토러스 네트워크에서 무교착 멀티캐스트 알고리즘의 성능분석

(Performance Analysis of Deadlock-free Multicast Algorithms in Torus Networks)

원 복 희 † 최 상 방 ††
(Bok-Hee Won) (Sang-Bang Choi)

요 약 본 논문에서는 양방향 토러스 네트워크와 웜홀 라우팅을 사용하는 다중컴퓨터에 대해 멀티캐스트 통신방법을 트리방식, 경로방식, 그리고 두 방식을 조합한 하이브리드방식으로 분류하였다. 경로방식으로는 동적분할 멀티캐스트 라우팅 알고리즘을 제안하였으며, 하이브리드방식으로는 라우팅의 첫 단계로 트리방식을 사용하고 두 번째 단계로는 경로방식을 사용하는 멀티캐스트 라우팅 알고리즘을 제안하여 성능을 분석하였다. 세가지 멀티캐스트 라우팅 알고리즘간의 성능은 메세지 길이에 따른 평균 지연시간을 사용하여 비교하였다. 그리고 웜홀 라우팅에서 플릿 버퍼 크기의 변화에 따른 성능을 가상 컷-스루와 비교하였으며, 경로방식의 알고리즘을 사용하여 버퍼 크기의 변화에 따른 지연시간을 기준으로 두 스위칭 방식의 성능관계를 분석하였다.

Abstract In this paper, we classify multicast methods into three categories, i.e., tree-based, path-based, and hybrid-based multicasts, for a multicomputer employing the bidirectional torus network and wormhole routing. We propose the dynamic partition multicast routing (DPMR) as a path-based algorithm. As a hybrid-based algorithm, we suggest the hybrid multicast routing (HMR), which employs the tree-based approach in the first phase of routing and the path-based approach in the second phase. Performance is measured in terms of the average latency for various message length to compare three multicast routing algorithms. We also compare the performance of wormhole routing having variable buffer size with virtual cut-through switching. The message latency for each switching method is compared using the DPMR algorithm to evaluate the buffer size trade-off on the performance.

1. 서 론

최근 수퍼 컴퓨터들의 경향은 프로세서의 수를 증가시켜 성능을 향상시킬 수 있는 확장 가능한 병렬컴퓨터로 설계 해오고 있다. 이런 시스템을 일반적으로 대규모 병렬처리 컴퓨터(Massively Parallel Computer: MPC) 또는 다중컴퓨터(multicomputer)라하며, 이것은 여러 개의 노드(node)들로 구성되어 있다. 각각의 노드는 자신의 프로세서, 지역 메모리(local memory) 그리고 라우터(router)등으로 구성되며, 각 노드사이의 통신은 통신 채널을 통하여 메세지(message)를 전달함으로 이루어진다. 메세지란 노드간에 정보를 교환하는 논리적인 단위이다. 이러한 메세지-전달 다중컴퓨터 시스템의 라우팅 알고리즘을 설계하는데 고려할 사항들은 다음과 같다[1]. 첫째로 서비스되는 통신 형태의 결정이고, 둘째로 네트워크의 위상(topology), 셋째로 메세지 전달 방법이다.

본 논문에서 고려한 시스템은 서비스되는 통신 형태로는 멀티캐스트를, 네트워크 위상으로는 양방향 토러스 네트워크를, 메세지 전달 방법으로는 웜홀(wormhole) 라우팅을 각각 사용한다[2]. 여기서 멀티캐스트란 하나

· 이 연구는 1998년도 인하대학교 연구비 지원에 의하여 수행되었음.
† 비 회 원 : (주)현대멀티캡 연구원
bhwon@multicav.co.kr
†† 정 회 원 : 인하대학교 전자공학과 교수
sangbang@inha.ac.kr
논문접수 : 1999년 4월 12일
심사완료 : 2000년 1월 25일

의 출발지 노드에서 다수의 목적지 노드로 동일한 정보를 전달하는 one-to-many 방식의 통신형태를 나타내며, 웜홀 라우팅은 메세지를 플릿(*flit*)이라는 작은 단위로 자른 후 파이프라인 방식으로 목적지 노드까지 메세지를 전달하는 방법이다. 최근 대부분의 다중 컴퓨터에서는 메세지가 목적지까지 도달하는데 걸리는 지연시간이 짧고, 각 노드에서 사용하는 버퍼가 작다는 장점 때문에 거의 웜홀 라우팅을 사용하고 있다.

본 논문에서는 멀티캐스트 라우팅 방식을 트리방식(tree-based), 경로방식(path-based), 하이브리드방식(hybrid-based)으로 분류하여 각 방식의 알고리즘 성능을 비교 분석하였다. 트리방식에서는 출발지 노드로부터 모든 목적지 노드까지 일련의 스텝에 따라 메세지가 진행할 때 전달되는 패턴이 트리 형태를 이루며, 매 스텝마다 두배씩 목적지 노드가 통신에 참여하게 된다 [3, 4]. 따라서 m-1 곳의 목적지에 메세지를 전달하기 위해서는 $\lceil \log_2 m \rceil$의 스텝이 걸리게 된다. 이런 트리방식은 패킷 스위칭이나 가상 컷-스루(virtual cut-through: VCT)를 사용하는 네트워크에서 비교적 좋은 성능을 나타내지만, 웜홀 라우팅을 사용하는 네트워크에서는 지연시간이 길어지고 교착상태가 발생할 수 있다.

경로방식이란 출발지 노드에서 모든 목적지 노드를 연결하는 최단 경로 설정하여 그 경로를 따라 메세지를 진행시키는 방식이다. Lin은 최초로 여러 내부 채널 쌍을 가진 다중포트 2-D 메쉬 구조에서 멀티캐스트 웜홀 라우팅을 사용하여 경로방식이 트리방식보다 좋은 성능을 갖는다고 주장하였으며[5], Malumbres는 캐쉬 일관성(cache coherence)을 유지하는 분산된 공유 메모리 다중프로세서 구조에서 공유 데이타의 무효화(invalidation)나 갱신(updating)과 같이 메세지 크기가 작은 경우 트리방식이 경로방식보다 좋은 성능을 갖는다는 것을 보였다[6]. Robinson은 단방향 토러스 구조에서 해밀톤 경로(Hamiltonian path)를 따라 메세지를 전송하는 멀티캐스트 알고리즘들을 제안하였다[7]. 하이브리드방식은 라우팅의 첫 단계로 트리방식을 사용하고 두 번째 단계로는 경로방식을 사용하는 트리방식과 경로방식이 혼합된 멀티캐스트 라우팅 알고리즘이다.

위의 세가지 라우팅 알고리즘간의 성능은 메세지가 출발지에서 목적지까지 도달하는데 걸리는 평균 지연시간을 사용하여 비교하였다. 트리방식으로는 유니캐스트를 이용하여 소프트웨어적으로 멀티캐스트를 수행하는 Robinson의 U-torus 알고리즘을 사용하였으며[3], 경로 방식과 하이브리드방식으로는 본 논문에서 제안한 동적분할 멀티캐스트 라우팅(Dynamic Partition Multicast Routing: DPMR) 알고리즘과 하이브리드 멀티캐스트 라우팅(Hybrid Multicast Routing: HMR) 알고리즘을 각각 사용하였다. 또한 경로방식에서 웜홀 라우팅의 버퍼의 크기를 VCT의 버퍼 크기가 될 때까지 점차 증가시켜 가면서 두 스위칭 방식의 성능을 비교하여 보았다. 웜홀 라우팅에서 버퍼의 크기가 메세지 길이와 같아지면 VCT가 된다. 이것은 버퍼 크기의 증가에 따른 웜홀 라우팅과 VCT의 성능관계를 알아보기 위함이다.

라우팅 알고리즘간의 성능을 비교하기 위한 시뮬레이션 결과, 16×16 네트워크에서 트래픽이 적고 메세지 길이가 짧으며 목적지 노드가 20개 이하인 경우에는 경로방식이 가장 좋은 성능을 보이며, 트리방식 그리고 하이브리드방식의 순서로 성능이 나타나고 있다. 그러나 목적지 노드의 수가 그 이상으로 증가하면 하이브리드방식이 트리방식보다 더 좋은 성능을 보이며, 100개 이상으로 증가하게 된다면 하이브리드방식이 다른 두 방식에 비해 가장 좋은 성능을 나타낸다. 그러나 네트워크에 트래픽이 많고 메세지 길이가 네트워크 크기의 4배 이하일 때는 하이브리드방식, 경로방식 그리고 트리방식의 순서로 성능을 나타내며, 메세지 길이가 그 이상 증가하면 경로방식이 가장 좋은 성능을 보였다. 웜홀 라우팅에서 버퍼크기가 성능에 미치는 영향을 분석하기 위하여 16×16, 32×32인 토러스 네트워크에서 플릿의 크기는 1, 메세지의 길이는 64로 설정하여 시뮬레이션을 수행하였다. 웜홀 라우팅의 버퍼의 크기가 증가함에 따라 메세지의 지연시간이 급격히 감소하나, 그 크기가 8이상 되면 지연시간은 거의 감소하지 않는 것을 볼 수 있다. 일반적으로 웜홀 라우팅의 버퍼 크기가 VCT 버퍼 크기의 1/8정도가 되면 VCT 방식과 거의 같은 성능을 나타내는 것을 알 수 있다.

