웜홀 방식 망에서의 효율적인 완전교환 통신 알고리즘

(Efficient All-to-All Personalized Communication Algorithms in Wormhole Networks)

김 시 관 † 맹 승 렬 †† 조 정 완 ††

(Si Gwan Kim) (Seung ryoul Maeng) (Jung Wan Cho)

요 약 완전교환 통신은 행렬전이, 푸리에변환 혹은 분산 테이블 검색과 같은 여러 가지 응용에서 아주 많이 활용되는 통신 방법이다. 본 논문은 웜홀 방식을 채용한 2차원 토러스에서의 개시 지연 시간을 줄이기 위하여 분할 및 합병(divide-and-conquer) 방식을 사용한 효율적인 완전교환 통신 알고리즘을 제안한다. 전체망을 2x2 형태의 기본셀로 분할한 뒤 각 기본셀에서는 마스터노드라고 불리는 특정 노드를 지정하여 기본셀내의 여타 노드들의 메시지를 이 마스터노드가 수집한다. 이 마스터노드들이 다른 모든 노드로 보내질 메시지를 수집한 뒤 각 기본셀내의 모든 마스터 노드들만이 가상 망을 형성하여 망의 크기가 N/2 x N/2으로 줄어든 상태로 완전 교환 알고리즘을 수행한다. 마스터노드들간의 완전교환 연산을 수행한 뒤 이 마스터노드들은 자기가 전담했던 여타 노드들의 메시지를 재분배해 줌으로써 주어진 완전교환 연산을 완료한다. 기존의 여러 가지 알고리즘과의 비교 분석을 제시하였으며 제시한 알고리즘이 약 2배 정도의 개시 지연시간 면에서 우수함을 보인다.

Abstract All-to-all personalized communication, or complete exchange, is at the heart of numerous applications, such as matrix transposition, fast Fourier Transform(FFT), and distributed table lookup. We present an efficient all-to-all personalized communication algorithm for a 2D torus in wormhole-routed networks. Our complete exchange algorithm adopts divide-and-conquer approach to reduce the number of start-up latency significantly, which is a good metric for network performance in wormhole networks. First, we divide the whole network into 2x2 basic cells. After specially designated nodes called master nodes have collected messages to transmit to the rest of the basic cell, only master nodes perform complete exchange with reduced network size, N/2 x N/2. When finished with this complete exchange in master nodes, these nodes distribute messages to the rest of the master node, which results in the desired complete exchange communication. After we present our algorithms, we analyze time complexities and compare our algorithms with several previous algorithms. And we show that our algorithm is efficient by a factor of 2 in the required start-up time which means that our algorithm is suitable for wormhole-routed networks.

1. 서 론

멀티컴퓨터[1]는 여러 개의 처리기가 특정한 형태로 연결되어 있는 컴퓨터이다. 각각의 처리기는 공통의 기억장치가 아닌 지역적인 기억장치를 가지고 있다. 그러므로, 처리기간에 정보 교환을 위하여 메시지 전달이 필요하며 이웃하지 않는 처리기들은 하나 이상의 중간 처리기를 통해서만 간접적으로 통신을 할 수 있다.

멀티컴퓨터에서 많이 사용되는 형태는 *k-ary n-큐브* 인데 이는 각 n 차원마다 k개의 노드들로 구성되어 있다. 메쉬, 토러스 및 하이퍼큐브는 *k-ary n-큐브*의 일종이라 할 수 있다. 2차원 메쉬는 Intel Paragon,

† 비 회 원 : 한국과학기술원 전산학과
sgkim@camars.kaist.ac.kr
†† 종신회원 : 한국과학기술원 전산학과 교수
maeng@cs.kaist.ac.kr
jwcho@camars.kaist.ac.kr
논문접수 : 1999년 7월 12일
심사완료 : 2000년 3월 8일

Touchstone DELTA과 Caltech Mosaic C에 채용되었고 3차원 메쉬는 J-Machine에서, 3차원 토러스는 CRAY T3D에서 그리고 하이퍼큐브는 nCUBE-2에 채용되었다.

대부분의 멀티컴퓨터는 단일전송(unicast)만 지원을 한다. 최근에는 집합체 통신(collective communication) 기능을 추가하는 추세인데 이는 같은 메시지가 다수의 원천지에서 다수의 목적지 노드로 전달되는 통신을 말한다. 메시지 전달 기법을 채용한 멀티컴퓨터에서는 부하 조절(load balancing), 사건 동기화(event synchronization), 자료 교환(data exchange)과 같은 여러 가지 작업을 위해서 처리기간에 서로 통신을 해야 할 필요가 발생한다. 송수신하는 처리기의 갯수에 따라 통신 형태는 일대일(unicast), 일대다(multicast), 방송형(broadcast), 다대다(all-to-all)로 분류할 수 있다. 또한, 보내야하는 메시지의 내용이 서로 다른 개별적(personalized) 통신과 메시지의 내용이 모두 같은 비개별적(non-personalized) 통신으로 구분된다.

행렬 전이, FFT, 분산 테이블 검색과 같은 분야에 응용이 되는 다대다 개별적 통신(All-to-All Personalized Communication)은 아주 중요한 통신 요소 중의 하나라고 할 수 있다. 이러한 형태의 통신에서는 N개의 처리기가 각각 길이는 같지만 내용이 서로 다른 메시지가 자기 이외의 $N-1$개의 처리기로 보내어진다. 따라서, 통신 형태와 통신량이 아주 복잡하다고 할 수 있다. 다대다 개별적 통신은 완전교환(complete exchange)이라고도 부른다.

완전 교환을 구현하는 알고리즘[2, 6, 12, 13, 14, 15, 18]은 직접(direct) 방식과 간접(indirect) 방식으로 분류할 수 있다. 직접 방식에서는 여러 개의 충돌이 없는 단계를 거쳐서 수행이 된다. 각 단계마다 서로 독립적인 경로를 통하여 메시지들이 원천지에서 목적지로 직접 전달이 된다. 이와 달리 간접 방식에서는 메시지가 목적지로 전달되는 도중에 중간 노드에서 버퍼링되거나 자료들이 재정렬될 수도 있다. 메쉬나 토러스에서는 일반적으로 직접 방식보다 간접 방식이 더 효율적이다. Bokhari[2]와 Sundar[15]는 2차원 메쉬에서의 다대다 개별적 통신에 관한 알고리즘을 제시하였다. [18]에서는 모든 노드가 링크 충돌을 피하기 위하여 대각선을 따라서 그룹화를 하였다. 이는 임의의 연속된 2개의 노드는 서로 다른 행과 열을 따라서 배치된다는 토러스의 특성을 이용한 것이다. 최근 [14]는 [18]와 유사한 알고리즘을 제안하였다.

본 논문에서는 메시지 개시지연시간을 줄이기 위하여 분할 후 합병 전략을 사용한 효율적인 알고리즘을 제시한다. 시스템의 성능면에서 본다면 메시지 개시 지연 시간을 줄이는 것은 웜홀 방식의 컴퓨터에서는 아주 중요한 요소이다. 새로운 알고리즘을 제시한 다음 기존의 알고리즘과의 비교를 통하여 제안한 알고리즘이 우수함을 보인다.

