論文 95-4-4-1

버퍼를 부가한 Dual 네트워크의 시뮬레이션 및 성능평가

正會員 崔 昌 勳**, 李 世 亨*, 金 聖 天**

Simulation and Performance Evaluation of Buffered Dual Network

Chang Hoon Choi**, Se Hyeong Lee*, Sung Chun Kim** Regular Members

본 연구는 과학재단 연구비지원에 의한 것임.

要 約

Dual 네트워크는 Gamma 네트워크와 비교할 때 하나 적은 스테이지수를 가지며 오류 허용 방법이 간단하기 때문에 패킷을 전달하는데 소요되는 시간이 적게 걸린다. 특히 중복 경로의 수도 Gamma 네트워크보다 많기 때문에 한 스테이지에서 다중 오류가 발생했을 때에 Gamma 네트워크는 이를 허용하지 못하는 경우가 많은 반면 Dual 네트워크는 다중 오류를 허용할 수 있다.

본 논문에서는 Dual 네트워크의 성능을 평가하기 위해 시뮬레이터를 개발하였다. 이 시뮬레이터는 다중버퍼(multi-buffered)를 가진 네트워크를 시뮬레이트할 수 있고 오류 모델은 스위칭 소자 오류를 가정하며 스위칭 방법으로는 패킷 스위칭을 사용한다. Gamma 네트워크는 Dual 네트워크와 거의 비슷한 스위치 복잡도(switch complexity)를 갖기 때문에 Dual 네트워크의 비교대상이 되었으며 시뮬레이터를 이용하여 성능을 비교한 결과 Dual 네트워크의 성능이 Gamma 네트워크보다 우수함을 입증하였다.

ABSTRACT

The Dual network has one less stage than the Gamma network and the fault tolerance scheme is simpler, so the time required for transfer of packet is faster than that of Gamma network. Especially, the Dual network allows multiple faults at one stage due to the many redundant paths, but the Gamma network does not.

In this paper, we developed a simulator to evaluate performance of the Dual network. This simulator simulates the multi-buffered networks using packet switching and the switch faults are assumed as fault model. The performances of the Dual network and the Gamma network are compared because the Gamma network has a corresponding switch complexity with the Dual's. As a result, it is verified by our simulator that the performance of the Dual network is better than that of the Gamma network

* 쌍용컴퓨터 시스템연구소
SsangYoug Computer System Co.
Computer System R&D Ins.
** 서강대학교 전자계산학과
論文番號 : 9487-0318
接受日字 : 1994年 3月 18日

I. 서 론

병렬처리 시스템(Parallel processing system)에서 방대한 자료의 전송과 교환을 위해 가장 필수적인 것은 다수 개의 프로세서와 다수 개의 메모리 모듈을 연결하는 상호연결 네트워크(Interconnection Network)이다. 따라서 상호연결 네트워크를 구성하고 있는 어떤 요소(component)의 오류(fault)는 전체 시스템의 성능과 신뢰도(reliability)에 심각한 영향을 끼칠 수 있기 때문에 이에 대한 연구가 많은 연구자에 의해서 꾸준히 진행되어 왔다[1,2,8,10,13,14].

병렬처리 시스템에서 사용되는 상호연결 네트워크의 종류에는 버스(bus)구조와 크로스바 스위치(crossbar switch), 그리고 다단계 상호연결 네트워크(Multistage Interconnection Network, 이하 MIN이라 표기함)가 있는데 MIN은 효율적인 성능과 적절한 하드웨어 복잡도(hardware complexity)를 갖기 때문에 주로 사용되고 있다. 현재까지 알려진 MIN은 Delta[6], Omega[11], Baseline[16]등 많은 종류가 있으며 오류를 허용(tolerant)할 수 있는 MIN도 Extra stage cube network[2], Gamma network[14] 등외에 여러가지가 발표되었다[8,10]. 대개 오류를 허용할 수 없는 NxN MIN은 기본적으로 2x2 크기의 스위칭 소자와 $\log_2 N$개의 스테이지를 가지며 각 스테이지는 N/2개의 스위칭 소자로 구성이 되어 있는데 이런 종류의 MIN은 임의의 근원지(source)와 목적지(destination)를 연결하는 경로(path)가 오직 하나만 존재하기 때문에UPP(Unique Path Property) MIN이라고 한다[9]. 기본적으로 이러한 UPP MIN을 이용해서 오류를 허용할 수 있는 능력을 가지게 하기 위해서는 스위칭 소자의 크기를 증가시키고, 스테이지의 수를 증가시키며 때로는 각 스테이지를 구성하는 스위칭 소자의 수도 증가시킨다. 이렇게 만들어진 MIN을 MMIN(Multiple-path MIN)이라고 부르고 MMIN은 UPP MIN에 비해 높은 성능과 신뢰성을 제공[10]하며 본 논문에서는 MMIN중에서 Gamma 네트워크와 Dual 네트워크를 선택하여 이들의 성능을 분석한다.

Dual 네트워크는 Gamma 네트워크를 변형시킨 것으로서 Gamma 네트워크보다 하나 적은 스테이지 갯수를 가지며 네트워크에서 오류가 발생하는 경우에 이를 허용하는 전략(strategy)이 후진 추적(backtracking)을 하는 Gamma 네트워크보다 간단하고 오류 허용 능력도 뛰어나지만 실제적으로 이를 입증하는 연구가 행해지지 않았기 때문에 본 논문에서는 Gamma 네트워크와 Dual 네트워크의 성능을 시뮬레이션을 통해 비교분석한다.

MIN의 성능평가 방법에는 수학적 해석방법(Mathematical analysis method)과 시뮬레이션에 의한 방법이 있는데 수학적 해석방법은 프로세서나 메모리의 수가 증가함에 따라 상당히 어렵고 복잡하며 정확한 성능측정이 불가능하기 때문에 시뮬레이션에 의한 방법이 효과적이다.

본 논문에서 개발한 시뮬레이터는 네트워크를 구성하는 스위칭 소자안에 임의 갯수의 버퍼(buffer)를 둘 수 있으며 스위칭 소자의 오류도 임의적으로 만들어 줄 수 있기 때문에 오류가 발생했을 때의 성능분석과 함께 다중버퍼를 부가한 네트워크의 성능도 분석할 수 있도록 개발되었다. 또한 오류가 발생했을 때에 패킷 스위칭(packet switching)방식을 사용하는 MMIN의 성능 비교를 위해 적합한 성능 척도로써 NAP, 순서지연을 제안하며 이들을 사용하여 성능을 분석하였다.

A. Dual 네트워크

Dual 네트워크(이하 DN이라 표기함)는 GN을 변형시킨 것으로써 GN이 갖는 여러 가지 문제점들을 해결한다. 즉 GN에서는 근원지 주소와 목적지 주소가 동일한 경우는 오직 하나의 경로만 존재하기 때문에 오류 허용이 불가능하지만 DN은 모든 근원지와 목적지 사이에 많은 갯수의 중복 경로가 존재하기 때문에 이 문제를 해결할 수 있게 된다. 또한 GN은 오류 발생시 후진 추적을 해야 하므로 네트워크 크기가 커질수록 오버헤드(overhead)가 커지는데 반해서 DN은 네트워크의 크기에 상관없이 일정한 오버헤드를 갖는다. 마지막으로 GN에서는 마지막 스테이지에서의 오류를 허용할 수 없지만 DN은 마지막 스테이지에서의 오류를 허용할 수 있다[17].

