

---

# 퍼지 컴퓨터와 퍼지 소프트웨어의 전망

---

김 동 춘 前 금성가전연구소 주임연구원  
이 근 철 제일전산훈련원 원장

## ■ 머리말

퍼지 컴퓨터에 관한 연구는 '80년대 중반부터 몇 가지 모델이 발표되면서 시작되었다.

하나는 기존의 디지털회로를 이용하여 퍼지집합을 표현하고 이것을 빨리 처리할 수 있도록 하는 추론기관을 VLSI화 하는 것이고 또 하나는 소속함수 값을 표현하는 0과 1 사이의 값을 아날로그회로의 0V에서 5V 사이의 값으로 대응시켜 처리하는 아날로그형 컴퓨터이다.

물론 후자의 경우가 처리속도가 빠르고 기억장소를 적게 사용하지만 디지털 시스템의 견고성과 프로그래밍의 용이성 때문에 전자의 방식을 사용한 퍼지 프로세서가 상품화되어 사용되고 있다.

한편 퍼지 컴퓨터에 관한 연구는 제한적인 범위내에서 이루어지고 있으며 현재까지 나온 퍼지 컴퓨터는 엄밀히 말해서 퍼지논리에 의한 퍼지 추론기관이라고 말할 수 있다.

또한 퍼지 연상용 소프트웨어도 PROLOG (Programming in Logic의 준말)와 LISP(List Processing의 약어)의 확장으로서 만들어지고 있다.

## 1. 퍼지 컴퓨터

인간은 0과 1만을 취급하는 불대수(2치논리)와 스위칭 디바이스(트랜지스터)를 축적 프로그램 방식이라는 매우 교묘한 방법으로 융합시켜 유용성이 높은 디지털 컴퓨터를 만들어 냈다.

이 디지털 컴퓨터는 본래부터 가지고 있는 잡음에 대한 강건성과 시스템의 확장성에 의해 오늘날 거대한 정보처리 시스템을 구축한 것이다.

그동안 컴퓨터 내부에서 신호처리방식은 축적 프로그램의 방식으로 현재까지 기본적 동작에는 아무런 변화가 없이 정보를 처리해 온 것이다.

그러나 컴퓨터를 반복해서 이용하면 할수록 무엇인가 부족함을 느끼게 하는데 이러한 사실은 인공지능(AI) 분야에서 현저하게 나타나고 있다.

이 분야에서는 수치계산보다는 논리의 전개에 중점을 두므로 If~then... 형식의 추론이 인기가 높는데 현재 인공지능에 대해서 행하여지고 있는 추론은 유니피케이션(단일화)이라고 하는 데이터·매칭의 수법이 기본으로 되어 있다.

따라서 결정론적인 데이터 즉 윤곽이 분명한 데이터끼리의 매칭이 가능하고 이를 위한 추론

전용 프로그램 언어와 하드웨어가 개발되고 있다.

그러나 인간이 취급하고 있는 정보는 반드시 윤곽이 확실한 것 뿐만이 아니며 오히려 윤곽이 분명하지 않은 정보가 대부분인 경우가 많다.

예를 들면 “고열이 수일간 이상 계속되면 폐렴이 되었을 가능성이 있다”와 “식후의 혈당치의 감소가 약한 경우는 인슐린의 분비량이 적다” 등이다.

이와 같은 애매한 언어정보로부터 구축된 지식 베이스와 입력 데이터의 유니퍼케이션을 실행하는 것은 불가능하며 애매한 정보와 소프트·매칭을 실행하고 애매하지만 매우 타당한 결론을 도출하는 수법이 요구되고 있다. 퍼지추론은 실제로 이것을 가능케 하며 현재까지의 인공지능에 채용되어온 수법에 큰 변화를 가져오고 있다.

디지털 컴퓨터는 만능기계로서 프로그램을 연구함으로써 퍼지추론도 실행할 수 있으나 그 처리는 오류가 많아 시간이 너무 걸린다.

한편 가능한 것과 최적인 것은 전혀 별개의 문제이므로 퍼지추론 전용의 하드웨어 즉 퍼지 컴퓨터의 출현이 절실히 요망되고 있다.

