Myrinet 클러스터링 시스템에서 순위차원 라우팅을 사용하는 통신들의 최적 스케줄링 방법

(Optimal Schedules for Dimension-Ordered Routing Communications in Myrinet Clustering Systems)

박 상 명 † 이 상 규 †† 문 봉 희 †††

(Sang-Myung Park) (Sang-Kyu Lee) (Bong-Hee Moon)

요 약 최근 병렬처리 시스템에 대한 연구는 마이크로 프로세서 제작 기술과 네트워크 기술이 발달함에 따라 고성능의 PC와 워크스테이션 여러대를 고속의 네트워크로 연결하여 구축하는 클러스터링 환경에 관심이 고조되고 있다. 그런데, 이러한 클러스터링 시스템의 성능은 수행되는 애플리케이션 프로그램의 병렬성이나 xdhtls 빈도 등의 특성에 따라 달라진다. 그러므로 클러스터링 시스템의 성능을 향상시키기 위해서는 애플리케이션의 이러한 특성을 고려하여 최상의 효과를 얻기 위한 조정작업이 필요하며 그 방법 중의 하나가 시스템 상에서 발생하는 통신들에 대하 스케줄링을 수행하는 것이다.

본 논문에서는 Myrinet 스위치를 사용하여 선형으로 구성한 클러스터링 시스템과 2차원 메쉬 형태로 구성한 클러스터링 시스템의 두 가지 모델을 가정하고, 이들 모델 상에서 특정 시간에 주어지는 통신 요청들에 대하여 순위차원 라우팅을 사용하여 메시지들을 최단시간에 전송할 수 있는 최적 통신 스케줄링 알고리즘을 제안한다.

시스템 상에서 같은 방향으로 동시에 링크를 공유하는 통신들의 개수의 최대값을 L_{max} 로, 시스템에서 하나의 메시지가 전달되는데 걸리는 시간을 T로 정의하면, 알고리즘에 의해 선형 네트워크에서의 통신 요청 집합에 대한 메시지 전송 완료 시간은 최대 $L_{max} \cdot T$, 메쉬 네트워크에서의 통신 요청 집합에 대한 메시지 전송 완료시간은 최대 $\frac{3}{2} L_{max} \cdot T$ 임을 증명하였다.

Abstract Recently, clustering of personal computers or workstation systems has been attracted by many researchers because of its high performance scale per unit cost. However, it is often found that the performance of PC/workstation clustering distributed processing is in various range depending on the computation and communication characteristics of application programs running on it. A perfect adjustment among those considerable factors in particular application is essential to high performance of the PC clustering distributed systems. One of those factors is scheduling communications among clustered system nodes.

In this thesis, we study communication scheduling in clustering environment using crossbar-based Myrinet as a communication medium. We carefully examine the characteristics of such clustered system model, and propose optimal communication scheduling algorithms that can transmit every communication requests in optimal time steps using dimension-ordered routing. The algorithm is designed for two network models: linear and 2D meshes.

We prove that the algorithms provide a solution for linear and 2D Mesh network topology with transmission time no larger than $L_{max} \cdot T$ and $\frac{3}{2} L_{max} \cdot T$, respectively, where L_{max} is the maximum load on a directed link and T is the unit time to transmit a message in Myrinet clustering systems. We also prove that the results we achieved are optimum.

† 비 회 원 : 숙명여자대학교 전산학과
sunmoon@chollian.net
†† 정 회 원 : 숙명여자대학교 전산학과 교수
sanglee@sookmyung.ac.kr
††† 종신회원 : 숙명여자대학교 전산학과 교수
moon@sookmyung.ac.kr
논문접수 : 2000년 1월 21일
심사완료 : 2000년 12월 15일

1. 서 론

최근 병렬처리 시스템에 대한 연구는 마이크로프로세서 제작기술과 네트워크 기술이 발달함에 따라 고성능의 PC와 워크스테이션 여러 대를 고속의 네트워크로 연결하여 구축하는 클러스터링 환경에 점차 관심이 고조되고 있다. 이는 클러스터링 환경이 가격 대 성능 비를 고려할 때 고성능 병렬컴퓨터와 같은 정도의 효과를 얻으면서 보다 저렴한 비용으로 대용량 데이타를 처리할 수 있는 기술로 인식되고 있기 때문이다. 이와 같은 연구의 결과로써 최근 대두된 웜홀 라우팅 랜은 높은 대역폭과 낮은 지연시간으로 메시지를 전송할 수 있는 특징 때문에 분산 컴퓨팅이나 클러스터링 컴퓨팅에 적용할 수 있는 매우 효과적인 시스템으로 주목받고 있다[1].

이러한 웜홀 라우팅 랜을 구축할 수 있는 대표적인 예가 바로 crossbar-based Myrinet이며, Myrinet은 기존의 Ethernet이나 FDDI와는 구별되는 여러 가지 특징으로 인해 새로운 형태의 랜을 구성한다[2, 3].

Myrinet은 여러 대의 호스트 컴퓨터를 하나의 클러스터링 시스템으로 묶을 수 있도록 하는 멀티포트 스위치, 컴퓨터와 스위치, 스위치와 스위치를 연결하는 링크와 호스트 인터페이스로 이루어진다. Myrinet 링크는 입력 채널과 출력 채널이 따로 존재하므로 양방향으로 동시에 2+2 Gigabit/second로 메시지를 전송할 수 있다.

Myrinet 스위치에는 4, 8, 16 포트를 가진 것이 있으며, 이러한 멀티포트 스위치로 구성할 수 있는 네트워크의 형태에는 특별한 제약이 없어 스위치의 포트가 허용하는 한도 내에서 어떠한 형태로도 구성이 가능하다. 그러나 Myrinet의 스위치 포트가 허용하는 한도 내에서 임의로 구성한 네트워크 형태와 특정한 형태로 정형화하여 구성한 네트워크를 비교하여 보면, 네트워크 상에서 노드간의 거리의 경우 전자의 네트워크 형태에서 더 작게 나타나지만 두 가지 형태의 네트워크간 성능에는 거의 차이가 없어 라우팅의 문제-더 간단하고 효과적인 라우팅 알고리즘을 적용할 수 있다-를 생각해 볼 때 네트워크의 형태를 정형화하는 것이 바람직하다.

이점을 고려하여 본 논문에서는 Myrinet 스위치를 사용하여 임의적으로 구성할 수 있는 네트워크 시스템의 형태들 중 가장 많이 사용되는 두 가지 종류의 Myrinet 모델[4]을 가정하고 이 모델 상에서의 통신 문제를 다룬다.

병렬 또는 분산처리를 위한 클러스터링 시스템에는 고려해야 할 많은 문제들이 있다. 이러한 문제들 중, 클러스터링 시스템에서 애플리케이션이 수행될 때 통신이 빈번하게 요구되는 경우 통신으로 인한 오버헤드 때문에 응답시간이 늦어진다거나 완료시간이 지연되는 문제가 발생할 수 있으며, 따라서 통신으로 인한 지연시간을 최소화 할 수 있는 방안이 모색되어야 한다. 관련 연구 [4, 5]에서는 시스템의 각 호스트간 통신 요청이 발생하면 요청이 발생한 시점에서 기존에 존재하는 통신들의 마감시간을 보장할 수 있는 범위 내에서 통신요청을 받아들이도록 하는 허가 제어(admission control)의 방법을 보여주고 있다. 그런데 이러한 허가 제어의 방법은 통신요청이 거부당했을 때 자원을 이용할 수 있을 때까지 프로그램 수행이 지연되는 문제가 발생할 수 있으며, 클러스터링 시스템 상에서 통신 요청이 많아질수록 통신요청에 대한 허가비율은 감소한다.[1] 이에 본 논문에서는 Myrinet으로 구성된 클러스터링 시스템에서 특정 시간에 호스트 컴퓨터들간에 요청되는 통신 집합(communication request set)이 존재할 때, 각각 요청되는 통신들에 대하여 소요되는 통신의 완료 시간을 최소화하기 위하여 통신 요청 집합 안의 통신들에 대한 스케줄링 방법을 제시한다. 스케줄링의 결과, 통신을 위해 지연되는 시간은 최적의 한도 시간을 초과하지 않음을 보장한다.