(1) 구성

NxN DN은 $n(=\log_2 N)$개의 스테이지와 스테이지당 N개의 스위칭 소자로 구성되며 처음 스테이지는 2x3 스위칭 소자로 구성되어 있고 중간 스테이지는 3x3 스

위칭 소자로 구성되어 있으며 마지막 스테이지는 2x2 스위칭 소자로 구성된다. 마지막 스테이지를 제외한 모든 스테이지의 스위칭 소자들은 같은 스테이지의 다른 스위칭 소자와 스테이지 내부링크(intrastage link)를 통해 연결되어 있으며 이 스테이지내부링크를 이용해서 충돌(conflict)이나 오류를 해결한다.〈그림 1〉에 N=8인 DN이 나타나 있다.

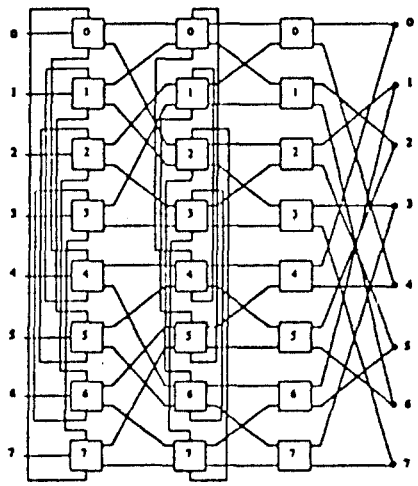


그림 1. N=8인 DN

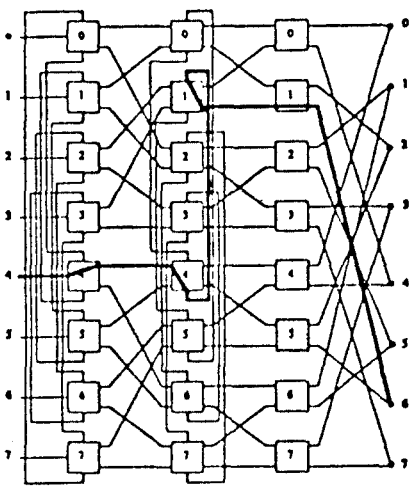


그림 2. DN의 중복경로

(2) 오류 허용 능력

DN에서는 오류가 발생했을 때 루프(loop) 형태의 스테이지내부링크를 이용한다. 스위치들의 루프는 교체가능(replaceable)한 모듈들로 구현된다. 따라서 어떤 스위치가 오류로 규정되면 그 스위치를 포함하고 있는 루프를 오류가 없는 루프로 교체한다[9,17].

DN의 오류 허용 전략은 GN과는 달리 오류가 발생했을 때 새로운 태그를 계산하지 않고 목적지 태그 라우팅(destination tag routing)방법을 그대로 사용하기 때문에 오류 허용 전략이 기존의 어느 전략보다도 간단하다. 〈그림 2〉는 스테이지 2에 있는 스위칭 소자 5번에서 오류가 발생했을 때 근원지 4에서 목적지 6으로의 연결을 나타낸다. NxN DN은 스테이지 i(0〈i〈n)에서 (N/2i - 1)개 스위칭 소자의 오류를 허용할 수 있으며 마지막 스테이지에서는 최소한 1개의 오류를 허용할 수 있다[17].

II. 시뮬레이터의 구성

A. 개요

본 논문에서 개발한 시뮬레이터는 오류를 가진 MMIN의 성능을 분석할 수 있다. 스위칭 방법으로는 패킷 스위칭을 사용하며 스위칭 소자의 각 입력 포트(port)는 2개씩의 버퍼를 가진다. 스위칭 소자안의 버퍼는 스위칭 소자안에서 충돌이 발생했을 때 선택되지 못한 패킷을 저장하게 되고 이것은 전체 네트워크의 성능을 향상시킨다. 즉 네트워크의 처리율(throughput)을 높이고 지연을 낮추게 된다. 하지만 버퍼갯수와 네트워크의 성능이 비례하지 않기 때문에 2개에서 3개의 버퍼를 사용할 때 가장 높은 성능의 향상을 얻을 수 있다.

따라서 본 논문에서는 버퍼의 수를 2개로 한정하여 네트워크의 성능을 분석하였다. 스위칭 소자안에 버퍼를 둠으로써 발생하는 현상은 파이프라인(pipeline) 효과이다. 한 패킷이 어떤 스테이지에서 옆의 스테이지로 라우트되는데 소요되는 시간 간격을 스테이지 싸이클이라고 할 때 〈그림 3〉은 각 스테이지 싸이클에서 파이프라인방식으로 전송되는 패킷들을 나타낸다.

DN은 스테이지 싸이클 4(=logN+1)일 때 목적지에 처음으로 패킷이 도착하며 GN은 스테이지 싸이클 5(=logN+2)일 때 처음으로 패킷이 도착한다. 처음 패

킷이 도착한 후에는 매 스테이지 싸이클마다 패킷이 도착할 수 있다. 본 시뮬레이션에서 사용되는 패킷의 형태는 〈그림 4〉과 같다.

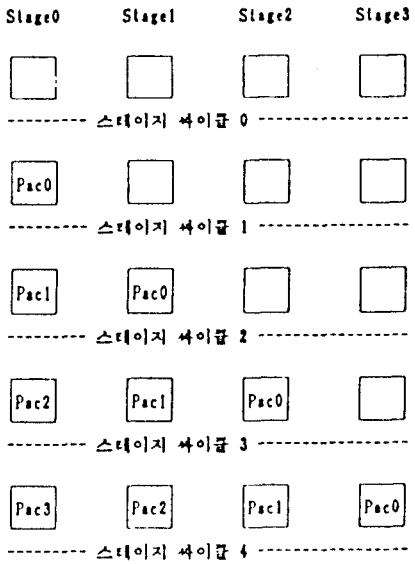


그림 3. 파이프라인 현상

근원지 주소 (sour. address)	목적지 주소 (dest. address)	지연 (delay)	데이타 (data)

그림 4. 패킷 형태

근원지 주소는 패킷을 생성한 프로세서번호를 나타내고 목적지 주소는 참조하려는 메모리 모듈을 나타내며 지연(delay)은 목적지에 도착하기까지 소요된 시간을 성능을 측정하기 위해 사용되며 초기값은 0이다. 데이타 필드는 목적지에 전송하는 데이타를 포함하며 본 시뮬레이션에서 데이타의 길이는 0으로 가정한다.

B. 모델

(1) 버퍼 모델

스위칭 소자안에 있는 버퍼의 가능한 상태는 〈그림 4〉와 같다. 버퍼가 full이 아니거나 full이면서 한 패킷이 다음 스테이지로 전송되는 경우 버퍼가 가용(available)하다고 하며 버퍼는 FIFO(First In First Out) 방식으로 운영되기 때문에 먼저 큐(queue)에 들어온 패킷이 먼저 출력된다.

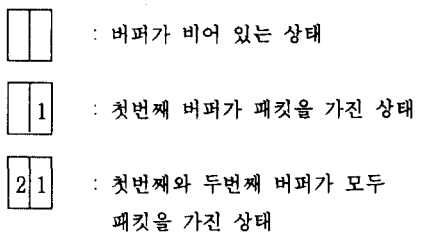


그림 5. 가능한 버퍼의 상태

스테이지 i-1의 첫번째 버퍼안의 패킷이 앞으로 진행할 수 있는 경우는 〈그림 6〉과 같다.