퍼지 컴퓨터는 인간이 디지털 컴퓨터에 대하여 생각했던 불만을 한번에 해결해 주는 것으로서 데이터에 내부구조나 하드웨어의 아키텍처, 더욱이 추론 알고리즘도 지금까지의 추론 머신과는 전혀 다르기 때문에 제6세대 컴퓨터라고 말할 수 있다.

### 1.1 퍼지추론

퍼지추론은 지식 베이스로서 보존되어 있는 추론규칙(이것을 단지 지식이라고 한다)과 주어진 사실로부터 새로운 결론을 도출하는 것이지만 통상의 추론과 다른 점은 명제중에 변수가 모두 퍼지변수, 즉 애매한 언어정보로 구성되어 있다는

점이다. 이것을 다음과 같이 기술할 수 있다.

〈지식〉 If  $x$  is  $A$ , then  $y$  is  $B$

〈사실〉  $x$  is  $A'$

〈결론〉  $y$  is  $B'$  (後向 퍼지 추론) (1)

또 다음과 같은 추론도 생각할 수 있다.

〈지식〉 If  $x$  is  $A$ , then  $y$  is  $B$

〈사실〉  $y$  is  $B'$

〈결론〉  $x$  is  $A'$  (前向 퍼지 추론) (2)

여기서  $A, A', B, B'$ 는 윤곽이 애매한 언어 정보(퍼지 집합)이다.

식 (1)과 식 (2)의 추론을 고전논리(불논리)에 의해서 실행하는 것은 매우 곤란하다.

애매한 언어정보  $A$ 는 예를 들면 그림 1(a)와 같이 멤버십(Membership) 함수라고 하는 특성함수로 표현되는데 이것은 그림 1(b)와 같이 샘플링 되어 다음과 같이 순서 對 집합으로서 표현된다.

$$A = \{(x_i, a_i)\}, x_j \in X \quad (3)$$

여기서  $X$ 는 전체집합이고  $a_i$ 는  $x_i$ 가 집합  $A$ (이 경우는 “키가 크다”라는 집합)에 속하는 정도이다.  $a_i$ 는 편의상 구간  $[0, 1]$ 의 값을 취하는 것으로 하며 퍼지집합  $A$ 는 벡터로 간주할 수도 있어 다음과 같이 기술할 수 있다.

$$A = (a_1, a_2, a_3, \dots, a_i, \dots, a_m), 0 \leq a_i \leq 1 \quad (4)$$

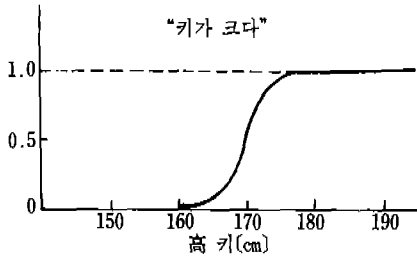
다만  $m$ 은 원소의 개수이고 유한개이다. 같은 방법으로 별도의 언어정보  $B$ 는 다음과 같이 표시할 수 있다.

$$B = (b_1, b_2, b_3, \dots, b_j, \dots, b_n), 0 \leq b_j \leq 1 \quad (5)$$

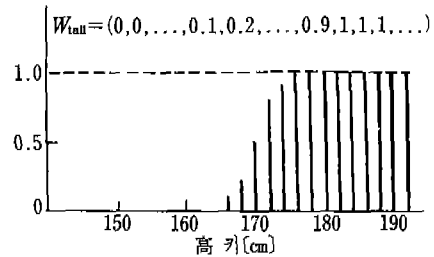
$A$ 를 원인,  $B$ 를 결과로 하는  $A$ 와  $B$ 의 인과 관계를 나타내는 행렬을 정의할 수 있고 이것을  $A$ 로부터  $B$ 로의 퍼지관계  $A$ 라고 말한다. 즉

$$R = \begin{bmatrix} r_{11} & \dots & r_{1n} \\ \vdots & & \\ r_{m1} & \dots & r_{mn} \end{bmatrix} \quad (6)$$