본 논문의 구성은 다음과 같다. 2장에서는 본 논문에서 사용된 네트워크 모델과, 무교착(deadlock-free)을 위한 가상채널(virtual channel) 모델, 그리고 그러한 가상채널 모델을 사용하여 고안된 동적분할 및 하이브리드 라우팅 알고리즘에 대해 설명한다. 3장에서는 시뮬레이션을 통하여 세가지 라우팅 방법의 성능을 비교하며, 마지막으로 4장에서는 제안된 알고리즘의 성능을 정리하여 결론을 맺는다.

2. 동적분할 및 하이브리드 라우팅 알고리즘

2.1 네트워크 모델

분산된 공유 메모리 다중프로세서는 여러 개의 노드로 구성되어 있으며, 각 노드는 계산을 위한 프로세서와

지역 메모리, 그리고 메세지를 다른 노드로 전송 할 수 있는 라우터로 구성되어 있다. 다중컴퓨터는 메모리가 물리적으로 분산되어 있기 때문에 노드들은 네트워크를 통해 메시지를 전송하여 서로 통신을 하게된다.

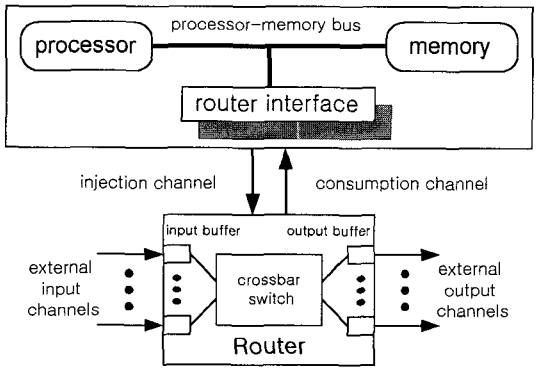


그림 1 단일포트 노드구조

그림 1은 한 쌍의 내부 채널(internal channel)을 가진 단일포트 노드의 구조를 나타낸 것으로, 주입(injection)채널은 네트워크를 통하여 다른 노드로 메세지를 전송하는데 사용하며, 소비(consumption)채널은 네트워크로부터 메세지를 받는데 사용한다. 단일포트란 네트워크의 위상과 관계없이 각 노드가 한번에 오직 하나의 메세지를 보내고 받을 수 있는 것을 말하며, 물리적인 내부채널의 수와 동일하다. 라우터 내의 크로스바 스위치는 입력 채널에서 출력 채널로 메세지를 전달하는데 사용된다.

본 논문에서 멀티캐스트 알고리즘의 성능을 비교하기 위한 네트워크 구조로는 K×K 토러스를 사용한다. 웜홀 라우팅에서 버퍼의 크기가 성능에 미치는 영향을 분석하기 위한 시뮬레이션에서는 그림 1의 출력 버퍼의 크기를 점차 증가시켜가면서 그에 따른 지연시간을 분석한다. 웜홀 라우팅에서 출력 버퍼의 크기를 증가시킨다고 그만큼 비례하여 성능이 향상되지는 않으므로, VCT에 비하여 성능이 크게 저하되지 않는 범위의 적절한 버퍼의 크기를 시뮬레이션을 통하여 구한다.

2.2 동적분할 멀티캐스트 라우팅 알고리즘

웜홀 라우팅에서는 노드간에 동기를 맞추어 파이프라인 방식으로 메시지가 전달되며, 일반적으로 단위 시간(클락)에 한 플릿씩 전달되는 것으로 가정한다. 따라서 경로방식을 따르는 알고리즘은 네트워크를 어떻게 분할하여 메시지를 전달하느냐에 따라서 성능이 다르게 나타난다. Lin은 다중포트(all-port) 네트워크 구조에서 출발지 노드의 위치에 따라 네트워크를 다르게 분할하는 방식을 사용하였으며, 네트워크를 두 부분으로 분할하는 알고리즘이 가장 좋은 성능을 보였다[5].

본 논문에서는 메세지의 길이에 따라 네트워크의 분할여부가 결정되는 동적인 분할방식을 사용한다. 메시지 길이는 그 메시지를 구성하는 플릿의 개수로 정의한다. 먼저 무교착 라우팅을 위해 네트워크의 각 노드에 해밀톤 경로를 따라 번호를 할당한다. 해밀톤 경로란 주어진 그래프내의 모든 노드를 정확히 한번 방문하는 것으로 경로방식의 멀티캐스트 알고리즘들은 주로 이 방법을 사용하여 각 노드들에 번호를 할당하고 있다. 해밀톤 경로방식의 알고리즘은 일단 네트워크가 해밀톤 경로를 가져야 한다는 단점이 있지만, 네트워크 크기가 작고 목적지 노드의 수가 많은 경우는 상당히 좋은 성능을 나타내는 방식이다. 본 논문에서 다루는 K×K 토러스 네트워크는 많은 해밀톤 경로를 가지고 있지만, 다음과 같은 식(1)을 사용하여 노드에 번호를 할당하였다.

$$l(x,y) = \begin{cases} x\times K+y, & \text{if } x \text{ is even,} \\ (x+1)\times K-y-1, & \text{if } x \text{ is odd.} \end{cases} \quad (1)$$

즉 원점 (0, 0)을 기준으로 오른쪽으로 진행하면 x좌표가 증가하고, 위쪽으로 진행하면 y좌표가 증가하는 정수 좌표(x, y)로 표현된 네트워크에 N(=K×K)개의 노드가 있다면, 원점노드를 0번으로 하여 y좌표가 증가함에 따라 번호를 할당하고, y좌표가 K-1에 도착하면 이번에는 x좌표를 1 증가한 후 다시 y좌표가 0이 될 때까지 번호를 할당한다. 결국 x좌표가 K-1이 되고 모든 노드에 번호를 할당할 때까지 반복한다. 토러스 구조의 교착상태를 제거하기 위하여 초기에는 네트워크의 일부를 사용하지 못하도록 제한을 두었지만, 이러한 방법은 성능에 많은 영향을 주고 최소 라우팅을 보장하지 못한다. 최소 라우팅(minimal routing)이란 라우팅 할 때 허용되는 채널의 집합이 최단경로를 보장하는 채널들로만 이루어지는 것을 말한다. 근래에는 Dally에 의해 제안된 가상채널 개념을 사용하는 것이 일반화 되어있다[8]. 즉 물리적인 채널 하나를 여러 개의 논리적인 가상채널이 공유하여 마치 여러 개의 채널이 연결되어있는 것처럼 사용하는 것이다. 가상채널을 사용하므로 생기는 장점은 교착상태의 제거뿐만 아니라, 라우팅의 적응성(adaptability)과 고장허용(fault tolerance)을 증가시켜주며, 물리적 채널 이용도도 증가 하게된다. 그러나 가상채널을 많이 사용하면 할수록 메세지들의 스케쥴링이 복잡해지며, 추가적인 하드웨어를 필요로 한다.

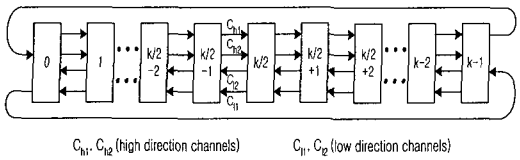


그림 2 동적분할 경로방식의 가상채널 구성

그림 2는 양방향 토러스 네트워크 구조의 교착상태를 제거하기 위하여 사용한 가상채널 모델로, 각각의 방향에 두개의 가상채널을 사용하여 사이클을 제거하고 있다. 채널 C_{h1}과 C_{l1}은 메세지를 전송하기 위해 먼저 사용되어지는 정규채널 층(normal channel layer)으로, C_{h1}은 현재 노드 번호 보다 높은 번호를 가진 노드로 라우팅 할 때, C_{l1}은 현재 노드 번호보다 낮은 번호를 가진 노드로 라우팅 할 때 사용된다. C_{h2}와 C_{l2}는 각각 C_{h1}과 C_{l1}의 교착상태를 제거하기 위해 사용되는 가상채널 층(virtual channel layer)이다. 만약 목적지 노드의 번호가 출발지 노드의 번호보다 큰 경우 높은 채널 서브네트워크(high-channel subnetwork)로 진행하게 되고, 만약 그렇지 않다면 낮은 채널 서브네트워크(low-channel subnetwork)로 진행하게 된다. 높은 채널 서브네트워크란 그림 2의 우측 방향 채널 즉 C_{h1}과 C_{h2} 들만 사용하여 라우팅하는 서브네트워크를 말하며, 낮은 채널 서브네트워크란 좌측 방향 채널 즉 C_{l1}과 C_{l2} 들만 사용하여 라우팅하는 서브네트워크를 말한다.