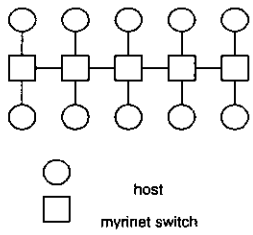


그림 1 선형구조

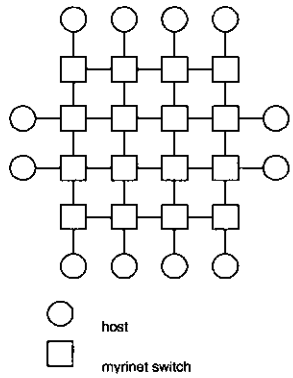


그림 2 2차원 메쉬구조

1) 자세한 내용은 참고문헌 4, 5를 참조.

그러면 이제 가정하고 있는 두 가지 클러스터링 시스템 모델을 제시한다. 첫 번째 모델은 4포트의 Myrinet 스위치를 선형으로 연결하고, [그림 1]에서 보는 것처럼 각 스위치에 호스트 컴퓨터를 연결하여 만든 네트워크이고, 두 번째 모델은 역시 4개의 포트를 가진 Myrinet 스위치를 2차원 메쉬 형태로 구성하여 메쉬의 사방 경계에 있는 스위치에 호스트 컴퓨터를 연결한 형태로 [그림 2]와 같다[4].

Myrinet에서 라우팅은 목적지 호스트의 위치에 따라 라우팅 경로를 선택하고 이 라우팅 정보를 Myrinet 패킷헤더에 모두 포함시켜 미리 정해진 경로로 메시지가 전달되게 하는 소스 라우팅 방식을 따르고 있다[2, 3]. 본 논문에서는 여러 가지 라우팅 방식 중 가장 단순하고 일반적이며 특성상 cyclic dependence가 없어 교착상태가 전혀 발생하지 않는 순위차원 라우팅[6] (dimension-ordered routing, 메쉬의 경우 row-column 라우팅)을 적용하여 Myrinet 패킷헤더에 라우팅 정보를 실어준다고 가정한다. 또한 본 논문에서는 메시지 전송 요청시 전송자 호스트가 하나의 목적지 호스트만을 갖는다고 가정한다.

다음 장에서는 Myrinet 클러스터링 환경 상에서 특정시간에 주어지는 통신 요청 집합의 스케줄링 시 전체 통신을 완료하는데 걸리는 총 통신시간의 최저 한도가 얼마인지 알아본다. 3장에서는 통신시간의 최저 한도 내에 통신 요청 집합에 속한 각각의 모든 통신을 마칠 수 있도록 보장하는 최적 스케줄링 방법을 소개하고, 마지막으로 4장에서 결론 및 향후 과제를 제시한다.

2. 통신시간 최저 한도

이 장에서는 앞장에서 가정한 두 가지 Myrinet 클러스터링 시스템 모델 상에서 특정 시간에 호스트 컴퓨터들 사이에 요청되는 통신들에 대하여 모든 메시지들을 전송하는데 소요되는 통신 시간의 최저 한도(최소한 얼마만큼의 시간이 소요되어야 하는가)를 알아본다. 시스템 상에서 같은 방향으로 동시에 링크를 공유하는 통신들의 개수의 최대 값을 L_{max}로, 시스템에서 하나의 메시지가 전달되는데 걸리는 시간을 T로 정의한다. 먼저 [그림 1]과 같이 Myrinet 스위치들을 선형으로 연결했을 경우를 본다.

정리 1. Myrinet 스위치들을 선형으로 연결한 클러스터링 시스템에서 순위차원 라우팅을 사용하는 통신 요청 집합 R에 대하여, 전체 메시지 전송 완료를 위해서는 최소한 $L_{max} \cdot T$ 시간이 소요된다.

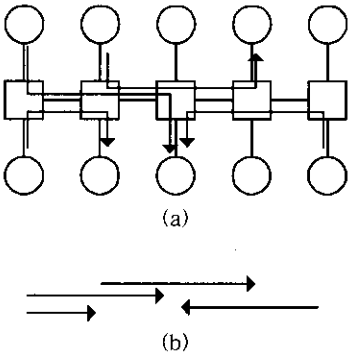


그림 3 coloring 문제

증명 [그림 3](a)는 선형 클러스터링 시스템에서 통신 요청 집합 R의 예를 보여주고 있다. [그림 3](b)는 [그림 3](a)에 나타나 있는 통신 요청들을 인터벌로 나타내어 인터벌 그래프의 형태로 표현한 것이다.

인터벌 그래프에서의 coloring은 각각의 인터벌들이 서로 교차하게 되면 그러한 인터벌간에는 다른 색들을 지정해 주어야 한다. 그런데, Myrinet의 링크는 양방향으로 메시지를 전달할 수 있으므로 스위치로 들어오는 통신과 나가는 통신이 링크를 공유한다고 해도 두 통신은 동시에 이루어질 수 있다. 이렇게 양방향으로 통신이 가능한 인터벌 그래프 상에서는 인터벌로 표시되는 각 통신들은 '같은 방향'으로의 통신 경로를 가지는 경우 그 통신들은 동시에 메시지를 보낼 수 없고 서로 다른 시간을 이용하여 전송이 이루어져야만 한다. 그러므로, 통신 요청 집합 R에 대한 통신 스케줄링의 문제는 시스템의 특성상 인터벌의 방향성이 존재한다는 것을 제외하고는 인터벌 그래프의 coloring 문제와 정확히 일치하고 있다[7].

coloring의 과정을 통해 같은 색으로 할당된 통신들은 동시에 통신이 가능하며, 따라서 특정시간에 네트워크에 존재하는 통신 요청 집합 R에 대한 전체 통신시간은 coloring에 사용된 색의 개수만큼과 단위시간의 곱으로 나타낼 수 있다. 그런데 위에서 언급한 인터벌 그래프에서의 coloring은 최저 한도가 max degree($= L_{max}$)로 널리 알려진 바이므로[8] 선형으로 이루어진 Myrinet 클러스터링 시스템에서 특정 시간에 주어진 통신 요청 집합 R 에 대한 메시지 전송 완료를 위해서는 최소한 $L_{max} \cdot T$ 시간이 소요되어야만 한다. ∎

다음은 Myrinet 스위치들을 2차원 메쉬 구조로 구성한 클러스터링 시스템에서 통신 요청 집합 R 에 대한

메시지 전송 시간의 최저 한도를 알아본다.

정리 2. Myrinet 스위치들을 2차원 메쉬구조로 연결한 클러스터링 시스템에서 순위차원 라우팅을 사용하는 통신 요청 집합 R에 대하여, 메시지 전송완료를 위해서는 최소한 $\frac{3}{2}L_{max} \cdot T$ 시간이 소요된다.

