한국 퍼지 및 지능시스템학회 논문지 2000, Vol. 10, No. 4 (2000)293~298

병렬컴퓨팅 환경에서의 대용량 퍼지 추론

Fuzzy Inference of Large Volumes in Parallel Computing Environments

김진일 · 이상구

Jin-Il Kim and Sang-Gu Lee

한남대학교 정보통신 · 멀티미디어공학부

요 약

대단히 많은 수의 퍼지 규칙을 갖거나 대용량의 퍼지 데이터를 갖는 퍼지 전문가 시스템 또는 퍼지 데이타베이스 시스템에서는 많은 추론 시간을 요구한다. 따라서 이러한 추론 시간을 줄이기 위해서는 고성능 병렬 퍼지 컴퓨팅 환경을 필요로 한다. 본 논문에서는 병렬 컴퓨팅 환경에서 병렬 퍼지 추론 기법을 제안한다. 여기에서 퍼지 규칙은 분산되어 있고 동시에 수행된다. ONE_TO_ALL 알고리즘은 모든 노드에 퍼지 입력 벡터를 broadcasting하는데 사용한다. MIN/MAX 연산의 결과는 ALL_TO_ONE 알고리즘에 의해 출력 프로세서로 전송된다. 퍼지 규칙 또는 데이터의 병렬 처리로 인해, 병렬 추론 알고리즘은 효과적인 병렬성의 추출 및 속도 향상을 가져온다.

ABSTRACT

In fuzzy expert systems or database systems that have huge volumes of fuzzy data or large fuzzy rules, the inference time is much increased. Therefore, a high performance parallel fuzzy computing environment is needed. In this paper, we propose a parallel fuzzy inference mechanism in parallel computing environments. In this, fuzzy rules are distributed and executed simultaneously. The ONE_TO_ALL algorithm is used to broadcast the fuzzy input vector to the all nodes. The results of the MIN/MAX operations are transferred to the output processor by the ALL_TO_ONE algorithm. By parallel processing of fuzzy rules or data, the parallel fuzzy inference algorithm extracts effective parallelism and achieves a good speed factor.

1. 서 론

퍼지이론이 제어분야 및 공학, 사회, 자연, 의료 등 많은 분야에 응용되고 있음에도 불구하고, 퍼지이론을 적용하기 위해 반드시 수반되는 전문가의 지식베이스의 규칙들을 쉽게 얻기 어렵고, 퍼지논리는 [0, 1] 사이의 실수 연산이 필요하기 때문에 수천 개 이상의 퍼지 규칙을 갖는 퍼지추론을 실시간 내에 수행하기 쉽지 않으므로, 퍼지 연산이 대규모로 실행되면서 실시간성이 요구되는 시스템이나 대용량의 전문가 시스템 등에서의 실시간 처리에는 아직 해결하여야 할 문제점들이 남아있다. 또한, 지금까지 개발된 대부분의 퍼지 하드웨어들은 AND/OR(Min/Max)의 연산은 병렬로 수행하고 있지만 퍼지규칙들에 대해서는 순차적으로 수행하고 있으므로, 대단위의 지식베이스 시스템 및 의료 진단 시스템에 활용할 수 있는 복합 지능 전문가 시스템을 위해서는 병렬컴퓨팅 환경에서의 대용량 퍼지추론을 필요로 한다.

퍼지논리 추론의 전용 하드웨어 개발에 대한 연구는 Yamakawa, Togai, Watanabe의 연구를 시작으로 일본의 OMRON, OKI, HITACHI, 미국의 American NeuraLogix, Togai Infralogic, VLSI Technology, 독일의 Siemens AG 등 여러 곳에서 진행되어, 프로세서 형태의 칩 또는 칩을 장착한 가속보드의 형태로 판매하고 있다. Jaramillo-Botero[1]는 퍼지 chip을 이용하여 일반 PC의 외부 확장 슬롯에 FP시리즈에 호환 가능한 디지털 인터페이스를 설계하여 여러 개의 rule chip을 달아 여러 개의 퍼지 유닛을 구성하였고, 이들의 출력을 모아 하나의 defuzzifier 유닛을 구성하여 PC에서 제어할 수 있는 시스템을 만들었다. Ungering과 Goser[2]는 64-비트의 퍼지추론 프로세서를 설계하였다. 또한 Ungering은 범용의 8-bit 마이크로컨트롤러에 퍼지 기능을 추가시킨 F166을 개발하였으며, Avodadro 등은 통상의 RISC 명령이외에 퍼지 응용을 위한 Max연산과 Min연산 명령을 추가한 RISC 프로세서 "FLORA"를 개발하여 범용의 프로세서에 퍼지 연산 기능을 부가하는 것에 의해, 범용성을 잃지 않고 퍼지 연산의 성능을 향상시킬 수 있다. Falchieri 등은 active rule들을 parallel-pipeline 방식으로 처리할 수 있는 아키텍처를 VHDL언어를 사용하여 0.7 μm CMOS ES2 공정으로 소규모의 고속 퍼지 칩을 개발하였다. 여기서는 입력변수를 2개, 출

