論文 89-26-5-10

다단상호 접속망의 Simulation Algorithm 개발에 관한 연구

(A Study on the Simulation Algorithm of the Multistage Interconnection Networks)

李 殷 卨,* 金 大 浩,* 林 濟 鐸*

(Eun Seol Lee, Dae Ho Kim and Chae Tak Lim)

要 約

다단상호 접속망의 성능평가를 위해 접속망을 모델링하는 방법과, 시뮬레이션 알고리듬을 제안하였으며, 이를 프로그램으로 구현하였다.

특히 여러개의 입력단에서 접속 요구가 동시에 발생한 경우를 처리하기 위해 상태변수를 설정하였으며, 각 단의 처리과정을 추적하기 위해 접속망 전체에 대한 정보를 갖는 자료 구조를 제안함으로써 순차적 처리를 하는 기존의 컴퓨터에 적용할 수 있었다.

이 방법에 의하면 다단상호 접속망에 대한 성능평가는 복잡한 수학적 해석에 의존하지 않고서 시뮬레이션으로 결과를 산출하게 되었다.

Abstract

To estimate a performance of MINs, a network modeling method and a simulation algorithm are proposed, and this algorithm is programmed by C language.

Especially, state variables are defined to process many concurrent requests at inputs and a data structure, which contains network informations, is proposed to keep track of each stage.

This simulation can be applied to computers which are designed for sequential processing.

This method can be used to estimate a performance of MINs instead of using complex mathematical method.

*正會員, 漢陽大學校 電子工學科

(Dept, of Elec. Eng., Hanyang Univ.)

接受日字 : 1989年 2月 15日

(※이 연구는 1988년도 문교부 지원 한국학술진흥재단의 자유공모과제 학술연구조성비에 의하여 연구되었음.)

I. 서 론

대량의 데이타를 고속으로 처리하기 위해서는 병렬처리 시스템의 개발이 필수적이며, 특히 병렬처리 시스템을 구성하는 많은 수의 프로세서와 메모리 혹은 프로세서와 프로세서를 연결하는데 있어서 하드웨어의 복잡도를 감소시키면서 충분한 정보교환 능력을

갖는 다단상호 접속망(MIN multistage interconnection networks)의 개발은 병렬처리 시스템 개발의 관건이 되고 있다.

MIN에 대한 연구는 1970년대 중반부터 발표되었으며 단일 경로를 갖는 접속망으로는 Generalized Cube 접속망[1], Omega 접속망[2], Indirect Binary n-cube 접속망[3], Baseline 접속망[2] 등이 있으며, 이중에서 가장 기본이 되는 것은 그림 1 의 Generalized Cube 접속망과 그림 2 의 Omega 접속망이다.

그리고 입력에서 출력까지 여러개의 경로를 갖는 다중 경로 접속망으로는 Extra Stage Cube 접속망[4], Benes 접속망[5], ADM (augmented data manipulator)[1], IADM (inverse ADM)[1], Gamma 접속망[6], F-접속망[7] 등이 있다.

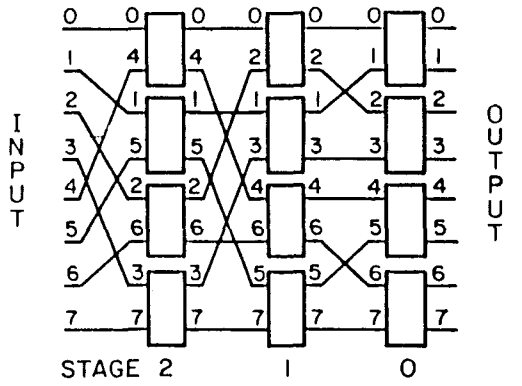


그림 1. N=8일 때 Generalized Cube 접속망
Fig. 1. Generalized cube network for N=8.

