토러스 망에서 라우팅 알고리즘에 대한 패킷 전송 효율의 시뮬레이션

°김현기** · 송호정* · 박준선* · 하기종*** · 이천희*
°충북전문대학 전자통신과** · 영동전문대학 전자통신과*** · 청주대학교 전자공학과*

Simulation of Packet Transmission Efficiency for Routing Algorithm on Torus-Networks

Kim Hyungi** · Song Hojeong* · Park Junseon* · Hah Gijong*** · Yi Cheonhee*
Chungbuk Junior Colleage** · Yeongdong Junior Colleage*** · Chongju University*

요약

본 논문은 토러스 망의 패킷 전송 효율을 평가하기 위해 각 노드가 에지당 2개의 입·출력 큐를 가지는 라우팅 알고리즘을 제시하고 시뮬레이션을 수행하여 그 결과를 분석하였다. 제시된 4가지 오름순차를 가지는 라우팅 알고리즘에 대해 시뮬레이션을 하기 위해 도착분포와 서비스분포는 지수분포를 갖도록하여 교착상태가 없는 패킷 전송 효율을 비교 평가하였다. 실험 결과로부터 무교착 상태로 패킷을 전달하기 위한 노드 큐의 크기를 알 수 있었고, 또한 입력 패킷 메시지의 분포에 따라 패킷의 전송 효율을 높이기 위한 버퍼의 크기를 예측할 수 있었다.

1. 서 론

병렬처리 컴퓨터의 성능은 통신망의 성능에 크게 의존한다. 따라서 병렬처리 컴퓨터(parallel computers)에서 통신 패킷 데이터 처리 및 영상 처리 응용 분야는 매우 효율적으로 이용할 수 있는 메쉬(mesh)와 토러스 네트워크(torus networks)등 여러 가지 네트워크들이 제공된다. 특히, 메쉬와 토러스는 짧은 패스를 사용해 d-차원으로 구성되어질 수 있다. 또한 이것들은 동일 보드상에 구축될 수 있으며, 보드 상호간의 연결을 위해서 적은 핀수가 요구된다. 따라서 이와 같은 잇점 때문에 대부분의 분산 메모리 병렬처리 컴퓨터는 메쉬나 토러스 네트워크를 이용한다. 반면에 메쉬와 토러스 네트워크는 비교적 큰 직경(diameter)과 작게 양분된(bisection) 밴드폭[1][2]을 갖는 결점이 있다. 따라서 이것은 전체 통신 능력을 제한한다.

그러나 같은 크기의 d-차원 메쉬 및 토러스 컴퓨터를 비교해 볼 때 토러스 컴퓨터는 메쉬 컴퓨터 직경의 대략 절반 정도이고 밴드폭은 2배가 된다. 더우기 토러스 네트워크는 대칭 노드이므로 토러스의 모든 노드는 동일시되며, 특히 토러스의 영역은 트래픽 충돌을 갖지 않는다. 반면에 메쉬 네트워크들은 대칭 노드가 아니므로 메쉬의 영역은 비대칭으로 인한 트래픽 충돌을 발생시킬 수 있다. 이와 같은 점으로 미루어 볼때 토러스 네트워크들은 차세대 병렬처리 컴퓨터에 중요한 역할을 담당할 것으로 기대된다.

따라서 본 논문에서도 효율적인 기억 용량을 가지고 무교착 상태로 패킷 정보를 처리하기 위한 방안으로 중앙 큐[2][3][4]를 사용하기 보다는 노드에 입 출력되는 라인마다 2개의 표준 큐(standard queue)[5][6]쌍을 사용하는 라우팅 알고리즘을 기술하고, 본 알고리즘을 이용하여 토러스 네트워크 상의 노드에 할당된 큐에 대해 시뮬레이션을 통해 알고리즘에서 우측 오름순차, 좌측 오름순차, 내측 오름순차 및 외측 오름순차에 대해서 각각 교착상태가 없는 패킷 전송 효율을 비교 평가하였다.

시뮬레이션 환경은 삼성 HP 워크스테이션에서 C언어로 실험 모델을 구현하여 시뮬레이션을 수행하였다.