본 논문은 다음과 같이 구성되어 있다. 2장에서는 본 논문에서 제시된 시스템 모델에 대하여 언급하고 3장에서는 완전 교환의 개념에 대해서 설명한 뒤 새로운 알고리즘을 제시하고 4장에서는 복잡도 분석 및 평가를 하고 마지막으로 5장에서 결론을 맺는다.

2. 시스템 모델(System Model)

다중컴퓨터 시스템에서의 완전교환 알고리즘의 성능을 측정하는 요소로는 메시지 개시지연(startup latency), 데이타 전송시간(data transmission time), 데이타 재정렬시간(data rearrangement time) 및 지연시간(propagational delay time)으로 구성되어 있다.

메시지 개시지연은 망에서 원천지 노드와 목적지 노드에서 메시지를 보내고 받을 때 처리하는데 소요되는 시간이고, 데이타 전송시간은 망을 통하여 실제 메시지가 전송되는데 소요되는 시간을 말한다. 데이타 재정렬 시간은 다음 단계를 위해 메시지들의 순서를 정렬하는데 필요한 시간이고 지연시간은 주어진 메시지가 목적지에 도달하기 위해 필요한 홉의 수를 측정하는 시간이다.

다중컴퓨터 시스템 구조는 메시지를 전달하는 스위칭 방식에 의해 결정된다. 대부분의 다중컴퓨터는 메시지를 여러개의 고정된 크기의 플릿(flit)으로 분할하여 각각을 네트워크를 통해서 파이프라인 형태로 목적지 노드까지 전달하는 웜홀(wormhole) 스위칭 방식이 널리 쓰이고 있다[4, 10].

웜홀 방식의 라우팅은 Intel Touchstone DELTA, Intel Paragon, MIT J-machine, Caltech MOSAIC 그리고 Cray T3D과 같은 차세대 병렬컴퓨터에 채용되었다. 메시지들이 각각의 처리기에서 저장되었다가 보내어지는 저장후전송방식(store-and-forward)과 달리 웜홀 방식은 메시지의 머리부분이 입력채널에서 출력채널로 바로 보내어진다. 각각의 처리기에서는 몇 개의 플릿만 버퍼에 저장이 된다. 메시지의 머리부분을 검사한 뒤 바로 다음 채널이 결정되고 이어 그 채널로 보내진다. 따라서, 이러한 과정에서 메시지가 원천지 노드에서 목적지 노드사이의 채널에 흩어져 있게 된다. 이런 과정에서 메시지의 헤더 플릿이 불통이 되면(blocked) 메시지

의 모든 다른 플릿이 나아갈 수 없고 현재 다른 메시지가 사용하는 채널을 요구하는 모든 다른 메시지들이 불통이 된다.

메시지의 길이가 길 경우에, 웜홀 라우팅의 파이프라인 효과로 인하여 메시지가 가야할 거리는 네트워크 지연에 큰 영향을 주지 않는다[9]. 메시지의 길이가 짧을 경우에, 메시지 개시지연은 단일전송 메시지에 대한 지연에 큰 영향을 미친다. 그러므로, 네트워크 자원에 대한 메시지들간의 경쟁(contention)이 없을 경우에는 웜홀 스위칭 방식에서 메시지 통신지연은 거의 거리에 무관하게 되며 메시지 개시지연 시간에 상당히 의존하게 된다. 웜홀 스위칭 방식의 가장 큰 문제점은 메시지들이 여러 채널을 점유하게 되므로 교착 상태(deadlock)가 발생할 가능성이 크다는 점이다.

다중컴퓨터 시스템 구조를 특징짓는 두번째 요소는 네트워크의 구조이다. 상호연결망의 구조를 그래프 $G = (V, E)$로 모델링할 때 본 논문에서 다루는 다중컴퓨터 시스템의 네트워크 구조인 2차원 메쉬(mesh)는 다음과 같이 정의된다.(여기서 V 집합의 각 노드는 처리기를를 의미하고 E 집합의 각 연결선은 통신 채널에 해당된다.) 즉, m x n 토러스는 $G = (V, E)$의 그래프로 표시할 수 있다. 여기에서 $V = \{(x, y) \in N \times N \mid 0 \le x < n \text{ and } 0 \le y < m\}$ 이고 $E = \{((x_i, y_i), (x_j, y_j)) \mid |x_i - x_j| + |y_i - y_j| = 1 \text{ or } (|x_i - x_j| = n-1 \text{ and } |y_i - y_j| = 0) \text{ or } (|x_i - x_j| = 0 \text{ and } |y_i - y_j| = m-1)\}$.

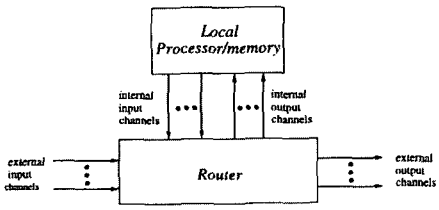


그림 1 일반적인 노드의 구조

다중컴퓨터 구조를 특징짓는 요소인 라우터(router)는 노드 처리기에 개별적으로 구성되어 있으며, 노드 처리기간의 통신을 처리한다. 그림 1의 노드 구조와 같이, 노드의 라우터가 여러 개의 외부 채널(external channels)들을 통하여 이웃의 라우터와 연결되는데 이러한 외부 채널이 연결되는 방식은 다중컴퓨터 네트워크의 구조에 의해 결정된다. 라우터로 들어오는 메시지들이 서로 다른 채널을 통해서 나가려고 한다면, 라우터는 여러 개의 메시지를 동시에 전달할 수 있다. 그와 더불어, 노드의 라우터와 이웃하는 라우터 사이에서 서로 반대의 방향으로 두개의 메시지를 동시에 전송하는 것도 가능하다. 라우터는 노드 처리기와 여러 개의 내부 채널(internal channels)들로 연결되어 있다. 내부 채널은 노드 처리기에서 들어오는 입력 채널과 노드 처리기로 들어가는 출력 채널로 구분된다.

다중컴퓨터 구조를 특징짓는 네번째 요소로 라우터가 동시에 복사할 수 있는 메시지의 갯수와 라우터로 노드 처리기가 동시에 보낼 수 있는 메시지의 개수인 통신 출구의 개수이다. 단출구 통신은 메시지를 보내는 단계마다 최대 두배만큼의 노드 처리기가 메시지를 받을 수 있기 때문에 m개의 다중전송의 목적지 노드로 메시지를 보내는 경우에 최소 $\lceil \log_2(m+1) \rceil$ 만큼의 단계가 메시지 전송 단계에 필요하다. 대부분의 다중전송 알고리즘은 단출구 통신만을 지원하는 라우터를 사용하며 본 논문에서도 이를 가정한다.

노드 처리기와 라우터가 동시에 여러 개의 메시지를 전송하기 위해서는 다음과 같은 기능이 필요하다. 노드 처리기는 그림 1에서와 같이 라우터에서 노드 처리기로 출력되는 메시지가 사용하는 출력 채널이 출력 버퍼로 향하는 채널과 입력 채널로 향하는 채널 두 가지로 구분된다. 라우터는 출력 채널로 들어오는 메시지를 여러 개 복사할 수 있는 기능과 출력 버퍼로 향하는 채널로 하나를 보내고 나머지를 입력 버퍼로 향하는 채널로 보내는 기능이 필요하다. 이러한 기능을 갖는 라우터로써 동시에 여러 개의 메시지를 전송할 수 있는 다출구 통신을 이용한 다중전송을 구현할 수 있다.