라우팅 알고리즘이 수행되기 전에 네트워크의 동적분할 여부와 각 영역에 포함되는 노드를 결정하기 위하여 그림 3과 같은 라우팅 준비작업 알고리즘이 선행되어진다. 다음 알고리즘에서 식 (3)은 출발지 노드에서 메세지가 먼저 출발하는 방향의 영역에 포함되는 노드들을 결정하는 식으로, 두번째 항의 l_m은 메세지 길이만큼 먼저 출발한다는 것을 의미하며 첫 번째 항은 전체경로에서 그 부분을 뺀 나머지 경로의 길이를 이등분하여 영역을 결정하는 것을 의미한다.

동적분할을 위한 라우팅 준비작업 알고리즘

Input: /* K×K 토러스 네트워크에서 $l(x_i, y_i)$는 식(1)에 의해 할당된 노드 번호 */
노드 집합 D = {$l(x_0, y_0)$, $l(x_1, y_1)$ ⋯ $l(x_i, y_i)$}; (x_0, y_0)는 출발지 노드,
(x_1, y_1) ⋯ (x_i, y_i)는 i개의 목적지 노드,
메시지 길이 l_m, 네트워크의 해밀톤 경로길이 l_{hp}.

Output: 분할되지 않은 네트워크 또는 분할된 두 개의 서브네트워크,
N_H: high-channel subnetwork, N_L: low-channel subnetwork.

Procedure:

1. 출발지 노드가 포함된 집합 D의 각 노드에 할당된 번호를 사용하여 오름차순으로 정렬한다.
2. **만약** $l_m \geq l_{hp}$ 면, 네트워크를 분할하지 않는다.
   그렇지 않으면, /* 분할된 영역 결정 */
   정렬된 집합 D에서 이웃노드 간의 좌표를 이용하여 총 경로길이를 구한다.

$$total\ path\ length = \sum_{n=1}^{K-1} (|x_n - x_{n+1}| + |y_n - y_{n+1}|) \quad (2)$$

   만약 $l(x_0, y_0) > l(x_{K-1}, y_{K-1})/2$ 면,
   N_H 에 포함되는 노드는 식 (3)에 의해 계산하고,
   나머지 노드는 N_L에 포함된다.
   그렇지 않으면,
   N_L에 포함되는 노드는 식 (3)에 의해 계산하고,
   나머지 노드는 N_H에 포함된다.

$$N_H(\text{or } N_L) = \left\lceil \frac{total\ path\ length - l_m}{2} \right\rceil + l_m \quad (3)$$

그림 3 경로방식의 동적분할을 위한 라우팅 준비작업 알고리즘

그림 4는 6×6 토러스 네트워크에서 출발지 노드가 (4, 3)이고, 목적지 노드가 16개, 그리고 메세지 길이가 10일 때 라우팅 준비작업 절차에 의해 네트워크가 양분할되어 각각의 방향으로 메세지를 전달하고 있는 예를 보여주고 있다. 메세지 길이(l_m=10)가 네트워크의 해밀톤 경로의 길이(l_{hp}=35)보다 짧으므로 네트워크는 분할되며, 식(1)에 의해 할당된 노드 번호 순서로 정렬된 집합 D = {(0, 2), (0, 5), ⋯ (5, 2), (5, 0)}에서 이웃노드끼리 경로길이를 식(2)에 의해 계산하면 총 경로길이는 31이 된다. 출발지 노드의 위치가 네트워크의 해밀톤 경로상에서 중간에 위치한 노드보다 위쪽에 존재하므로 N_H영역으로 먼저 출발하게 되고, 식(3)에 의해 N_H의 영역에는 출발지 노드를 중심으로 21 만큼의 거리에 존재하는 노드들이 포함되며 나머지 노드는 N_L의 영역에 포함되게 된다. N_H 방향으로 먼저 출발하게 되는 이유는 네트워크에 트래픽이 집중되지 않고 모든 채널을 고르게 사용하기 위함이다. 만약 경로 차단이 발생하지 않는다면, N_H 영역의 최종 목적지인 (2, 2)노드까지 지연시간은 30, N_L 영역의 최종 목적지인 (2, 3)노드까지 지연시간은 31로 시간상으로 볼 때 거의 동시에 도달하는 것을 알 수 있다. 네트워크를 동적으로 분할하는데 있어서 메세지의 길이는 중요한 역할을 하고 있다. 그림 4의 경우라면 메세지의 길이가 점점 길어질수록 N_H가 차지하는 영역이 넓어 지게되며, 메세지의 길이가 35이상이 되는 경우에는 네트워크가 분할되지 않고 한쪽 방향으로만 진행하게 되고 그 경우 해밀톤 경로방식과 동일하게 진행된다.

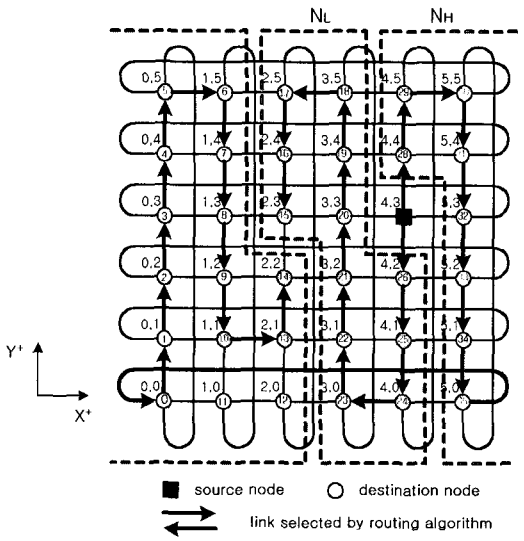


그림 4 동적분할 경로방식 라우팅 예

그림 5는 16×16 네트워크에서 목적지 노드 100개를 임의로 10번 선택하여 각각에 대하여 DPMR 알고리즘을 사용하여 라우팅을 행한 후, Y축 채널의 평균 채널 이용도(utilization)를 계산하여 그래프로 표현한 것이다. 이 그래프는 메세지의 길이에 따라 각 채널이 어느 정도 사용되는가를 보여주고 있다. 여기서 메세지 길이가 32, 64, 128일 때는 네트워크가 알고리즘에 의해 양분할되어 라우팅이 수행되며, 메세지 길이가 256인 경우에는 네트워크가 분할되지 않고 해밀톤 경로방식으로 수행된다. 네트워크의 분할 여부에 상관없이 모든 경우에 각 채널이 고르게 사용되는 것을 그래프로부터 알 수 있다. 다시 말하면 트래픽이 네트워크의 전체 채널에 고르게 분포되어 특정 채널에 집중되지 않는다는 것을 설명해 준다. 위에서 예를 들어 설명한 경로방식의 DPMR 알고리즘을 형식에 맞추어 표현하면 그림 6과 같다.

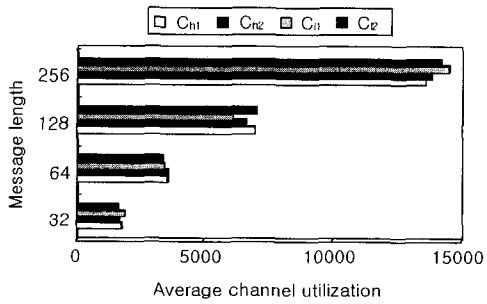


그림 5 동적분할 경로방식의 채널이용도

동적분할 경로방식의 라우팅 알고리즘

Input: 현재노드 n, 출발지 노드 s, 정렬된 목적지 노드의 집합 $D = \{d_1, d_2, \cdots\cdots d_k\}$.

Procedure:

1. 라우팅 준비작업 알고리즘에 의해 네트워크의 분할 여부를 결정한다.
2. **만약** 네트워크가 분할되면,
   만약 출발지 노드의 위치가 해밀톤 경로상에서 중간에 위치한 노드보다 위쪽에 존재하면,
   1) N_H네트워크로 먼저 메시지를 보낸다.
   2) 마지막 플릿을 보낸 후 곧바로 동일 메시지를 N_L네트워크로 진행시킨다.
   그렇지 않으면,
   1) N_L네트워크로 먼저 메시지를 보낸다.
   2) 마지막 플릿을 보낸 후 곧바로 동일 메시지를 N_H네트워크로 진행시킨다.
   그렇지 않으면, /* 네트워크가 분할되지 않는 경우 */
   만약 출발지 노드의 위치가 해밀톤 경로상에서 중간에 위치한 노드보다 위쪽에 위치하면, N_H네트워크로 메시지를 보낸다.
   그렇지 않으면,
   N_L네트워크로 메시지를 보낸다.
3. 네트워크의 모든 목적지 노드로 메세지가 전달될 때까지 다음을 반복한다.
   /* d_1은 현재 집합 D에 남아있는 첫 번째 목적지 노드 */
   만약 $n = d_1$ 면, $D = D-\{d_1\}$이 되고 메세지를 현재 노드에 복사한다.
   그렇지 않으면, $D = D$로 한다.
   만약 $D = 0$ 면, 메세지 전송을 끝낸다.
   그렇지 않으면, 아래 라우팅함수 $R(n, d_1)$을 사용하여 다음 노드 n' 을 결정한 후 메시지를 그 노드로 전송한다.