증명 정리 1의 선형 네트워크에서와 마찬가지로 2차원 메쉬구조를 이루는 네트워크에서의 통신 스케줄링의 문제도 그래프 coloring 문제로 바꾸어 생각해 볼 수 있다.

정리 2에 대한 증명은 이러한 경우-순위차원 라우팅을 사용하는 메쉬 구조의 Myrinet에서 통신 요청 집합 R에 대한 coloring을 수행하려면 적어도 $\frac{3}{2}L_{max}$개의 색들이 필요한 경우-의 통신 요청 집합 R이 존재함을 예를 보임으로써 증명하겠다.

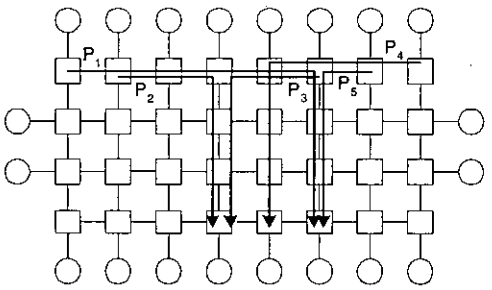


그림 4 $L_{max}=2$일 때 3 coloring의 예

[그림 4] 에서 보는 것처럼 4×8 메쉬 형태의 Myrinet에서 통신 요청 집합 $R=\{P_1, P_2, P_3, P_4, P_5\}$ 일 때의 coloring을 생각해 본다. [그림 4] 에서 보는 것처럼 $L_{max}=2$임을 쉽게 알 수 있다. coloring을 수행할 때 할당할 수 있는 색들을 $Color_k$ (k= *integer*)라 하면, 현재 색을 할당하기 위해 고려 중인 통신에는 링크를 공유하는 다른 통신에서 이미 사용되지 않은 색 $Color_k$ (k= *integer*) 중 가장 작은 첨자(index) k를 갖는 색을 선택하여 할당하도록 하겠다.

먼저, P_1에 $Color_1$을 할당한다. P_2와 P_1은 링크 충돌(link contention)이 발생하므로 P_1을 위해 사용되지 않은 색 중 첨자값이 가장 작은 색 $Color_2$를 P_2에 할당하고, P_3는 P_2에서 이미 사용된 $Color_2$는 사용할 수 없으나 P_1과는 다른 채널을 사용하여 링크 충돌이 발생하지 않으므로 $Color_1$을 할당한다. 다음으로 P_4는 P_3와 같은 색을 할당받으면 안되므로 $Color_2$를 할당한다. 그런데 마지막 P_5의 경우, $Color_2$를 받은 P_4, $Color_1$의 P_1과 링크를 공유하고 있으므로 $Color_1$, $Color_2$ 모두 사용할 수 없고 새로운 색이 필요하므로 P_5에는 할당할 수 있는 색 중 가장 작은 첨자 값을 가지는 색 $Color_3$을 할당해야만 한다. 이 예에서 어떠한 방법으로도 두 개의 색만을 가지고는 coloring을 할 수 없으므로 $L_{max}=2$ 일 때 3개의 색, 즉 $\frac{3}{2}L_{max}$개의 색이 필요하다. 따라서, 순위차원 라우팅을 따르는 메쉬구조의 Myrinet에서는 최소한 $\frac{3}{2}L_{max}$개의 색을 사용해야만 통신 요청 집합 R에 대하여 통신 스케줄링을 위한 coloring을 수행할 수 있으므로, 최소한 $\frac{3}{2}L_{max} \cdot T$ 시간이 소요되어야만 메시지 전송을 완료할 수 있다. 이것으로써 정리 2를 증명하였다. ∎

3. 최적 통신 스케줄링

이 장에서는 앞서 제시한 두가지 시스템 모델 상에서 coloring의 방법을 사용하여 특정 시간에 주어지는 통신 요청 집합 R에 대한 최적 스케줄링 방법을 제안한다.

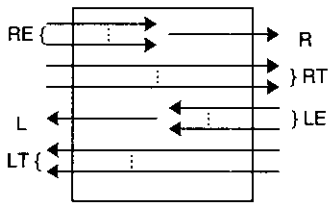


그림 5 선형 통신경로의 형태

3.1 선형 네트워크

먼저 Myrinet 스위치가 선형으로 연결된 클러스터링 시스템에서 통신 요청 집합 R에 대한 통신 완료 시간을 최소화 할 수 있는 통신 스케줄링 알고리즘을 알아본다.

앞장에서, 전체 시스템 상에서 동시에 같은 방향으로 링크를 공유하는 통신의 개수의 최대값을 L_{max} 라 정

표 1 선형 네트워크에서 각 스위치에 존재할 수 있는 통신 형태

L_i	스위치 i에서 출발하여 왼쪽 방향으로 나가는 통신
LE_i	스위치 i의 오른쪽에서 들어와 더 이상 다른 스위치로의 통신경로를 가지지 않는 통신
LT_i	스위치 i의 오른쪽으로부터 들어와서 왼쪽으로 통과해 나가는 통신
R_i	스위치 i에서 출발하여 오른쪽방향으로 나가는 통신
RE_i	스위치 i의 왼쪽에서 들어와 더 이상 다른 스위치로의 통신 경로 를 가지지 않는 통신
RT_i	스위치 i의 왼쪽으로부터 들어와 오른쪽으로 통과해 나가는 통신

```
알고리즘 Assign Color_Linear
입 력 : 통신 요청 집합, R
출 력 : 각 통신에 색을 할당한 결과

for i=1 to m {
  C(R_i) := Color_min{k | k=integer, Color_k ∉ C(RT_i)}
    /* RT_i에 사용되지 않은 Color_k 중에서 가장 작은
     첨자 값을 가지는 색을 선택하여 할당한다. */

  C(LE_i) := Color_min{k | k=integer, Color_k ∉ C(LT_i)}
    /* LT_i에 사용되지 않은 Color_k 중에서 가장 작은
     첨자 값을 가지는 색을 선택하여 할당한다. */
}
```

의했다. 그러면 선형의 Myrinet 클러스터링 시스템에서는 최대 L_{max}개의 색만을 사용하면 항상 coloring이 가능하며, *AssignColor_Linear* 알고리즘에서 보는 것처럼 간단한 방법으로 수행할 수 있다.

알고리즘 *AssignColor_Linear*는 스위치들을 기준으로 각 스위치를 통과하는 통신들에 색을 할당하는 방법을 사용하므로 알고리즘의 설명에 앞서 각 스위치에서 존재할 수 있는 통신의 유형에 대해 살펴본다. 선형의 Myrinet 클러스터링 시스템에서 모든 통신들은 스위치의 왼쪽과 오른쪽 양방향으로만 통과하여 목적지 호스트에 도착할 수 있으므로 각 스위치를 통과하는 통신들은 목적지 호스트의 위치에 따라 [그림 5]과 같이 6가지 유형 중 하나의 형태로 나타날 수 있다. 스위치를 기준으로 보아 왼쪽 방향으로 향하는 통신에는 L(left)을, 오른쪽으로 향하는 통신에는 R(right)을 사용하였고, 스위치의 외부로부터 들어와서 스위치에서 더 이상 다음 스위치로의 통신 경로를 가지지 않는 통신에는 E(end)를, 스위치를 통과해 다음 스위치로의 통신 경로를 갖는 통신에는 T(through)를 사용하여 통신의 유형을 명명하였다.