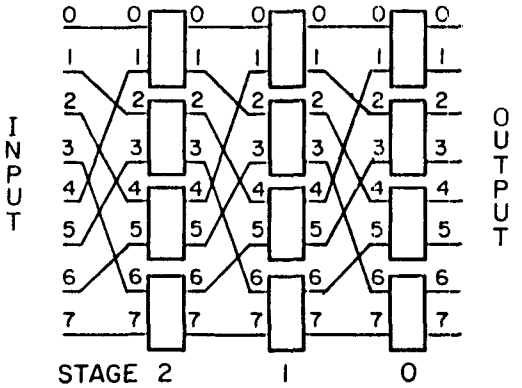


그림 2. N=8일 때 Omega 접속망
Fig. 2. Omega network for N=8.

이와 같은 MIN들의 성능을 평가하는 방법에는 수학적 해석 방법과 시뮬레이션에 의한 방법이 있는데, 이중 수학적 해석 방법은 프로세서나 메모리의 수가 증가함에 따라 상당히 어렵고 복잡해지기 때문에 시뮬레이션에 의해 처리되어야만 한다.[11,12] 그러나 MIN을 시뮬레이션하기 위해서는 접속 요구, 경로 설정, 데이타 전송등과 같이 병렬로 처리되어야 하는 사항들을 순차적 처리를 하는 기존의 컴퓨터에 적용시킬 수 있도록 해야만 한다.

그러므로 본 연구에서는 이와 같은 문제를 해결하기 위하여 각 스위치에 상태변수를 설정하고, 접속망에 대한 정보를 갖는 자료 구조를 제안하였으며, 서로 다른 링크 연결 방식을 갖고 있는 MIN을 시뮬레이션 할 수 있도록 접속망 모델링 방법을 제안하였고, 경로 설정 태그를 사용해 경로를 설정하는 알고리듬과 시뮬레이션 알고리듬을 제안한 후 이를 C언어로 프로그램화 하였다.

본 연구에서 제안한 시뮬레이션 알고리듬을 여러 종류의 MIN중 가장 기본이 되는 Generalized Cube 접속망과 Omega 접속망에 적용하여 실질적 활용이 가능함을 입증하였다.

II. 접속망 모델링

일반적으로 MIN의 특성은 각 단(stage)을 연결하는 방법, 스위칭 소자의 입·출력 단자수, 경로 설정 태그의 생성 방법 및 경로 설정방법 등에 의하여 결정된다. 따라서 임의의 MIN을 시뮬레이션하기 위해서는 이와 같은 내용들을 시뮬레이션에 적합한 형태로 모델링을 해야 하는데, 먼저 각 스위칭 소자의 입·출력 단자에 의해 각 단을 연결하는 접속망 모델링에 대해 알아보도록 하겠다. 각 단을 구성하는 스위칭 소자들의 연결 방법은 접속망마다 서로 다르기 때문에 이를 모델링하기 위해서는 다음과 같은 사항을 정의해야 한다.

(1) MIN을 구성하는 단의 갯수를 산출한다.
(2) 각 단에 번호를 부여한다.
(3) 각 단을 구성하는 스위칭 소자의 갯수를 산출한다.
(4) 각 단의 각 스위칭 소자를 연결하는 링크 관계를 구한다.

이와 같은 사항을 일반적인 Generalized Cube 접속망과 Omega 접속망에 적용시켜 접속망을 모델링 하도록 하자.

1. Generalized Cube 접속망

(1) 입력 단자가 N개일 경우 단의 갯수 $S=LOG_2(N)$ 이다.

(2) 단 번호는 우측에서 좌측으로 0부터 S-1까지 순차적으로 할당한다. (S-1, S-2, …, 1, 0)

(3) 각 단은 N/2개의 스위칭 소자로 구성된다. (2×2 스위치로 구성)

(4) 링크 연결

(가) 임의의 k단→k-1단으로 연결(단, 0<k<S)

(a) 스위치 번호 산출

입력 단자 번호(In_term_no)의 k-1번째 Bit를 제외한 나머지 Bit들의 합이 k-1단에서의 스위치 번호가 된다.