$N \times N$ 토러스에서의 완전교환 통신에 필요한 하한값(lower bound)은 다음과 같이 주어진다. $N \times N$ 토러스를 이등분(bisection)하는 $2N$개의 링크가 각 방향으로 존재한다. 이등분한 $N^2/2$개의 노드들은 다른 이등분한 $N^2/2$개의 노드에게 $(N^2/2)^2$개의 메시지 전송이 필요하다. 그러므로, 데이타 전송시간의 하한값은 $(N^{4/4}/4x2N)t_d=(N^3/8)t_d$ 로 주어진다(단, 메시지의 길이는 1바이트로 가정하고 t_d는 1바이트를 전송하는데 소요되는 시간을 의미). 그리고, 메시지 개시지연은 하나의 노드가 자기 이외의 노드로 메시지를 보낼 수 있으므로 하한값은 $(\log_2 N^2)t_s$ (단, t_s는 하나의 메시지를 준비하는데 필요한 시간을 의미)로 주어진다.

3. 완전교환 통신 알고리즘

이 장에서는 먼저 기존 연구에 대하여 간략히 설명한 뒤 본 논문에서 제시하는 알고리즘에 대한 여러 가지

필요한 가정과 용어들에 대하여 설명한다. 3절에서는 2차원 알고리즘에 대한 자세한 내용을 제시한다.

3.1 기존 연구

Bokhari [2]와 Sundar [15]는 간접(indirect) 방식을 사용하여 2차원 메쉬에서의 다대다 개별적 통신에 관한 알고리즘을 제시하였다. [18]에서는 채널 충돌을 피하기 위하여 모든 노드를 대각선을 따라서 4개의 그룹(G_0, G_1, G_2, G_3)으로 나눈 다음 토러스는 4방향($H+$, $H-$, $V+$, $V-$, 여기서 H는 수평방향의 채널, V는 수직방향의 채널, +는 각 채널의 양의 방향, -는 음의 방향을 의미함)으로 채널이 배치된다는 특성을 이용하여 메시지를 채널 충돌이 없도록 4개의 방향으로 메시지를 전송하도록 하였다. [14]는 성능은 더 우수하면서도 망의 크기가 4의 배수에서도 적용할 수 있는 [18]과 유사한 알고리즘을 제안하였다.

3.2 기본 사항

먼저 알고리즘에 필요한 가정은 다음과 같다.

- 웜홀 라우팅 방식이 사용된다.
- 메시지가 전달되기 위해서 중간 노드들에서 버퍼링, 재정렬 및 전달과정이 이루어지는 간접 방식을 가정한다.
- 노드에서 한번에 하나의 메시지만을 주고 받을 수 있는 단일 출구(one-port) 통신을 가정한다.

2차원 토러스에서의 각 노드는 $P_{i,j}$로 표시한다.(여기에서 $0 \le i, j < N$이다.) 노드 $P_{i,j}$는 노드 $P_{i-1,j}$, $P_{i+1,j}$, $P_{i,j-1}$와 $P_{i,j+1}$에 연결되어 있다.

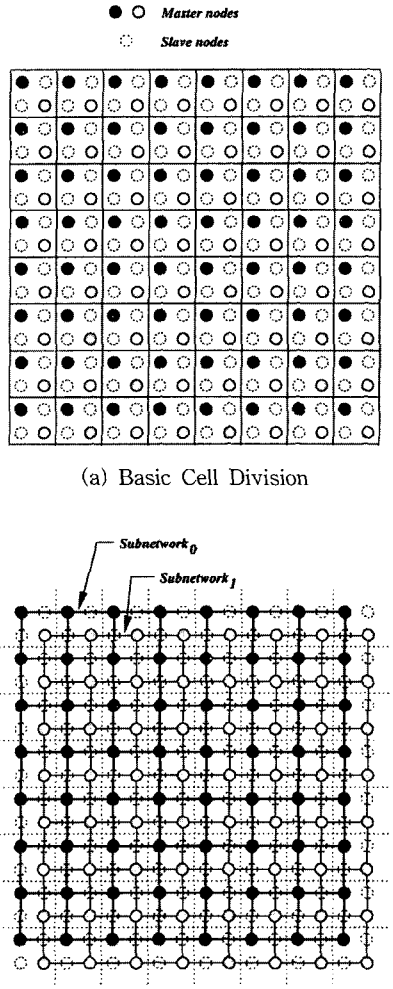
정의 1 노드그룹 NG_i는 $NG_i = \{P_{2j+i,0}, P_{2j+i,1}, P_{2j+i,2}, \ldots, P_{2j+i,N-1}\}$를 만족하는 노드들의 집합이다.(단, $0 \le i < 2$ 이고 $0 \le j < N/2$) 즉, 행/열 방향으로 짝수번째(또한 홀수번째)의 노드들은 NG_0(또한 NG_1)로 분류된다.

정의 2 기본셀 BC는 전체 토러스망을 $N/2 \times N/2$의 형태로 분할시키는 2×2 크기의 망이다. 전체망을 기본셀로 분할시키면 각각의 기본셀 단위들은 메쉬망의 형태로 동작을 한다.

예를들어, 그림 2(a)의 16 x 16 토러스망에서는 실선으로 분할된 64개의 기본셀을 가지며 짝수번째의 열과 홀수번째의 열은 각각 노드그룹 NG_0와 NG_1을 나타낸다.

정의 3 기본셀내에서 역방향 대각선으로 위치한 노드들을 **마스터 노드** MN_j라고 부른다.

즉, $MN_j = \{P_{2i+j,j} \mid 0 \le i \le N/2-1$ 이고 $j = 0, 1\}$. 그 이외의 노드들은 ***슬레이브 노드***라고 정의한다.



(a) Basic Cell Division

(b) Subnetworks

그림 2 16x16 토러스망에서의 기본셀과 마스터 노드 예

기본셀이 $P_{0,0}$, $P_{0,1}$, $P_{1,0}$ 및 $P_{1,1}$의 노드들로 구성된 예를 든다. $P_{0,0}$와 $P_{1,1}$은 마스터 노드이고 $P_{0,1}$와 $P_{1,0}$은 $P_{0,0}$와 $P_{1,1}$은 마스터 노드의 각각 슬레이브 노드이다. 그림 2에서 마스터 노드 MN_0와 MN_1는 각각 검정색 원과 점선 원으로 표시되어 있다. 그리고, 슬레이브 노드들은 점선 원으로 표시되어 있다.

정의 4 전체망에서 마스터 노드 MN_j의 집합은 ***서브네트워크***를 구성한다.

그러므로, 모든 토러스망에서는 2개의 서브네트워크

($subnetwork_0$와 $subnetwork_1$)를 가질 수 있다.

예를들어. 그림 2(b)에서는 2개의 서브네트워크를 나타내는데 각각 실선으로 된 것은 MN_0의 집합, 점선으로 된 것은 MN_1의 집합으로 된 서브네트워크의 예이다.