알고리즘 *AssignColor_Linear*는 **for loop**에서 통신 요청 집합에 대하여 coloring을 수행하고 있는데, 첨자 i 는 스위치의 위치를 나타내고 m은 선형으로 이루어진 스위치들의 개수를 나타낸다. coloring의 수행 시 사용할 수 있는 색들을 $Color_k(k=integer)$라 하고, 특정 통신 유형에 속하는 모든 개별 통신 각각의 색의 집합을 $C(통신유형)$ 라 하겠으며, 색을 선택하는 방법으로는 다른 통신과의 링크 충돌을 발생시키지 않도록 할 수 있는 많은 색들 중 가장 작은 값의 첨자 k 를 할당하는 갈방(greedy)법을 사용한다[7]. coloring의 수행은 일렬로 구성된 각 스위치를 가장 왼쪽에서 오른쪽으로 옮겨가면서 [그림 5]에 나타낸 통신의 유형 중 스위치에서 처음 나타나서 아직 색을 할당받지 못한 - 통신경로가 스위치에서 시작하거나 끝나는 경우를 의미 - 통신에 대해 적당한 색을 선택하여 할당한다. 가장 왼쪽 스위치를 통과하는 통신부터 시작하여 오른쪽 마지막 스위치를 통과하는 통신으로 색의 할당을 진행하므로 스위치 i에서 처음 나타나 색을 할당받지 못한 통신은 R_i, LE_i이며 스위치에서 R_i와 LE_i가 존재하면, 다음과 같이 coloring을 한다.

R_i는 $i-1$ 스위치에서 들어와서 i 스위치를 통과하는 통신 RT_i와 링크를 공유하므로 RT_i 에서 사용되지 않은 $Color_k$중 가장 작은 첨자 값을 갖는 색을 할당하고, LE_i에는 R_i와는 반대로 $i+1$에서 들어와서 i 스위치를 통과하는 통신과 링크를 공유하므로 LT_i 에서 사용되지 않은 $Color_k$중 가장 작은 첨자 값의 색을 할당한다. i가 m일 때까지 이 과정을 반복하면 전체 통신 요청 집합에 대해 적당한 색을 할당할 수 있다.

그러면 이제 제안한 알고리즘에 의해 통신 스케줄링을 수행한 결과로 보장될 수 있는 메시지 전송 완료시간의 최고 한도에 대하여 알아본다.

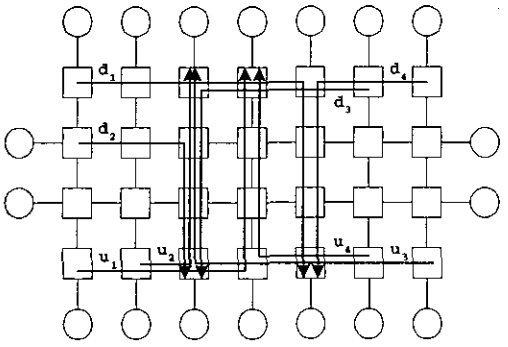
정리 3. Myrinet 스위치들을 일렬로 연결한 클러스터링 시스템에서 순위차원 라우팅을 사용하는 통신들의 메시지 전송은 $L_{max} \cdot T$ 시간 내에 완료할 수 있다.

증명 *AssignColor_Linear* 알고리즘을 적용하여 coloring을 수행하는 것을 생각해 보자. 만약 i번째 스위치에서 통신 경로 R_i에 있는 통신 경로 중 하나에 색을 지정해야 할 때 $Color_{L_{max}}$ 가 지정된다면, 이것은 통신 경로 R_i의 왼쪽 끝점이 시작되는 링크를 $Color_1$에서부터 $Color_{Lmax-1}$ 까지의 색을 지정 받은 통신 경로들과 함께 공유하는 경우뿐이다(즉, RT_i에 $Color_1$로부터 $Color_{Lmax-1}$의 색이 할당된 경우). 마찬가지로, 만약 통신 경로 R_i에 $Color_{Lmax+1}$ 를 할당해야 한다면 이 링크에 L_{max} 보다 많은 수의 통신 경로들이 존재하는 것이고 이러한 경우는 L_{max}의 정의에 위배되므로 발생할 수 없다. 따라서 *AssignColor_Linear* 알고리즘은 $L_{max} \cdot T$ 시간 내에 요청된 통신을 완료 할 수 있다.■

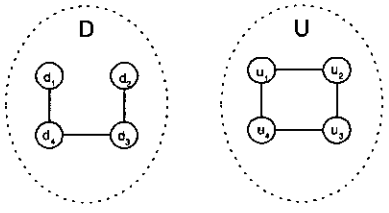
3.2 메쉬 네트워크

다음으로 스위치의 연결이 $n \times m$ 2차원 메쉬구조로 이루어진 Myrinet 클러스터링 시스템에서 통신 요청 집합 R에 대한 최적 통신 스케줄링에 대해 생각해 본다.

2차원 메쉬구조에서의 path coloring은 NP-complete로 잘 알려진 ARC COLORING 문제로 전환시킬 수 있으므로[9, 10, 11] NP-complete의 문제이다. 그러나 본 논문이 다루는 2차원 메쉬형태의 Myrinet 클러스터링 시스템은 메쉬를 형성하고 있는 전체 스위치들 어디에서나 통신의 송/수신이 가능한 것이 아니라 2차원 메쉬의 사방 경계에 위치하는 스위치들과 연결된 호스트들만이 통신의 송/수신에 참여할 수 있기 때문에 앞서 언급한 NP-complete 문제인 2차원 메쉬구조-모든 노드에서 송/수신이 가능-에서의 path coloring과는 다른 문제이다.



(a)



(b)

그림 8 경로 그래프

이 형태에서의 클러스터링 시스템에서 요청될 수 있는 통신의 유형들은 선형의 시스템에서보다 복잡하다. [그림 6](a)는 이러한 클러스터링 시스템 구조에서 요청될 수 있는 통신들의 예를 보여주고 있다. 통신 요청 집합 $R=\{d_1, d_2, d_3, d_4, u_1, u_2, u_3, u_4\}$ 이고, R에 속하는 각 통신들을 노드로 하고 각 통신 사이에 링크 충돌이 발생하면 노드사이에 edge로 연결하여 표현한 그래프를 경로 그래프라 하면, [그림 6](a)의 예는 [그림 6](b)와 같이 두 개의 분리된(disjoint) 집합 D와 U로 구분되어진다. 여기서 D는 아래로 향하는 통신 경로를 가지는 통신들만을 포함하고 있으며 U는 위로 향하는 통신 경로를 가지는 통신들만을 포함하고 있음을 알 수 있다. 전체 네트워크에 존재하는 어떠한 통신 요청 집합 R에 대해서도 이와 같이 두 개의 분리된 집합 D와 U로 분리할 수 있는데, 그 이유는 Myrinet 링크가 양방향 전송 방식을 지원하고, 순위차원 라우팅을 사용하기 때문이다. 이 장에서 제안한 *AssignColor_Mesh* 알고리즘은 [그림 6](b)의 D 통신에 대한 최적 스케줄링을 생성한다. [그림 6](b)의 U에 포함된 통신들에 대한 스케줄링은 U의 통신들과 D의 통신들 사이에 링크 충돌이 전혀 발생하지 않으므로 [그림 6](b)의 D 통신들과 분리하여 180도 회전시킨 후 *AssignColor_Mesh* 알고리즘을 적용하면 네트워크에서 요구되는 전체 통신 집합에 대해 정확하게 coloring을 할 수 있다. 특정 시간에 통신 요청 집합이 주어졌을 때 알고리즘 *AssignColor_Mesh*는 최대 $\frac{3}{2}L_{max}$개의 색만을 사용하여 2차원 메쉬구조의 클러스터링 시스템에 요청된 모든 통신에 적당한 색을 할당할 수 있다. 필요한 색들의 최대 개수가 $\frac{3}{2}L_{max}$를 초과하지 않는다는 것은 다음 절에서 증명하겠다.