(예) In_term_no(입력 단자 번호)
=I(s-1) I(s-2)…I(k) I(k-1)…
I(1) I(0)(단 I(k)=0 or 1)

여기서 k-1번째 Bit를 제외한 나머지 Bit로 스위치 번호를 산출하면 다음과 같은 식으로 표현할 수 있다.

```
Switch_number_comp( )
{
  int temp1, temp2, Switch_no;
    temp1=(In_term_no>>k)<<(k-1)
    /*>> : shift right 연산자, << : shift left 연산자*/
    temp2=In_term_no & (2**(s-1)-1))
      /*& : Bit단위 AND 연산자*/
    Switch_no=temp1 | temp2
      /*| : Bit단위 OR 연산자*/
}
```

(b) 해당 스위치의 In_term_no 선택(k-1번째 Bit를 조사)

```
if(In_term_no(k-1))
  select 하단부(In_term_no(k-1)=1인 경우)
else
  select 상단부(In_term_no(k-1)=0인 경우)
```

그림 3은 위의 (b)항을 이용하여 다음단의 상, 하단부 링크를 선택하는 경우이다.

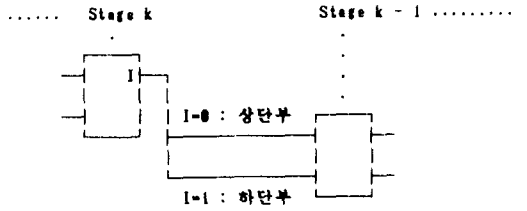


그림 3. 다음 단의 링크 선택
Fig. 3. Select link of next stage.

2. Omega 접속망

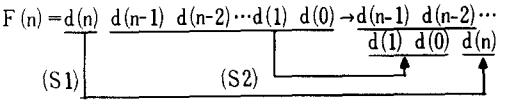
(1) 입력 단자가 N개일 경우 단의 갯수 $S=LOG_2(N)$ 이다.

(2) 단 번호는 우측에서 좌측으로 0부터 S-1까지 순차적으로 할당한다. (S-1, S-2, …, 1, 0)

(3) 각 단은 N/2개의 스위칭 소자로 구성된다. (2×2 스위치로 구성)

(4) 링크 연결

각 단은 1-SHUFFLE로 연결되므로 일반적인 표현식은 다음과 같다.



1-SHUFFLE 전의 표현식 : $F(n)=S1+S2$
$(S1=d(n)*2^{**}n)$

1-SHUFFLE 후의 표현식 : $F(n)=(S2*2) \bmod N + d(n)$

즉,

$F(n)\%(2^{**}n) \rightarrow S2$

$F(n)/(2^{**}n) \rightarrow d(n)$

(가) 임의의 k단→k-1단으로 연결 (단 0<k<S)

$d(n)=Out\_term\_no/Mod\_base$ (단, $Mod\_base=2^{**}n$)

k단의 출력 단자 번호

(a) 스위치 번호 산출

$Switch\_no=(2*S2+d(n))/2$
$=In\_term\_no/2$
(각 스위치에 할당된 번호는 0부터 순차적으로 부여)
$S2+d(n)/2$→연산 결과 정수값은 S2이므로 $Input_no\%(2^{**}n) \rightarrow S2$와 동일하다.

(b) 해당 스위치의 In_term_no 선택

```
In_term_no=2*S2+d(n)
if(In_term_no%2)
  select 하단부(In_term_no가 홀수인 경우)
else
  select 상단부(In_term_no가 짝수인 경우)
```

이와 같은 방법으로 접속망을 모델링할 수 있으며, 다음 단계로는 시뮬레이션시 데이타가 전송될 경로를 설정하는 방법에 대해 알아 보기로 하겠다.