본 논문의 전체적인 구성은 2장에서 토러스 네트워크의 노드 순차를 소개하고 3장에서 네크워크에 대한 라우팅 알고리즘의 모델과 정의를 기술하며 4장에서는 제시된 알고리즘을 이용해 시뮬레이션을 통한 실험결과를 고찰하고,

마지막으로 결론을 제시하였다.

2. 토러스 네트워크의 노드 순차

본 논문에서는 토러스 네트워크의 라우팅 알고리즘 정의와 교착상태를 해결하기 위한 방안으로 네트워크의 전체 구성 노드들에 대한 순차(ordering) 정의를 이용하게 된다. 이와 같은 순차 정의는 우측 오름순차(right-increasing ordering), 좌측 오름순차(left-increasing ordering), 내측 오름순차(inside-increasing ordering) 및 외측 오름순차(outside-increasing ordering)인 4가지로 분류할 수 있다. 여기서 5×5 토러스 노드에 대해 예를 들면 우측 오름순차는 그림 1(a)와 같이 노드들에 대한 표준 순차이다. 좌측 오름순차는 그림 1(b)와 같이 단순히 우측 오름순차의 역 형태이며, 내측 오름순차는 그림 1(c)와 같이 토러스와 가장자리 근처에 있는 노드들에 가장 작은 값들을 할당한다. 또한 외측 오름순차는 그림 1(d)와 같이 내측 오름순차의 역 형태가 된다. 이와 같은 순차에 대한 일반화한 정의는 다음과 같다. 여기서 정수 i는 $0 \le i < d$로 주어진다.

$$g(i) = \prod_{j=0}^{i-1} n_j$$

여기서 g(0) = 1이 된다. 토러스 노드 라벨 형태는 (a_{d-1}, a_{d-2}, ⋯ , a_0)라 할때

$$Eval(a_{d-1}, a_{d-2}, \cdots, a_0) = \sum_{i=0}^{d-1} g(i)\, a_i$$ 이다.

그러므로 함수 Eval은 노드의 각각에 0에서 n-1 범위까지 단일 정수로 표시하며 노드 라벨로 이용된다.

토러스 노드에 대한 4개의 전체 순차는 어떤 함수에 의해 다시 라벨링하므로서 노드가 얻어지며, 이 경우에 새로운 라벨이 얻어진다. $n_i \ge 2$와 a_i인 어떤 정수가 주어질때 (단, $0 \le a_i \le n_i$), f_R, f_L, f_I, 및 f_O는 다음과 같이 표현할 수 있다.

$$f_R(a_i, n_i) = a_i$$

$$f_L(a_i, n_i) = n_i - a_i - 1$$

$$f_I(a_i, n_i) = \begin{cases} a_i, & a_i < [n_i/2] \\ [3n_i/2] - a_i - 1, & \text{그외} \end{cases}$$

$$f_O(a_i, n_i) = \begin{cases} n_i - a_i - 1, & a_i < [n_i/2] \\ a_i - [n_i/2] & \text{그외} \end{cases}$$

여기서 함수 f_R은 좌측에서 우측 오름순차로 0에서 n_i - 1 숫자를 순서화하고, f_L은 우측에서 좌측 오름순차로 순서화한다. 함수 f_I는 외측에서 내측으로 오름순차로 순서화하고, f_O는 내측에서 외측으로 오름순차로 순서화한다. 따라서 토러스 노드가 (a_{d-1}, a_{d-2}, ⋯ , a_0)로 주어질때, 토러스 노드들의 4가지 순서는 다음과 같이 나타낼 수 있다.

Right(a_{d-1}, a_{d-2}, ⋯ , a_0) = Eval($f_R(a_{d-1}, n_{d-1})$,$f_R(a_{d-2}, n_{d-2})$, ⋯ ,$f_R(a_0, n_0)$)

Left(a_{d-1}, a_{d-2}, ⋯ , a_0) = Eval($f_L(a_{d-1}, n_{d-1})$,$f_L(a_{d-2}, n_{d-2})$,⋯ ,$f_L(a_0, n_0)$)

Inside(a_{d-1}, a_{d-2}, ⋯ , a_0) = Eval($f_I(a_{d-1}, n_{d-1})$,$f_I(a_{d-2}, n_{d-2})$,⋯ ,$f_I(a_0, n_0)$)

Outside(a_{d-1}, a_{d-2}, ⋯ , a_0) = Eval($f_O(a_{d-1}, n_{d-1})$,$f_O(a_{d-2}, n_{d-2})$,⋯ ,$f_O(a_0, n_0)$)

이와 같은 토러스 노드들의 순서는 실제 패킷을 전송할때 패킷의 전송 흐름 순서를 나타낸다. 예를 들면 노드들이 우측 오름순차로 되어 있을때, 노드 a에서 인접 노드 b로 패킷 전송은 우측에서 발생됨을 의미한다.