표 2 2차원 메쉬구조의 네트워크에서 각 스위치에 존재할 수 있는 통신 형태

$D_{i,j}$	통신경로가 스위치 (i, j)에서 시작해서 아래방향으로 향하는 통신으로 메쉬의 마지막 행에서는 존재할 수 없다.
$DT_{i,j}$	스위치(i, j)의 위로부터 들어와서 아래방향으로 향하는 통신. 메쉬의 첫 번째 행에서는 존재할 수 없다.
$DE_{i,j}$	스위치 (i, j)의 위로부터 들어와서 더 이상 다른 스위치로의 통신경로를 가지지 않는 통신.
$L_{i,j}$	스위치 (i, j)에서 시작해서 왼쪽 방향으로 진행하는 통신.
$LT_{i,j}$	스위치 (i, j)의 오른쪽에서 들어와서 스위치를 통과해 왼쪽으로 나가는 통신.
$LD_{i,j}$	스위치 (i, j)의 오른쪽에서 들어와서 스위치의 아래방향으로 향하는 통신.
$LE_{i,j}$	스위치 (i, j)의 오른쪽으로부터 들어와서 더 이상 다른 스위치로의 경로를 가지지 않는 통신.
$R_{i,j}$	통신경로가 스위치 (i, j)에서 시작되어 오른쪽 방향으로 진행하는 통신.
$RT_{i,j}$	스위치 (i, j)의 왼쪽에서 들어와서 스위치를 통과해 오른쪽으로 나가는 통신.
$RD_{i,j}$	스위치 (i, j)의 왼쪽에서 들어와서 아래방향으로 향하는 통신.
$RE_{i,j}$	스위치 (i, j)의 왼쪽에서 들어와서 더 이상 다른 스위치로의 통신 경로를 갖지 않는 통신

선형의 클러스터링 시스템에서와 마찬가지로 이 형태의 클러스터링 시스템에서도 스위치를 기준으로 각 스위치를 통과하는 통신에 대해 적당한 색을 선택하여 할당하는 방법을 사용한다. 따라서 각각의 Myrinet 스위치들에서 존재할 수 있는 통신 경로를 살펴보면 표2에서 설명하고 있는 것과 같이 몇가지 유형으로 나타내 볼 수 있으며, 이중 $R_{i,j}$, $L_{i,j}$, $RE_{i,j}$, $LE_{i,j}$, $RT_{i,j}$, $LT_{i,j}$ 는 선형에서와 같고 스위치를 기준으로 보아 아래방향으로 향하는 통신의 명칭에는 D를 사용하였다. *AssignColor_Linear* 알고리즘에서와 마찬가지로 coloring의 수행 시 사용할 수 있는 색들은 $Color_k(k=integer)$라 하고, 특정 통신 유형에 속하는 개별 통신 각각에 대한 색의 집합을 $C(통신유형)$ 이라 하겠으며, 색을 선택하는 방법으로는 갈망(greedy)법을 사용한다.

이제 알고리즘 *AssignColor_Mesh*에 의해 [그림 6] (b)의 D 집합과 같은 종류의 통신들에 대한 coloring 방법을 살펴보자. 이 시스템에서 호스트들은 모두 2차원 메쉬의 가장자리에 있는 스위치에 연결되어 있기 때문에 통신은 항상 가장 바깥쪽에 있는 스위치들을 거치게 되므로 이 스위치들을 세 부분으로 나누고 순서에 따라 각 부분을 거치는 통신에 대하여 coloring을 수행하도록 한다.

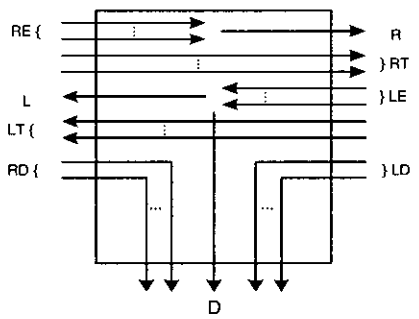


그림 7 1행 스위치에서의 통신 형태

먼저, 첫 번째 행을 거치는 통신에 대하여 coloring을 수행한다. 첫 번째 행에 위치하는 스위치들을 지나는 통신경로의 종류는 [그림 7]과 같으며 왼쪽 스위치 $(1, 1)$을 지나는 통신에서부터 시작해서 오른쪽 마지막 $(1, m)$에 위치한 스위치를 지나는 통신까지 적당한 색을 할당하도록 한다. 첫 번째 **for loop**이 이를 수행하고 있는데, 먼저 j가 1로 스위치 $(1, 1)$일 때의 coloring 방법을 본다. 스위치 $(1, 1)$에서는 위치의 특성상 $R_{1,1}$, $LE_{1,1}$, $D_{1,1}$, $LD_{1,1}$종류의 통신만이 존재할 수 있으며, 만약 이들 통신이 존재한다면 이 4 종류의 통신에 다음과 같은 방법으로 색을 할당한다. $R_{1,1}$ 에는 $Color_k$중 가장 작은 값을 가지는 $Color_1$(아직 사용한 $Color_k$ 가 전혀 없으므로)을 할당하고, $LE_{1,1}$에는 $Color_1$부터 시작하여 $Color_k$ 중 가장 작은 첨자 값을 가지는 색들을 $LE_{1,1}$에 속한 통신의 개수만큼 선택하여 할당하며, $D_{1,1}$에는 $R_{1,1}$과 마찬가지로 $Color_1$을 할당한다. 이때 각 호스트들은 단지 하나의 목적지 호스트만을 가질 수 있으므로 $D_{1,1}$와 $R_{1,1}$의 두 통신은 동시에 존재할 수 없으며, 각각 $D_{1,1}$와 $R_{1,1}$ 에 속하는 통신의 개수도 만약 존재한다면 하나뿐이다. $LD_{1,1}$에는 $D_{1,1}$에서 사용하는 $Color_1$과 $LE_{1,1}$ 에서 사용하는 색(즉, $C(LE_{1,1})$)을 받을 경우 충돌이 발생하게 되므로 이를 제외한 $Color_k$ 중 가장 작은 첨자 값의 색들을 선택하여 $LD_{1,1}$의 개수만큼 지정해준다. 1행 스위치를 오른쪽 방향으로 진행하면서 coloring을 수행하므로 스위치 $(1, 1)$을 통과하는 통신들에 대한 coloring이 끝나면, 다음으로 스위치 $(1, 1)$의 오른쪽 스위치, 즉 $j=2$로 스위치 $(1, 2)$에서의 coloring을 수행한다. $(1, 2)$를 지나면서 색을 할당받지 못한 통신에는 $R_{1,2}$, $LE_{1,2}$, $D_{1,2}$, $LD_{1,2}$, $LT_{1,2}$, $L_{1,2}$, $RD_{1,2}$, $RT_{1,2}$, $RE_{1,2}$의 통신의 종류가 존재할 수 있다. 그런데 이러한 9개의 통신의 형태를 살펴보면, 이들 중 $RD_{1,2}$와 $RT_{1,2}$, $RE_{1,2}$는 $R_{1,1}$에 의해 그리고 $LT_{1,2}$와 $LE_{1,2}$는 $L_{1,1}$에 의해 이미 색을 할당받은 상태이다. 따라서 현재 스위치 $(1, 2)$에서 색을 지정해 주어야 하는 통신은 $R_{1,2}$, $LE_{1,2}$, $D_{1,2}$, $LD_{1,2}$ 로 스위치 $(1, 1)$에서 색을 할당해 주어야 하는 통신의 종류와 같음을 알 수 있다.