Ⅲ. 경로 설정

MIN에서 경로를 설정하기 위해서는 경로 설정 태

그를 사용하는데, 일반적으로 사용하는 방식에는 수신지 태그와 Exclusive OR 연산에 의한 태그 방법 등이 있다. 수신지 태그란 입력 단자에서 데이타를 전송하고자 하는 출력 단자의 번지를 그대로 사용하는 방법이고, Exclusive OR에 의한 태그는 입력 단자의 번지와 출력 단자의 번지를 Exclusive OR 연산에 의해 산출된 값을 사용하는 방법인데, 여기서는 수신지 태그를 이용하기로 한다. 경로 설정 태그를 수식으로 표현하면, Rt=수신지 번지이므로 Rt=D(s-1) D(s-2)…D(k) D(k-1)…D(1) D(0) (단 D(k)=0 or 1)와 같으며, 각 단의 각 스위치 소자에서는 해당 Bit를 조사하여

```
if(Rt(i))
  select 하단부
else
  select 상단부
```

의 경로를 선택하면 된다.

Ⅳ. 시뮬레이션

MIN에서는 모든 실행이 병렬처리 되므로 기존의 컴퓨터를 사용하여 시뮬레이션을 하기 위해서는 병렬처리 효과를 얻을 수 있도록 다음과 같은 조건을 부여 하였으며, 성능 평가를 위한 요소로는 RST(request service time)와 Throughput을 이용하였다.

1. 시뮬레이션 조건

(1) 프로세서의 갯수(=입출력단 갯수)는 사용자가 선택한다.
(2) 데이타 전송 접속 요구(request)가 발생할 확률은 사용자가 선택하며, 입력단과 출력단이 선택될 확률은 동일하게 한다.
(3) 입력단 번호와 출력단 번호는 Random하게 처리한다.
(4) 블로킹 발생시는 역추적 하는 것으로 한다.
(5) MIN의 진행상태는 경로 설정중, 데이타 전송중, 역추적중의 3가지 상태로 구성된다.
(6) 1개의 실행 싸이클(cycle)은 경로 설정시, 역추적시 1개의 단을 이동하는 시간이며, 데이타 전송량은 1개의 실행 싸이클 동안에 1개의 데이타를 입력에서 출력까지 보낼 수 있는 단위다.
(7) 총 실행 싸이클수는 사용자가 선택한다.
(8) 역추적시는 할당된 스위치를 해제시킨다.
(9) 스위칭 방식은 회선 스위칭(circuit swithcing) 방식을 사용한다.
(10) RST(request service time) =

$$\frac{\text{경로를 설정하는데 소요되는 총 시간(cycle)}}{\text{총 Request 발생수}}$$

RST는 입력단에서 접속 요구(request)가 발생하였을때 경로 설정에 소요된 평균 시간(cycle)을 나타낸다.

(11) Throughput =

$$\frac{\text{전송된 총 데이타 량}}{[N^*(\text{Request 발생 확률})]^*\text{총 실행 싸이클수}}$$

Throughput은 접속 요구 처리에 소요된 총 시간 중 데이타 전송에 소요된 시간의 비율을 나타낸다.

일반적으로 접속망에 대한 성능 평가는 수학적 방법과 시뮬레이션에 의한 방법이 모두 사용되지만 수학적인 방법은 많은 부분을 간략화하여 적용하므로 결과치에 대한 정확성이 결여 될 수 있으므로 시뮬레이션에 의한 방법이 이와 같은 문제점을 해결할 수있다.

(14), (15)의 방법에서는 블로킹 발생시 접속 요구를 포기하는 방식과 경로 설정시 데이타 전송 시간을 고려하지 않고 적용하였지만, 본 논문에서는 블로킹 발생시 재 접속 요구를 하도록 하였으며, 데이타 전송 시간을 고려하여 시뮬레이션을 하였다.