표 1. 병렬 퍼지 추론 시스템
Table 1. Parallel fuzzy inference systems

Architecture	Hypercube 등 병렬컴퓨터	Transputer	FPGA
Degree of Parallelism	Partitioned rules	Fuzzy rules	전건부와 후건부의 부분 fuzzy rule
Needed number of processors	$N(2^n)$	$N(2^n)$	전건부와 후건부 변수의 최대값
Inference rules	All rules	Only active rules	All rules
Size level	병렬 컴퓨터	보드 레벨	칩 레벨
Merits	대용량의 퍼지규칙 또는 많은 전건부, 후건부 변수	실시간 추론	MISO, MIMO
Applications	Expert system, Fuzzy database system	로봇 제어, 영상처리	일반용

력변수를 1개로 제한하였다.

최근에는 퍼지연산에서 병렬처리 기법들에 대한 연구가 진행되고 있으며, 퍼지컴퓨팅에서 병렬퍼지 추론 시스템의 아키텍처 및 각 특성, 응용분야는 표 1과 같이 요약할 수 있다[3,4].

2. 병렬 퍼지 추론

2.1 퍼지 규칙들의 분산

복잡한 퍼지 지식베이스 시스템일수록 이에 필요한 제어규칙들이 증가하는 특징을 가지고 있다. 추론과정에서 병렬처리 기법을 사용함으로써 순차적처리 방법보다 빠른 퍼지제어를 할 수 있다. 따라서, 병렬퍼지 추론을 사용하기 위하여 추론에 필요한 제어 규칙들을 재구성하여 병렬시스템의 각 노드에 분산시켜 저장해야 된다. 제어 규칙들을 분산시키기 위하여 $N(=2^n)$개의 노드로 구성된 n-dimensional cube에 대해 N-modulo function을 사용하면 N으로 나눈 나머지가 i인 제어 규칙은 P_i번째 노드의 MB_i 기억장소에 저장된다. 많은 노드들의 기억장소에 분산되어 저장된 제어 규칙들은 추론과정에서 2가지 잇점을 가지고 있다. 첫째, 병렬컴퓨팅 시스템에서의 병렬처리를 가능하게 해준다. 각 노드에는 R개의 전체 제어 규칙 중 $[R/N]$개의 규칙만을 갖게 된다. 분산 저장된 제어 규칙들에 대하여 다른 제어 규칙들을 추가하는 것은 modulo function에 의하여 쉽게 각 기억장소에 분산되어 저장될 수 있다. 두 번째, Min 연산을 사용한 퍼지 추론에서 매우 효과적으로 수행할 수 있다.

2.2 자료전송과 병렬 Min 동작

분산 저장된 제어 규칙들에 대하여 입력 벡터 X의 값이 P_0(input node) 노드에 들어오면 ONE_TO_ALL broadcasting 알고리즘에 의하여 P_0 노드에서부터 다른 모든 노드에 X를 전송한다. ONE_TO_ALL