한편 퍼지이론의 제창자인 캘리포니아대학 교



(a) "키가 큰(tall)"이라는 언어의 멤버쉽 함수



(b) (a)를 샘플링한 이산표현

<그림 1 >

수였던 A. Zadeh는 (1)식의 추론결과  $B'$ 는 다음 식에서 구해진다고 하였다.

$$B' = (b'_1, b'_2, b'_3, \dots, b'_j, \dots, b'_n)$$

$$= (a'_1, a'_2, a'_3, \dots, a'_j, \dots, a'_m) \circ \begin{bmatrix} r_{11} \dots r_{1n} \\ \vdots \\ r_{m1} \dots r_{mn} \end{bmatrix}$$

$$= A' \circ B \quad (7)$$

$$\text{또는 } b'_j = \bigvee_i a'_i * r_{ij} \quad (8)$$

여기서 \*는 예를 들면 퍼지 논리곱(MIN)이나 대수곱 등인데 Zadeh 교수는 MIN을 제창하여 오늘날도 널리 사용되고 있으며 그 타당성이 입증되고 있다.

퍼지관계에 대해서는 여러 가지 연산이 제안되고 있는데 Zadeh 교수는 다음과 같은 2가지 타입의 퍼지관계를 제안하였다.

$$r_{ij} = (a_i \wedge b_j) \vee (1 - a_i) \quad (9)$$

$$r_{ij} = 1 \wedge (1 - a_i + a_j) \quad (10)$$

그리고 Mamdani 교수는 다음과 같은 별도의 타입의 퍼지관계를 제안하고 실용적인 응용에서 그 타당성을 보여 주었다.

$$\text{즉 } r_{ij} = a_i \wedge b_j \quad (11)$$

이것을 기준으로 평가하여 주관적인 추론에 보다 적합한 퍼지관계를 얻고 있으며 식 (1)의 전향 퍼지추론 혹은 식 (2)의 후향 퍼지추론을 2진 디지털회로가 아니라 퍼지 전용회로로서 병렬로 실

행하는 非노이만형(비축적 프로그램 방식)의 컴퓨터를 퍼지 컴퓨터라고 정의하였다.

즉 식 (1)과 식 (2) 중에서 사용되는 애매한 언어정보  $A, A', B, B'$  등은 그림 1(a)와 같은 멤버쉽 함수로 표현할 수 있기 때문에 이것을 그림 1(a)와 같이 샘플링하여

$$W_{\text{tall}} = (0, 0, \dots, 0.1, 0.2, \dots, 0.9, 1, 1, 1, \dots) \quad (12)$$

와 같은  $m$ 개의 요소(각각은 0~1의 값)로부터 되는 벡터로 한다. 이것을 통상의 2진 워드에 대하여 퍼지워드라고 부르기도 하여 각 요소의 진리치(0~1)를 예를 들면 0V~5V 혹은  $0\mu\text{A}$ ~ $100\mu\text{A}$ 에 대응시켜, 10의 퍼지워드를  $m$ 개의 신호라인(데이터 버스)으로 취급하기로 한다.

기본적으로 퍼지 컴퓨터는 애매한 언어정보를 기억하는 퍼지 메모리와 그 정보를 이용하여 퍼지추론을 실행하는 퍼지추론 엔진으로 구성되어 있고 필요에 따라서 디퍼지파이어(Defuzzifier)가 부가된다.

이상의 3가지 블록은 모두 연속량으로서 전기 신호를 취급하지 않으면 안되기 때문에 퍼지 전용 전자회로로 설계할 필요가 있다.

## 1.2 퍼지논리 회로

퍼지논리 함수는 2치 논리와 다치 논리와는 달

리 본질적으로 무한히 정의할 수 있지만 현재 널리 알려져 있는 것은 십여 종류이다.

이 중에서 퍼지추론에 가장 기본이 되는 트랜지스터를 이용한 퍼지 논리합(MAX)과 퍼지 논리곱(MIN)의 입출력 특성을 살펴보기로 한다.

전자회로는 취급할 신호의 모드에 따라서 전류 모드회로와 전압 모드회로로 나눌 수 있으며 전류 모드는 정보신호를 전류가 흐르는 방향과 전류의 크기로 표현하는 것인데 I<sup>2</sup>L(Integrated Injection Logic)은 대표적인 예이다.