3.3 라우팅 알고리즘 SEM

이 장에서는 N x N, $N=2^n$으로 구성된 2차원 토러스에서의 완전교환 알고리즘을 제시한다. 알고리즘 SEM은 3개의 스테이지로 구성되어 있다. 이 알고리즘의 개요가 그림 3에 설명되어 있다. 각 스테이지는 메시지를 수집, 교환 및 분배하는 것으로 생각할 수 있다.

P와 P'를 x 혹은 y 좌표가 서로 다른 두개의 노드라고 할 때 P에서 P'로 같은 차원을 따라 메시지를 전송하는 것을 $P \xrightarrow{d} P'$ 으로 표시할 수 있다. (여기에서 $d \in \{+, -\}$이다.) 만약 d = "+" (그리고 "-")이면 라우팅이 해당 차원을 따라서 순방향(그리고 역방향)으로 일어나는 것을 나타낸다.

완전교환 통신에서는 각 노드마다 자기 이외의 $N^2 - 1$개의 노드에게 메시지를 보내야하기 때문에 설명을 간단히 하기 위하여 자기 노드에게 보내는 1개의 가상 메시지를 포함시키면 각 노드 $P_{i,j}$는 N^2개의 메시지가 필요하게 된다. 이 메시지들은 배열 $M_{i,j}[0 : N-1, 0 : N-1]$에 각각 저장되어 있다.

Procedure:

---{스테이지 1(분할(*Split*) 단계)}

1. 주어진 망을 2 x 2의 기본망으로 분할한다.
2. 기본망의 각 마스터 노드들은 슬레이브 노드의 메시지를 수집한다.
3. 스테이지 2에서는 슬레이브 노드를 제외시킨다.

---{스테이지 2(교환(*Exchange*) 단계)}

1. 마스터 노드들로 된 서브네트워크별로 완전 교환 알고리즘을 수행한다.

---{스테이지 3(합병(*Merge*) 단계)}

1. 각 마스터 노드는 슬레이브 노드에게 메시지를 재분배한다.

그림 3 완전교환 알고리즘 SEM의 형태

스테이지 1에서는 기본셀을 독립단위로 하여 마스터 노드들이 슬레이브 노드의 메시지를 수집하게 된다. 메시지들의 라우팅 형태는 그림 4에 표시되어 있다. 예를들어 $P_{0,0}$, $P_{0,1}$, $P_{1,0}$ 및 $P_{1,1}$로 구성된 기본셀에서는 NG_0(또한 NG_1)가 목적지인 모든 메시지들은 마스터 노드 $P_{0,0}$(또한 $P_{1,1}$)로 수집된다. 이 스테이지는 2단계를 필요로 한다. 마스터 노드들은 스테이지 1에서 기본 셀내의 각 열에 위치한 노드들의 대표 노드로서 메시지를 수집하고 스테이지 3에서는 메시지들을 분배시키게 된다.

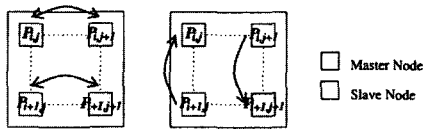


그림 4 스테이지 1에서 각 기본셀의 메시지 라우팅 형태

스테이지 1 :

// 각 마스터 노드에서 메시지 수집을 한다.

단계 1: // 차원 0을 따라 메시지를 수집한다.

parbegin

If j *mod* 2 = 0, then $P_{i,j} \longrightarrow P_{i,\,(j+1)}$ mod N.

If j mod 2 = 1, then $P_{i,j} \longrightarrow P_{i,\,(j-1)}$ mod N.

parend

단계 2: // 차원 1을 따라 메시지를 수집한다.

parbegin

If $i \neq j$, then $P_{i,j} \longrightarrow P_{(i+1)\ mod\ N,\ j}$.

$P_{i,j} \longrightarrow P_{(i-1)\ mod\ N,\ j}$.

parend

그림 5 완전 교환 알고리즘 - 스테이지 1

스테이지 1이 끝나게 되면 모든 기본 셀내의 2개의 마스터 노드에는 다른 노드로 보내질 메시지들이 준비된 상태이다. 슬레이브 노드의 메시지들이 마스터 노드에서 수집되기 때문에 메시지 배열의 크기는 $2N$ x N가 된다. 다음 스테이지에서는 각 기본셀의 2개의 마스터 노드가 망의 크기가 $N/2$ x $N/2$으로서 완전교환을 수행하게 된다.

각 스테이지가 진행됨에 따라서 $M_{i,j}$의 원소 값들이 변하게 되는데 이를 $M_{i,j}^{<s,p,t>}$로 표기한다. 여기에서 s는 각 단계, p는 페이즈, t는 스테이지를 의미한다. (적절한 값이 없는 경우는 '*' 로 표기한다.) $M_{i,j}$의 초기값은 $M_{i,j}^{<0,0,0>}$로 표기한다.. 다음의 소정리는 주어진 메시지 $M_{i,j}$가 어떻게 변천하는지를 보여준다.

소정리 1 N x N 토러스에서 $M_{i,j}^{<*,*,*>}[0{:}N-1, 0{:}N-1]$을 고려한다. 스테이지 1을 완료한 후에 $M_{i,j}^{<*,*,1>}[a, b]$ $(0 \leq a < N, 0 \leq b < 2N)$는 다음과 같이 주어진다.

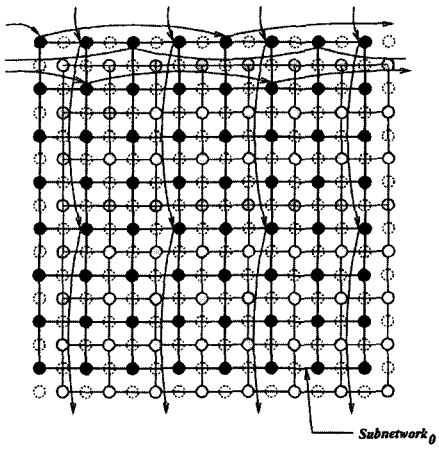
$$Mi,j\langle *,*,1\rangle[a,b] = \begin{cases} M_{i,j}^{\langle 0,0,0\rangle}[a,b/2], & \text{if } a = even \text{ and } b = even, \\ M_{i\pm1,j}^{\langle 0,0,0\rangle}[a,b/2], & \text{if } a = even \text{ and } b = odd, \\ M_{i,j\pm1}^{\langle 0,0,0\rangle}[a,b/2], & \text{if } a = odd \text{ and } b = even, \\ M_{i\pm1,j\pm1}^{\langle 0,0,0\rangle}[a,b/2], & \text{if } a = odd \text{ and } b = odd, \end{cases}$$

단, ±는 i = j = *even*일 때 "+", i = j = *odd*일 때 "−".