0	1	2	3	4
5	6	7	8	9
10	11	12	13	14
15	16	17	18	19
20	21	22	23	24

(a) 우측 오름 순차

24	23	22	21	20
19	18	17	16	15
14	13	12	11	10
9	8	7	6	5
4	3	2	1	0

(b) 좌측 오름순차

0	1	4	3	2
5	6	9	8	7
20	21	24	23	22
15	16	19	18	17
10	11	14	13	12

(c) 내측 오름순차

24	23	20	21	22
19	18	15	16	17
4	3	0	1	2
9	8	5	6	7
14	13	10	11	12

(d) 외측 오름순차

(그림 1) 5 × 5 토러스 노드 순차

3 라우팅 알고리즘의 모델과 정의

본 논문에서 사용될 라우팅 모델은 모델을 단순화시키시 위해 축적 제어 라우팅을 사용하며 라우팅 모델에서의 각 노드들은 하나의 패킷에 대해 충분한 기억 장소를 제공하는 큐(Queue)를 가진다. 따라서 토러스 네트워크에서 각각의 노드들은 큐의 세트로 구성된다. 즉 하나는 전달(delivery) 큐이고 다른 하나는 삽입(injection) 큐라하며, 나머지는 표준(standard) 큐라고 한다.

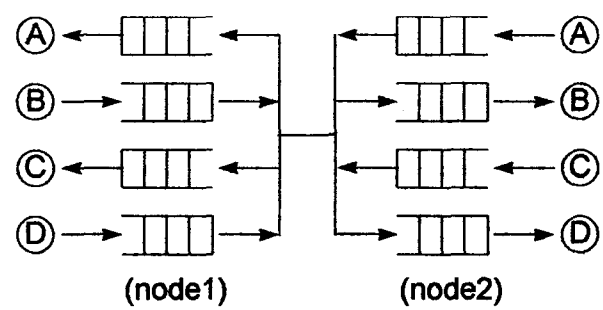
패킷은 소오스 노드에서 빈 삽입 큐에 놓여지게 되므로 토러스 네트워크에 들어갈 수 있다. 그리고 패킷정보들이 목적지 노드의 전달 큐에 있을때, 네트워크로부터 제거될 수 있다. 삽입 큐와 전달 큐는 모델의 설명을 간단히 하기 위해 도입되고, 실제 네트워크에서 나타내지 않으며 실현하는데 중요하다. 결과적으로 라우팅 알고리즘에 요구되는 노드당 큐의 수를 셀때 표준 큐만을 포함하게 된다. R. Cypher와 L. Gravano는 기존에 토러스 네트워크에서 노드당 오직 2개의 중앙 큐(Central Queue)를 사용함으로써 최소의 기억 용량으로 교착상태를 해소시킬 수 있음을 증명하였다[7][8]. 따라서 본 논문에서도 효율적인 기억 용량으로 교착상태를 해결하기 위한 방안으로 중앙 큐를 사용하기 보다는 노드에 입출력되는 라인마다 2개의 표준 큐(standard queue) 쌍을 사용하여, 교착상태를 해결하기 위한 라우팅 알고리즘을 기술하기로 한다. 여기서 2개의 표준 큐쌍을 각 노드에서 라인당 입력 버퍼용 1쌍과 출력 버퍼용 1쌍을 할당하게 된다. 그러므로 각각의 버퍼쌍으로 구성된 큐는 인접 노드간의 유입(incoming) 패킷과 유출(outgoing) 패킷의 상호 교환용으로 이용한다.