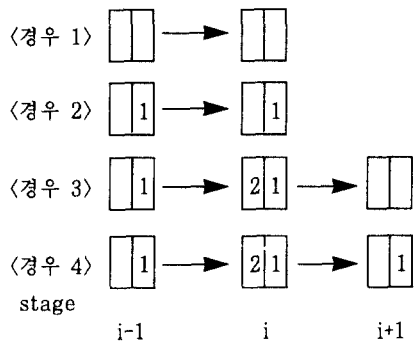


그림 6. 패킷 진행이 가능한 경우

〈경우 1,2,3,4〉는 모두 스테이지 i-1의 버퍼가 empty가 아닌 경우에 스테이지 i로 스테이지 i-1의 패킷이 전송될 수 있는 가능한 상태를 나타내고 있다. 네트워크는 동기적(synchronously)으로 작동한다고 가정하기 때문에 위의 패킷 이동 가능 상태를 알아내기 위해서는 마지막 스테이지로부터 시작하여 처음 스테이지까지 각 스테이지의 모든 스위치의 상태를 조사해야 한다. 따라서 위의 〈경우 3〉과 〈경우 4〉같은 패킷 이동이 가능해지며 스테이지 i를 검사할 때 스테이지 i+1로 스테이지 i의 패킷이 이동하기 때문에 스테이지 i-1의 패킷이 스테이지 i로 이동할 수 있게 된다.

만약 처음 스테이지에서 마지막 스테이지로의 순서로 각 스테이지의 모든 스위치들을 조사한다면 동기적으로

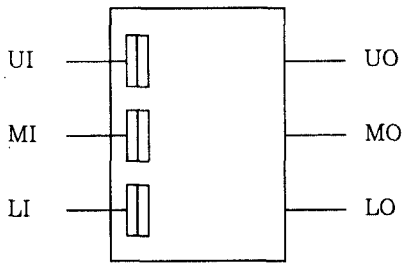
동작하는 네트워크를 시뮬레이트할 수 없게 된다. 그 이유는 위의 〈경우 3〉과 〈경우 4〉같은 경우는 패킷이동이 불가능해지기 때문이다.

(2) 스위칭 소자 모델

후진추적을 하는 MIN은 후진추적을 하지 않는 MIN보다 낮은 하드웨어 복잡도를 갖는 것이 대부분이지만 후진추적을 하는 MIN은 구현하기가 힘들다. 그 이유는 양방향 경로(bidirectional path)와 역방향 큐를 요구하기 때문이다. 따라서 오류 허용을 위해 후진추적을 해야하는 GN의 경우는 양방향 경로를 필요로 하며 DN은 후진추적을 하지 않기 때문에 단방향 경로를 사용한다.

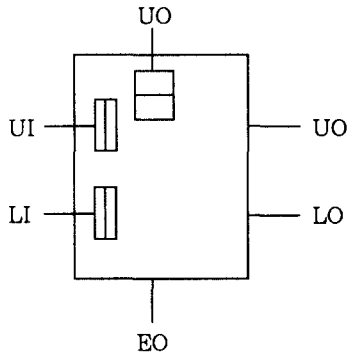
오류로 인하여 앞으로 전진할 수 없는 패킷은 후진 큐를 이용하여 전스테이지(previous stage)로 후진하게 된다. 또한 GN은 양방향 링크를 사용하기 때문에 전이중(full duplex)방식을 가정한다.〈그림 7〉은 GN의 중간단계에서 사용되는 3x3 스위칭 소자의 형태를 나타내는데 왼쪽의 UI,MI,LI는 각각 상위,중위,하위입력링크를 나타내고 오른쪽의 UO,MO,LO는 각각 상위,중위,하위출력링크를 나타낸다.

DN의 중간스테이지에서 사용되는 3x3 스위칭 소자의 형태는 〈그림 8〉과 같다. 왼쪽의 UI,LI는 입력포트이고 EI는 스테이지내부링크와 연결된 입력포트이며, 오른쪽의 UO,LO는 출력포트이고 EO는 같은 스테이지의 다른 스위칭 소자와 연결되는 출력포트이다.



UI : Upper input link UO : Upper output link
MI : Middle input link MO : Middle output link
LI : Lower input link LO : Lower output link

그림 7. GN에서 버퍼를 가진 3x3 스위칭 소자의 형태



UI : Upper input link UO: Upper output link
LI : Lower input link LO : Lower output link
EI : Extra input link EO : Extra output link

그림 8. DN에서 버퍼를 가진 3x3 스위칭 소자의 형태

각 스위칭 소자에서는 버퍼의 운영관리를 위해서 〈그림 9〉와 같은 버퍼 제어 테이블(buffer control table)을 필요로 한다.

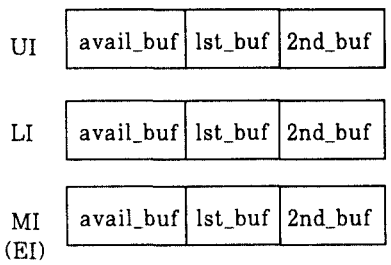


그림 9. 버퍼 제어 테이블

1st_buf와 2nd_buf는 각각 첫번째와 두번째 버퍼의 상태를 나타내는데 버퍼가 비어있을 때는 0의 값을 가지며 버퍼가 사용중일때는 1의 값을 갖는다. 또한 avail_buf는 버퍼의 가용상태를 나타낸다. 즉, 두 버퍼가 모두 사용중일때는 1의 값을 갖고 그 외에는 0의 값을 갖게 되어 버퍼의 가용상태를 나타낼 수 있다.

〈그림 10〉은 스위칭 소자의 각 입력에 있는 버퍼모듈(buffer module)과 버퍼 제어 테이블을 나타낸다. 첫

번째 버퍼가 비게되면 두번째 버퍼에 있는 패킷이 첫번째 버퍼로 이동하게 된다. 버퍼 모듈의 크기를 M으로 할 경우에는 다음과 같은 형태의 버퍼 제어 테이블과 버퍼모듈을 갖게 된다.

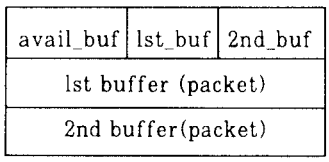


그림 10. 버퍼 제어 테이블과 버퍼모듈.

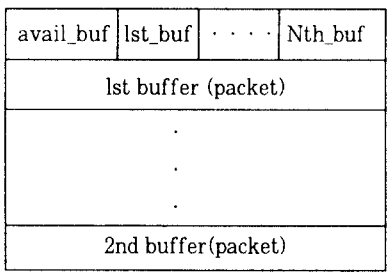


그림 11. M개의 버퍼를 가진 경우

각 입력측이 가진 전체 버퍼를 버퍼 모듈(module)이라고 할 때 avail_buf가 1로 세트(set)되는 경우는 버퍼 모듈이 full일 때이다. 그 이외에는 avail_buf는 0의 값을 가지고 각 버퍼는 full일 때와 empty일 때 각각 1과 0의 값을 갖게 된다. 또한 스위칭 소자의 오류 상태를 나타내기 위한 오류 플래그(flag)가 필요하게 되는데 이 오류 플래그의 초기값은 0이고 오류가 발생했을 때는 오류 플래그가 1로 세트된다.

(3) 네트워크 모델

GN과 DN을 구성하기 위해 앞에서 소개한 버퍼모델과 스위칭 소자 모델을 이용한다.

(a) GN

GN은 $PM2^i$[14]형태로 연결되므로 이에 따라 스위칭 소자들의 출력과 입력을 연결한다. 스테이지 i($0 \le i < \log N$)에 있는 스위칭 소자 j($0 \le j < N$)의 UO는 스테이지 i+1의 스위칭 소자 $(j - 2^i) \bmod N$의 LI의 버퍼모듈로 연결되고, MO는 스테이지 i+1의 스위칭 소자 j의 MI의 버퍼모듈, 그리고 LO는 스테이지 i+1의 스위칭 소자 $(j + 2^i) \bmod N$의 UI의 버퍼모듈로 연결된다.

(b) DN

크기가 N인 DN은 크기가 N/2인 baseline 네트워크의 연결을 사용하여 스위칭 소자들을 연결한다. $n = \log N$이라 하면 스위칭 소자 j($0 \le j < N$)는 $S_nS_{n-1}..S_1$으로 표시되고 스위칭 소자 j의 UO와 LO는 각각 $S_nS_{n-1}..S_i0$와 $S_nS_{n-1}..S_i1$로 표시된다.