라우팅 함수 $R(n, d)$에 의한 노드 선택:

/* 현재 노드 좌표 (n_x, n_y), 목적지 노드 좌표(d_x, d_y) */

(1) **만약** $d_x - n_x > 0$ 면, 출발지의 오른쪽 채널을 사용하여 전송한다.
(2) **만약** $d_x - n_x < 0$ 면, 출발지의 왼쪽 채널을 사용하여 전송한다.
(3) **만약** $d_x - n_x = 0$ **그리고** $d_y - n_y > 0$ 면, 출발지의 위쪽채널을 사용하여 전송한다.
(4) **만약** $d_x - n_x = 0$ **그리고** $d_y - n_y < 0$ 면, 출발지의 아래쪽 채널을 사용하여 전송한다.

그림 6 동적분할 경로방식의 라우팅 알고리즘

네트워크에서 교착상태의 발생 가능성은 다음과 같이 두 가지 경우가 있을 수 있다. 첫 번째 경우, 서로 다른 채널 층 사이에서 교착상태가 발생될 수 있다. 그러나 그림 2의 채널모델에서 각각의 방향에 가상채널을 사용하여 실제 라우팅 할 때 같은 채널 층을 두 번 사용하지 않게 하였으므로 채널 층 사이에서는 사이클이 존재하지 않게 되어 교착상태가 발생하지 않는다. 두 번째 경우, 자신의 채널 층 내에서 교착상태가 발생할 수 있다. 그림 7은 4개의 노드 n_0, n_1, n_2, n_3이 그림과 같이

네트워크 내에 위치하고 $n_0 < n_1 < n_2 < n_3$의 순서로 노드번호가 할당될 때, (a)는 라우터 내의 각 방향 출력 버퍼만을 표시한 것이며, (b)는 C_{h1}채널 층에서 (c)는 C_{l1}채널 층에서 각각 무교착 라우팅임을 보여주고 있다.

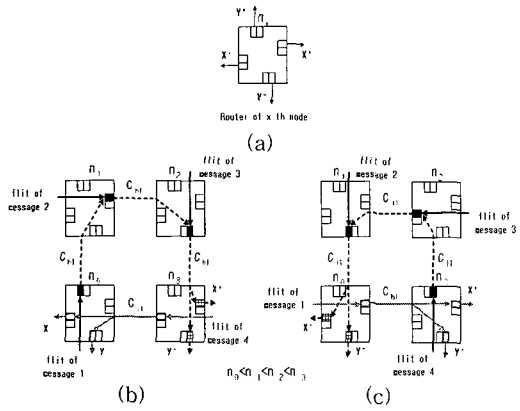


그림 7 각각의 채널 층 내에서 무교착 라우팅 : (a) 라우터내의 각 방향 출력버퍼, (b) C_{h1}채널 층의 무교착, (c) C_{l1}채널 층의 무교착

그림 7 (b)의 경우 C_{h1}채널 층에서 사이클이 형성되어 교착상태가 발생하려면 $n_0 \rightarrow n_1 \rightarrow n_2 \rightarrow n_3 \rightarrow n_0$ 노드사이에 모두 C_{h1}채널을 사용하여 라우팅을 해야한다. 즉, n_0노드로 들어오는 첫 번째 메세지의 플릿은 C_{h1}채널을 사용하여 $n_0 \rightarrow n_1 \rightarrow n_2$로 진행하려하나 현재 두 번째 메세지의 플릿에 의해 경로가 차단되어 n_0노드에서 대기하고 있고, n_1노드로 들어오는 두 번째 메세지의 플릿은 C_{h1}채널을 사용하여 $n_1 \rightarrow n_2 \rightarrow n_3$로 진행하려하나 현재 세 번째 메세지의 플릿에 의해 경로가 차단되어 n_1노드에 대기하고 있는 상태를 나타낸다. 그러나 n_2노드로 들어오는 세 번째 메세지의 플릿은 $n_2 \rightarrow n_3$으로 진행 할 때 동적분할 라우팅 알고리즘에 의해 점선으로 표시되는 방향의 채널 버퍼 즉 X^+ 나 Y^- 방향으로만 진행하게 되고, n_3 노드로 들어오는 네 번째 메세지의 플릿은 $n_3 \rightarrow n_0$로 진행할 때 노드번호가 감소하므로 C_{l1}채널을 사용하여 진행하게되어 C_{h1}채널 층 내에서 사이클이 발생되지 않는다. 그림 7 (c)의 경우도 (b)경우와 동일하게 C_{l1}채널 층에서 교착상태가 발생하려면 $n_3 \rightarrow n_2 \rightarrow n_1 \rightarrow n_0 \rightarrow n_3$ 노드사이에 모두 C_{l1}채널을 사용하여 라우팅을 해야 하지만, n_1 노드로 들어오는 두 번째 메세지의 플릿은 $n_1 \rightarrow n_0$으로 진행할 때 제안된 라우팅 알고리즘에 의해 X^- 나 Y^- 방향으로 진행하게 되고, n_0노드로 들어오는 첫 번째 메세지의 플릿은 $n_0 \rightarrow n_3$로 진행할 때 노드번호가 증가하므로 C_{h1}채널을 사용하여 진행하게되어 사이클이 발생하지 않는다. C_{h2}채널 층과 C_{l2}채널 층에서도 각각 그림 7 (b), (c)의 경우와 동일하게 된다. 결론적으로 C_{h1}(또는 C_{h2})채널은 X^+, Y^+, Y^- 방향으로만 진행하며, C_{l1}(또는 C_{l2})채널은 X^-, Y^+, Y^- 방향으로만 진행하게되어 자신의 채널 층 내에서 사이클이 형성되지 않는다. 그러므로 DPMR 알고리즘은 무교착 라우팅 알고리즘이다.

2.3 하이브리드 멀티캐스트 라우팅 알고리즘

경로방식을 사용하면 트리방식이 스텝별로 동기화되어 진행해야하기 때문에 발생하는 성능감소는 줄일 수 있다. 그러나 헤더 플릿(header flit)이 한번 차단되면 연속적으로 따라오는 나머지 플릿들은 더 이상 진행 할 수 없게 된다. 또한 이 방식의 가장 큰 문제점은 일단 해밀톤 경로가 존재해야 한다는 것이다. 최근 다중 컴퓨터의 경향이 점차 적응형 라우팅을 지원하는 방향으로 가고 있기 때문에, 적응형 알고리즘과 공존할 수 없는 해밀톤 경로방식은 적용 범위가 줄어들 것이다. 따라서 본 논문에서는 트리방식과 경로방식의 단점을 서로 보완하여 토러스 네트워크에 적용할 수 있는 일종의 혼합방식인 HMR 알고리즘을 제안한다. 이 방식을 직관적으로 설명하면 처음 몇 스텝은 트리방식으로, 이후 나머지 노드는 경로방식을 사용하여 보내는 것으로, 최초로 Panda가 메쉬구조의 네트워크에서 사용하였다[9]. 하이브리드방식은 한번의 스텝으로 보낼 수 있는 목적지 노드끼리 먼저 그룹을 만들고, 그 그룹의 대표노드에게 트리방식으로 메세지를 전달한 후, 대표노드가 나머지 노드에게 경로방식으로 메세지를 보내는 방식이다. 최악의 경우 이 방식은 트리방식과 동일한 스텝 수를 갖지만, 네트워크에서 목적지 노드수가 점차 증가할수록 스텝 수는 어떤 일정한 값으로 수렴하기 때문에 트리방식보다 좋은 성능을 나타낼 수 있다. 그러나 이 방식은 많은 소비채널을 필요로 하며, 소비채널로 인한 교착상태도 제거를 해야만 한다. 최근에 소비채널로 인한 성능감소를 완화하기 위해 다중 소비채널을 사용하는 개념이 제안되었으며, 현재 기술로 노드 당 2~4개의 소비채널을 사용하는 것이 가능하나 전체 비용을 고려할 때 2개를 사용하는 것이 가장 바람직한 것으로 여겨지고 있다[10].

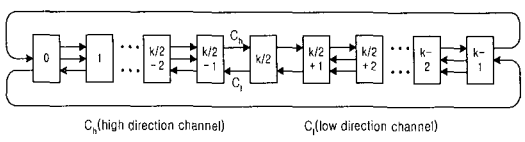


그림 8 하이브리드방식의 가상채널 구성

그림 8은 하이브리드방식의 네트워크 교착상태를 제거하기 위하여 본 논문에서 사용한 가상채널 모델이다. 하이브리드방식은 네트워크를 여러 방식으로 분할 할 수 있기 때문에 네트워크 교착상태를 제거하기 위한 가상채널 모델은 여러 가지 있을 수 있으나, 본 논문에서는 교착상태를 제거하면서 최소한의 버퍼를 사용하는 모델을 제시한다. 이것은 경로방식으로 메세지를 전달할 때 서로 다른 행이나 열로 진행하지 않고, 오직 자신의 행이나 열로만 진행하게 되므로 그림 2와 비교하여 볼 때 채널을 그림 8과 같이 줄여도 교착상태를 제거 할 수 있게 된다. 하이브리드방식은 메세지를 라우팅하기 전에 그림 9와 같은 라우팅 준비작업 알고리즘이 선행되어진다.