한편 전압 모드회로는 정보신호를 전압의 극성과 크기로 표현하는 것으로 디지털회로와 아날로그회로가 이에 해당된다.

전류 모드회로는 가감산에 단순한 리드선의 결선으로 실현 가능하고 여분의 트랜지스터 이외에는 어떠한 회로요소도 필요로 하지 않는다는 큰 특징을 갖고 있지만 하나의 회로는 하나의 부하밖에 구동할 수 없다(Fan Out 1)는 결점도 있다.

이와 반대로 전압 모드회로는 팬아웃이 풍부하지만 가감산에는 적합하지 않다. 그러나 전류 모드회로는 전원 전압 변동에는 매우 강하다.

그림 2와 그림 3에 각각 MAX 회로와 MIN 회로를 나타내며 이 전압 모드회로에서는 진리치 0~1를 0V~5V로 대응시키고 있다. MAX 회로와 MIN 회로는 비교되어 최대치 -0.7V 혹은 최소치 +0.7V가 에미터 결합된 NPN 트랜지스터 어레이(MAX 회로) 또는 PNP 트랜지스터 어레이의 에미터(MIN 회로)에 각각 나타난다.

비교부의 모든 트랜지스터가 에미터로 결합되어 있기 때문에 그림 2와 그림 3은 에미터 결합 퍼지논리 게이트(ECFL Gate : emitter coupled fuzzy logic gate)라고 부르기도 한다.

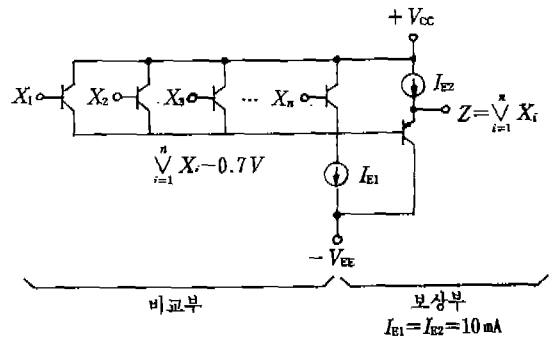
비교부의 에미터 접합의 전압 0.7V는 다음 단계의 보상부에 의해서 보정된다. 이와 같이 입력 전압의 최대치와 최소치가 그림 2와 3의 출력 Z

로서 나타난다.

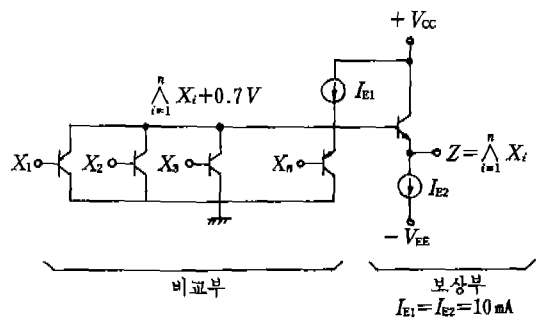
보상부는 非部位 전압 시프트만을 보정할 뿐만 아니라 비교부 에미터 접합 전압의 열적 드리프트도 동시에 보상하고 있다.

한편 ECFL 게이트의 다른 큰 특징은 팬아웃(Fan Out)을 많이 취할 수 있으며 전원 전압 변동에 매우 강하다.

또한 ECFL 게이트 +V<sub>CC</sub>가 +6V~+51V의 범위에서 그리고 -V<sub>EE</sub>가 -1V~46V의 범위에서 변화하여도 완전히 정상으로 동작하기 때문에 통상의 2진·디지털회로 등은 비교가 안될 정도로 강건한 회로이다.



<그림 2> 퍼지 논리합회로(MAX)



<그림 3> 퍼지 논리곱회로(MIN)

### 1.3 퍼지 메모리 디바이스

퍼지 메모리는 그림 1(a)에서 나타내는 멤버십 함수 혹은 그림 2의 샘플링한 함수를 기억하는 것이다. 즉 식 (12)와 같은 퍼지워드를 하나의 단위로 기억하고 읽어내는 것이다.

