규칙번호> <관계연산자> <규칙>

<규칙> ::= <규칙번호> | <관계>

<관계번호> ::= Rel1 | Rel2 | _ | Rel_n

<관계연산자> ::= SEQ | ITR | PAR | FORK | JOIN | ME | MI */각각 순차, 반복, 병행, 분기, 결합, 상호 배타 및 상호 보완을 나타냄 */

부록 B. TFPN 수행 알고리즘

Procedure *Execution*()
Define symbols and functions
$p_i\#$ = {0,1} // 플레이스 p_i내의 토큰수(토큰마킹) //
cp_j^* = $\mu(sen_j) > \alpha_i$ // 조건플레이스 cp_j의 퍼지마킹 함수 //
δ_k = $\pi(ct_k)$ = $AF_k^T(ip_i^*)$ // 제어트랜지션 ct_k의 퍼지점화 함수 //
TM_0, TM_c //initial and current Token Marking //
ETS // enabling transition set //
CETS // concurrent enabling transition set //
Input
sen_j // cp_j가 모형화한 센서의 값 //
Output
δ_k // ct_k가 모형화한 구동기의 제어시간 구간 //
Reachability graph
Initialization
Edit TFPN (graphic or textual), internal_data_structures, place_transition_membership_batabase
Get TM_0
for $\forall p_i \in P$, do p_i^*=0 **end_do** // p_i의 초기치 //
print "intial marking is", TM_0.
repeat
// making procedure //
for $\forall sen_i \in SEN$ **concurrently_do** // get Fuzzy Marking //
Get sensor input sen_i.
Get fuzzy making. (i.e., **for** $\forall cp_j \in \pi_{sen_i}$ **do** $cp_j^* = \mu(sen_i)$ **end_do**)
end_do
Update TM_c. (i.e, **for** $\forall p_i \in P$, **if** $p_i^* > 0$ **then** $\#p_i$=1.)
Find CETS. (i.e., **for** $\forall t_i \in T$ **do (for** $\forall ip_j \in I(t_i)$, $\forall op_j \in O(t_i)$, **if** ($\#ip_j$=1 **and** $\#op_j$=0) **then** $t_i \in$ETS.)
end_do.
for $\forall t_i$, $t_j \in$ETS **do (if** ($I(t_i) \cap I(t_j) = \phi$ **and** $O(I(t_i)) = \{t_j\}$ **and** $O(I(t_j)) = \{t_i\}$ **then** $t_i, t_j \in$CETS)
end_do.)
Partition CETS by act_i. (i.e., CFact_i = {t_j | $t_j \in$CETS **and** $t_j \in \pi_{sen_i}$})
// firing procedure //
for $\forall t_i \in$CETS **concurrently_do** // Breadth-First Reachability Graph generation //
if $t_i \in NT \cup MT$
then Inference. (i.e., **if** $t_i \in$NT **then for** $\forall ip_j \in I(t_i)$, $\forall op_j \in O(t_i)$
$op_j^* = \mathrm{Min}(\{ip_j^*\})$ **else** $op_j^* = \mathrm{Max}(\{ip_j^*\})$ **end_if.**
$\forall ip_j \in I(t_i)$ $\forall ip_j \in I(t_i)$
ip_j^*=0, $\#ip_j$=0, $\#op_j$=1) // update marking //
print "when", t_i, "is fired next marking is", TM_c.
else (i.e., $ct_i \in$CFact_k)
Fuzzy firing. (i.e., **for** $\forall ip_j \in I(t_i)$, $\forall op_j \in O(t_i)$, $\delta_j = \pi(ct_j) = AF_j^T(ip_j^*)$)
Defuzzyfication. (i.e., Get τ_i from δ_i's.)
Control actuator act_i by τ_i.
CETS = CETS - CFact_k.
Reset all tokens. (i.e., **for** $\forall p_j \in P$, $\#p_j$=0, p_j^*=0)
Get new marking. (**for** $\forall op_j \in O(t_i)$, $\#op_j$=1, $op_j^* = \delta_j$)
print "when", CFact_k "is(are) fired with delay of", τ_i, "next marking is", TM_c.
end_if
end_do
until no more sensor input or enabled transition.