토러스 네트워크에서 표준 큐는 A, B, C, D 4개로 분류하여 나타내었다. 여기서 패킷이 내측으로 이동하는 에지는 B와 D큐가 이용되고, 패킷이 외측으로 이동하는 에지는 A와 C큐를 이용하며 이에 대한 구성은 그림 2과 같다. 여기서 노드 x와 인접노드 y간의 패킷 전송은 노드 x와 패킷 목적지 노드간에 노드 x, y사이의 라인을 포함하면서 최소 경로가 존재한다면, 이것은 합당한 패킷 전송이 된다. 이때 패킷에 대해 현재 노드로 유입 및 유출하는 라인이라면, 이 패킷은 내측 및 외측으로 이동하게 된다. 그러므로 라우팅 알고리즘은 소스 노드와 목적지 노드간에 최소 경로를 따라 각각의 패킷을 라우팅하는 것으로, 이에 대한 정의는 다음과 같다. 하나의 패킷이 전달 큐에 저장될때 패킷의 대기 세트(waiting set)는 빈 상태가 된다. 만약 어떤 패킷이 목적지 노드에 있는 전달 큐외의 다른 큐에 저장된다면, 패킷의 대기 세트는 그 노드에서 전달 큐로 구성된다. 이외의 모든 다른 경우에는 다음과 같은 규칙이 이용된다. 어떤 패킷이 삽입 큐인 A 또는 B큐에 저장될때, 패킷의 대기 세트는 내측으로 이동하는 최적의 패킷의 이동을 수행하므로서 입력을 위한 C큐로 구성된다. 패킷이 C큐에 저장될때, 패킷의 대기 세트는 B와 C큐를 구성한다. 이것은 외측으로 적어도 하나의 최적 이동을 가지므로서 제공된다. 이외의 최적 패킷이동을 수행하므로서 입력될수 있는 대기 세트는 D큐를 구성한다. 패킷이 D큐에 저장될때 대기 세트는 최적의 이동을 수행하므로서 D큐를 구성한다. 그리고 교착상태를 해결하기 위해서는 다음과 같이 정의된다.

① 알고리즘에 의해 라우팅된 패킷을 p라하고, (a_{d-1}, a_{d-2}, … , a_0)은 p가 현재 기억된 노드를 표시한다.

② p는 목적지 노드의 전달 큐에 기억되고, p의 대기 세트는 현재 p가 기억된 큐보다 더 높은 순위를 가진 큐를 포함한다.

③ p는 목적지 노드의 전달 큐에 저장되기 전에 유한 수가 큐에 기억된다.