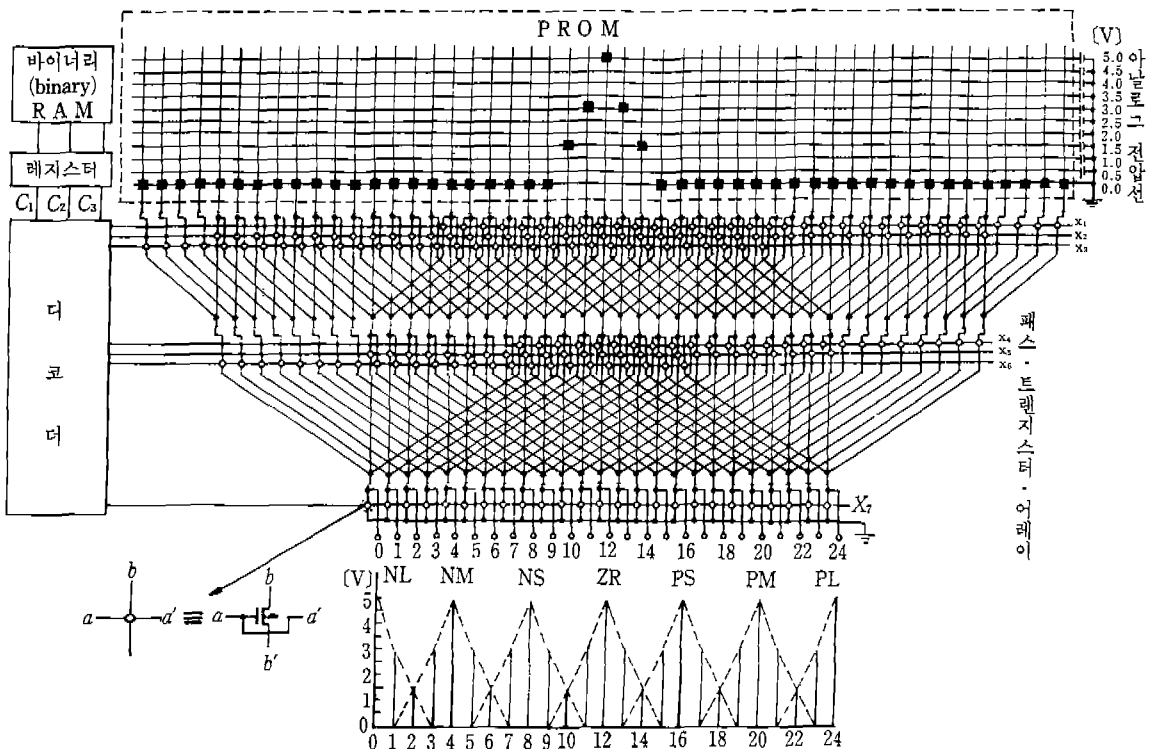
그림 4에 퍼지 메모리 디바이스의 구성을 나타낸다. 여기서 메모리가 취급하는 애매한 정보는

- 負보다 크다(NL : Negative Large)
- 負에서 중간 정도(NM : Negative Medium)
- 負보다 작다(NS : Negative Small)
- 약, 제로(ZR : Approximately Zero)
- 正보다 작다(PS : Positive Small)

- 正에서 중간 정도(PM : Positive Medium)
- 正보다 크다(PL : Positive Large)

의 7가지로 한다.

여기서 NL, NM, NS, ZR, PS, PM, PL은 언어정보에 붙여진 라벨로서 퍼지 메모리는 이 용자가 PROM(Programmable Read Only Memory)부에 ZR의 멤버십 함수를 써넣고 외부의 2치 메모리에 저장된 000~111의 라벨(PL : 111, PM : 110, PS : 101, ZR : 100, NS : 011, NM : 010, NL : 001, NOT SPECIFIED : 000)을 1 to 8 디코더로 7비트 워드로 변환하고 패스·트랜지스터·어레이를 제어하여 아날로그 전



<그림 4> 퍼지 메모리 디바이스의 구성

압(전리치 0~1은 0V~5V로 대응)을 25개의 출력단자(출력 워드)로 집어내는 것이다.