만약 $R_{1,2}$, $LE_{1,2}$, $D_{1,2}$, $LD_{1,2}$의 통신이 존재한다면, 색의 할당 방법은 스위치 $(1, 1)$에서와 동일하지만 링크를 공유하는 다른 통신 때문에 사용할 수 없는 색이 존재할 수 있으므로 이를 고려하여 각 통신에 대하여 다음과 같이 coloring을 수행한다. $R_{1,2}$ 에는 $RT_{1,2}$가 받은 색을 고려하여 링크 충돌이 발생하지 않도록 $Color_k$중 가장 작은 첨자값의 색을 할당하고, $LE_{1,2}$에는 $LT_{1,2}$에서 이미 사용되지 않은 $Color_k$ 중 가장 작은 첨자 값의 색을 할당하며, $D_{1,2}$에는 같은 링크를 공유하고자 하는 $RD_{1,2}$가 존재한다면 전 단계에서 이미

알고리즘 *AssignColor_Mesh*

입 력 : 통신 요청 집합, R
출 력 : 각 통신에 색을 할당한 결과
/*첫 번째 행을 지나는 통신들에 대한 coloring*/
for $j=1$ to m {
$C(R_{1,j})$:= $Color_{\min\{k \mid k=integer,\ Color_k \notin C(RT_{i,j})\}}$
/* $RT_{i,j}$에 사용되지 않은 $Color_k$중에서 가장 작은 첨자값을 가지는 색을 선택하여 $R_{1,j}$ 에 할당한다. */
$C(LE_{1,j})$:= $Color_{\min\{k \mid k=integer,\ Color_k \notin C(LT_{1,j})\}}$
/* $LT_{1,j}$에 사용되지 않은 $Color_k$중에서 가장 작은 첨자값을 가지는 색을 선택하여 $LE_{1,j}$에 할당한다. */
$C(D_{1,j})$:= $Color_{\min\{k \mid k=integer,\ Color_k \notin C(RD_{1,j})\}}$
/* $RD_{1,j}$에 사용되지 않은 $Color_k$ 중에서 가장 작은 첨자값을 가지는 색을 선택하여 $D_{1,j}$에 할당한다.*/
$C(LD_{1,j})$:= $Color_{\min\{k \mid k=integer,\ Color_k \notin C(L_{1,j})\cup C(D_{1,j})\cup C(LT_{1,j})\}}$
/* $L_{1,j}$, $D_{1,j}$, $LT_{1,j}$ 에 사용되지 않은 $Color_k$ 중에서 가장 작은 첨자값의 색을 선택하여 $LD_{1,j}$에 할당한다.*/

/*마지막 행에 아래방향으로 도착하는 통신들에 대한 coloring*/
for $j=2$ to $m-1$
if $|C(DE_{n,j})|<|DE_{n,j}|$ then /*coloring이 되지 않은 $DE_{n,j}$가 존재하면*/
$C(DE_{n,j})$:= $Color_{\min\{k \mid k=integer,\ Color_k \notin C(D_{1,j})\cup C(RD_{1,j})\cup C(LD_{1,j})\}}$
/* $D_{1,j}$, $RD_{1,j}$, $LD_{1,j}$에 사용되지 않은 $Color_k$ 중에서 가장 작은 첨자 값을 가지는 색을 선택하여 할당한다. */

/*첫 번째와 마지막 m번째 열을 지나는 통신들에 대한 coloring*/

for $i=2$ to $n-1$ {
$C(D_{i,1})$:= $Color_{\min\{k \mid k=integer,\ Color_k \notin C(DT_{i,1})\}}$
/* $D_{i,1}$에는 $Color_k$ 중 $DT_{i,1}$에 사용되지 않은 가장 작은 첨자값을 가지는 색을 할당한다. */
$C(LE_{i,1})$:= $Color_1$ /* $LE_{i,1}$에는 $Color_1$을 할당한다.*/
$C(LD_{i,1})$:= $Color_{\min\{k \mid k=integer,\ Color_k \notin C(DT_{i,1})\cup C(D_{i,1})\}}$
/* $LD_{i,1}$에는 $Color_k$중 $DT_{i,1}$ 와 $D_{i,1}$에서 사용되지 않은 가장 작은 첨자 값을 가지 는 색을 선택하여 할당한다.*/
$C(D_{i,m})$:= $Color_{\min\{k \mid k=integer,\ Color_k \notin C(DT_{i,m})\}}$
/* $D_{i,m}$에는 $Color_k$ 중 $DT_{i,m}$에서 사용되지 않은 가장 작은 첨자값을 가지는 색을 할당한다.*/
$C(RE_{i,m})$:= $Color_1$ /* $RE_{i,m}$에는 $Color_1$을 할당한다.*/
$C(RD_{i,m})$:= $Color_{\min\{k \mid k=integer,\ Color_k \notin C(DT_{i,m})\cup C(D_{i,m})\}}$
/* $RD_{i,m}$에는 $Color_k$ 중 $DT_{i,m}$과 $D_{i,m}$에서 사용되지 않은 색 중 가장 작은 첨자 값을 가지는 색을 선택하여 할당한다.*/
}

$R_{1,1}$에 의해 coloring 되어있는 상태이므로 $RD_{1,2}$에 사용되지 않은 색 중 가장 작은 첨자 값을 갖는 색을 할당한다. $LD_{1,2}$는 아래로 내려가는 방향의 링크를 $D_{1,2}$, $RD_{1,2}$와 함께 공유하게 되므로 $D_{1,2}$, $RD_{1,2}$에서 사용되지 않은 $Color_k$중 가장 작은 첨자 값의 색을 할당하도록 한다.

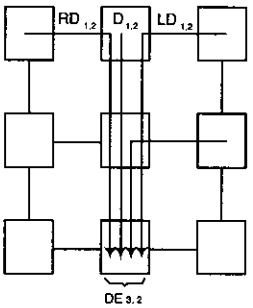


그림 8 n행 스위치에서의 통신 형태

스위치 (1,1), (1,2)를 통과하는 통신들에 대한 coloring에서도 보았듯이 각 스위치에서 색을 할당해 주어야 하는 통신들은 $R_{1,j}$, $LE_{1,j}$, $D_{1,j}$, $LD_{1,j}$ 이다. 앞서 설명한 방법으로 j값을 증가시켜 1행의 스위치를 왼쪽에서 오른쪽으로 진행해 가면서 링크 충돌이 일어나지 않도록 다른 통신들이 이미 받은 색을 고려하여 현재 스위치에서 coloring되지 않은 4종류의 통신에 적당한 색을 할당한다.

메쉬의 첫 번째 행에 대한 coloring이 끝나면 다음으로 마지막 n번째 행에 대한 coloring을 수행한다. [그림 6](b)의 D집합에 속하는 통신 중 n번째 행에 있는 스위치를 지나는 통신은 [그림 8] 에서 볼 수 있듯이 위치의 특성상 위로부터 들어와 더 이상 다른 스위치로의 경로를 가지지 않는 $DE_{n,j}$의 통신만 존재한다. 이중 어떤 통신은 이전 for loop, 즉 첫 번째 행의 coloring을 수행할 때 아래방향으로 내려오는 통신인 $D_{1,j}$, $RD_{1,j}$, $LD_{1,j}$에 의해 이미 coloring이 되어 있는 것이 있을 것이다. 따라서 n번째 행의 스위치에 이르는 통신에 대해 coloring을 할 때에는 아직 색을 할당받지 못한 $DE_{n,j}$에 대해 $D_{1,j}$, $RD_{1,j}$, $LD_{1,j}$에서 사용되지 않은 $Color_k$중 가장 작은 첨자 값을 가지는 색을 선택하여 할당하도록 한다. 두 번째 **for loop**이 이를 수행하고 있다.