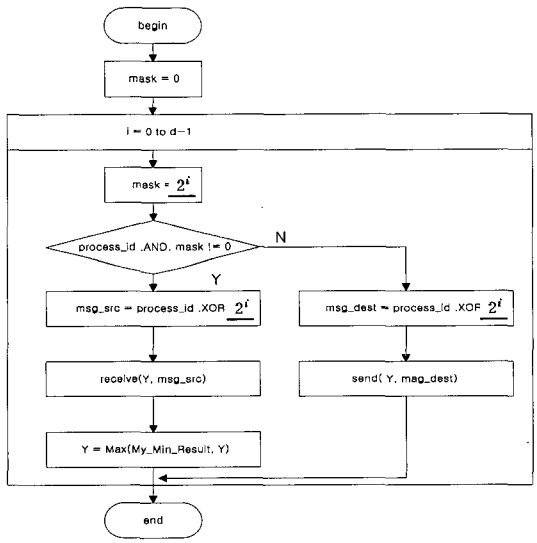

그림 1. ONE_TO_ALL broadcasting 알고리즘
Fig. 1. ONE_TO_ALL broadcasting algorithm

알고리즘은 그림 1에 나타나 있다. 각 노드는 같은 값의 입력벡터 X를 입력받게 되고 입력벡터 X는 store-and-forward routing 방법에 따라 broadcasting 알고리즘이 끝날 때까지 이웃한 노드로 전달된다.

입력퍼지 데이터인 입력벡터 X는 log N 스텝에 P_0에서부터 모든 노드로 전송된다. 각 노드의 주소는 n 비트로 이루어진 비트열로 나타낼 수 있다. ONE_TO_ALL broadcasting 알고리즘은 AND 연산을 사용하여 비트열을 masking 하고, XOR 연산을 사용하여 입력벡터 X의 다음 목적지 노드를 결정한다. 퍼지 추론에 대하여 제어부분에서 퍼지 입력값과 전건부의 소속함수 사이에 병렬 Min 동작이 사용된다.

2.3 Max 동작과 비퍼지화

Min 연산은 기억장소에 저장되어 있는 제어 규칙들

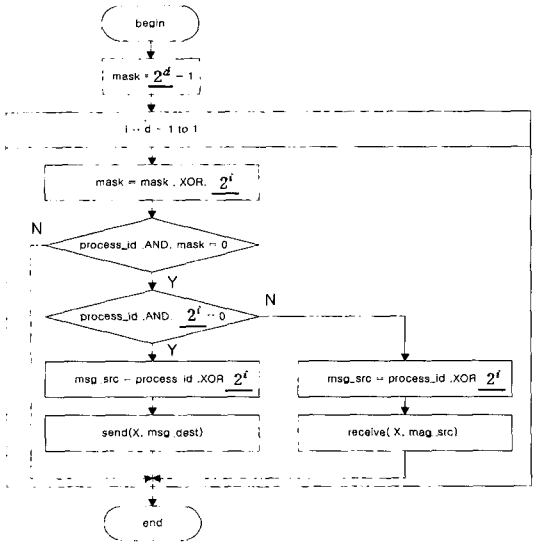

그림 2. ALL_TO_ONE 알고리즘
Fig. 2. All_TO_ONE Algorithm

과 입력벡터 X를 사용하여 각 노드에서 병렬로 처리될 수 있기 때문에 Min과 Max 연산을 동시에 사용할 수 없다. 보통 Max 연산은 각 노드에서의 Min 연산에 의한 결과값을 사용하여 수행되기 때문에 Min과 Max 연산의 병렬처리는 불가능하다.

Max 연산은 벡터 Y가 P_{N-1} 노드로 전달되면서 단계적으로 수행되기 때문에 본 연구에서 Max 연산의 병렬처리는 부분적으로 가능하다. output node는 제어하고자 하는 장치에 직접 연결할 수 있으며 비퍼지화 연산을 수행한다. 벡터 Y는 각 노드의 Min 연산 결과값이다. ALL_TO_ONE 알고리즘도 log N 스텝에 수행된다.

그림 2는 ALL_TO_ONE 알고리즘을 나타낸다. ALL_TO_ONE 알고리즘은 각 노드에서 동시에 실행된다. P_{N-1}에서의 결과값이 모이면 마지막 Max 연산 수행된다.

제안된 시스템에서는 입력과 출력 노드가 분리되어 있다. P_0는 입력값을 받아들이고 P_{N-1}는 장치에 대한 퍼지 제어의 출력부분의 역할을 수행한다. 입력과 출력 노드의 분리는 프로세서에 대한 부하를 균등하게 해주고, 불필요한 routing과 waiting을 줄여준다. 이와 같은 구조에서는 pipelining의 새로운 가능성을 제공한다.