하이브리드방식을 위한 라우팅 준비작업 알고리즘

Input: 노드들의 집합 $D = \{(x_s, y_s), (x_1, y_1) \cdots (x_k, y_k)\}$,
출발지 노드 좌표 (x_s, y_s), 목적지 노드 좌표 $(x_1, y_1) \cdots (x_k, y_k)$.

Output: 메세지를 전송하기 위한 목적지 노드들의 그룹:
행(또는 열) 그룹 $D_{g1} = \{D_1, D_2 \cdots D_k\}$,
D_{g1}그룹 중 행(또는 열) 대표노드 그룹 D_{g2},
D_{g1}그룹 중 열(또는 행) 대표노드 그룹 D_{g3}.

Procedure:

1. 출발지 노드의 H라인 (horizontal line) V라인 (vertical line)을 각각 검사하여 목적지 노드가 많이 존재하는 라인을 기준 라인으로 택한다.
2. 만약 V(또는 H)라인이 선택되었다면, y(또는 x) 좌표가 동일한 목적지 노드끼리 행(또는 열) 그룹을 만들어 D_{g1}그룹이라 한다 (예를 들어 $D_1, D_2 \cdots D_k$).
3. D_{g1}내의 각각의 행(또는 열) 그룹에서 x_s(또는 y_s) 좌표와 동일한 목적지 노드가 존재한다면 그 그룹의 행(또는 열) 대표노드로 결정하고, 대표노드가 결정되지 않은 그룹은 x_s(또는 y_s) 좌표와 가장 가까운 목적지 노드를 행(또는 열) 대표노드로 결정한다.
4. D_{g1}내의 각각의 행(또는 열)그룹들에서 결정된 행(또는 열) 대표노드들 끼리 그룹을 만들어 D_{g2}그룹이라 한다.
5. 만약 D_{g2}그룹의 행(또는 열) 대표노드 중 동일한 x(또는 y) 좌표를 가진 노드가 하나이상 존재하면, 그 노드들 중 y_s(또는 x_s) 좌표와 가장 가까운 목적지 노드를 열(또는 행) 대표노드로 결정하고, D_{g3}그룹에 포함시킨다. 그렇지 않은 나머지 노드들은 그대로 D_{g3}그룹에 포함시킨다.

그림 9 하이브리드방식을 위한 라우팅 준비작업 알고리즘

그림 10은 6×6 토러스 네트워크에서 출발지 노드가 (4, 3)이고, 목적지 노드가 16개인 경우 하이브리드방식으로 라우팅하는 과정을 도식화한 것이다. 이 경우는 H라인 보다 V라인에 목적지 노드가 하나 더 많으므로 V라인이 기준 라인이 된다. 그러므로 D_{g1}내의 그룹들은 행별로 그룹이 되며, 각각의 행에 모두 목적지 노드가 분포해 있으므로 D_{g1}으로 다음과 같이 모두 6개의 행그룹이 다음과 같이 만들어지게 된다. $D_{g1} = \{D_1 = \{(4, 0), (5, 0)\}, D_2 = \{(1, 1), (2, 1), (4, 1)\}, D_3 = \{(0, 2), (2, 2), (3, 2), (5, 2)\}, D_4 = \{(2, 3), (4, 3)\}, D_5 = \{(2, 4), (3, 4), (5, 4)\}, D_6 = \{(0, 5), (2, 5), (5, 5)\}\}$ 다음은 D_{g1}내에 6개의 행그룹 별로 출발지의 x_s 좌표와 동일하거나 가장 가까운 노드를 행 대표노드로 결정하여, $D_{g2} = \{(3, 2), (4, 0), (4, 1), (4, 3), (5, 4), (5, 5)\}$ 그룹을 만든다. 마지막으로 D_{g2}내에 동일한 x 좌표를 가지고 있는 노드들이 존재하므로 (4, 0), (4, 1), (4, 3) 노드들 중에서는 출발지 노드인 (4, 3)이, (5, 4), (5, 5) 노드들 중에서는 출발지의 y_s와 가장 가까운 노드인 (5, 4)가 각각 열 대표(column representation) 노드가 되고, 나머지 노드인 (3, 2)는 그대로 열 대표 노드에 포함시킨다. 따라서 $D_{g3} = \{(3, 2), (4, 3), (5, 4)\}$ 그룹이 만들어진다. 그림 10에서 D_{g3}그룹의 목적지 노드는 트리방식으로 메세지를 전송하고, D_{g2}그룹부터는 경로방식으로 메세지를 전송하는 것을 볼 수 있다. 출발지 노드는 마지막 그룹인 D_{g3}에 언제나 존재하게 되며, 만약 D_{g3}에 출발지 노드 하나만 존재한다면 경로방식으로만 메세지를 전송하게 되어 가장 이상적인 경우가 된다. 위에서 예를 들어 설명한 HMR 방법을 알고리즘으로 표현하면 그림 11과 같다.

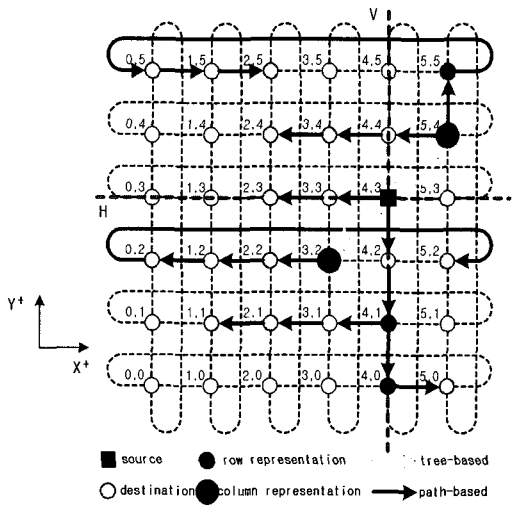


그림 10 하이브리드방식의 라우팅 예

하이브리드방식의 라우팅 알고리즘

Input: D_{g1}, D_{g2}, D_{g3}그룹

Procedure:

1. 라우팅 준비작업 알고리즘에 의해 D_{g1}, D_{g2}, D_{g3}그룹들이 만들어진다.
2. D_{g3}그룹은 노드의 좌표를 사전편집순서(lexicographical ordering)로 정렬을 시킨다.
3. **만약** D_{g3}에 출발지 노드만 존재한다면,
   1) D_{g3}에 대표노드들을 각각 열(또는 행)경로를 따라 경로방식으로 D_{g2}의 노드들에게 메세지를 전달한다

   그렇지 않고 D_{g3}에 2개이상의 노드가 포함되어 있다면,
   1) D_{g3}에서 출발지 노드가 처음에 위치하도록 정렬된 노드를 회전(rotation)시킨다.
   2) D_{g3}의 대표노드들을 트리방식(U-torus 알고리즘사용)으로 메세지를 전달한다.
   3) D_{g3}의 대표노드들을 각각 열(또는 행)경로를 따라 경로방식으로 D_{g2}의 노드들에게 메세지를 전달한다.
4. D_{g2}의 대표노드들을 각각 행(또는 열)경로를 따라 경로방식으로 D_{g1}의 노드들에게 메세지를 전달한다.

그림 11 하이브리드방식의 라우팅 알고리즘

네트워크의 교착상태에 관한 문제는 2.2 절에서도 언급했듯이 두 가지 경우가 존재하게 된다. 첫 번째 경우, 서로 다른 채널 층들 사이에서 교착상태가 발생될 수 있다. 그러나 그림 8의 채널 모델에서 실제 라우팅 할 때 같은 채널 층을 두 번 사용하지 않으므로 교착상태가 발생하지 않게 된다. 두 번째, 자신의 채널 층 내에서 교착상태가 발생하는 경우는 하이브리드 방식으로 라우팅을 하기 때문에 두 가지 방식, 즉 트리와 경로방식 모두 고려해야 한다. 그러나 트리방식의 라우팅은 X-Y 라우팅을 기본으로 하고 있으므로 절대 교착상태가 발생할 수 없으며, 경로방식의 라우팅은 서로 다른 행이나 열로 진행하지 않고 오직 자신의 행이나 열로만 진행하게 되므로 역시 자신의 채널 층에서는 교착상태가 발생되지 않는다는 것을 알 수 있다. 그러므로 HMR 알고리즘은 무교착 라우팅 알고리즘이다.