2. 시뮬레이션 알고리듬

시뮬레이션을 하기 위해서 앞에서 제안한 접속망 모델링 방법과 시뮬레이션 조건을 C언어를 사용하여 프로그램화 하였으며, 모두 4개의 모듈로 구성되어 있다. 그림 4는 본 프로그램의 구성도를 나타내며, 그림 5는 MAIN.C 프로그램의 일부를 나타낸다. 각 프로그램의 기능은 다음과 같다.

(1) MAIN.C : 전체 프로그램을 관리하며, 접속망의 모델링과, 각 스위치에 필요한 정보 Table을 생성한다.
(2) PATHPROC.C : 경로를 설정하는 기능과, 역추적, 데이타 전송 등의 기능을 수행한다.
(3) DATAIN.C : Key Board로 부터의 입력 및 Menu로 부터 선택된 데이타를 처리한다.
(5) ALLSUB.C : Video, Key Board 등 각종 인터럽처리 기능을 한다.

그림 6은 경로 설정 및 성능 평가를 위해 필요한 데이타 구조를 나타낸다. req_stat는 해당 입력단에서 데이타 전송 요구가 발생하였는지의 여부를 나타내며, input_no는 입력단의 번지를 output_no는 입력단에서 데이타를 전송하고자 하는 출력단의 번지를 나타낸다. tag_no는 경로 설정용 태그로 각 단에서는 해당 태그를 조사하여 스위치의 출력단을 결정한다. tot_stat는 현재 접속망의 상태 즉, 각 단의 스위치들이 데이타 전송중, 경로 설정중, 혹은 역 추적중인 상태를 나타내며 path_stage_count는 현재 접속망의 상태를 처리하는 과정이 어느 단까지 수행 되었는가를 나타낸다.

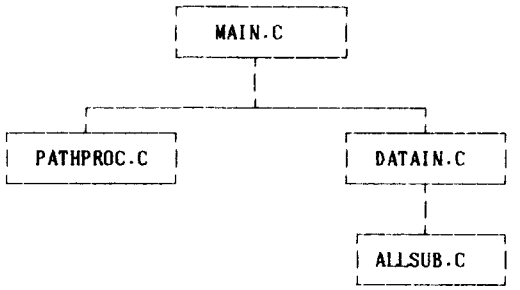


그림 4. 프로그램의 구성도
Fig. 4. Program configuration.

```
main()
{
  long i;
  while(1){
    if(menu_selection() == -1)          /*항목 선택 및 데이타 입력*/
      exit(0);
    initial_request();                  /*각 변수의 초기화,
                                          태그, Table 생성*/
    for(i=0;i<C_count;i++){             /*링크 단자 할당 정보 갱신*/
      next_current();
      status_end_process();             /*경로 설정, 데이타 전송,
                                          역추적 등 종료 상태 처리*/
      path_datatrance_process();        /*경로 설정, 데이타 전송,
                                          역추적 등 진행 상태 처리*/
    }
    write_total();                      /*최종 결과 디스플레이*/
  }
}
```

그림 5. 주 프로그램의 일부
Fig. 5. Partition of main program.

```
typedef  struct   stage_info    {    /*network structrue*/
         TINY     req_stat;          /*request 발생여부*/
         COUNT    input_no;          /*입력 번지*/
         COUNT    tag_no[MAX_STAGE]; /*각 단에 사용되는 경로 설정
                                       태그*/
         COUNT    output_no;         /*출력 번지*/
         TINY     tot_stat;          /*접속망의 상태*/
         COUNT    path_stage_cnt;    /*현재 처리중인 단번호*/
         LONG     path_setup_no;     /*경로 설정이 완료된 횟수*/
         LONG     tot_setup_cycle;   /*경로 설정에 소요된 싸이클수*/
         COUNT    data_trans_cycle;  /*데이타 전송 싸이클수*/
         LONG     data_tran_no;      /*데이타 전송이 완료된 횟수*/
         LONG     tot_data_trans_cycle; /*데이타 전송에 소요된 싸이클수*/
         LONG     request_no;        /*request 횟수*/
};
typedef  struct   proc_info     {    /*processor status table*/
         TINY     cur_proc_st;       /*현 상태*/
         TINY     next_proc_st;      /*다음 상태*/
};
```