그림 3은 위의 2가지 알고리즘에 의한 입력벡터 X와 Min 연산 결과의 벡터 Y에 대한 노드들 사이의 경로를 보여주고 있다. P_{N-1}에서의 Min 연산 결과값과 P_{N-1}을 제외한 노드들의 Min 연산 결과값인 벡터 Y에

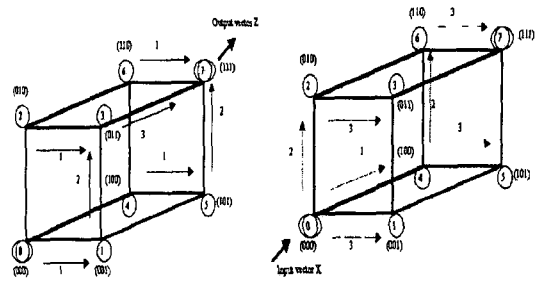

그림 3. 메시지 전송
Fig. 3. Message broadcasting

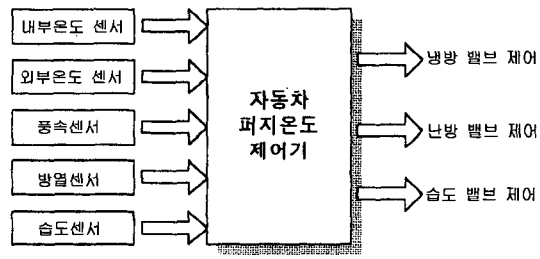

그림 4. 자동차에서 퍼지 온도 습도 제어기
Fig. 4. Fuzzy controller for Air conditioning system

대한 마지막 Max 연산이 P_{N-1}에서 수행된다. 비퍼지화는 Max 연산의 결과값으로 무게중심법을 사용하여 수행된다.

제안된 병렬 퍼지 추론 시스템에 대한 전체 수행과정은 다음과 같다.

1. 입력벡터 X를 P_0 노드에 인가한다.
2. 입력벡터 X는 ONE_TO_ALL 알고리즘에 의해 모든 노드들에 전달된다.
3. 각 노드에서는 입력벡터 X와 input fuzzy 변수 사이에 Min 연산이 수행된다. 결과값은 중간 결과값 벡터 Y에 저장된다.
4. 각 노드에서 벡터 Y는 ALL_TO_ONE 알고리즘에 의하여 P_{N-1}에 전달된다.
5. P_{N-1} 노드(output node)에서 자신의 Min 연산 결과값과 vector Y사이에 마지막 Max 연산이 수행된다.
6. P_{N-1} 노드에서 Max 연산의 결과값을 사용하여 비퍼지화 연산을 수행하여 출력벡터 Z를 구한다.

3. 병렬 퍼지 추론 모델

그림 4는 본 논문에서 제시한 병렬 컴퓨팅 환경에

서 병렬 퍼지 추론 알고리즘의 효율성을 증명하기위해 사용한 예제의 병렬 퍼지 추론 모델이다. 이 모델은 자동차의 내부 온도와 습도를 제어하기 위한 퍼지 제어 시스템으로 출력 데이터를 얻기 위해 다섯 개의 센서를 사용하고 온도 및 습도 제어를 위해 세 개의 밸브를 사용한다. 사용되는 입력변수는 내부온도센서(Inside_T), 외부온도센서(Outside_T), 풍속센서(Wind_S), 방열센서(*R*), 습도센서(*H*)로 하였고 출력변수는 냉방 밸브 제어(CV), 난방 밸브 제어(HV), 습도 밸브 제어(Hu_V)로 하였다. 즉, 퍼지규칙은 전건부의 변수 5개, 후건부의 변수 3개이다. 이러한 제어시스템은 MIMO(multiple input, multiple output)시스템으로 본 논문에서는 단지 병렬컴퓨팅환경에서의 병렬 추론 알고리즘, 효과적인 병렬성의 추출 및 속도향상을 위해 성능평가를 목적으로 사용한다.