3. 시뮬레이션 및 성능비교

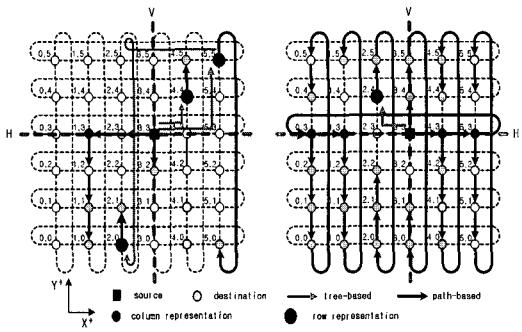
3.1 라우팅 알고리즘의 성능비교

트리방식으로는 유니캐스트를 이용하여 소프트웨어적으로 멀티캐스트를 수행하는 Robinson의 U-torus 알고리즘을[3], 경로방식과 하이브리드방식으로는 본 논문에서 제안한 DPMR 알고리즘과 HMR 알고리즘을 사용하여 각 라우팅 방법의 성능을 비교하였다. Visual C++ 5.0을 사용하여 16×16과 32×32 토러스 네트워크를 모델링하고 라우팅 알고리즘을 시뮬레이션하였다. 지연시간은 출발지의 모든 노드가 각각의 최종 목적지 노드까지 메세지를 전달하는데 걸리는 총 소요시간으로 계산하였으며, 이를 반복 수행하여 평균지연시간을 계산하였다. 시뮬레이션에서는 다음과 같은 세 가지 관점에서 알고리즘의 성능을 분석하였다. 첫째, 출발지 노드의 수를 일정하게 설정한 후 각 출발지 노드가 메시지를 멀티캐스트할 목적지 노드의 개수를 변화시켜 평균 지연시간을 구하였다. 일정한 출발지 노드 수와 메세지 길이에 대하여 네트워크의 트래픽 변화에 따른 각 라우팅 방법의 성능을 비교할 수 있으며, 본 시뮬레이션에서는 목적지 노드 수를 10~200개까지 변화시키며 지연시간을 측정하였다. 둘째, 출발지 노드의 수와 목적지 노드 수를 일정하게 설정한 후 메세지 길이의 변화에 따른 각 라우팅 알고리즘의 평균 지연시간을 구하였다. 셋째, 목적지 노드의 수가 일정할 때 출발지 노드 수의 변화에 따른 각 라우팅 방법의 평균 지연시간을 구하였다.

첫 번째 관점인 목적지 노드 개수의 변화에 따른 지연시간의 시뮬레이션은 16×16 토러스에서 수행하였으며, 메세지 길이는 16과 256, 그리고 동시에 멀티캐스트할 출발지 노드의 수는 5개, 20개, 50개인 경우에 대하여 각 출발지 노드당 목적지 노드의 개수를 최대 200개까지 증가시켜가며 평균 지연시간을 얻었다. 여기서 출발지 노드와 목적지 노드는 임의(random)로 분포한다고 가정하였고, 출발지 노드들이 모든 목적지 노드에 메세지가 전송되면 시뮬레이션이 끝나게 된다.

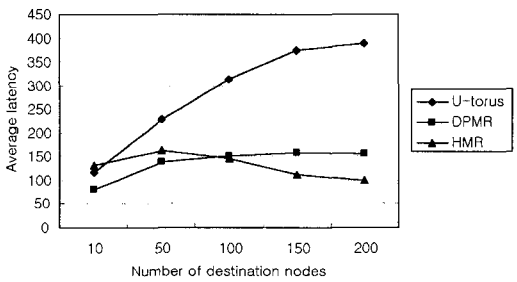
하이브리드방식에서 목적지 노드의 개수가 어느 정도 이상 증가하면 효율적인 그룹화 작업으로 트리방식으로 보내는 스텝의 수가 줄어들어 전체적인 지연시간이 줄어들 수 있다. 그림 12는 그러한 예를 6×6 토러스 네트워크에서 출발지 노드가 (3, 3)인 경우에 대하여 설명한 것이다. 그림 12 (a)에서 목적지 노드의 수가 10개일 때 트리방식으로 (3, 3)→(5, 5)로 보내는데 한 스텝, (3, 3)→(4, 4), (5, 5)→(2, 0)로 보내는데 한 스텝, 그리고 경로방식으로 행과 열로 보내는데 각각 한 스텝씩 필요하므로 전체 네 스텝만에 10개의 목적지 노드에게 메세지를 전달하게 된다. 그러나 목적지 노드의 수가 30개로 증가하는 경우(그림 12 (b)) 트리방식으로 (3, 3)→(2, 4)로 보내는데 한 스텝, 경로방식으로 두 스텝이 필요하여 전체 세 스텝만에 모든 목적지 노드에 메세지를 전달하게된다. 결국 목적지수가 많아지면 그림 9의 라우팅 준비작업 알고리즘에 의해서 그룹화 작업이 효율적으로 이루어지게 되어 지연시간이 오히려 줄어들게 된다.

그림 13, 14, 15에서 트리방식의 U-torus 알고리즘은 네트워크의 상태와 관계없이 목적지 노드의 개수가 증가함에 따라 다른 방식의 알고리즘에 비해 지연시간이 급격히 증가하는 것을 볼 수 있다. 트리방식은 당연히 목적지 노드가 많아질수록 스텝 수 또한 증가하게 되고, 스텝의 동기화에 따른 각 노드들의 지연시간이 그만큼

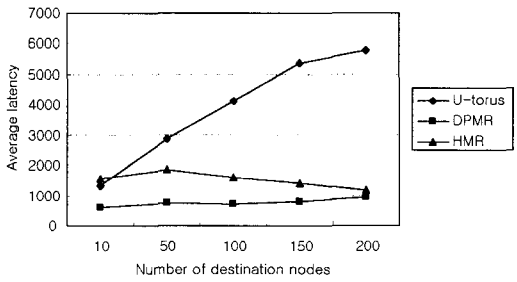


(a) (b)

그림 12 하이브리드방식의 메세지 패턴 : (a) 목적지 노드의 수 : 10, (b) 목적지 노드의 수 : 30



(a)

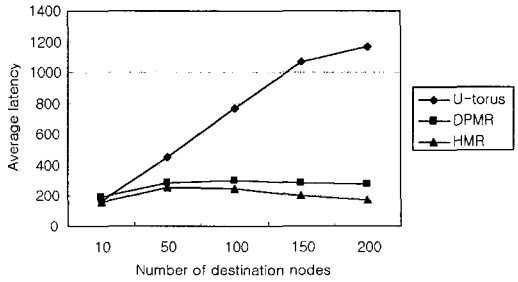


(b)

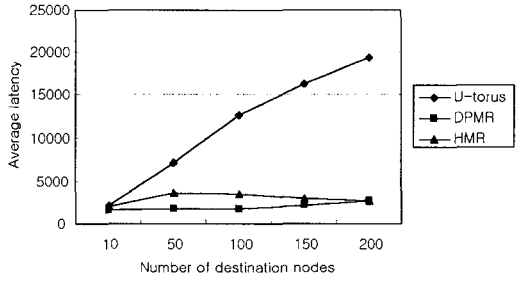
그림 13 출발지 노드가 5일 때 목적지 노드의 개수에 따른 성능비교 : (a) 메세지 길이 : 16, (b) 메세지 길이 : 256

많이 필요하게 되기 때문이다. 반면에 경로방식의 DPMR은 목적지 노드 수의 변화에 크게 상관없이 50개 정도 이후부터 지연시간이 거의 일정하게 되는 것을 볼 수 있다. 또 메세지 길이가 길 수록 다른 두 방식 보다 좋은 성능을 나타내고 있다. 하이브리드방식의 HMR은 메세지의 길이에 상관없이 목적지 노드의 개수가 50개 이상 증가하면 그림 12에서 설명한 것과 같이 점차 지연시간이 줄어들고 있음을 확인할 수 있다.

그림 13에서 HMR은 목적지 노드가 20개 이하일 경우 트리방식보다 더 나쁜 성능을 보이고 있지만, 메세지 길이가 16으로 짧은 경우(그림 13 (a)) 목적지 노드가 100개 이후부터는 DPMR보다 더 좋은 성능을 보이고 있다. 그러나 메세지 길이가 256으로 길어지는 경우(그림 13 (b)) DPMR보다 나쁜 성능을 보이며, 목적지 노드가 점차 증가함에 따라서 DPMR에 접근하여 가는 것을 알 수 있다.



(a)



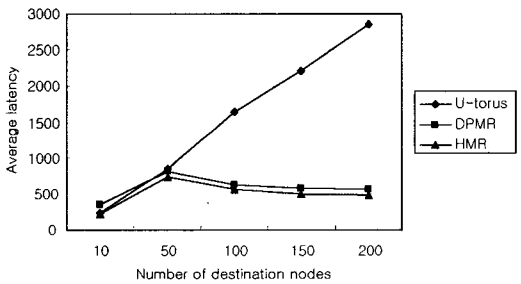
(b)

그림 14 출발지 노드가 20일 때 목적지 노드의 개수에 따른 성능비교 : (a) 메세지 길이 : 16, (b) 메세지 길이 : 256

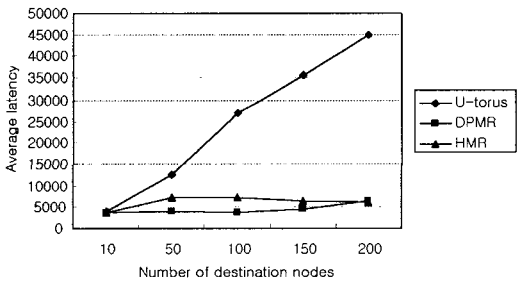
그림 14는 목적지 노드가 10개 일 때 세 가지 방식의 지연시간이 서로 비슷하지만, 목적지 노드의 개수가 증가할 수록 그림 13과 비슷한 성능을 보이고 있다. 그러나 메세지의 길이가 16인 경우(그림 14 (a)) HMR이 가장 좋은 성능을 보이고 있다. 이것은 HMR 알고리즘으로 네트워크를 분할 할 때 행(또는 열) 별로 목적지 노드를 그룹을 만들고 최종적으로는 행(또는 열) 경로를 따라 진행하게 되는데, 이때 만약 경로 차단이 발생하게

된다면 메세지가 길어질수록 행(또는 열)별로 대기하는 시간은 점점 증가하게되고, 메세지 길이가 짧아질 수록 행(또는 열)별로 대기하는 시간이 감소하게 되기 때문이다. 메세지 길이가 256인 경우(그림 14 (b)) DPMR은 일정한 지연시간을 유지하면서 가장 좋은 성능을 보이고 있다. 이것은 메세지의 길이가 해밀톤 경로보다 길어져 목적지 노드의 개수에 상관없이 네트워크가 분할되지 않고 해밀톤 경로방식으로 진행하기 때문이다. 또 목적지 노드 개수가 증가할수록 DPMR과 HMR의 지연시간은 근소한 차이가 나고 있음을 볼 수 있다.