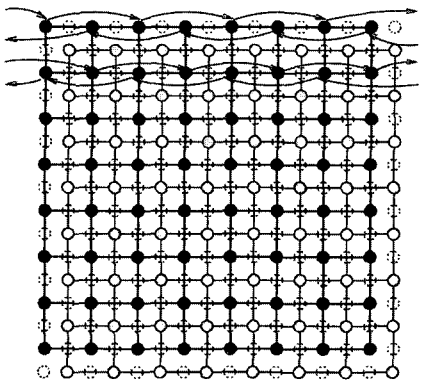
스테이지 2에서는 2개의 독립적인 서브네트워크가 각각 완전교환을 수행한다. 이 스테이지는 4단계로 구성되어 있다. 임의의 노드에서 모든 메시지들이 최대한 자기 목적지보다 적어도 8홉이내에 배치하기 위하여 페이즈 1과 2에서는 *(N/8-1)* 단계동안 각 차원을 따라 8 홉만큼 떨어진 노드로 메시지를 전송한다. 페이즈 3에서는 2단계동안 각 차원을 따라 4 홉만큼 떨어진 노드로 메시지를 전송한다. 페이즈 4에서는 2 단계동안 각 차원을 따라 2 홉마큼 떨어진 노드로 메시지를 전송한다. 스테이지 2는 망의 크기가 16 x 16이상인 경우에만 실행이 된다.

이 단계에서는 마스터 노드들만 통신을 수행한다. 즉, $\{P_{i,j} \mid 0 \le i \le N-1$ 이고 $j=2i+m$, 여기서, $m = 0$ 혹은 $1\}$. 아래의 표기에서는 $p=i/2$, $q=j/2$('/' 는 정수형 나눗셈)임을 나타낸다.

라우팅 형태는 그림 6에 나타나 있다. 이 그림에서는 간단하게 $subnetwork_0$에 대한 페이즈 1과 3만을 표시하였다.



(a) Routing Patterns for Phase 1



(b) Routing Patterns for Phes 3

그림 6 16x16 토러스의 라우팅 형태(스테이지 2)(참고:라우팅 형태는 일부만 표시)

Stage 2:
  Phase 1 :
    For *Step=1* to *N/8-1*
      parbegin
        If *(p+q) mod 4 = 0*, then $P_{i,j} \rightarrow P_{i,\ (j+8)\ mod\ N}$.
        If *(p+q) mod 4 = 1*, then $P_{i,j} \rightarrow P_{(i+8)\ mod\ N,\ j}$.
        If *(p+q) mod 4 = 2*, then $P_{i,j} \rightarrow P_{i,(j-8)\ mod\ N}$.
        If *(p+q) mod 4 = 3*, then $P_{i,j} \rightarrow P_{(i-8)\ mod\ N,\ j}$.
      parend
    End for
  Phase 2 :
    For *Step=1* to *N/8-1*
      parbegin
        If *(p+q) mod 4 = 0*, then $P_{i,j} \rightarrow P_{(i+8)\ mod\ N,\ j}$.
        If *(p+q) mod 4 = 1*, then $P_{i,j} \rightarrow P_{i,(j+8)\ mod\ N}$.
        If *(p+q) mod 4 = 2*, then $P_{i,j} \rightarrow P_{(i-8)\ mod\ N,\ j}$.
        If *(p+q) mod 4 = 3*, then $P_{i,j} \rightarrow P_{i,(j-8)\ mod\ N}$.
      parend
    End for

그림 7 완전 교환 알고리즘 - 스테이지 2(페이즈 1과 2)

Stage 2:
  Phase 3:
    Step 1:
      parbegin
        If *(p+q) mod 4 = Even* and *q = 0* or *1*, then $P_{i,j} \rightarrow P_{i,\ (j+4)\ mod\ N}$.
        If *(p+q) mod 4 = Even* and *q = 2* or *3*, then $P_{i,j} \rightarrow P_{i,\ (j-4)\ mod\ N}$.
        If *(p+q) mod 4 = Odd* and *q = 0* or *1*, then $P_{i,j} \rightarrow P_{(i+4)\ mod\ N,\ j}$.
        If *(p+q) mod 4 = Odd* and *q = 2* or *3*, then $P_{i,j} \rightarrow P_{(i-4)\ mod\ N,\ j}$.

```
    parend
  Step 2:
    parbegin
      If (p+q) mod 4 = Even and q = 0 or 1, then
        P_{i,j} +→ P_{(i+4) mod N, j}.
      If (p+q) mod 4 = Even and q = 2 or 3, then
        P_{i,j} -→ P_{(i-4) mod N, j}.
      If (p+q) mod 4 = Even and q = 0 or 1, then
        P_{i,j} +→ P_{i, (j+4) mod N}.
      If (p+q) mod 4 = Even and q = 2 or 3, then
        P_{i,j} -→ P_{i, (j-4) mod N}.
    parend
Phase 4:
  Step 1:
    parbegin
      If (p+q) mod 2 = 0, then P_{i,j} +→ P_{i, (j+2) mod N}.
      If (p+q) mod 2 = 1, then P_{i,j} -→ P_{i, (j-2) mod N}.
    parend
  Step 2:
    parbegin
      If (p+q) mod 2 = 0, then P_{i,j} +→ P_{(i+2) mod N, j}.
      If (p+q) mod 2 = 1, then P_{i,j} -→ P_{(i-2) mod N, j}.
    parend
```

그림 8 완전 교환 알고리즘 - 스테이지 2(페이즈 3 과 4)

소정리 2 *NxN* 토러스에서 $M_{i,j}^{<*,*,*>}[0:N-1, 0:N-1]$을 고려한다. 스테이지 2를 완료한 후에 $M_{i,j}^{<k,1,2>}[a,b](k=1, ..., N/8-1, , 0 \le a < N, 0 \le b < 2N)$는 다음과 같이 주어진다.

$$M_{i,j}^{<k,1,2>}[a,b] = \begin{cases} M_{i,j}^{<0,1,2>}[a,b], & \text{if } j \le b < j+8, \\ M_{i,j+N-8m}^{<0,1,2>}[a,b-8m], & otherwise \end{cases} \quad (1)$$

(단, if $(b-j) \bmod N \ge 8k$ then $m = k$ else $m = b/8$(정수형 나눗셈)). 그리고, $M_{i,j}^{<k,2,2>}[a,b](k=1, ..., N/8-1, , 0 \le a < 2N, 0 \le b < N)$는 다음과 같이 주어진다.

$$M_{i,j}^{<k,2,2>}[a,b] = \begin{cases} M_{i,j}^{<0,2,2>}[a,b], & \text{if } i \le a < i+8, \\ M_{i+N-8m,j}^{<0,2,2>}[a-8m,b], & otherwise \end{cases} \quad (2)$$

(단, if $(a-i) \bmod N \ge 8k$ then $m = k$ else $m = a/8$($정수형 나눗셈)).

증명:

식 1와 식 2과의 다른 점은 식 1은 수평방향으로의 메시지 이동을 의미하고 식 2는 수직 방향인 점이다. 그러므로, 여기서는 식 1만 증명을 한다.