그림 6. 시뮬레이션을 위한 데이타 구조
Fig. 6. Data structure for simulation.

path_setup_no는 프로그램의 종료시까지 경로 설정이 완료된 횟수를 나타내며, tot_setup_cycle은 프로그램의 종료시까지 경로 설정을 위해 소요된 총 싸이클수가 된다.

data_trans_cycle은 1개의 데이타를 입력단에서 출력단까지 전송하는데 필요한 시간을 나타내며, data_trans_no는 데이타 전송이 완료된 횟수를, tot_data_trans_cycle 은 프로그램이 종료할 때까지 데이타 전송에 소요된 시간을 나타내며, request_no는 데이타 전송요구가 발생한 횟수를 나타낸다.

그리고 경로를 설정해 나가는 중에 동일한 스위치의 출력단을 요구하는 문제, 혹은 이미 할당된 출력단을 요구하는 문제 등을 처리하기 위해 각 스위치에는 현재의 상태와 다음 상태를 나타내는 상태 변수를 할당하여 다음과 같이 정의하였다.

현 상태	다음 상태	처 리 내 용
FREE	FREE	경로 설정 가능
FREE	ALLOCATE	경로 설정 가능 (우선 순위에 의해 할당)
ALLOCATE	FREE	역추적중이므로 경로 설정 불가능
ALLOCATE	ALLOCATE	데이타 전송중이므로 경로 설정 불가능

그러므로 이 상태 변수를 이용해 경로 설정 여부는 그림 7 과 같은 순서에 의해 처리된다.

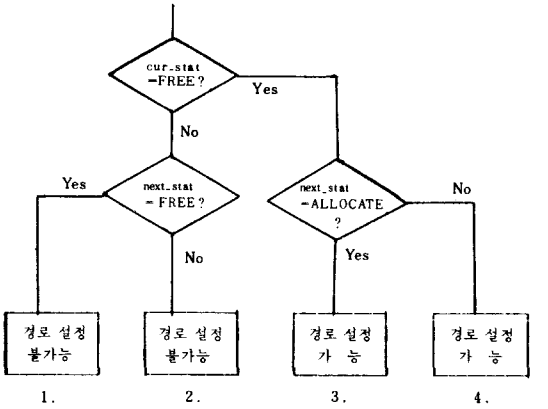


그림 7. 경로 설정 순서도
Fig. 7. A flowchart for path setup decision.

그림 7 의 1과 2의 경우는 역추적을 해야하며, 3의 경우는 우선 순위에 따라 경로 설정을 해야 한다. 이와 같은 내용을 가장 일반적인 Omega 접속망과

표 1. Omega 접속망과 Generalized Cube 접속망의 RST
Table 1. RST for Omega network & Generalized Cube network.

입출력 단수 (N)	Request 확률 (r)	Data				Cycle (d)			
		10		30		50		100	
		OMEGA	GCN	OMEGA	GCN	OMEGA	GCN	OMEGA	GCN
16	0.2	4.73	4.82	6.58	6.80	7.78	8.21	8.90	9.24
	0.5	7.92	9.38	17.53	17.61	27.76	28.62	47.67	49.03
	1.0	21.02	21.03	44.35	45.73	72.10	71.90	122.04	122.28
32	0.2	6.69	7.32	10.20	10.64	12.58	12.26	21.87	24.10
	0.5	11.19	13.16	25.64	26.27	35.51	37.26	62.88	67.58
	1.0	27.55	28.32	57.58	58.34	86.55	87.60	161.19	161.46
64	0.2	8.11	9.03	13.55	12.73	17.90	17.82	31.53	27.88
	0.5	14.18	17.56	32.62	31.67	47.74	47.52	81.09	78.92
	1.0	33.77	34.06	68.70	67.32	101.42	99.14	181.04	178.98
128	0.2	11.75	11.80	17.64	18.46	24.84	24.09	34.42	36.98
	0.5	21.83	22.20	39.70	39.60	56.76	58.84	99.12	98.58
	1.0	40.85	41.19	78.70	78.78	114.91	114.13	204.37	202.67

표 2. Omega 접속망과 Generalized Cube 접속망의 Throughput
Table 2. Throughput for Omega network & Generalized Cube network.