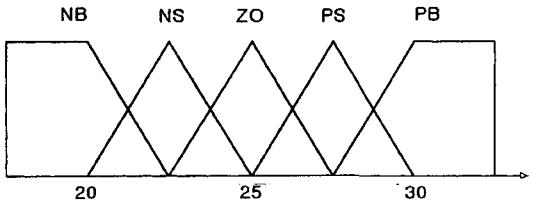

그림 5. 입력 변수에 대한 퍼지소속함수-내부 온도 센서
Fig. 5. Fuzzy membership function for input variable- inside temperature sensor

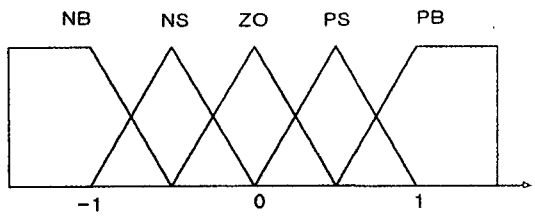

그림 6. 출력 변수에 대한 퍼지소속함수-냉방밸브
Fig. 6. Fuzzy membership function for output variables-cooling valve

각각의 입력변수와 출력변수의 퍼지 소속함수의 linguistic term은 각각 5개로 구성하였으며 그림 5, 그림 6과 같다. 이와 같이 정의된 입력변수와 출력변수의 퍼지 집합을 이용하여 표 2와 같이 제어 규칙을 구성하였다. 이것은 가능한 전체 퍼지규칙의 수 320,625개 중에서 2,048개의 퍼지 규칙만을 생성하여 성능 측정을 하였다. 퍼지 제어기의 출력은 무게 중심법에 의해 플랜트에 입력될 실수값으로 변환하였다.

3. 성능측정 및 비교

본 연구를 위하여 SGI/Cray사에서 개발한 T3E 병렬컴퓨터를 사용한다. T3E는 분산메모리 방식을 갖고 있으며, 네트워크의 구성은 3차원 Torus 구조를 갖고 있다. 각 노드마다 DEC alpha EV 5.6 마이크로 프로세서가 1개씩 장착된 128개의 노드로 구성되어 있으며, PE(processing element)당 450 MHz의 클록속도와 128Mbyte의 메모리를 가지고 있다. T3E 시스템은 128개의 User PE외에 8개의 Support PE가 추가로 장착되어 136개의 노드를 가지고 있지만, 이중 병렬처리를 위해 사용될 수 있는 노드는 128개이며, 나머지는 사용자의 명령어를 수행하는데 사용된다. 하나의 PE의 구조는 그림 7과 같다.

각 노드마다 1개의 PE, 1개의 메모리 서브 시스템이 장착되며 이들 노드들은 분산 메모리 Interconnect 네트워크로 연결된다. 메모리 서브 시스템은 물리적으로는 분산되어 있지만 논리적으로는 공유되어 있어서 모든 시스템 메모리는 단일 어드레스 주소체계를 가지며 모든 PE에서 직접 액세스가 가능하다.

표 2. 퍼지 제어 규칙
Table 2. Fuzzy control rules

control rule 1 :	if Inside_T is PB and Outside_T is PB and Wind_S is PB and R is PB and H is PB **then** CV is PB and HV is NB and Hu_V is NB
control rule 2 :	if Inside_T is PS and Outside_T is PB and Wind_S is PB and R is PB and H is PB **then** CV is PS and HV is NB and Hu_V is NB
control rule 3 :	if Inside_T is ZO and Outside_T is PB and Wind_S is PB and R is PB and H is PB **then** CV is PS and HV is PS and Hu_V is NB
	---
control rule n :	if Inside_T is NS and Outside_T is PB and Wind_S is PB and R is PB and H is PB **then** CV is ZO and HV is PS and Hu_V is NB
	---
control rule 2048 :	if Inside_T is NB and Outside_T is NB and Wind_S is NB and R is NB and H is NB **then** CV is NB and HV is PB and Hu_V is PB

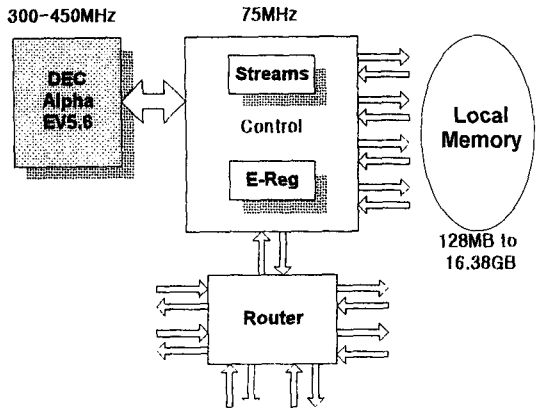

그림 7. T3E 노드의 블럭다이어그램
Fig. 7. Block diagram of a node in T3E node

```
#!/bin/csh -f
#QSUB -r carfuzzy
#QSUB -l mpp_p=32
#QSUB -lT 60
#QSUB -ro -re -eo -o carfuzzy.out
.....
ja
.....
module load mpt
f90  -o carfuzzy.exe carfuzzy.f
# -- Run the executable on 32 processors.
mpprun -n32 ./carfuzzy.exe
```