그림 15는 동시에 출발하는 출발지 노드의 수를 50개로 하여 네트워크에 트래픽을 더욱 증가시킨 경우이다. 대부분 그림 14와 비슷한 특성을 보이고 있지만, 메세지 길이가 16인 경우 (그림 15 (a)) 목적지 노드가 50개까지는 DPMR과 HMR의 지연시간이 급격히 증가하는 것을 볼 수 있다.



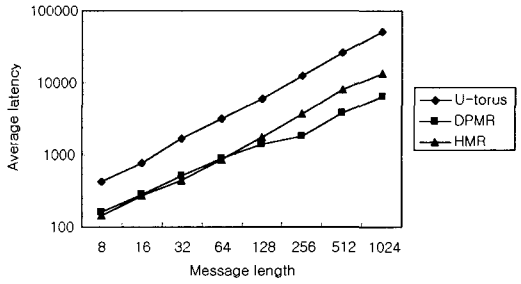
(a)



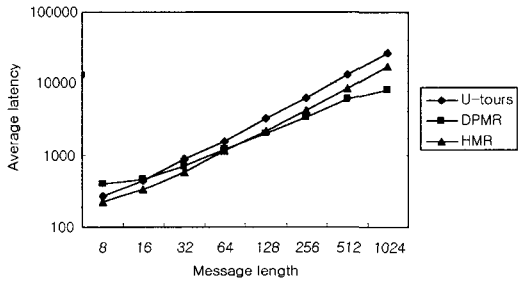
(b)

그림 15 출발지 노드가 50일 때 목적지 노드의 개수에 따른 성능비교 : (a) 메세지 길이 : 16, (b) 메세지 길이 : 256

그림 16은 네트워크의 크기를 일정하게 두고 메세지 길이를 8~1024 까지 변화시켜 가면서 평균 지연시간을 구한 후 로그스케일로 표현한 것이다. 네트워크의 크기가 16×16일 때(그림 16 (a)) 메세지 길이가 약 32 정도까지는 HMR이 DPMR보다 더 좋은 성능을 보이나 메세지 길이가 약 64정도 이후에서는 DPMR이 HMR보다 좋은 성능을 나타낸다. 네트워크의 크기가 32×32일 때(그림 16 (b)) 메세지의 길이가 약 64 정도까지는 HMR이 좋은 성능을 보이나 메세지 길이가 약 128정도 이후에서 DPMR이 더 좋은 성능을 나타내고 있다. 따라서 본 시뮬레이션에서는 일반적으로 메세지 길이가 네트워크의 크기에 4배정도가 될 때까지는 DPMR보다 HMR의 성능이 좋다가, 그 이후 메세지의 길이가 더 길어질 경우는 DPMR의 성능이 항상 좋은 것으로 나타났다.



(a)

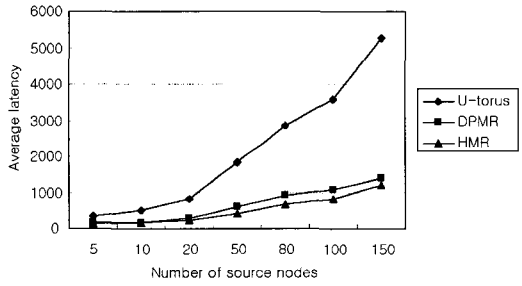


(b)

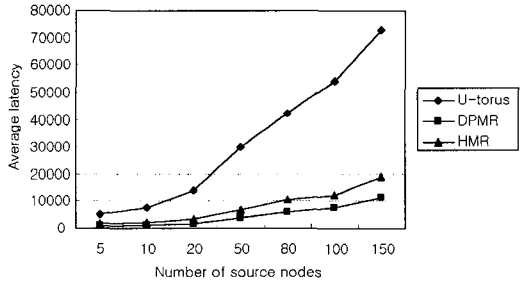
그림 16 출발지 노드가 20일 때 메세지 길이의 변화에 따른 성능비교 : (a) 네트워크 크기 16×16, (b) 네트워크 크기 32×32

그림 17은 목적지 노드의 수를 네트워크 크기의 1/2, 즉 128개로 일정하게 할 때 출발지 노드의 변화에 따른 성능을 보여주고 있다. U-torus 알고리즘은 메세지 길이가 16일 때는 출발지 노드의 수가 20개 이후부터, 메

세지 길이가 256일 경우는 출발지 노드의 수가 10개 이후부터 지연시간이 급격히 상승하는 것을 볼 수 있다. 따라서 U-torus 알고리즘은 출발지 노드 수가 증가할수록 그리고 메세지 길이가 길어질수록 경로 차단이 발생할 확률이 높게되어 성능에 많은 영향을 주는 것을 알 수 있다. 반면에 DPMR과 HMR은 모두 출발지 노드의 증가에 따른 영향보다는 메세지 길이의 따라 성능 차이가 나게 된다. 메세지 길이가 16일 경우 (그림 17 (a)) HMR이 DPMR보다 더 좋은 성능을 보이고 있고, 메세지 길이가 256인 경우 (그림 17 (b)) DPMR이 HMR보다 더 좋은 성능을 보이고 있다. 그리고 이런 결과는 이미 그림 16에서도 볼 수 있었다.



(a)



(b)

그림 17 목적지 노드가 128개 일 때 출발지 노드 개수에 따른 성능비교 : (a) 메세지 길이 : 16, (b) 메세지 길이 : 256

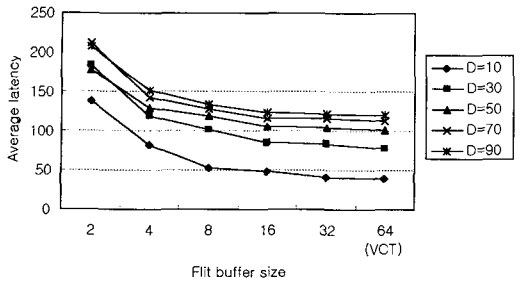
3.2 버퍼 크기에 따른 웜홀 라우팅의 성능비교

VCT 스위칭 방식은 메세지의 경로가 차단되면 모든 플릿을 그 경로 차단이 일어난 노드의 버퍼에 저장하여 네트워크 상에서 다른 메시지에 길을 만들어 주어 채널을 사용할 수 있도록 한다. 따라서 웜홀 라우팅보다 네트워크의 메시지 전송율은 높지만, 플릿 전체를 저장할 수 있는 큰 버퍼를 필요로 하는 단점이 있다[11][12]. 따라서 본 논문에서는 웜홀 라우팅의 버퍼의 크기를 VCT 버퍼 크기가 될 때까지 점차 증가시켜 가면서 두 스위칭 방식의 성능을 비교하여 보았다. 웜홀 라우팅에서 버퍼의 크기가 메세지 길이와 같아지면 VCT가 된다. 플릿 버퍼의 크기를 무조건 증가시킨다고 해서 그만큼 성능이 향상되지는 않으므로, 시뮬레이션을 통하여 VCT와 비슷한 성능을 갖는 적절한 버퍼의 크기를 얻었다. 시뮬레이션에서는 경로방식인 DPMR을 사용하였으며, 버퍼의 증가로 더 많은 실리콘 영역(silicon area)을 사용하게 되지만 그로 인한 전달 지연(propagation delay)과 클럭 주파수의 감소는 무시한다.