입출력 단수 (N)	Request 확률 (r)	Data				Cycle (d)			
		10		30		50		100	
		OMEGA	GCN	OMEGA	GCN	OMEGA	GCN	OMEGA	GCN
16	0.2	0.68	0.68	0.82	0.82	0.87	0.86	0.91	0.91
	0.5	0.54	0.51	0.63	0.62	0.65	0.63	0.70	0.66
	1.0	0.32	0.32	0.39	0.39	0.41	0.41	0.44	0.44
32	0.2	0.60	0.58	0.75	0.73	0.80	0.80	0.86	0.81
	0.5	0.46	0.43	0.54	0.52	0.58	0.57	0.61	0.59
	1.0	0.26	0.26	0.34	0.34	0.37	0.36	0.37	0.37
64	0.2	0.54	0.53	0.69	0.70	0.74	0.74	0.76	0.77
	0.5	0.40	0.36	0.48	0.48	0.51	0.51	0.55	0.55
	1.0	0.22	0.22	0.30	0.30	0.33	0.33	0.35	0.35
128	0.2	0.46	0.46	0.62	0.62	0.67	0.67	0.74	0.73
	0.5	0.31	0.31	0.43	0.43	0.46	0.45	0.50	0.50
	1.0	0.19	0.19	0.27	0.27	0.30	0.30	0.32	0.32

Generalized Cube 접속망에 적용하여 시뮬레이션을 하여 얻은 결과가 표 1과 표 2 이다.

표 1 은 Request Service Time을 나타내며, 표 2 는 Throughput을 나타낸다. Generalized Cube 접속망과 Omega 접속망은 링크 방식에서만 차이가 있는 동일한 접속망 구조를 갖고 있으므로 시뮬레이션 결과치에 큰 차이는 없지만 Omega 접속망이 RST, Throughput에서 약간 우수함을 알 수 있다.

V. 결 론

MIN의 성능 평가를 위해 접속망을 모델링하는 방법과 시뮬레이션을 위한 알고리듬을 제안하였고, 이를 C언어로 프로그램화 하였다.

특히 여러개의 입력단에서 동시에 접속요구가 발생한 경우 이를 병렬 처리하기 위해 상태 변수를 설정하므로써 순차적 처리를 하는 기존의 컴퓨터에 적용할 수 있었으며, 접속망 모델링은 2입력×2출력 스위치를 사용하는 단일경로 MIN중에서 가장 기본이 되는 Generalized Cube 접속망과 Omega 접속망에 적용하였다.

또한 링크 연결 부분과 경로 설정 태그의 처리부분을 변환하므로써 3입력×3출력, 4입력×4출력, 혹은 여러가지 크기(size)를 갖는 스위치들이 혼합된 다중경로 MIN에도 적용이 가능하게 되었다.

그리고 입력 단자수, 총 실행 싸이클수, 접속 요구의 발생 확률, 데이타 전송량을 사용자가 입력할 수 있도록 함으로써 시뮬레이션의 일반성을 높였다. 이로써 MIN에 대한 성능 평가는 복잡한 수학적 해석 방법에 의존하지 않고서도 본 연구에서 제안한 시뮬레이션을 사용함으로써 그 결과를 쉽게 산출할 수 있게 되었으며, 단일 경로 MIN뿐만 아니라 다중 경로 MIN에도 활용할 수 있게 되었다.