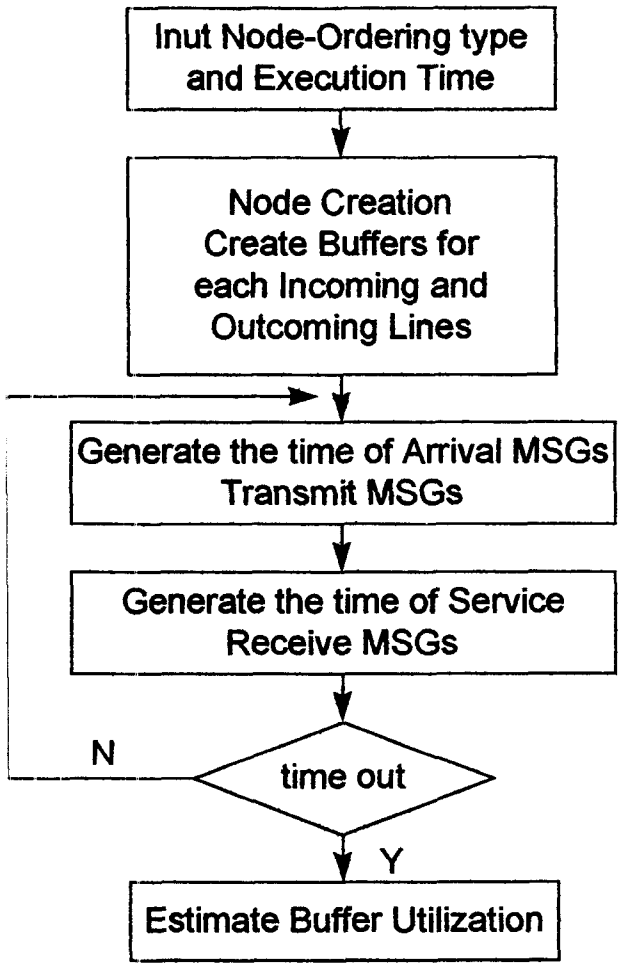


(그림 2) 노드에 대한 유입 및 유출 큐의 구성

4 실험 및 고찰

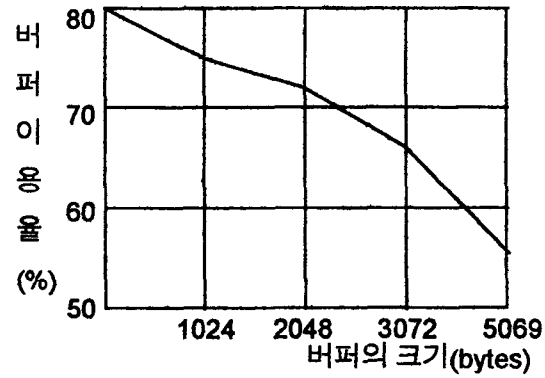
본 장에서는 앞에서 기술된 라우팅 알고리즘에 대해 시뮬레이션 환경을 위해 그림 3과 같이 구현하였다. 여기서 네크워크는 편의상 2차원 4×4(4-ary 2-cube)로 구성하고 각 노드당 표준 큐는 유입 및 유출 라인당 각각 2개의 큐를 가지도록 하였다. 라우팅 알고리즘 정의와 교착상태를 해결하기 위해 2장에서 제시한 네트워크의 전체 구성 노드들에 대한 순차는 우측 오름순차, 좌측 오름순차, 외측 오름순차 및 내측 오름순차를 각각 적용해 구성하였고 여기서 노드들의 순서는 실제 패킷을 전송할때

■킷의 전송 흐름 순서로 결정된다.

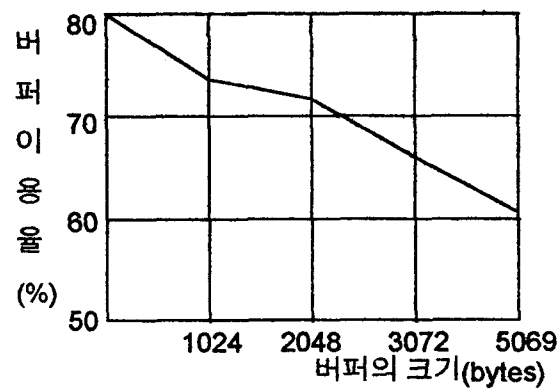


(그림 3) 시뮬레이션 모델 기능 구성도

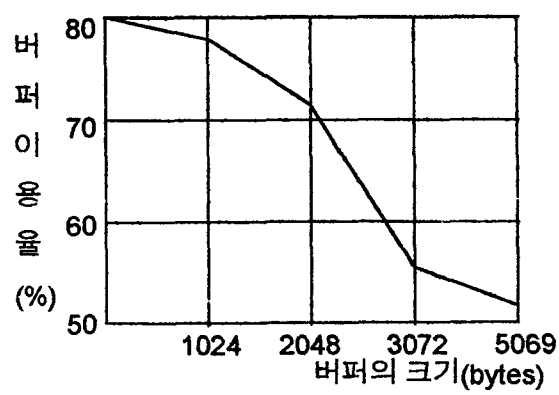
이와 같이 구성된 네트워크에 대한 도착 메시지 발생과 서비스 분포는 지수 분포를 가지며, 패킷의 사이즈는 128 bytes 단위로 구성하여 전송하였으며, 이때 각 노드에 포함된 각각의 큐들의 크기는 512 - 5096 bytes로 구성하였다. 입력 메시지에 대한 노드상의 특정 큐의 성능 분석 결과는 그림 4, 그림 5, 그림 6 및 그림 7과 같은 결과를 얻을 수 있었으며, 이 결과로부터 표 1과 같이 제시된 4가지 알고리즘을 비교하였다. 따라서 버퍼의 크기가 512 bytes인 경우는 내측 오름순차가 가장 높았고, 좌측 오름순차가 가장 낮았다. 또한 버퍼의 크기가 5096 bytes인 경우는 좌측 오름순차가 가장 높았고, 내측 오름순차가 가장 낮았다. 따라서 실험결과로 부터 입력 패킷 메시지의 분포에 따른 효율적인 버퍼의 사이즈를 예측할 수 있다.