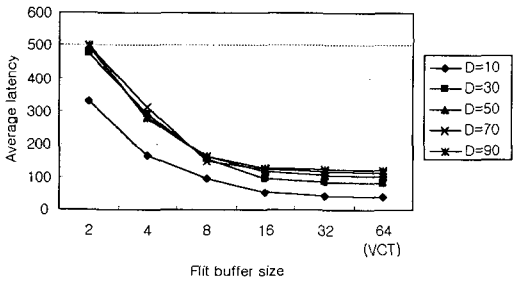
그림 18은 네트워크의 크기가 16×16인 경우, 멀티캐스트를 완료하는데 소요되는 평균지연시간을 플릿 버퍼 크기에 따른 그래프로 나타낸 것이다. 메세지 길이가 64일 때, 웜홀의 버퍼를 2~64까지 변화시켜 보았다. 여기서 웜홀의 버퍼 크기가 64인 경우 VCT 방식과 동일하게 된다. 그림 18 (a)와 (b)는 각각 출발지 노드가 5개, 20개로 웜홀의 버퍼가 8일 때까지는 지연시간이 급격히 감소하나, 그 이후 버퍼가 64가 될 때까지는 2배씩 증가시켜도 지연시간은 거의 일정한 것을 알 수 있다. 또한 그림 18 (b)는 목적지 노드가 10개인 경우를 제외하고 30~90개까지 거의 동일한 지연시간을 보이고 있다. 그것은 목적지가 30개 이상 계속 증가하여도 거의 비슷한 경로를 따라 메시지가 진행하기 때문이다. 그림 18 (c)는 (b)와 대부분의 조건들은 동일하게 하고, 단지 네트워크 크기만 32×32로 증가시킨 경우이다. 역시 웜홀의 버퍼가 8보다 더 커지는 경우 지연시간은 거의 감소하지 않는 것을 볼 수 있다. 네트워크의 크기를 증가시켰으므로 목적지 노드의 변화에 따른 지연시간도 16×16에 비하여 늘어난 것을 알 수 있다.

본 시뮬레이션을 통하여 플릿 버퍼의 크기가 VCT 버퍼 크기의 1/8 정도만 되어도 웜홀은 VCT 방식과 어느 정도 비슷한 성능을 나타내는 것을 알 수 있다. 그러나 플릿 버퍼의 크기를 결정하기 위한 일반적인 법칙을 얻기 위해서는 정확한 해석적 분석과 시뮬레이션이 필요하다. 또한 라우터의 설계에서 플릿 버퍼의 크기는 네트워크의 흐름제어 정책(flow control policy)도 고려되어야 한다. 웜홀 라우팅을 사용하여 실제로 구현된 시스템인 Cray T3D의 경우 플릿 버퍼 1개의 크기는 8개의 16-bit phits로 구성되며, Cray T3E는 5개의 16-bit phits로 구성된다[13]. 여기서 phits는 한 사이클당 물

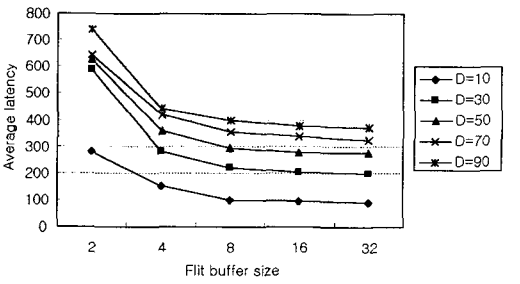
리적 채널을 통해 전송할 수 있는 정보의 기본 단위이다.



(a)



(b)



(c)

그림 18 메세지 길이가 64일 때 플릿 버퍼 크기에 따른 성능비교 : (a) 네트워크 크기 : 16×16, 출발지 노드 : 5, (b) 네트워크 크기 : 16×16, 출발지 노드 : 20, (c) 네트워크 크기 : 32×32, 출발지 노드 : 20

4. 결 론

본 논문은 다중 컴퓨터 시스템에서 프로그램의 수행 속도를 향상시키기 위하여 필요한 멀티캐스트 통신을 라우팅 방법에 따라 트리방식, 경로방식, 그리고 하이브리드방식으로 분류하였다. 경로방식으로는 DPMR 알고리즘을, 하이브리드방식으로는 HMR 알고리즘을 각각 제안하고, 트리방식의 U-torus 알고리즘과 성능을 비교하였다. 각각의 방식에 대해 목적지 노드 개수의 변화, 메세지 길이의 변화, 그리고 출발지 노드 개수의 변화에 따른 평균 지연시간을 기준으로 하여 성능을 비교 분석하였다.

동적분할 경로방식의 특징은 메세지의 길이에 따라 네트워크의 분할 여부를 동적으로 결정하는 방식이며, 네트워크의 트래픽이 한쪽으로 집중되지 않고 모든 채널 층들이 고르게 사용됨을 알았다. 하이브리드방식은 트리방식과 경로방식의 단점을 서로 보완한 혼합 방식이다. 이 방식의 라우팅 알고리즘은 경로방식을 따를 때 메세지를 서로 다른 행이나 열로 진행시키지 않고, 선택된 대표노드의 행이나 열로만 진행시키므로 메세지의 길이와 네트워크의 크기에 특히 민감한 것을 알았다. 시뮬레이션 결과 16×16 네트워크에서 트래픽이 낮고 메세지 길이가 짧은 경우, 목적지 노드가 20개 이하에서는 경로방식이 가장 좋은 성능을 보였고, 두 번째로 트리방식, 그리고 하이브리드방식의 순서로 나타났지만, 목적지 노드의 수가 점차 증가하면 하이브리드방식이 트리방식보다 더 좋은 성능을 보이고 있다. 그리고 목적지 노드의 수가 100개 이상으로 증가하게 되면 하이브리드방식이 가장 좋은 것으로 나타났다. 그러나 네트워크에 트래픽이 많고, 메세지 길이가 네트워크의 크기에 네 배 이하일 때는 하이브리드방식, 경로방식, 그리고 트리방식의 순서로 성능을 보이며, 메세지 길이가 그 이상 증가하면 경로방식이 더 좋은 성능을 나타냄을 보였다.

웜홀 라우팅에서 버퍼크기가 성능에 미치는 영향을 분석하기 위하여 16×16, 32×32인 토러스 네트워크에서 시뮬레이션을 수행하였다. 메시지의 크기가 64인 경우 웜홀 라우팅의 버퍼의 크기가 증가함에 따라 메세지의 지연시간이 급격히 감소하나, 그 크기가 8이상 되면 지연시간은 거의 감소하지 않는 것을 볼 수 있다. 일반적으로 웜홀 라우팅의 버퍼 크기가 VCT 버퍼 크기의 1/8정도가 되면 VCT 방식과 거의 같은 성능을 나타내는 것을 알 수 있다.

참 고 문 헌

[1] L.M. Ni and P.K. McKinley, "A survey of wormhole routing techniques in direct networks," *IEEE Comput.*, vol. 26, pp. 62-76, Feb. 1993.

[2] W.D. Dally, "Performance analysis of k-ary n-cube interconnection networks," *IEEE Trans. Computers*, vol. 39, no. 6, pp. 775-785, June 1990.
[3] D.F. Robinson, P.K. McKinley, and B.H.C. Cheng, "Optimal multicast communication in wormhole-routed torus networks," *IEEE Trans. Parallel and Distributed Systems*, vol. 6, no. 10, pp. 1029-1042, Oct. 1995.
[4] P.K. McKinley, Y. Tsai, and D.F. Robinson, "Collective communication in wormhole-routed massively parallel computers," *IEEE Comput.*, vol 28, no. 12, pp. 39-50, Dec. 1995.
[5] X. Lin, P.K. McKinley, and L.M. Ni, "Deadlock-free multicast wormhole routing in 2D-mesh multicomputers," *IEEE Trans. Parallel and Distributed Systems*, vol. 5, no. 8, pp. 793-804, Aug. 1994.
[6] M.P. Malumbres, J. Duato, and J. Torrellas, "An efficient implementation of tree-based multicast routing in distributed shared-memory multiprocessors," *Proc. of the 8th IEEE Symp. on Parallel and Distributed Processing*, pp. 186-189, Oct. 1996.
[7] D.F. Robinson, P.K. McKinley, and B.H.C. Cheng, "Path-based multicast communication in wormhole-routed torus networks," *J. of Parallel and Distributed Computing*, vol. 45, pp. 104-121, 1997.
[8] W.J. Dally and D.L. Seitz, "Deadlock-free message routing in multiprocessor interconnection networks," *IEEE Trans. Comput.*, vol. C-36, pp. 547-553, May 1987.
[9] D.K. Panda, S. Singal and R. Kesavan, "Multi-destination message passing mechanism conforming to base wormhole routing scheme," *IEEE Trans. Parallel and Distributed Systems*, vol. 10, no. 1, pp. 76-96, Jan. 1999.
[10] D. Basak, and D.K. Panda, "Alleviating consumption channel bottleneck in wormhole-routed k-ary n-cube systems," *IEEE Trans. Parallel and Distributed Systems*, vol. 9, no. 5, pp. 481-496, May 1998.
[11] J. Duato, S. Yalmanchili, and L.M. Ni, *Interconnection Networks: An Engineering Approach*, IEEE CS Press, Los Alamitos, CA, 1997.
[12] P. Kermani and L. Kleinrock, "Virtual cut-through: A new computer communication switching technique," *Comput. Networks*, vol. 3, no. 4, pp. 267-286, 1979.
[13] S.L. Scott and G.M. Thorson, "The Cray T3E network: adaptive routing in a high performance 3D torus," *Proc. of HOT Interconnects Symposium IV*, Aug. 1996.

원 복 희
1997년 수원대학교 전자공학과 공학사. 1999년 인하대학교 전자공학과 공학석사. 1999년 ~ 현재 (주)현대멀티캡 연구원. 관심분야는 컴퓨터구조, 라우팅, 무선데이타 통신

최 상 방
정보과학회논문지 : 시스템 및 이론
제 27 권 제 2 호 참조