$j \le b < j+8$일 때는 명백하다. b가 이 범위가 아닐 때는 k에 대하여 귀납법을 사용하여 증명한다. 먼저, $(b-j) \bmod N \ge 8k$인 경우를 고려한다. $k = 1$일 때는 식 1은 $(b-j) \bmod N \ge 8$인 관계를 유지하면서 명백하다. 식 1이 $k = p$인 경우가 참이라고 가정하자. 즉, 아래의 식은 참이 된다.

$$M_{i,j}^{<p,1,2>}[a,b] = M_{i,j+N-8m}^{<0,1,2>}[a,b-8m], \text{ 단, } (b-j) \bmod N \ge 8p$$

$k=p+1$일 때는 먼저 메시지 $M_{i,j}^{<p,1,2>}[a,b]$를 p스텝만큼 수평방향으로 옮긴 뒤 한 스텝(즉, 8 홉)을 옮긴 것과 동일하다. 이것은 다음과 같이 다시 쓸 수 있다.

$$\begin{aligned} M_{i,j}^{<p+1,1,2>}[a,b] &= \text{Move } M_{i,j}^{<p,1,2>}[a,b] \text{ 8 hops, 단, } (b-j) \bmod N \ge 8p \\ &= \text{Move } M_{i,j+N-8m}^{<0,1,2>}[a,b-8m] \text{ 8 hops, 단, } (b-j) \bmod N \ge 8p \\ &= M_{i,j+N-8m}^{<0,1,2>}[a,b-8m], \text{ 단, } (b-j) \bmod N \ge 8p+8. \end{aligned}$$

마찬가지로 $(b-j) \bmod N < 8k$일 때도 같은 방법으로 증명할 수 있다. 그러므로 식 1이 증명된다.

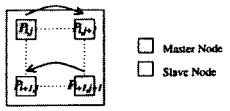


그림 9 각 기본셀에서의 라우팅 형태(스테이지 3)

스테이지 3에서는 기본셀내에서 각 마스터 노드들은 해당 슬레이브 노드에게 메시지를
분배한다. 이 스테이지는 1 단계만 소요된다. 라우팅 형태는 그림 9에 나타나 있다.

```
Stage 3 :
  //각 마스터 노드는 메시지를 분배한다.
  parbegin
    If i mod 2 = 0 and j mod 2 = 0, then P_{i,j} +→ P_{i, j+1}.
      If i mod 2 = 1 and j mod 2 = 1, then P_{i,j} -→ P_{i, j-1}.
    parend
```

그림 10 완전 교환 알고리즘 - 스테이지 3

3.4 메시지의 이동과 정렬

이 장에서는 각 단계마다 전송되는 메시지 내용들을 예를 들어 설명한다. 완전교환 통신에서는 토러스내의

각 노드들은 자기 이외의 N^2 - 1개 만큼의 노드들에게 전송할 메시지를 가지고 있다. 일반적으로 각 노드 $P_{i,j}$는 자기 자신을 포함하여 N^2개의 전송 메시지를 가지고 있다. 이러한 메시지들은 $M_{i,j}[0 : N-1, 0 : N-1]$ 배열에 각 한개씩 메시지가 저장되어 있다. 본 절에서는 16 x 16 토러스를 사용하여 노드 $P_{0,0}$에서의 메시지 이동을 중심으로 설명한다.

그림 11에서 그림 20에 표시된 기호 "√"는 새로운 메시지가 현재의 행이나 열에 도착하였음을 표시하고 각 단계에서 표시된 숫자 'pqrs'는 메시지가 원천지 노드 $P_{p,q}$에서 목적지 노드 $P_{r,s}$로 전송됨을 의미한다.

스테이지 1에서는 $P_{0,0}$가 기본셀내에서 마스터 노드이기 때문에 홀수 열이 목적지인 모든 메시지들이 $P_{0,1}$로 전송되고 짝수 열이 목적지인 모든 메시지들이 $P_{0,1}$로부터 수신이 된다.

이것이 단계 1이며 그림 12에 나타나 있다. 단계 2에서는 $P_{1,0}$와 $P_{1,1}$로부터 메시지가 전송되어 세번째와 네번째 열에 위치되는 것을 그림 13에서 볼 수 있다. 단계 2가 끝나게 되면 기본셀 내의 각 노드들의 메시지 중 짝수 열로 향하는 모든 메시지들은 마스터 노드 $P_{0,0}$에 모이게 된다.

스테이지 2에서는 마스터 노드들만이 통신을 한다. 페이즈 1에서는 메시지들이 차원 0(X축 방향)을 따라서 8 홉만큼 떨어진 노드들과 메시지를 주고 받는다. 그림 14에서는 노드 $P_{0,8}$에서의 메시지가 행 8부터 F사이에 도착하는 것을 볼 수 있다. 마찬가지로 차원 1(Y축 방향)을 따라서 8홉만큼 떨어진 노드간에 메시지가 송수신된다.

그림 15는 노드 $P_{2,0}$에서의 메시지가 행 4에서 행 7, C에서 행 F, 행 14에서 17, 행 1C에서 1F 사이에 도착하는 것을 볼 수 있다. 스테이지 3에서는 홀수 행의 메