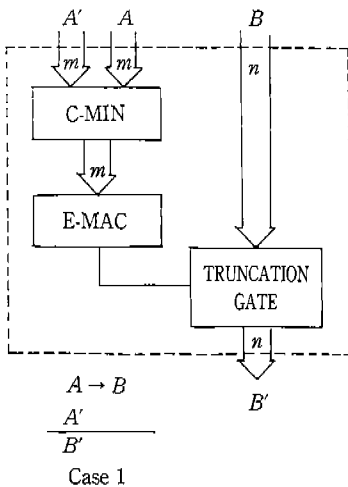
실제로 試作된(AI 게이트 PMOS 프로세스) 특성을 보면 PMOS이면서 CMOS와 동일한 동작을 하기 때문에 정상오차가 거의 없다고 하며 응답 속도에 대해서는 제조장치의 제한으로 인하여 늦은 결과를 얻었다.

### 1.4 퍼지추론 엔진 및 퍼지 컴퓨터 구성

식 (1)의 전향 퍼지추론을 식 (8)을 이용하여 실행한 경우의 퍼지추론 엔진 블록도는 그림 5와 같다.

여기서 퍼지워드  $A, A'$ 의 요소수와  $B, B'$ 의 요소수를 각각  $m, n$ 으로 하였다. C-MIN은  $m$ 개의 2입력 MIN 회로를 병렬로 한 것이고 E-MAX는  $m$ 입력 1출력의 MAX 회로이다.

TRUNCATION GATE는  $n$ 개의 2입력 MIN 회로를 병렬로 하고 2입력중에 한편을 모두  $a$ 에



<그림 5> 퍼지추론 엔진의 블록도

연결한 것이다.

$A, B$ 를 지식베이스로부터 If  $x$  is  $A$ , then  $y$  is  $B$ 로서 입력하고  $A'$ 에 사실 정보로서  $x$  is  $A'$ 로 입력하면  $B'$ 에는  $y$  is  $B'$ 로서  $n$ 개의 신호라인(혹은 출력단자)에 샘플링된 멤버쉽 함수의 형태로 전압이 나타난다.

이것은 증후(Evidence) 명제와 결론(Hypothesis) 명제의 퍼지추론을 실행하기 때문에 기본 퍼지추론 엔진이라고 하며 1개의 추론 엔진보드에서는 288개의 바이폴라 트랜지스터가 사용되었다. 추론속도는 100ns 즉 매초 10000000추론이 가능하다고 한다.

한편 퍼지 논리합과 퍼지 논리곱회로를 이용하여 퍼지 컴퓨터를 구성하는 경우 그림 6과 같이 Fine Grain Parallelism을 사용할 필요가 있다.

즉  $r$ 개의 퍼지추론을 동시에 실행하기 위해  $r$ 개의 퍼지추론 엔진을 병렬로 배열하고 퍼지워드도 병렬로 입력된다.

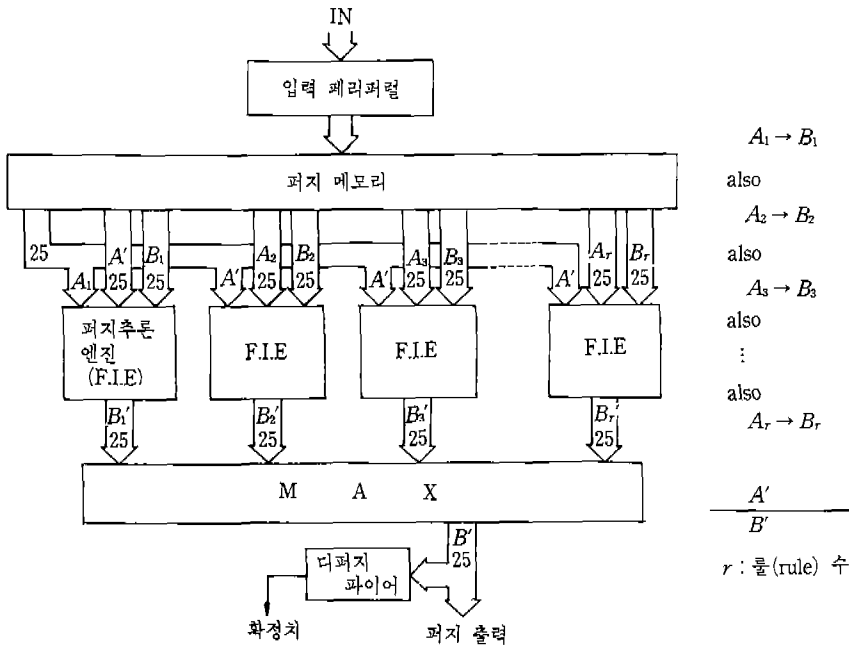
또 1개의 추론에 대해서 각 퍼지워드가  $m$ 개 또는  $n$ 개(여기서는 25개)의 신호라인에 전압으로서 분포하여 1개 1개의 신호가 완전히 병렬로 처리된다.