스테이지 i($0 \le i < n-1$)에 있는 스위칭 소자 $S_nS_{n-1}..S_1$의 $S_nS_{n-1}..S_i0$은 스테이지 i+1에 있는 스위칭 소자 $S_n..S_{n-i+1}0S_{n-i}..S_2$로 연결되고 $S_nS_{n-1}..S_i1$은 스테이지 i+1에 있는 스위칭 소자$S_n..S_{n-i+1}1S_{n-i}..S_2$로 연결된다. j가 짝수이면 UO와 LO는 다음 스테이지에 있는 해당 스위칭 소자의 UI버퍼모듈에 연결되고 홀수이면 LI에 연결된다. j가 N/2보다 작을때는 EO가 같은 스테이지에 있는 스위칭 소자 j+(N/2)의 EI버퍼모듈로 연결되고, j가 N/2보다 크거나 같을때는 j가 k번째 분할에 속할 때 같은 스테이지의 스위칭 소자 $(j-N/2+1) \bmod ((N/2)/2^i)+((N/2)/2^i)*k$의 EI버퍼모듈로 연결된다. 스테이지 n-1에 있는 스위칭 소자 $S_nS_{n-1}..S_1$의 $S_nS_{n-1}..S_i0$은 목적지$0S_{n-1}..S_2S_1$로 연결되고 $S_nS_{n-1}..S1_1$은 목적지 $1S_{n-1}..S_2S_1$로 연결된다.

목적지에는 세마포어(semaphore)를 가정하여 2개 이상의 패킷이 같은 목적지를 요구할 때 먼저 요구한 패킷을 받아들이고 나머지 패킷은 버퍼안에서 한 스테이지 싸이클동안 지연된다. 본 시뮬레이션에서는 낮은 번호의 스위칭 소자가 우선순위(priority)를 갖도록 하였다.

(4) 오류 모델

일반적으로 사용되고 있는 오류 모델은 세가지가 있다. 즉 스위칭 소자를 동작시키는 제어가 있음에도 불구하고 스위칭 소자가 특정 상태에 머무르는 stuck-at 오류 모델과 스위칭 소자는 동작하고 링크만 오류인 링크 오류 모델, 그리고 스위칭 소자 오류 모델이 있는데 이 세가지 중에서 스위칭 소자 오류 모델은 스위칭 소자 전체를 사용하지 못하는 것으로서 가장 심각한 오류 모델이다[9]. 본 논문에서는 오류 모델로서 스위칭 소자 오류 모델을 사용하고 오류 형태를 오류가 발생하는 스테이지와 오류의 갯수에 따라 네가지로 분류한다.

〈오류 형태 1〉 첫번째 스테이지와 마지막 스테이지를

제외한 중간 스테이지에 하나씩의 오류가 발생한 경우이다.

〈오류 형태 2〉 스테이지 1에서 N/2-1개의 오류가 발생한 경우이다. NxN DN은 스테이지 1에서 최소한 $N/2^1$-1개의 오류를 허용할 수 있기 때문에 오류의 수를 DN에 맞추어서 정하였다.

〈오류 형태 3〉 마지막 바로 전 스테이지에서 다중 오류가 발생한 경우이다. GN의 경우에는 차 logN-1 스테이지에서 $N/2\log^{N-1}$개의 오류가 발생하게 되며 DN의 경우는 logN-2 스테이지에서 $N/2\log^{N-2}$개의 오류가 발생하게 되고 GN과 DN 모두 동등 하게 공평(fair)한 원칙에 의해 오류의 수가 정해진다.

〈오류 형태 4〉 마지막 스테이지에서 다중오류가 발생한 경우이다. 이 때는 네트워크 크기의 절반에 해당하는 스위칭 소자의 오류를 가정한다.

(5) 성능 척도

(a) 환 경

오류가 네트워크의 성능에 미치는 가장 큰 영향은 단절(disconnection)이다. 네트워크에서 오류가 발생하지 않은 경우에 가장 일반적으로 사용되고 있는 성능척도로는 대역폭(bandwidth)이 사용된다. 이 때 대역폭은 대칭적 트래픽(symmetric traffic)을 가정하며 네트워크에 있는 모든 프로세서와 메모리에 대한 평균 성능을 나타내지만 네트워크에 오류가 발생하면 대칭적 성질이 파괴되고 네트워크의 여러 부분에서 부하의 차이가 생기기 때문에 네트워크 대역폭은 적당한 성능 척도가 될 수 없다[10].

오류를 허용하는 MIN을 모델링하는데는 성능과 신뢰성뿐만 아니라 서비스의 질(quality of service)의 요구도 명확히 하는 것이 필요하다. 예를 들어 다중 컴퓨터의 네트워크에서 한 프로세서의 저하된 서비스는 병렬처리의 병목(bottleneck)이 될 수 있다. 많은 경우에 있어서 MIN의 성능은 얼마나 많은 정보가 주어진 시간 주기안에 네트워크를 통과하는지 그리고 패킷이 네트워크를 통과하는데 걸리는 시간으로 특성지어진다. 패킷을 생성하는 태스크(task) 사이의 종속(dependence)이 중요한 시스템의 성능을 모델링하기 위해서는 지연척도(delay measure)가 중요하다.

(b) 성능 척도 정의

(정의) d_{ij}는 스테이지 싸이클 i에 생성되고 목적지 주소 j에 도착한 패킷의 지연을 나타낸다. 만약 스테이지 싸이클의 제한횟수(limit)까지 패킷이 도착하지 않으면 d_{ij}는 0의 값을 갖는다. ($1 \le i \le limit$, $1 \le j \le N$)

o 순서지연(Seq_delay) : 스테이지 싸이클 1에서 생성된 패킷들이 목적지에 도착할 때까지 소요된 평균 시간 지연.

$$\text{순서지연} = \sum_{j=1}^{N} d1j / N$$

o 평균지연(Ave_delay) : 근원지에서 목적지까지 도착한 패킷들의 평균 시간 지연.

$$\text{평균지연} = \sum_{i=1}^{limit} \sum_{j=1}^{N} dij / limit \times N$$

o NAP (Not Arrive Packet) : NxN 크기의 네트워크상에서 임의 갯수의 오류가 발생했을 때 도착하지 못하는 패킷의 평균 갯수

$$NAP = N - arrive$$

여기서 arrive는 목적지에 도착한 패킷들의 합이다.

o PA (Probability Acceptance) : 전체 메모리 모듈에 대해서 요구된 횟수와 참조된 횟수의 비율.

$$PA = arrive / limit \times N$$

각 프로세서에 의해서 생성되는 패킷들은 여러 스테이지를 거쳐서 목적지에 도착하기까지 여러 스테이지 싸이클을 요구하게 된다. 특히 GN의 경우에는 새로운 라우팅 태그를 산출하기 때문에 지연이 발생하게 된다. 또한 오류로 인하여 목적지에 도착하지 못하는 패킷들도 생길 수 있으며 이런 경우는 타임아웃이나 일련번호(Sequence number)등을 이용하여 탐지해내고 패킷을 재전송(retransmission)해야하는 오버헤드를 갖는

다.