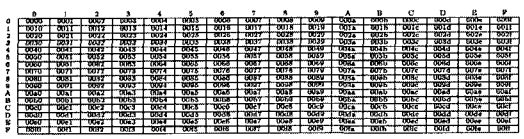


그림 11 초기 메시지의 형태

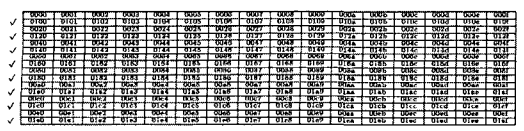


그림 12 스테이지 1 중 단계 1이 끝난 후의 상태

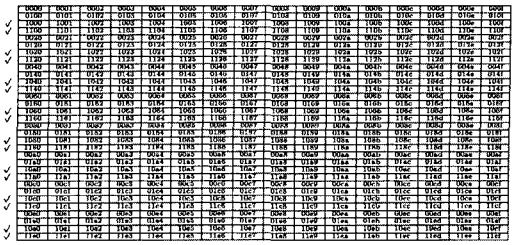


그림 13 스테이지 1 중 단계 2가 끝난 후의 상태

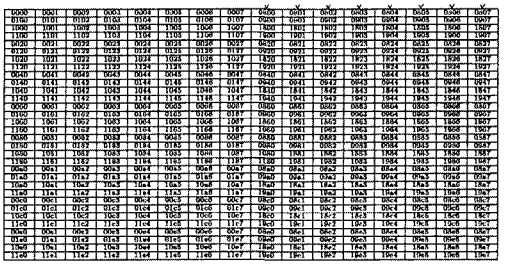


그림 14 스테이지 2 중 페이즈 1이 끝난 후의 상태

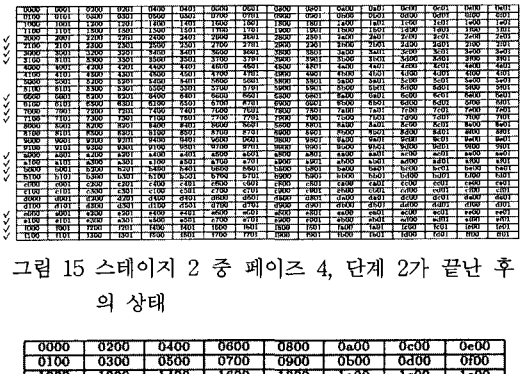


그림 15 스테이지 2 중 페이즈 4, 단계 2가 끝난 후의 상태

0000	0200	0400	0600	0800	0a00	0c00	0e00	1000	1200	1400	1600	1800	1a00	1c00	1e00
0100	0300	0500	0700	0900	0b00	0d00	0f00	1100	1300	1500	1700	1900	1b00	1d00	1f00
2000	2200	2400	2600	2800	2a00	2c00	2e00	3000	3200	3400	3600	3800	3a00	3c00	3e00
2100	2300	2500	2700	2900	2b00	2d00	2f00	3100	3300	3500	3700	3900	3b00	3d00	3f00
4000	4200	4400	4600	4800	4a00	4c00	4e00	5000	5200	5400	5600	5800	5a00	5c00	5e00
4100	4300	4500	4700	4900	4b00	4d00	4f00	5100	5300	5500	5700	5900	5b00	5d00	5f00
6000	6200	6400	6600	6800	6a00	6c00	6e00	7000	7200	7400	7600	7800	7a00	7c00	7e00
6100	6300	6100	6700	6900	6b00	6d00	6f00	7100	7300	7100	7700	7900	7b00	7d00	7f00
8000	8200	8400	8600	8800	8a00	8c00	8e00	9000	9200	9400	9600	9800	9a00	9c00	9e00
8100	8300	8100	8700	8900	8b00	8d00	8f00	9100	9300	9100	9700	9900	9b00	9d00	9f00
a000	a200	a400	a600	a800	aa00	ac00	ae00	b000	b200	b400	b600	b800	ba00	bc00	be00
a100	a300	a100	a700	a900	ab00	ad00	af00	b100	b300	b100	b700	b900	bb00	bd00	bf00
c000	c200	c400	c600	c800	ca00	cc00	ce00	d000	d200	d400	d600	d800	da00	dc00	de00
c100	c300	c100	c700	c900	cb00	cd00	cf00	d100	d300	d100	d700	d900	db00	dd00	df00
e000	e200	e400	e600	e800	ea00	ec00	ee00	f000	f200	f400	f600	f800	fa00	fc00	fe00
e100	e300	e500	e700	e900	eb00	ed00	ef00	f100	f300	f500	f700	f900	fb00	fd00	ff00

그림 16 스테이지 3이 끝난 후의 상태

시지들은 슬레이브 노드의 메시지임을 알 수 있다. 각 마스터 노드가 해당 슬레이브 노드에게 메시지를 보내고 나면 원하는 완전교환 통신이 완료된다. 바로 전 메시지의 상태가 그림 16에 표시되어 있다.

4. 복잡도 분석과 평가

본 절에서는 개시지연시간(start-up time)과 전송시간(data transmission time)등의 관점에서 제시된 알고리즘에 필요한 복잡도를 분석한다.

•개시지연시간(*startup latency*) : 이 시간은 채널간에 통신을 위해 소요되는 시간을 말한다. 스테이지 1은 2단계로 구성되어 있다. 스테이지 2에서의 페이즈 1과 2는 각각 *N/8-1* 단계가 소요된다. 그리고, 페이즈 3과 4는 각 2단계로 구성되어 있다. 그러므로, 스테이지 2에서 필요한 개시지연 시간은 *N/4+2* 단계이다. 스테이지 3은 단지 1개의 단계만 필요하다. 그러므로, 제시된 알고리즘이 필요한 개시지연시간은 $(N/4+5)t_s$(t_s는 한 개의 메시지에 필요한 개시지연시간을 의미)이다.

•데이타 전송 시간(*data transmission time*) : 이는 망을 통하여 실제 메시지가 전송되는데 소요되는 시간을 말한다. 스테이지 1은 2단계로 구성되어 있는데 각 단계마다 $N^2/2$와 N^2 만큼의 메시지 전송이 필요하다. 스테이지 2에서는 각 마스터 노드에 배치되어 있는 메시지 배열의 형태가 $2N^2$이다. 이 스테이지에서 페이즈 1과 2는 각각 *N/8-1* 단계로 되어 있으며 각 단계마다 N^2개의 메시지 전송이 필요하다. 마찬가지로 페이즈 3에서는 각 단계마다 N^2개의 메시지를 전송한다. 스테이지 3에서는 N^2개의 메시지들이 전송된다. 그러므로, 제시된 알고리즘에서 필요한 총 데이타 전송시간은 $(N^2(N+10)/4)mt_d$(단, m은 메시지의 길이, t_d는 1 바이트의 메시지 전송에 필요한 시간을 의미)이다.

•재정렬 시간(*data rearrangement time*) : 이는 다음 단계를 위해 메시지들의 순서를 정렬하는데 필요한 시간이다. 스테이지 1은 $2N^2$개의 메시지를 재정렬하여야 한다. 스테이지 2는 $6N^2$개의 메시지를, 스테이지 3은 $2N^2$개의 메시지를 재정렬하여야 한다. 그러므로, 제시된 알고리즘에서 필요한 총 데이타 재정렬 비용은 $10N^2mmt_r$(단, m은 메시지의 길이, t_r은 메모리내에서 1 바이트를 정렬하는데 소요되는 시간을 의미)이다.

•지연 시간(*propagation delay time*) : 이는 주어진 메시지가 목적지에 도달하기 위해 필요한 홉의 수를 측정하는 시간이다. 스테이지 1에서는 주어진 메시지는 단계 1과 2는 각 1번의 홉만 필요하다. 스테이지 2의 페이즈 1과 2에서는 각각 *N/8-1* 번 동안 8번의 홉을 거치게 된다. 또, 페이즈 3에서는 각 단계마다 4번의 홉을 필요로 한다. 그러므로, 스테이지 2에서는 총 *2N-8* 번의 홉이 필요하다. 스테이지 3에서는 1번의 홉만 필요하다. 그러므로, 제시된 알고리즘에서 필요한 총 지연시간은 $(2N-5)t_l$(단, t_l는 한 개의 플릿이 링크를 통과하는 시간)이다.

제시된 알고리즘과 기존의 알고리즘들의 시간 분석에 관한 비교를 표 1에 나타나 있다. 제시된 알고리즘 *SEM*은 메시지 전송시간 면에서는 최적이라는 것을 알 수 있다. 그리고, 개시지연 시간은 비교된 모든 알고리즘이 $O(N)$이지만 *SEM*은 약 2배정도 우수함을 알 수 있다.

2차원 토러스의 경우 Suh[14]와 Tseng et. al.[18]가 간접적인 방법(indirect solution)을 사용한 알고리즘을 제안하였다. 비록 제시된 알고리즘이 스테이지 1에서 마스터 노드에 수집되는 메시지를 보관하기 위해 기존의 제시된 알고리즘보다 약 2배정도의 기억 공간을 필요로 하지만 웜홀 방식을 사용하는 망에서 개시지연시간의 최소화가 중요하다는 관점에서 본다면 우수한 방법이라고 할 수 있다. Suh의 알고리즘은 망의 크기 제한의 관점에서 볼 때 Tseng et. al.의 알고리즘보다 더 우수하다는 것을 알 수 있다. 제시된 알고리즘은 망의 크기를 2의 멱승으로 가정을 하였지만 주어진 망의 크기가 2의 멱승이 되지 않을 때는 가상 노드를 추가시켜 문제를 해결할 수 있다. 이 경우 개시지연시간이 조금 더 필요하지만 여전히 성능이 기존의 알고리즘보다 우수한 것을 알 수 있다. 또, 망의 크기가 아주 큰 경우(예를 들어, 512 x 512 혹은 그 이상)에는 기본셀의 크기를 4 x 4로 수정하여 제시된 알고리즘을 수행하면 기억공간이 더 필요하더라도 개시지연시간을 상당히 줄일 수 있다.