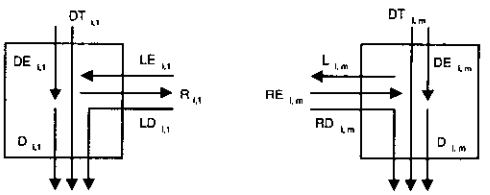


그림 9 1열과 m열 스위치에서의 통신 형태

마지막으로, 메쉬의 양쪽 열에 위치한 스위치를 지나는 통신에 대한 coloring 을 수행한다.

소스 호스트는 목적지 호스트를 하나만 가질 수 있기 때문에 [그림 9] 에서 보는 것처럼 $D_{i,1}$과 $R_{i,1}$, $D_{i,m}$과 $L_{i,m}$의 각 두 경우들은 동시에 발생할 수 없다. 또한 $L_{i,m}$은 $LE_{i,1}$또는 $LD_{i,1}$ 로, $R_{i,1}$은 $RE_{i,m}$ 또는 $RD_{i,m}$ 의 통신유형으로 나타나기 때문에 $LE_{i,1}$과 $LD_{i,1}$, $RE_{i,m}$과 $RD_{i,m}$도 각각 두 개의 통신 중 하나의 통신만이 존재 할 수 있다. 즉, 스위치 (i, 1)에서는 $DE_{i,1}$, ($D_{i,1}$ 또는 $R_{i,1}$), ($LE_{i,1}$ 또는 $LD_{i,1}$), $DT_{i,1}$ 통신들이 존재할 수 있으며, 스위치 (i, m)에서는 $DE_{i,m}$, ($D_{i,m}$ 또는 $L_{i,m}$), ($RD_{i,m}$ 또는 $RE_{i,m}$), $DT_{i,m}$이 존재할 수 있다. 이중 $DE_{i,1}$, $DT_{i,1}$, $DE_{i,m}$, $DT_{i,m}$은 이미 색이 할당된 상태이므로 이들 통신을 제외한 통신들에 대해 다음과 같은 방법으로 색을 할당한다. 세 번째 **for loop** 에서처럼 각각 $D_{i,1}$에는 $Color_k$ 중 $DT_{i,1}$ 에 사용되지 않은 가장 작은 첨자 값의 색을 할당하고, $LE_{i,1}$에는 $Color_1$을, $LD_{i,1}$에는 $Color_k$ 중 $DT_{i,1}$과 $D_{i,1}$에서 사용되지 않은 가장 작은 첨자의 색을 선택하여 할당한다. $D_{i,m}$에는 $Color_k$ 중 $DT_{i,m}$에서 사용되지 않은 가장 작은 첨자 값의 색을 할당하고, $RE_{i,m}$ 에는 $Color_1$을, $RD_{i,m}$ 에는 $Color_k$ 중 $DT_{i,m}$ 과 $D_{i,m}$에서 사용되지 않은 가장 작은 첨자값의 색을 할당한다. 알고리즘을 수행하면서 스위치 (n, 1) 과 (n, m)을 지나는 통신에 대한 coloring을 제외한 이유는 ($n-1$, 1), ($n-1$, m)까지의 coloring이 이들을 모두 포함할 수 있기 때문이다.

이제 *AssignColor_Mesh* 에 의해 스케줄링 한 결과를 통해 순위차원 라우팅을 사용하는 통신들의 메시지 전송 완료를 보장할 수 있는 최고한도(upper bound) 시간을 알아본다.

정리 4. Myrinet 스위치들을 2차원 메쉬 형태로 연결한 클러스터링 시스템에서 순위차원 라우팅을 사용하는

통신들의 메시지 전송은 최대 $\frac{3}{2}L_{max} \cdot T$ 시간 내에 완료할 수 있다.

증명 알고리즘 *AssignColor_Mesh*에서 coloring은 세 단계를 통해 색들을 지정하게 된다. 1행과 n행, 그리고 1열, m열의 스위치를 통과하는 통신들에 대해 수행하고 있는데, 1행 스위치를 지나는 통신들을 coloring 할 때에는 현재 스위치에 존재하는 여러 통신 중 아직 색을 할당받지 못한 $R_{i,j}$, $LE_{i,j}$, $D_{i,j}$, $LD_{i,j}$에 적당한 색을 할당하였고, n행과 1열, m열 스위치들을 통과하는 통신들에 대한 coloring도 역시 스위치에서 아직 색을 할당받지 못한 통신들에 링크 충돌이 없도록 하여 색을 지정하였다. 그런데 이중 n행과 1열, m열 스위치들을 통과하는 통신들의 coloring은 선형 네트워크에서와 동일한 경우이므로, 최대 L_{max}개의 색이면 양쪽 열과 마지막 행에 위치한 스위치들을 지나는 통신들에 대한 coloring을 수행할 수 있다. 그런데 1행 스위치를 지나는 통신들에 대한 coloring의 경우, [그림 8]에서 보는 것처럼 1행 스위치를 통과하는 통신 중 $LD_{i,j}$는 $RD_{i,j}$, $LE_{i,j}$, $LT_{i,j}$, $D_{i,j}$ 에서 사용된 색을 받을 수 없고,

$$|\ (RD_{i,j}\text{를 위해 사용된 색}) \cup (LE_{i,j}\text{와 } LT_{i,j}\text{를 위해 사용된 색}) \cup (D_{i,j}\text{를 위해 사용된 색})\ | = C \quad (1)$$

라 하면 L_{max}의 정의에 의하여,

$$C \le L_{max} \quad (2)$$

이다.

따라서, 만약

$$C + |LD_{i,j}| \ge L_{max} \quad (3)$$

이라면 , $LD_{i,j}$에는

$$[\ |LD_{i,j}| - (L_{max} - C)\ ] \quad (4)$$

식(4)의 개수만큼 새로운 색을 더 사용해야만 한다.

(3)의 경우가 되어 $LD_{i,j}$에 L_{max}이외의 새로운 색을 식(4)의 개수만큼 더 할당해야 할 경우, 필요한 색의 최대값을 구해보자. 필요한 색 식(4)의 최대값은 (2)에서 $C = L_{max}$인 경우이며, 따라서 $L_{max} - C = 0$이 되어 (4)의 최대값을 구하는 것은 $|LD_{i,j}|$의 최대값을 구하는 것과 같게 된다.

$$|RD_{i,j}| + |LD_{i,j}| + |D_{i,j}| \le L_{max} \quad (5)$$

$$|LE_{i,j}| + |LT_{i,j}| + |LD_{i,j}| \le L_{max} \quad (6)$$

$$|LE_{i,j}| + |LT_{i,j}| + |RD_{i,j}| + |D_{i,j}| \ge L_{max}\ (\because C = L_{max}) \quad (7)$$

그런데 $|D_{i,j}| = 0$ 또는 $|D_{i,j}| = 1$이므로 최대값을 구하기 위해 $|D_{i,j}| = 0$으로 한다.