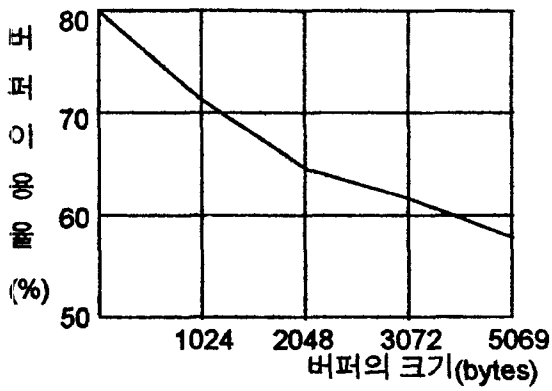
(그림 4) 우측 오름순차를 이용한 버퍼이용률에 따른 최적 버퍼 시뮬레이션



(그림 5) 좌측 오름순차를 이용한 버퍼이용률에 따른 최적 버퍼 시뮬레이션 결과



(그림 6) 내측 오름순차를 이용한 버퍼이용률에 따른 최적 버퍼 시뮬레이션 결과



(그림 7) 외측 오름순차를 이용한 버퍼이용률에 따른 최적 버퍼 시뮬레이션 결과

(표 1) 우측 오름순차, 좌측 오름순차, 내측 오름순차 그리고 외측 오름순차를 이용한 버퍼이용률 비교

버퍼크기 (bytes) / 버퍼이용율 (%)	512	2048	5096
우측오름순차	77.5	72	56
좌측오름순차	75	72.5	60.5
내측오름순차	79	71.5	52
외측오름순차	75.5	65	58

5 결론

본 논문에서는 토러스 네트워크에 대해 입력된 패킷이 교착상태없이 네트워크상의 노드사이에 효율적으로 전송될 수 있도록 노드에 유입 및 유출 큐를 각각 2개씩 할당한 패킷 라우팅 알고리즘을 제시하였다. 그리고 실제 패킷을 전송할 때 네트워크상에서 패킷의 전송 흐름 순서는 2장에서 제시된 토러스 네트워크의 전체 구성 노드들에 대한 순차, 즉 우측 오름순차, 좌측 오름순차, 내측 오름순차 및 외측 오름순차로 노드들을 배열하여 각각에 대한 패킷 전송 효율을 시뮬레이션을 통해 비교 평가하였다.

실험결과는 도착 분포와 서비스 분포가 지수 분포를 갖도록하여 입력 패킷에 대한 노드에 할당된 큐들을 무교착 상태로 패킷 전송이 이루어짐을 확인하였고, 또한 큐에 할당된 버퍼의 크기는 사용 기억 용량에 대한 효율로서, 입력 패킷 메시지와 분포에 따라 효율적인 패킷 전송을 할 수 있음을 알 수 있었다.

참고문헌

1. Clark D. Thompson. Area-time complexity for VLSI. In Proc. 11th Annual Symposium on Theory of Computing, pp.81-88, 1979.
2. C.R.Jeshope, P.R.Miller, and J.T.Yantchev. High performance communication in processor networks. In Proc. 16th Intl. Symposium on Computer Architecture, pp.150-157, 1989.
3. J.Yantchev and C.R.Jesshope. Adaptive, low latency, deadlockk-free packet routing for networks of processors. IEEE Proc, Pt.E,136(3):178-186,May 1989.
4. G.-M. Chiu, S. Chalasani, and C. S. Raghavendra. Flexible, faulttolerant routing criteria for circuit-switched hypercubes. In Proc. 11th International Conference on Distributed Computing Systems. IEEE, 1991.
5. Sergio A. Felperin, Hernán Laffitte, Guillermo Buranits, and Jorge L.C.Sanz. Deadlock-free minimal packet routing in the torus network. Technical Report TR:91-22, IBM Argentina, CRAAG, 1991.
6. W. H. Dally and C. Seitz. Deadlock-free message routing in multiprocessor interconnection networks, IEEE Transactions on Computers, 36:547-553, May 1987.
7. Robert Cypher and Luis Gravano. Requirements for deadlock-free adaptive packet routing. In Proc. 11th Annual ACM Symposium on Principles of Distributed Computing, pp.25-33, 1992.
8. K.D.Günther. Prevention of deadlocks in packet-switched data transport systems. IEEE Transactions on Communications, 29(4), April 1981.