표 1 완전교환 알고리즘의 복잡도 분석

분류	개시지연시간	데이타 전송시간	데이타 정렬시간
알고리즘 *SEM*	$(\frac{1}{4}N+5)t_s$	$\frac{1}{4}(N^3+10N^2)mt_d$	$10N^2mt_r$
알고리즘 *[18]*	$(\frac{1}{2}N+2)t_s$	$\frac{1}{4}(N^3+4N^2)mt_d$	$\frac{1}{2}(N^3+2N^2)mt_r$
알고리즘 *[14]*	$(\frac{1}{2}N+2)t_s$	$\frac{1}{4}(N^3+4N^2)mt_d$	$3N^2mt_r$

5. 결 론

본 논문에서는 웜홀 방식의 2차원 병렬 컴퓨터에서 사용되는 효율적인 완전 교환 알고리즘을 제안하였다. 제시한 알고리즘은 웜홀 방식의 컴퓨터에서 개시 지연

시간을 줄이기 위하여 분할 및 합병(divide-and-conquer) 방식을 채용하였다. 먼저 전체망을 2x2 형태의 기본셀로 분할한 뒤 각 기본셀에서는 마스터노드라고 불리는 특정 노드를 지정하여 기본셀내의 여타 노드들의 메시지를 이 마스터노드가 수집하여 전송할 수 있도록 지정하였다. 이 마스터노드들이 다른 모든 노드로 보내질 메시지를 수집한 뒤 각 기본셀내의 모든 마스터노드들만이 가상 망을 형성하여 망의 크기가 $N/2$ x $N/2$으로 줄어든 상태로 완전 교환 알고리즘을 수행한다. 마스터노드들간의 완전교환 연산을 수행한 뒤 이 마스터노드들은 자기가 전담했던 여타 노드들의 메시지를 재분배해 줌으로써 주어진 완전교환 연산을 완성할 수 있다. 기존의 여러 가지 알고리즘과의 비교 분석을 제시하였으며 제시한 알고리즘이 약 2배정도의 기동시간 면에서 우수함을 보였다.

향후 과제로서 첫째, 메시지 전송 시간을 줄이는 것인데, 제시된 알고리즘의 페이즈 3과 4의 각 단계에서 수평(그리고, 수직) 방향의 채널이 메시지를 전송할 때 수직(수평) 방향의 채널이 사용되지 않는 점을 고려할 때 스테이지 1에서의 메시지 전송을 적절히 배치한다면 메시지 전송 시간을 더 줄일 수가 있을 것이다. 둘째, 비록 2차원 토러스인 경우에만 한정지었지만 제시된 알고리즘을 쉽게 3차원 토러스로 확장시키는 것이다.

참 고 문 헌

[1] W.C. Athas and C.L. Seitz, "Multicomputers: Message-Passing Concurrent Computers," *IEEE Computers*, Vol. 21, No. 8, pp.9-24, Aug. 1988.

[2] S. Bokhari and H. Berryman, "Complete exchange on a circuit switched mesh," *Proc. of the 1992 Scalable High Performance Computing Conference*, pp.300-306, 1992.

[3] J. Bruck, C. T. Ho, and D. Weatherby, "Efficient Algorithms for All-to-All Communications in Multi-Port Message-Passing Systems," *Proc. of Symposium on Parallel Algorithms and Architectures*, pp. 298-309, 1994.

[4] W. J. Dally and C. L. Seitz, "The Torus Routing Chip," *Journal of Parallel and Distributed Computing,"* vol. 1, no. 3, pp. 187--196, 1986.

[5] S. L. Johnsson and C. T. Ho, "Optimum Broadcasting and Personalized Communications in Hypercubes," *IEEE Transactions on Computers*, 38(9):1249--1268, Sep. 1989.

[6] S. G. Kim, S. R. Maeng and J. W. Cho, "Complete Exchange Algorithms in Wormhole-Routed Torus Networks: A Divide-and-Conquer Strategy," *Proc. of Int'l Symposium on Parallel Architectures, Algorithms and Networks*, pp. 296-301, 1999.

[7] P.K. McKinley, Y.J. Tsai and D.F. Robinson, "A Survey of Collective Communication in Wormhole-Routed Massively Parallel Computers," Technical Report, MSU-CPS-94-35, Michigan State University, June 1994.

[8] Message Passing Interface Forum, "Document for standard message-passing interface," Technical Report CS-93-214, University of Tennessee, Nov. 1993.

[9] L.M. Ni and P.K. McKinley, "A Survey of Wormhole Routing Techniques in Direct Networks," *IEEE Computer*, Vol. 26, No. 2, pp.62-76, Feb. 1993.

[10] D.A: Reed and R.M. Fujimoto, *Multicomputer Networks: Message Based Parallel Processing*, MIT Press, Cambridge, MA, 1987.

[11] D.F. Robinson, D. Judd, P.K. McKinley, and B.H.C. Cheng, "Efficient Collective Data Distribution in All-Port Wormhole-Routed Hypercubes," *Proc. of Supercomputing '93*, Nov. 1993, pp.792-801.

[12] D. S. Scott, "Efficient All-to-All Patterns in Hypercube and Mesh Topologies," *Proc. of the 6th Conference Distributed Memory Concurrent Computers*, pp. 398-403, 1991.

[13] S. R. Seidel, "Circuit-Switched vs. Store-and-Forward Solutions to Symmetric Communication Problems," *Proc. of the 4th Conference Hypercube Concurrent Computers Applications*, pp. 253-255, 1989.

[14] Y. Suh and S. Yalamanchili, "Efficient Algorithms for Complete exchange in 2D Tori," *Proc. of the 9th IASTED Int'l Conference Parallel and Distributed Computing and Systems*, pp.113-119, 1997.

[15] N. Sundar, D. Jayasimha, D. Panda, and P. Sadayappan, "Complete exchange in 2D Meshes," *Proc. of the 1994 Scalable High Performance Computing Conference*, pp.406-413, 1994.

[16] R. Thakur and A. Choudhary, "All-to-all communication on meshes with wormhole routing," *Proc. of the 1994 International Parallel Processing Symposium*, pp.561-565, 1994.

[17] Y.C. Tseng and S. Gupta, "All-to-All Personalized Communication in a Wormhole-Routed Torus," *IEEE Tran. on Parallel and Distributed Systems*, Vol. 7, No. 5, pp.498-505, May 1996.

[18] Y.C. Tseng, T.H. Lin, S. Gupta and D.K. Panda, "Bandwidth-Optimal Complete Exchange on Wormhole-Routed 2D/3D Torus Networks: A Diagonal-Propagation Approach," *IEEE Tran. on Parallel and Distributed Systems*, Vol. 8, No. 4, pp. 380-396, Apr. 1997.

김 시 관

1882년 경북대학교 전자공학과 졸업. 1984년 한국과학기술원 전산학과 석사학위 취득. 1984년 ~ 1995년 삼성전자, LG정보통신 근무. 1994년 ~ 현재 한국과학기술원 전산학과 박사과정. 관심분야는 컴퓨터구조, 병렬처리, Wireless ATM등

맹 승 렬

정보과학회논문지 : 시스템 및 이론
제 27 권 제 4 호 참조

조 정 완

정보과학회논문지 : 시스템 및 이론
제 27 권 제 4 호 참조