그림 6의 MAX라는 블록은 추론규칙중의 also를 실행하는 블록으로 각 추론엔진으로부터 얻어진 추론결과와 논리합을 취하는 곳이다.

퍼지출력(추론결과)  $B'$ 는 분포된 전압신호 그대로 실제로 사용할 수 없는 경우가 많기 때문에 디퍼지퍼어에 의하여 확정치가 출력된다.

퍼지 컴퓨터 전체의 동작을 同期시키기 위한 컨트롤 유닛과 컨트롤버스는 그림 6에서는 생략되었다.

1987년 일본 동경에서 개최된 퍼지 시스템 국제학회에서 발표한 퍼지 컴퓨터의 추론보드를 보면 입출력 주변장치는 종래의 디지털 시스템을 이용하고 있기 때문에 시스템 전체의 속도는 디



<그림 6> 퍼지 컴퓨터의 기본적 구조

지털 시스템의 속도로 결정된다고 한다.

최근 컴퓨터의 이용은 수치계산으로부터 추론으로 중심이 이동하고 있으며 인공지능분야에서는 추론을 위한 전용 프로그램과 전용 하드웨어가 널리 연구되고 있다.

1965년에 Zadeh 교수가 퍼지집합론을 제창한 이후 27년째가 되며 반도체기술이 진전된 오늘날 시스템을 구체화하는 것은 시간문제일 것이며 1995년쯤에는 실용적인 퍼지 컴퓨터가 디지털 컴퓨터와 융합한 형태로 시장에서 발매될 것이다.

## 2. 퍼지 소프트웨어와 전망

퍼지 소프트웨어는 퍼지집합을 취급할 수 있는 퍼지 프로그래밍 언어와 프로그램의 제어를 퍼지 이론을 이용하여 행할 수 있는 퍼지 프로그래밍 언어로 나눌 수 있다.

퍼지집합은 멤버십 함수를 이용하여 표현되어 있기 때문에 퍼지집합을 종래의 프로그래밍 언어로 기술할 수 있다. 이를 위해서는 퍼지집합들의 각종 연산에 대하여 복잡한 절차를 거쳐야 하므로 퍼지집합들간의 각종 연산이 짜 놓여져 있는 것이 바람직하다.

이와 같은 목적으로 개발된 것이 전자의 퍼지 집합을 취급할 수 있는 퍼지 프로그래밍 언어이며 이것은 정수와 변수로서 퍼지집합과 똑같이 통상의 크리스프(Crisp)적인 것도 표현할 수 있는 것이 보통이다.

후자의 프로그래밍 제어에 퍼지이론을 이용한 퍼지 프로그래밍 언어에서는 퍼지집합을 정의하고 이용할 수 있는 것은 당연하지만 관심의 대상은 각 스테이트먼트에 부가된  $[0, 1]$ 의 사이값, 또는  $[0, 1]$ 상의 퍼지집합(진리치, 확실도, 애매도 등으로 해석된다)에 의해 프로그램의 분기를

제어한다든가, 精度部의 결론을 얻는다든가 하는 것에 있다.