그러면, (7)으로부터

$$|LE_{i,j}| + |LT_{i,j}| \ge L_{max} - |RD_{i,j}| \quad (8)$$

(6)으로부터

$$|LD_{i,j}| \le L_{max} - (|LE_{i,j}| + |LT_{i,j}|) \quad (9)$$

(8), (9) 로부터

$$|LD_{i,j}| \le |RD_{i,j}| \quad (10)$$

그리고 (5)에 의해

$$|LD_{i,j}| \le L_{max} - |RD_{i,j}| \quad (11)$$

이므로 (10, (11)에 의해 $|LD_{i,j}| \le \frac{1}{2}L_{max}$로 $|LD_{i,j}|$의 최대값은 $\frac{1}{2}L_{max}$이다. 이는 즉, L_{max}개의 색과 최대 $\frac{1}{2}L_{max}$개의 색을 추가하여 사용하면 2차원 메쉬구조의 클러스터링 시스템에서 1행 스위치를 통과하는 통신들에 대해 모두 coloring을 수행 할 수 있음을 나타낸다. 그러므로 최대 $\frac{3}{2}L_{max}$개의 색이면 메쉬 형태의 Myrinet 클러스터링 시스템에서 특정한 시간에 주어지는 통신 요청 집합 R에 대하여 적당한 색을 할당할 수 있다. 이로써 정리 4를 증명하였다. ∎

5. 결론 및 향후과제

본 논문에서는 Myrinet으로 구성된 클러스터링 시스템에서 네트워크의 형태가 선형으로 이루어진 경우와 2차원 메쉬구조로 이루어진 경우, 특정시간에 시스템에 주어지는 통신요청들에 대하여 순위 차원 라우팅을 사용하여 모든 메시지의 전송을 최단시간에 완료할 수 있도록 하는 통신 스케줄링의 최적 알고리즘을 제안하였다.

통신 스케줄링의 최적 알고리즘을 찾는 이 문제는 그래프 이론의 path coloring 문제로, 링크를 공유하는 통신경로들에 대해서는 다른 색을 지정해야만 하고, coloring의 결과 서로 다른 색이 지정된 통신들은 각각 다른 시간에 메시지의 전송이 이루어지게 된다.

시스템에서 주어진 통신 요청 집합에 대하여 같은 방향으로 동시에 링크를 공유하는 통신들의 최대 개수를 L_{max}, 하나의 메시지가 전달되는데 걸리는 시간을 T라고 정의하면, 본 논문에서 제안한 *AssignColor_Linear*, *AssignColor_Mesh* 알고리즘을 적용한 결과로 얻은 통신 스케줄링에 의해 통신 요청 집합의 모든 통신을 완료하는데 소요되는 시간은 선형 네트워크인 경우 $L_{max} \cdot T$, 2차원 메쉬구조의 네트워크인 경우에는 $\frac{3}{2}L_{max} \cdot T$의 시간을 초과하지 않았다. 또한 앞서 제시한 두 가지 시스템에서의 통신 시간은 본 논문에서 제안한 알고리즘에서 보장하고 있는 통신을 위해 소요되

는 시간의 최고 한도로써 통신을 완료하기 위한 최적의 시간이며 이에 대한 증명은 정리 3, 4에 잘 나타나 있다.

본 논문에서는 4포트의 Myrinet 스위치를 사용하여 2차원 메쉬구조를 이루는 클러스터링 시스템에서의 통신 스케줄링 문제를 다루었다. 8포트의 Myrinet 스위치를 사용하면 3차원 메쉬구조의 클러스터링 시스템을 구축할 수 있으므로 이러한 형태의 모델 상에서의 통신 스케줄링 방법도 생각해 볼 수 있으며, Myrinet으로 구성할 수 있는 클러스터링 시스템에는 특별한 제한이 없으므로 다른 시스템 모델 상에서의 통신 스케줄링 문제도 고려해 볼 수 있다. 또한 본 논문에서는 전송자인 호스트 컴퓨터가 하나의 목적지 호스트 컴퓨터에만 메시지를 전송할 수 있는 유니캐스트 방식을 가정하였는데, 이에 비해 동시에 여러 개의 목적지를 가지는 멀티캐스트의 경우 어떠한 방법으로 최적의 통신 스케줄링을 수행할 수 있는지의 문제도 생각해 볼 수 있을 것이다.

참 고 문 헌

[1] Mario Gerla, Prasasth Palnati, Simon Walton, "Multicasting Protocols for High-Speed, Wormhole-Routing Local Area Networks," http://www.cs.ucla.edu/~simonw/sigcom/sigcom.html.

[2] Myricom Inc., Myricom Homepage, "http://www.myri.com/myrinet/," April 1998.

[3] Nanette J. Boden, Danny Cohen, Robert E. Felderman, Alan E. Kulawik, Charles L. Seitz, Jakov N. Seizovic, and Wen-King Su "Myrinet --A Gigabit-per-Second Local-Area Network," IEEE-Micro, Vol. 15, No.1, pp. 29-36, February 1995.

[4] Sharad Sundaresan, Riccardo Dettati, "Distributed Adimission Control for Wormhole-Routed Networks," Technical Report 96-015, Texas A&M University, July 1996.

[5] Sharad Sundaresan, Riccardo Bettati, "Distributed Connection Management for Real-Time Communication over Wormhole-Routed Networks," Technical Report 96-021, Texas A&M University, July 1996.

[6] Lionel M. Ni, Philip K. Mckinley, "A Survey of Routing Techniques in Wormhole Networks," Technical Report MSU-CPS-ACS-46, October 17, 1991.

[7] Douglas B. West, "Introduction of graph theory," Prentice Hall, 1996.

[8] Martin C. Carlisle, Errol L. Lloyd, "On k-coloring a set of intervals," Discrete Applied Mathematics 59, pp. 225-235, 1995.

[9] Alan Tucker, "Coloring a family of circular arcs," SIAM J. APPL. MATH, Vol 29., No 3., November 1975.

[10] Klaus Jansen, "NP-completeness of path coloring in meshes with fixed row-column routing," preprinted paper.

[11] M. R. Garey, D. S. Johnson, G. L. Miller, and C. H. Papadimitriou, "The complexity of coloring circular arcs and chords," SIAM J. Algebraic Discrete Methods, 1(2), pp. 216-227, 1980.

박 상 명

1997년 숙명여자대학교 전산학과 졸업. 1999년 숙명여자대학교 대학원 전산학과 졸업. 1999년 ~ 현재 숙명여자대학교 자연과학연구소 연구원. 관심분야는 컴퓨터구조, 병렬처리, 웹어플리케이션

이 상 규

1989년 University of Southern Califonia Dept. of Computer Science (공학사). 1991년 George Washington University Dept. of Electrical Engineering and Computer Science(공학석사). 1995년 George Washington University Dept. of Electrical Engineering and Computer Science(공학박사). 1995년 ~ 1996년 George Washington University Dept. of Electrical Engineering and Computer Science 박사후 과정. 1997년 ~ 현재 숙명여자대학교 전산학과 교수. 관심분야는 병렬/분산 처리 시스템, 컴퓨터이론, 컴퓨터 통신, 인터넷 프로그래밍.

문 봉 희

1981년 서울대학교 자연과학대학 계산통계학과(이학사). 1983년 서울대학교 대학원 계산통계학과(이학석사). 1992년 서울대학교 대학원 전산과학과(이학박사). 1984년 울산대학교 전자계산학과 전임강사. 1985년 ~ 현재 숙명여자대학교 전산학과 교수. 관심분야는 컴퓨터구조, 웹프로그래밍