같은 시간에 생성된 패킷들이 서로 다른 시간에 목적지에 도착하는 이유는 스위칭 소자에서 충돌이나 오류가 발생하기 때문이다. 따라서 이들 패킷들이 근원지에서 목적지까지 도착하기 위해 얼마만큼의 시간이 소요되었는지를 평가하여 네트워크의 성능을 측정하는 척도로 삼는다. 순서지연과 NAP은 각 프로세서에서 한 개씩의 패킷을 생성하여 N개의 패킷이 모두 도착하거나 스테이지 싸이클의 수가 제한된횟수에 이를때까지 수행하고 PA와 평균지연은 각 프로세서에서 고정된 갯수의 패킷을 생성하여 스테이지 싸이클의 수가 제한횟수에 이를때까지 수행한 결과를 측정한 것이다. 네트워크의 모든 스위칭 소자에서 충돌이나 오류가 발생하지 않는 경우는 가장 이상적인 경우이고 이 때는 순서지연과 평균지연이 같은 값을 갖게 된다. 오류가 발생한 경우에 순서지연과 평균지연을 측정하면 평균지연은 순서지연보다 크거나 같은 값을 갖는다. 그 이유는 순서지연은 네트워크가 초기화되어있는 상태에서 측정되기 때문이며 평균지연은 네트워크의 부하가 있는 경우에 측정되기 때문이다. PA는 처리율과 비슷한 척도로 사용되는데 NAP과는 아래와 같은상관관계를 가진다.

$$arrive = N - NAP$$

$$PA = (N - NAP)/limit \times N$$

N,limit는 고정된 수이므로 NAP이 네트워크의 처리율에 직접적인 영향을 끼침을 알 수 있다.

III. 시뮬레이션의 수행

A. 개요

본 시뮬레이션에서는 스위칭 방식으로 패킷 스위칭을 사용하기 때문에 네트워크의 트래픽(traffic)에 따라 각 스위칭 지점에서 임의의 지연을 갖는다. 제어방식(control method)에는 집중식(centralized)과 분산(distributed)방식이 있는데 집중식은 모든 제어신호가 한 근원지에서 나오게 되고 중앙 제어기(central controller)는 시스템 병목현상을 유발시키고 전체 시스템의 성능과 신뢰도에 직접적인 영향을 미치게 된다. 따라서 본 시뮬레이션에서는 분산식 제어방식을 사용하여 집중식 제어방식에서 발생하는 단점을 없앤다. 그리고 네트워크는 동기적으로 동작한다고 가정하기 때문에 전역클럭(global clock)으로 구현된다[4].

어떤 네트워크가 버퍼나 후진 능력을 가지고 있지 않으면 하나의 네트워크 싸이클은 n개의 스테이지 싸이클과 동일하나 그 이외의 경우에 있어서는 하나의 네트워크 싸이클이 n개의 스테이지 싸이클보다 크다. 파이프라인 방식을 사용하지 않을 때에는 각 근원지에서 각 네트워크 싸이클의 시작에서 패킷을 생성할 수 있고 파이프라인 방식에서는 각 스테이지 싸이클의 시작에서 패킷을 생성할 수 있다[12].

B. 가정

[가정 1] 시뮬레이션에서 가정하는 것은 다음과 같다.

(1) 네트워크는 동기적으로 동작한다.
(2) 초기에 네트워크의 모든 버퍼는 empty이다.
(3) 스테이지 싸이클은 5ns 주기로 발생한다.
(4) N개의 프로세서는 각 스테이지 싸이클에서 최대입력부하 확률로 패킷들을 독립적으로 생성한다.
(5) 프로세서에 의해 각 스테이지 싸이클에서 생성되는 각 패킷의 목적지는 균일 분산(uniformly distributed)된다.
(6) 각 스위칭 소자의 입력 포트에 2개의 버퍼가 있다.
(7) 버퍼는 FIFO에 의해서 운영된다.
(8) 모든 스위치는 두 개 이상의 패킷이 서로 다른 출력링크를 가리키는 한 병렬적으로 패스(pass)할 수 있는 능력을 가진다. (만약 같은 출력을 가리키면 임의로 하나를 선정한다.)
(9) 버퍼로 패킷을 넣거나 빼는데 걸리는 시간은 패킷이 전송되는 시간과 함께 한 스테이지 싸이클안에 포함된다.
(10) 다음 스테이지의 오류의 상태를 알 수 있다.

[가정 2] GN의 동작과 이에 필요한 가정은 아래와 같다.

(1) 한 스위칭 소자에서 두 개 이상의 입력 패킷이

같은 출력 링크를 요구하면 한 개의 패킷이 선택되어 출력 링크로 출력되고 나머지는 버퍼에 남게 된다.

(2) 새로운 태그를 계산하는데 소요되는 시간은 하나의 스테이지 싸이클이다.

(3) 후진 추적에 소요되는 시간은 후진 깊이 만큼의 스테이지 싸이클이다.

〔가정 3〕 DN의 동작과 이에 필요한 가정은 아래와 같다.

(1) 두 개의 패킷이 동시에 같은 출력을 요구하면 하나를 선택하여 출력 링크로 출력하고 다른 하나는 스테이지내부링크로 스위치한다.

(2) 세 개의 패킷이 동시에 같은 출력을 요구하면 하나를 선택하여 출력 링크로 출력하고 나머지 두 개 중에서 하나를 선택하여 스테이지내부 링크로 스위치하며 선택되지 못한 패킷은 버퍼에 남게 된다.

(3) 스테이지내부 링크를 이용하여 같은 스테이지에 있는 다른 스위칭 소자로 패킷을 전송할 때 하나의 스테이지 싸이클이 소요된다.

IV. 비교 분석

본 논문에서 구현한 시뮬레이터를 이용해서 버퍼를 부가한 DN과 GN의 성능을 비교한다. DN과 GN 모두 패킷 스위칭 방법을 사용하며 오류를 허용할 수 있는 MIN이기 때문에 앞에서 정의한 오류 모델에 따라서 성능을 평가하며 성능 평가의 기준은 기존의 성능척도인 평균지연외에 본 논문에서 제안한 NAP,순서지연등을 사용한다. 다중 프로세서 시스템을 단일 프로세서 시스템에서 시뮬레이트하는 것은 그리 쉬운 일이 아니지만, 본 논문에서 구현한 시뮬레이터는 동기적으로 동작하는 네트워크를 시뮬레이트하였으며 그 과정에 있어서 많은 시간과 기억 장소가 필요하였고 시뮬레이션 프로그램은 PASCAL 언어를 사용하였으며 SUN-3/160M 시스템에서 실행하였다.

각 오류가 발생하는 형태에 따라서 순서지연,PA등을 측정하였으며 변화요인은 네트워크의 크기이고 버퍼의 크기는 2로 고정시켰다. 네트워크의 입력 부하는 1로써 매 스테이지 싸이클마다 프로세서에 의해서 패킷들이 생성되어 네트워크에 입력된다. 순서지연과 NAP은 N개의 패킷을 가지고측정되기 때문에 낮은 부하에서의 성능을 측정할 수 있고 평균지연과 PA는 임의의 정수 m(〉1)에 대해서 Nxm개의 패킷을 가지고 측정되기 때문에 높은 부하에서의 성능을 나타낸다.

패킷 스위칭을 사용하는 네트워크에서는 패킷의 도착 순서가 보내진 순서와 다를 수 있으며 SIMD(Single Instruction stream Single Data stream)환경에서는 패킷을 잃어버리는 경우에 미리 도착한 패킷들이 잃어버린 패킷이 재전송되어 목적지에 도착할 때까지 기다려야 하기 때문에 네트워크의 성능을 저하시키게 된다. 순서지연과 평균지연은 일반적인 성능척도가 될 수 있고 NAP은 SIMD환경에서 네트워크의 성능을 평가하는데 사용될 수 있으며 MIMD(Multiple Instruction stream Multiple Data stream)환경에서는 PA가 네트워크의 성능을 평가하는데 사용될 수 있다.