그림 8. 32개 노드로 퍼지 추론 포트란 프로그램을 실행하기 위한 NQS 스크립트
Fig. 8. NQS script that runs the fuzzy inference Fortran program by 32 nodes

T3E에는 자료전달을 위해, MPI, PVM, SHMEM의 3가지 라이브러리가 제공되고 있으나, 본 논문에서는 MPI(Message Passing Interface)를 사용한다.

본 논문에서는 16, 32개의 노드을 이용하여 퍼지제어 시스템을 병렬처리하여 벤치마크 테스팅을 하였다. 각 노드를 위한 프로그램을 FORTRAN으로 작성하고, 자료전달함수는 call MPI_ISEND(data, N, B), call MPI_RECV(data, N, B)를 사용하여 각각의 PE간의 일대일 자료 전달 또는 여러 PE간의 다대일 혹은 다대다 자료전송기능을 이용하였다. 그림 8은 NQS Batch 스크립트의 예이다. NQE(Network Queuing Environment)는 T3E 시스템 내에서 일괄작업의 제어와 실행을 제공한다. NQE에 작업을 제출(Submit)하기 위해서는 qsub 명령어를 사용한다.

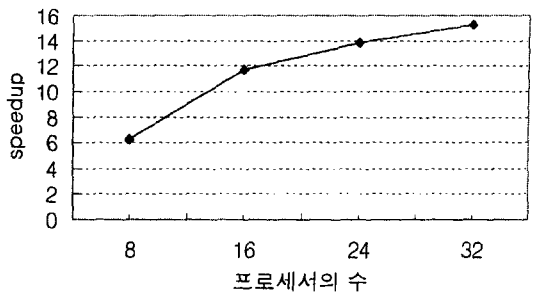

그림 9. 성능평가
Fig. 9. Performance

그림 9는 단일 프로세서 시스템에서 퍼지 추론을 수행한 결과에 비해 병렬 처리 시스템상에서 수행했을 때의 결과를 비교하여 프로세서의 수에 따른 속도 향상(Speedup)을 나타낸다. 프로세서의 수가 변화함에 따라 본 논문에서 제안한 병렬 퍼지추론 알고리즘의 성능이 얼마나 향상되었는지를 평가하기 위하여 속도 향상(Speedup)을 파라메터로 하였다.

$$Speedup = \frac{T_1}{T_n}$$

여기서, T_1은 단일 프로세서 상에서 병렬 퍼지 추론 모델의 수행시간이고, T_n은 n개의 프로세서 상에서 병렬 퍼지 추론 모델의 수행시간이다.

그림 9에서 프로세서의 수가 작은 경우에도 병렬 퍼지 추론 알고리즘의 성능향상률이 좋으며 프로세서의 수가 증가할수록 성능향상률이 점진적으로 증가함을 알 수 있다. 그러나, 노드의 수를 16개로 한 경우에 비해 32개로 한 경우의 성능 향상률이 크게 향상되지는 않았다. 이렇게, 이론상의 성능향상이 되지 않는 이유는 제안된 알고리즘을 이용하여 퍼지 규칙 또는 데이터를 각 분산된 메모리에 전송하고 그 결과를 다시 join하는 메시지 패싱 시간이 각 PE들이 실제 프로그램을 실행시키는 시간에 비해 더 많은 시간을 필요로 하기 때문이다. 즉, 16개의 노드을 사용하는 경우, 각 PE당 퍼지 추론 규칙이 128개가 할당되고, 32개의 노드를 사용하는 경우, 64개의 퍼지 추론 규칙만이 할당되므로 실제 추론처리 시간의 차이는 아주 작다. 그러므로, 퍼지 제어 규칙의 수에 따라 적절한 노드 수의 선택이 전체 시스템의 성능향상의 요인이 될 수 있음을 알 수 있다. 실험결과 이러한 병렬 환경에서 32개의 노드를 사용한 경우 약 600 MFLIPS의 성능을 얻었다. 결과적으로는, 퍼지 추론 모델을 병렬화 하는 것이 더 효과적임을 알 수 있으며, 이를 통해 제안된 병렬 퍼지 추론 기법이 병렬처

리에 적합함을 알 수 있다.