추론용의 프로그래밍 언어에 적용되고 있는 경우가 거의 대부분이며 여기에는 퍼지 프로덕션 시스템과 퍼지 프롤로그(PROLOG) 등이 있다.

퍼지집합을 취급할 수 있는 퍼지 프로그래밍 언어는 취급하는 대상이 퍼지집합이고 실행을 위한 스테이트먼트는 확정적인 것에 대하여 이것들을 각 스테이트먼트 그 자체에 퍼지 진리치를 줄 수 있다.

한편 실제로 가동되고 있는 퍼지 소프트웨어는 현재까지는 적으나 앞으로는 종래의 프로그래밍 언어에 퍼지집합과 퍼지 진리치를 취급할 수 있는 능력을 부가하는 방향으로 진전될 것이다.

스피드의 문제는 종래의 디지털회로로 구성되어 부가한다는 방향으로 나갈 것이며 퍼지 이론이 애매한 대상을 명확하게 취급하는 이론이라는 것을 고려한다면 장래의 퍼지 소프트웨어를 실행하는 하드웨어는 퍼지적인 대상을 아날로그로, 명확한 제어는 디지털로 하는 아날로그와 디지털이 혼합된 퍼지 컴퓨터가 될 것이다.

Zadeh 교수에 의하여 퍼지 알고리즘의 개념이나 자연언어에 의한 명제를 퍼지논리를 이용하여 기술하는 것이 제안되고 있다.

끝으로 인공지능과 응용분야에서 널리 응용되고 있는 PROLOG와 LISP 프로그래밍 언어에 대하여 간단히 소개한다.

PROLOG는 비절차형 언어로서 기초는 一階의 述語논리에 두고 있어서 추론규칙으로 도출원리를 이용하고 있다.

일계 술어 논리치는 2차 논리의 체계이고 각 명제나 술어에 포함되는 논리 변수의 값은 항상 진(1)이라든가 거짓(0)이든가의 어느쪽인가로 가정되어 있다.

특히 PROLOG의 각 스테이트먼트는 하나의

논리식에 상당하고 있어 이들 스테이트먼트 즉 사실이나 물은 항상 진(1)로 가정되어 있다.

그런데 퍼지논리에서는 각 논리변수의 진리치는 2차 논리와 같이 진(1)과 거짓(0)만이 아니라 폐구간  $[0, 1]$ 의 임의의 값을 취하는 것을 허락하고 있다.

따라서 PROLOG의 각 스테이트먼트에  $[0, 1]$ 의 임의의 값을 할당하고 그 수치를 진리치나 확신도로서 해석하면 애매한 사실이나 확신을 가질 수 없는 물로부터 추론할 수 있도록 될 것이다.

$[0, 1]$  사이의 값을 가진 결론을 얻음으로써 그 수치를 이용하여 프로그램의 제어를 바꿀 수 있는데 이것이 퍼지 PROLOG 연구의 동기가 된 것이다.

1970년대 초에 알랭폴메르가 개발한 PROLOG는 앞으로 숫자 계산보다는 인공지능 분야에서 논리적인 추론, 패턴매칭 및 리스트처리 등에 적합한 5세대 컴퓨터 언어가 될 것이다.

한편 LISP 언어는 1960년 MIT의 J.MacCarthy 등에 의해 만들어진 언어로서 Lambda 代數에 이론적 근거를 둔 적용형 언어로 프로그램과 데이터가 모두 S식(S Expression)이라는 일반화된 리스트 형태로 표현되므로 프로그램이 데이터처럼 취급될 수 있다.

또한 프로그램이 함수로 선언되며 일반적으로 함수의 재귀호출(Recursive Call)로 필요한 작업을 수행한다는 것이 특징이다.

주로 인공지능 분야와 식의 전개, 인수분해, 미적분, 정리증명, 로봇문제 및 자연어 처리 등에 많이 응용되고 있다.

현재 커먼 리스프(Common LISP), 프란츠 리스프(Frantz LISP) 및 LISP 1.5 등 원래의 리스프에 여러 가지 기능을 첨가한 언어가 사용되고 있으나 순 리스프(Pure LISP)는 실제로 거의 사용되지 않고 있다.