A. 오류 형태 1이 발생한 경우

〈그림 12〉에서 (a)는 N에 의한 NAP의 영향을 보여주고 있는데 처음 스테이지와 마지막 스테이지를 제외한 모든 중간 스테이지에 하나씩의 오류가 발생했기 때문에 GN의 경우에는 목적지에 도착하지 못하는 패킷들이 있었고 DN의 경우에는 생성된 모든 패킷들이 목적지에 도착하였다. (b)는 N에 의한 평균지연의 영향을 보여주고 있는데 DN의 평균지연이 GN의 평균지연보다 작은 것을 알 수 있다. (c)는 N에 의한 순서지연의 영향을 보여주고 있는데 (b)와 마찬가지로 DN의 순서지연이 GN의 순서지연보다 작은 것을 알 수 있다. DN은 모든 패킷이 목적지에 도착한데 반해서 GN은 목적지에 도착하지 못한 패킷이 있지만 DN의 평균지연과 순서지연이 GN보다 작은 이유는, DN이 GN보다 하나 적은 스테이지 갯수를 갖고 오류 허용 방법이 간단하면서도 지연이 적게 걸리기 때문이다. GN의 오류 허용 방법은 후진 추적을 사용하기 때문에 N이 커질수록 후진 추적의 깊이가 깊어지고 이것이 GN의 평균지연과 순서지연을 증가시키는 원인이 된다. (d)는 N에 의한 PA의 영향을 보여주고 있는데 N이 증가할수록 GN의 PA가 1에 수렴하는 것을 볼 수 있다. N이 증가할수록 스테이지마다 발생한 하나의 오류는 네크워크의 크기와 비교할 때 작은 영향을 끼치기 때문에 GN의 PA가 커지게 된다.

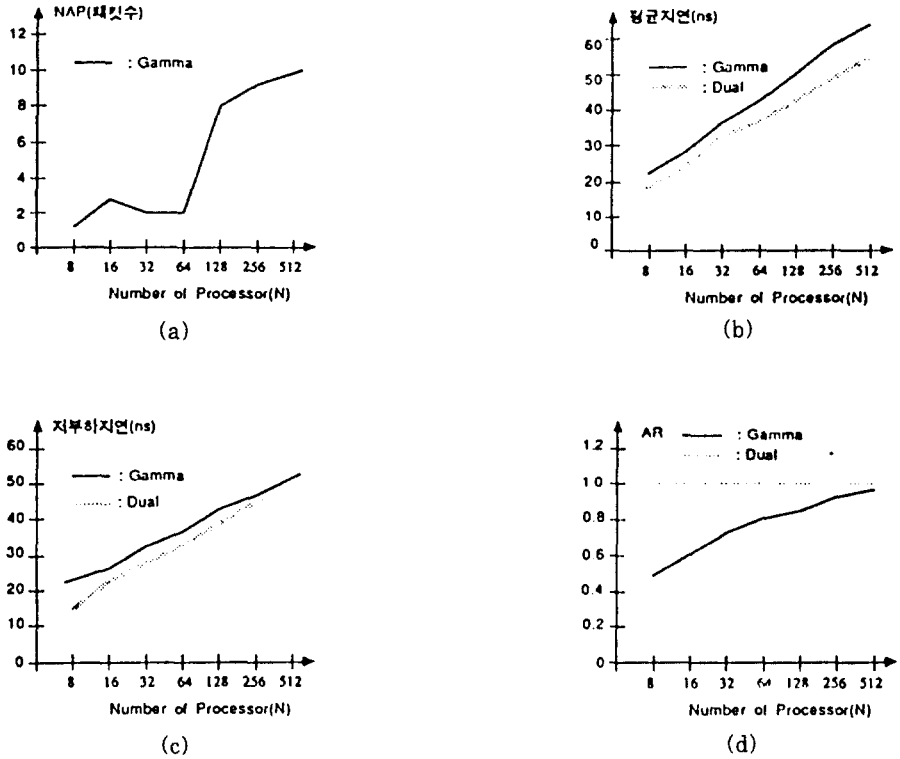


그림 12. 각 스테이지에 하나씩의 오류가 발생한 경우

B. 오류 형태 2가 발생한 경우

〈그림 13〉에서 (a)는 N에 의한 NAP의 영향을 보여주고 있는데 N이 커질수록 NAP이 지수적으로 증가함을 알 수 있다. N = 512일 때 NAP = 250이므로 거의 N/2에 가까운 패킷이 목적지에 도착하지 못하는 것을 알 수 있으며 이것은 근원지에 가까운 스테이지에서 다중 오류가 발생할 때에 네트워크의 성능에 심각한 영향을 끼친다는 것을 의미한다. (b)는 N에 의한 평균지연의 영향을 보여주고 있는데 오류 형태 1의 결과와는 달리 DN의 평균지연이 GN보다 큰것을 나타낸다. 그 이유는 GN의 경우에는 제대로 도착한 패킷들(N/2개 정도)만의 지연이 측정된데 비해 DN은 대부분의 패킷(N개 정도)들이 오류로 인한 지연까지 측정되기 때문이다. (c)는 N에 의한 순서지연의 영향을 보여주고 있는데 N이 커질수록 DN의 순서지연이 GN의 순서지연과 거의 일정한 차이를 두고 단조 증가함을 알 수 있는데 그 이유는 (b)에서 관찰한 것과 같다. (d)는 N에 의한 PA의 영향을 보여주고 있는데 DN과 GN의 차이가 거의 일정한 것을 알 수 있으며 DN은 0.8 에서 0.95 정도의 PA를 갖는데 반해 GN은 0.2에서 0.35 정도의 PA를 갖는다.

GN은 라우팅 태그로써 목적지 주소에서 근원지 주소를 뺀 값을 사용한다. 이 때 이 태그가 짝수이면 스테이지 0의 태그 비트가 0이기 때문에 스테이지 1에서 오류가 발생하는 것은 네트워크의성능에 아주 치명적이다. 그 이유는 스테이지 0에서 스테이지 1로 가는 경로가 오직 MI밖에 없기 때문에 다수개의 중복경로는 유명무실하게 되어 오류의 허용이 불가능해진다. DN은 목적지 주소 라우팅을 사용하고 중복경로가 $\pi_{i=2,n}2^i$(N=2^n, n

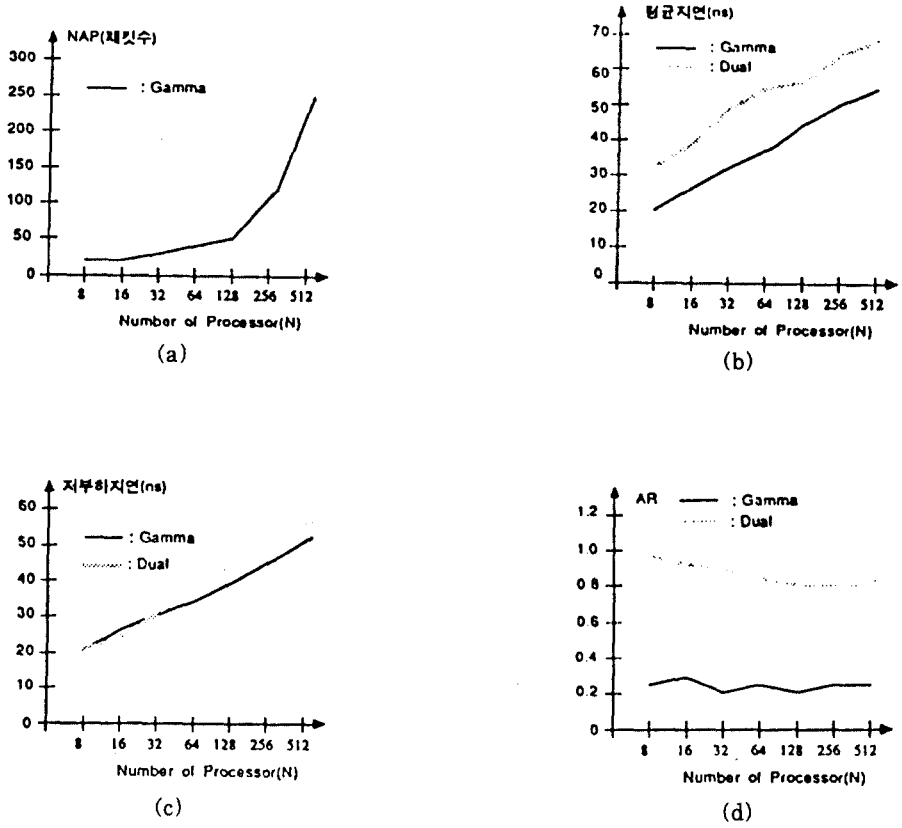


그림 13. 스테이지 1에서 N/2-1 개의 오류가 발생한 경우

≥2)개이기 때문에 오류의 허용이 가능하며 단지 지연이 증가하게 된다.