參 考 文 獻

[1] H.J. Siegel, Interconnection Network for Large-Scale Parallel Processing, Lexington book, 1985.

[2] C.L. Wu and T.Y. Feng, "On a class of multistage interconnection network," *IEEE Trans. Computer*, pp. 694-702, Aug. 1980.

[3] M.C. Pease, "The indirect binary n-cube microprocessor array," *IEEE Trans. Computers*, pp. 458-473, May 1977.

[4] G.B. Adams and H.J. Siegel, "The extra stage cube: a fault-tolerant interconnection network for supercomputer." *IEEE Trans. Computers*, pp. 443-454. May 1982.

[5] D. Nassimi and S. Sahni, "A self-routing benes network and parallel permutation algorithm," *IEEE Trans. Computers,, pp.* 241-249, May 1981.

[6] D.S. Parker and C.S. Raghavendra, "The gamma network," *IEEE Trans. Computers*, pp. 367-373, April 1984.

[7] G.B. Adams and D.P. Agrawal and H.J. Siegel, "Fault-tolerant multistage interconnection network," *IEEE Trans. Computers*, pp. 14-27, June 1987.

[8] McMillen, R.J. and Siegel, H.J. "Routing schemes for the augmented data manipulator network in an MIMD system," *IEEE Trans. Computer*, pp. 184-196, Dec. 1982.

[9] H.J. Siegel and R.J. McMillen, "The multistage cube:a versatile interconnection network," *IEEE Computer*. vol. 14, pp. 66-76, Dec. 1981.

[10] H.J. Siegel and R.J. McMillen, "Using the augmented data manipulator network in PASM," *IEEE Computer*. vol. 14, pp. 25-33, Feb. 1981.

[11] A.L. Overving, "The simulation of the generalized cube interconnection network," Purdue Univ. M.S., 1982.

[12] D.M. Dias and J.R. Jump, "Analysis and simulation of buffered delta networks,". *IEEE Trans. Computer*, pp. 273-282, April, 1981.

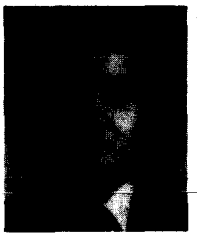
[13] C.P. Kruskal and M. Snir, "The performance of multistage interconnection networks for multiprocessor," *IEEE Trans. Computer*, pp. 1091-1098, Dec. 1983.

[14] J.H. Patel, "Processor-memory interconnections for multiprocessors," 6th Symp Computer Arch., 1979, pp. 168-177.

[15] J.H. Patel, "Performance of processor memory interconnections for multiprocessor," *IEEE. Trans. Computer* pp. 771-780, Oct. 1981.

[16] 김대호, 임제탁, "다중 경로를 갖는 다단상호 접속망에 관한 연구", 전자공학회 논문집, 제 25권 제 5 호, pp. 70-77. 5. 1988. *

著 者 紹 介



李 殷 鳥 (正會員)

1979年 2月 한양대학교 전자공학과 졸업 학사학위 취득. 1981年 2月 한양대학교 대학원 전자공학과 졸업 석사학위 취득. 1987年 8月 한양대학교 대학원 전자공학과 박사과정 수료. 1981年 12月～1984年 9月 육군종합행정학교 교수부 전산학처 교관 및 프로그램작성 장교 근무. 1985年 7月～1989年 2月 Software개발 회사 근무. 1986年 3月～1989年 2月 안양공업전문대학 전자계산학과 산업체 전임. 1989年 3月～현재 안양공업전문대학 전자계산학과 전임. 주관심분야는 컴퓨터 아키텍처, 병렬처리 시스템 등임.

金 大 浩 (正會員) 第25卷 第8號 參照

현재 안양공업전문대학 전자계산학과 교수

◆

林 濟 鐸 (正會員) 第25卷 第8號 參照

현재 한양대학교 전자공학과 교수