4. 결론 및 향후 연구

본 논문에서는 병렬컴퓨팅 환경에서 대용량의 퍼지 데이터 또는 많은 수의 퍼지 규칙을 갖고 있는 시스템을 위한 효율적인 병렬퍼지 추론방법을 제안하였다. 분산된 메모리 시스템과 메시지 패싱 방식을 갖는 병렬컴퓨터에서 ONE_TO_ALL 알고리즘과 ALL_TO_ONE 알고리즘을 사용하여 퍼지 규칙 또는 퍼지 데이터들을 병렬로 추론한다. 제안된 방법은 실시간에 고속추론을 요하는 시스템이나 MIMO와 같이 전건부, 후건부에 많은 변수를 갖고 있는 시스템에 특히 효율적이다. 또한 입력노드와 출력노드를 분리하여 데이터의 전송에 따른 파이프라인 처리도 가능하다. 향후의 연구로서는 대용량의 퍼지 데이터베이스를 위한 병렬컴퓨팅 환경에 적합한 퍼지질의 처리 시스템의 구현에 대한 연구가 필요하다.

감사의 글

본 논문은 2000년도 한남대학교 교내 학술연구비 지원에 의해 수행되었습니다.

참고문헌

[1] A. Jaramillo-Botero, "Parallel, high-speed PC fuzzy control," *IEEE Micro,* p. 63, Dec. 1995.

[2] A. P. Ungering and K. Goser, "Architecture of a 64-bit fuzzy inference processor," *Proc. FUZZ-IEEE WCCI*, Vol. 26, No. 6, pp. 1776-1780, 1994,

[3] Sang Gu Lee and K. Akizuki, "Design of an effective parallel architecture for fuzzy information processing," *Trans. of IEE Japan*, Vol. 118-C, No. 7/8, pp. 1190-1195, 1998.

[4] Sang Gu Lee and K. Akizuki, "A parallel inference method of fuzzy rules by using Transputers," *Trans. of IEE Japan*, Vol. 118-C, No. 7/8, pp. 1176-1182, 1998.

[5] Y. D. Kim and H. Lee-Kwang, "High speed flexible fuzzy hardware for fuzzy information processing," *IEEE Trans. on Syst., Man, Cybern.,* Vol. 27, No. 1, pp. 45-56, Jan. 1997.

[6] M. Sasaki, F. Ueno, and T. Inoue, "7.5 MFLPS fuzzy microprocessor using SIMD and logic-in-memory structure," *Proc. IEEE Int. Conf. on Fuzzy Systems*, pp. 527-534, 1992.

[7] T. Chiueh, "Optimization of fuzzy logic inference architecture," *IEEE Computer Magazine,* pp. 67-71, May 1992.

[8] Sang Gu Lee, "Parallel Inference Method of Fuzzy Rules," *Journal of Electrical Engineering and Information Science,* Vol. 4, No. 3, pp. 357-363, 1999. 6.

[9] 이상구, "퍼지 정보처리를 위한 효율적인 병렬 퍼지 아키텍처의 설계", 한국정보과학회 논문지(C), Vol. 4, No. 4, pp. 567-574, 1998. 8.

[10] 김진일, 이상구외 2인, "병렬 컴퓨팅 환경에서의 대용량 퍼지 추론," 한국 퍼지 및 지능시스템학회 2000 춘계학술대회 발표논문집, Vol. 10, No. 1, pp. 13-16, 2000. 5.

김 진 일 (Jin-Il Kim)

1991년 : 한남대학교 컴퓨터공학과 학사
1993년 : 한남대학교 컴퓨터공학과 석사
2000년 : 한남대학교 컴퓨터공학과 박사
관심분야 : 컴퓨터구조, 퍼지 제어, 병렬 처리

이 상 구 (Sang-Gu Lee)

제 9권 5호 참조