C. 오류 형태 3이 발생한 경우

〈그림 14〉에서 (a)는 N에 의한 NAP의 영향을 보여주고 있는데 N의 크기에 상관없이 GN의 NAP은 0에서 1개 임을 알 수 있고 다중 오류가 발생한 스테이지가 목적지에 가까울수록 네트워크의 성능에는 거의 영향을 끼치지 않음을 알 수 있다. 그 이유는 후진 추적의 깊이가 오류 형태 2보다 깊기 때문에 중복 경로의 수도 많아지고 따라서 목적지에 도착하는 패킷들의 수가 오류 형태 2에 비해서 많아진다. (b)는 N에 의한 평균지연의 영향을 보여주고 있는데 N의 크기가 커질수록 DN과 GN의 차이가 점점 커짐을 알 수 있다. 이것은 후진 추적을 오류 허용 방법으로 사용하는 MIN의 오버헤드때문이다. 따라서 중복 경로를 제공하지만 많은 지연이 걸리는 GN보다 중복 경로를 제공하면서도 적은 지연이 걸리는 DN의 성능이 뛰어나다고 할 수 있다. (c)는 N에 의한 순서지연의 영향을 보여주고 있는데 N의 크기에 상관없이 차이가 거의 일정하게 나타난다. (d)는 N에 의한 PA의 영향을 보여주고 있는데 N이 커질수록 GN의 PA가 거의 1에 수렴하는 것을 알 수 있고 그 이유는 목적지에 가까운 스테이지에서 발생하는 오류의 경우가 근원지에 가까운 스테이지에서의 오류보다 많은 중

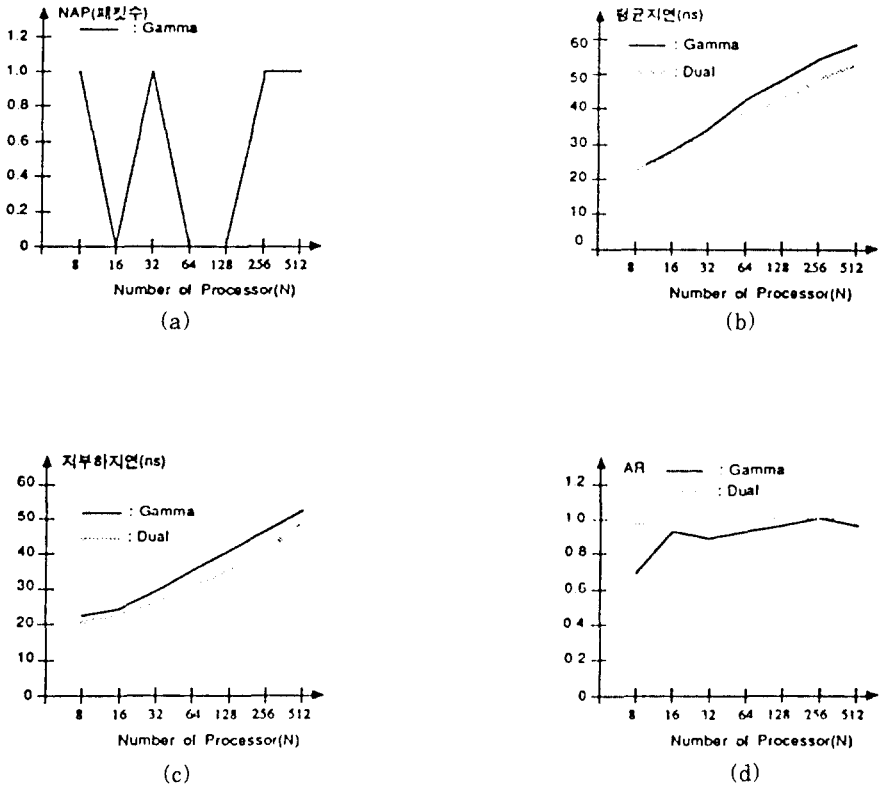


그림 14. 마지막 바로 전 스테이지에서 다중오류가 발생한 경우

복경로를 제공하기 때문이며 GN의 평균지연이 증가하는 이유이기도 하다. 이것은 〈그림 12〉의 (d)와 비교할 때에 1에 수렴하는 속도가 더욱 빠르게 나타나는 것으로도 알 수 있다.

D. 오류 형태 4가 발생한 경우

〈그림 15〉에서 (a)는 N에 의한 NAP의 영향을 보여주고 있는데 GN은 마지막 스테이지에서 발생하는 스위칭 소자의 오류를 허용하지 못하기 때문에 요구된 패킷의 절반이상이 목적지에 도착하지 못한다. 이와는 달리 DN은 마지막 스테이지에서 발생하는 다중오류를 허용할 수 있기 때문에 NAP은 0이다. (b)는 N에 의한 평균지연의 영향을 보여주고 있는데 DN은 N개의 패킷에 대한 지연이 측정되고 GN은 N/2정도의 패킷에 대한 지연이 측정되었기 때문에 DN은 오류를 허용하는 전략에 따라 스테이지내부 링크를 이용하는데 걸리는 지연이 포함되었고 GN은 대부분의 패킷이 오류를 포함하지 않는 패킷들의 지연을 포함하고 있다. 따라서 DN의 평균지연이 GN의 평균지연보다 오래 걸리지만 전달된 패킷의 수를 고려한다면 두 배에 가까운 패킷이 전달될 때 평균지연은 두 개의 스테이지 싸이클 정도 차이가 발생하므로 DN의 성능은 GN의 두 배에 가깝다고 할 수 있다. (c)는 N에 의한 순서지연의 영향을 나타내는데 DN과 GN의 차이가 거의 발생하지 않는다. NAP이