부록 C. 직렬 모터 제어의 수행 결과 수치표

I (전류)	조건플레이스의 퍼지값	제어시간범위 δ, 면적 a 및 무게중심 m	본연구결과 τ (RPM)	문헌 [31]	인간 조작
1	cp1*=0.5, cp2*=0.5,	δ1=[1800,2000], δ2=[1400,1800]	1678	1673	2000
		$a1$=150, $m1$=1833, $a2$=300, $m2$=1600,			
1.5	cp1*=0.25, cp2*=0.75,	δ1=[1709,2000], δ2=[1500,1700]	1641	1625	
		$a1$=87.5, $m1$=1816, $a2$=375, $m2$=1600,			
2	cp2*=1,	δ2=[1600,1600]	1600	1600	1850
		$a1$=400, $m1$=1600			
2.5	cp2*=0.75, cp3*=0.25,	δ2=[1500,1700], δ3=[900,1500]	1473	1484	
		$a1$=375, $m1$=1600, $a2$=175, $m2$=1600			
3	cp2*=0.5, cp3*=0.5,	δ2=[1400,1800], δ3=[1000,1400]	1400	1400	1550
		$a1$=300, $m1$=1600, $a2$=300, $m2$=1200			
3.5	cp2*=0.25, cp3*=0.75,	δ2=[1300,1900], δ3=[1100,1300]	1327	1315	
		$a1$=175, $m1$=1600, $a2$=375, $m2$=1200			
4	cp3*=1,	δ3=[1200,1200]	1200	1200	1350
		$a1$=400, $m1$=1200			
4.5	cp3*=0.75, cp4*=0.25,	δ3=[1100,1300], δ4=[500,1100]	1073	1200	
		$a1$=375, $m1$=1200, $a2$=175, $m2$=800			
5	cp3*=0.5, cp4*=0.5,	δ3=[1000,1400], δ4=[600,1000]	1000	1000	1100
		$a1$=300, $m1$=1200, $a2$=300, $m2$=800			
5.5	cp3*=0.25, cp4*=0.75,	δ3=[900,1500], δ4=[700,900]	927	915	
		$a1$=175, $m1$=1200, $a2$=375, $m2$=800			
6	cp4*=1,	δ4=[800,800]	800	800	900
		$a1$=400, $m1$=800			
6.5	cp4*=0.75, cp5*=0.25,	δ4=[700,900], δ5=[400,700]	759	775	
		$a1$=375, $m1$=800, $a2$=87.5, $m2$=583			
7	cp4*=0.5, cp5*=0.5,	δ4=[600,1000], δ5=[400,600]	722	726	800
		$a1$=300, $m1$=800, $a2$=150, $m2$=567			
7.5	cp4*=0.25, cp5*=0.75,	δ4=[500,1100], δ5=[400,500]	671	560	
		$a1$=175, $m1$=800, $a2$=187.5, $m2$=550			
8	cp5*=1,	δ5=[400,400]	533	500	700
		$a1$=200, $m1$=533			
8.5	cp5*=0.75, cp6*=0.25,	δ5=[400,500], δ6=[400,700]	561	511	
		$a1$=187.5, $m1$=550, $a2$=87.5, $m2$=567			
9	cp5*=0.5, cp6*=0.5,	δ5=[400,600], δ6=[400,600]	567	528	550
		$a1$=150, $m1$=567, $a2$=150, $m2$=567			
9.5	cp5*=0.25, cp6*=0.75,	δ5=[400,700], δ6=[400,500]	561	511	
		$a1$=87.5, $m1$=583, $a2$=187.5, $m2$=550			
10	cp6*=1,	δ6=[400,400]	533	500	550
		$a1$=200, $m1$=533			

이 강 수

1981년 홍익대학교 전자계산학과 졸업(학사)
1983년 서울대학교 대학원 계산통계학과 (이학석사)
1987년 서울대학교 대학원 계산통계학과 (이학박사)
1985년~1989년 국립대전산업대학 전자계산학과 전임강사
1992년~1993년 미국 일리노이대학교 객원교수
1995년 한국전자통신연구소 초빙연구원
1987년~현재 한남대학교 전자계산공학과 부교수
관심분야: 소프트웨어공학, 실시간 시스템 모형화 방법론, 패트리넷 응용, 보안프로토콜 모형화, 멀티미디어 동기화 등

김 소 연

1990년 한남대학교 전자계산공학과 졸업
1992년 한남대학교 대학원 전자계산공학과(공학석사)
1994년~현재 한남대학교 전자계산공학과 박사과정
관심분야: 소프트웨어공학, 패트리넷 응용, 병행시스템 모형화

윤 정 모

1968년 광운대학교 응용전자공학과 졸업
1971년 성균관대학교 산업대학원 전산학전공 (경영학석사)
1993년 일본 오사카부립대학교 대학원 전자계산학전공 (공학박사)
1966년~1982년 한국전력(주) 근무
1986년~현재 서울산업대학교 전자계산학과 부교수
관심분야: 패트리넷 응용, 소프트웨어공학 등