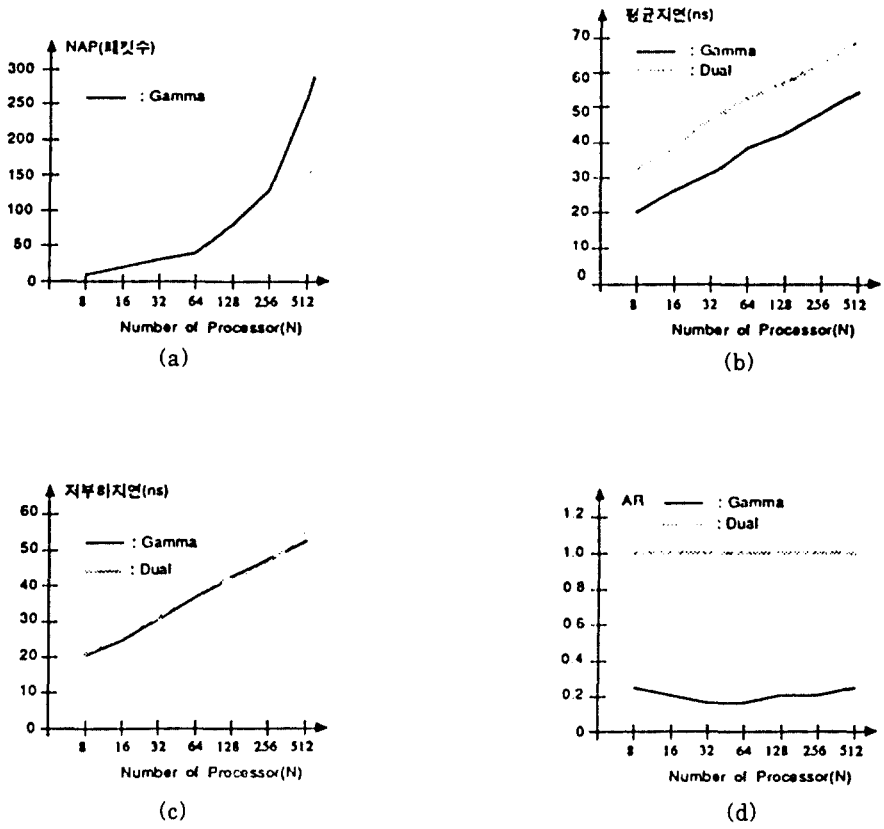


그림 15. 마지막 스체이지에서 N/2개의 오류가 발생한 경우

GN의 두 배이면서도 지연의 차이가 거의 발생하지 않은 이유는 DN의 스테이지수가 GN보다 하나 적기 때문이다. (d)는 N에 의한 PA의 영향을 나타내는데 DN은 NAP이 0이므로 요구된 참조의 수와 실제로 참조된 수가 동일하지만 GN은 NAP이 N/2정도이고 이 패킷들이 네트워크의 부하를 높이기 때문에 PA는 작은 값을 갖게 된다.

V. 결론

지금까지 많은 MIN들이 소개되었고 그들의 성능도 분석되었지만 오류가 발생했을 때에는 뚜렷한 성능기준 없이 기존의 오류가 발생하지 않았을 때의 성능 척도(performance measure)를 그대로 사용하였기 때문에 정확한 성능 비교를 할 수가 없었다. 오류 형태도 주로 하나의 오류가 발생한 경우이거나 임의적인 다중 오류를 가정한 상태에서 성능평가가 행해졌다. 또한 분석 모델(analytic model)을 이용하여 성능을 평가하는 것은 복잡한 수식 계산을 필요로 하며 그 결과도 정확하지 못한 것이 대부분이었고 오류가 발생했을 때의 성능분석은 더욱 복잡하기 때문에 분석 모델을 사용하는 것은 매우 어렵게 된다.

따라서 본 논문에서는 MIN의 성능을 분석할 수 있는 시뮬레이터를 개발하여 Gamma 네트워크와 Dual 네

트워크의 성능을 분석하였다. 성능 척도로서는 순서지연, NAP,평균지연,PA를 사용하였으며 오류 형태의 크기는 네트워크의 크기에 비례하도록 하여 성능을 분석하였고, Dual 네트워크의 성능이 모든 경우에 있어서 Gamma 네트워크보다 뛰어났다. Dual 네트워크는 Gamma 네트워크와 비교할 때 하나 적은 스테이지를 갖기 때문에 지연이 적고 오류 허용능력이 더 크기 때문에 처리율이 Gamma 네트워크보다 뛰어났다. 또한 Daul 네트워크는 중복경로의 수가 많아서 여러 오류 형태가 발생했을 때에도 모든 패킷이 목적지에 도착하지만 Gamma 네트워크의 경우는 중복경로가 존재함에도 불구하고 근원지에 가까운 스테이지에서 다중오류가 발생하는 경우에는 이를 허용하지 못하고 네트워크 크기의 절반에 가까운 패킷들이 목적지에 도착하지 못하였다. 또한 마지막 스테이지에서 발생하는 다중 오류도 DN은 허용하였고 오류 허용 전략에 따른 오버헤드도 적게 나타났다.

본 논문에서 개발한 시뮬레이터는 MIN의 여러 가지 특성을 실험할 수 있다. 즉 네트워크의 부하상태, 패킷 전송시 발생하는 충돌횟수, 그리고 목적지에 도착하는 패킷의 순서등을 알 수 있어서 MIN의 특성을 연구하는데 사용될 수 있다.

참고문헌

1. G.B.Adams III, D.P.Agrawal and H.J.Siegel, "Fault-Tolerant Multistage Interconnection Networks," IEEE Compt., pp.14-27, June., 1987.
2. G.B.Adams III and H.J.Siegel, "The Extra Stage Cube : A Fault-Tolerant InterconnectionNetwork for Supersystems," IEEE Trans. Compt., pp.443-454, May., 1982.
3. P.Banerjee and A.Dugar, "The Design, Analysisand Simulation of a Fault-Tolerant Interconnection Network Supporting the Fetch-and-Add Primitive," IEEE Trans. Compt., pp.30-45, Jan., 1989. 4. L.N.Bhuyan,Q.Yang and D.P.Agrawal, "Performance of Multiprocessor Interconnection Networks," IEEE Compt., pp.25-37, FEb., 1989.
5. D.M.Dias and J.R.Jump, "Packet Switching Interconnection Networks for Modular Systems," IEEE Compt., pp.43-53, Dec., 1981.
6. D.M.Dias and J.R.Jump," Analysis and Simulation of Buffered delta Networks," IEEE Trans. Compt., pp.273-282, Apr., 1981.
7. T.Feng, "A Survey of Interconnection Networks," IEEE Compt., pp.12-27, Dec., 1981.
8. S.C.Kothari,G.M.Prabhu and R.Roberts, "A Multipath Network ith Cross links," Jounal of Parallel and Distributed Computing 5, pp.185-193., 1988.
9. V.P.Kumar and S.M.Reddy, "Augmented shuffle-Exchange Multistage Interconnection Networks," IEEE Compt., pp.30-40, June., 1987.
10. V.P.Kumar and A.L.Reibman, "Failure Dependent Performance Analysis of a Fault-Tolerant Multistage Interconnection Network," IEEE Trans. Compt., pp.1703-1713, Dec., 1989.
11. D.H.Lawrie, "Access and Alignment of Data in an Array Processor," IEEE Trans. Compt., pp.1145-1155, Dec., 1975.
12. K.Y.Lee and H. YOON, "The B-Network : A Multistage Interconnection Network with Backward Links," IEEE Trans. Compt., pp.966-969, July., 1990.
13. R.J. Mcmillen and H.J.Siegel, "Routing Schemes for the Augmented Data ManipulatorNetwork in an MIMD System," IEEE Trans. Compt., pp.184-196, Dec., 1982.
14. D.S.Parker and C.S.Raghsvendra, "The Gamma Network," IEEE Trans. Compt., pp.367-373, Apr., 1984.
15. J.H.Patel, "Performance of Processor-Memory Interconnections for Multiprocessors," IEEE Trans. Compt., pp.771-780, Oct., 1981.

16. C.L.Wu and T.Y.Feng, "On a Class of Multistage Interconnection Networks," IEEE Trans. Compt., pp.694-702, Aug., 1980.

17. 최 창훈, "병렬 처리 시스템에서의 Dual 네트워크 설계 및 오류 허용 라우팅 전략," 서강 대학교 공학석사 학위논문, 1989.

李 世 亨 (Se Hyeog Lee) 정회원
1967년 12월 15일생
89년 2월 : 서강대학교 전자계산학과 졸업(학사)
91년 2월 : 서강대학교 전자계산학과 졸업(공학석사)
91년 ~ 현재 : 쌍용컴퓨터 시스템연구소 근무
• 관심분야 : 병렬처리 컴퓨터 구조 분산DB, 멀티미디아