

# 분산 공유메모리 환경에 효율적인 동적 라우팅 알고리즘

홍강운<sup>U</sup>, 전창호

한국전자통신연구원<sup>U</sup>, 한양대학교 전자컴퓨터공학부  
gwhong@etri.re.kr<sup>U</sup>, chjeon@cse.hanyang.ac.kr

## An Efficient Dynamic Routing Algorithm for a Distributed Shared Memory Environment

Kang-Woon Hong<sup>U</sup>, Chang-Ho Jeon

Electronics and Telecommunication Research Institute<sup>O</sup>,  
School of Electrical Engineering and Computer Science, Hanyang University

### 요 약

다단계 버스 네트워크 기반의 분산공유메모리 환경에서 효율적인 동적 라우팅 알고리즘을 제안한다. 버디특성으로부터 생기는 잉여경로들을 활용하여 네트워크 상의 스위치들의 트래픽을 동적으로 분산시키고, 동적 라우팅이 선택될 확률을 높이기 위해 기존의 최적 경로결정 알고리즘을 개선한 것이다. 시뮬레이션을 통해 제안된 동적 라우팅 알고리즘의 성능과 다단계 버스 네트워크에서의 잉여경로들을 고려하지 않는 기존의 라우팅 알고리즘들의 성능을 비교하여 우리가 제안한 동적 라우팅 알고리즘이 기존의 라우팅 알고리즘들보다 평균 대기 작업수와 평균 응답시간 면에서 효율적으로 동작한다는 것을 보인다.

### 1. 서론

본 논문은 분산 공유메모리 환경의 다중 프로세서 시스템에서 스위치들의 트래픽을 고려한 동적 라우팅 알고리즘을 제안한다. 기존의 연구에서는 거쳐야 할 스위치의 수가 최소가 되는 라우팅 방법을 사용하고 참조 요청을 하는 프로세서와 대상 메모리 모듈로 이루어진 쌍이 같다면 그 참조요청들은 항상 동일한 경로를 갖는다 [1]. 그러나 U자형 선회 라우팅 방법에서 버디특성[2]을 이용하면 참조요청을 하는 프로세서와 대상 메모리 모듈의 쌍이 같고 동일한 라우팅 방법이 선택되더라도 데이터를 다른 경로로 라우팅시킬 수 있는 다단계 연결 네트워크의 특성을 고려하지 않고 있다.

이에 본 논문에서는 스위치들의 트래픽에 따라 동적으로 데이터 전송경로를 설정하고 트래픽 정도가 높은 스위치를 피해서 전송시킴으로써 스위치들의 트래픽을 분산시키는 동적 라우팅 알고리즘을 제안한다.

본 논문의 2장에서는 기존의 라우팅 알고리즘과 최적 경로 결정 알고리즘의 문제점을 지적한다. 3장에서는 동

적 라우팅 알고리즘을 제안하고 수정된 최적 경로결정 알고리즘을 제시한다. 4장에서는 성능을 측정하기 위한 시뮬레이션 결과를 분석하고 5장에서 결론을 맺는다.

### 2. 문제점

애플리케이션이나 알고리즘에 따라 차이는 있지만 다중 프로세서 시스템의 모든 프로세서들이 항상 메모리 참조 요청을 발생시키지는 않는다. 그래서 스위치들 간에 메모리 참조 요청이나 데이터로 인해 발생하는 스위치 트래픽 면에서 차이가 생기게 되고 응답시간에도 차이가 생긴다. 또한, 버디 특성을 만족하는 다단계 버스 네트워크들은 U자형 선회 라우팅시 연속된 단계들 사이에는 항상 두 개 이상의 경로를 갖게 된다. 그러나 기존의 라우팅 알고리즘은 동적으로 경로를 변경할 수 없어 스위치의 트래픽을 분산시킬 수 있는 가능성을 배제하고 있다. 또한, 기존에 제안된 최적 경로 결정 알고리즘에서는 포워드/백워드 라우팅과 포워드/백워드 U자형 선회 라우팅시 거치는 스위치들의 수가 동일한 경우 포

워드/백워드 라우팅을 선택한다[3,4]. 하지만, 동적 라우팅 알고리즘이 최대한 높은 확률로 선택되게 하기 위해서는 거쳐야 하는 스위치들의 수가 동일한 경우 선회 라우팅이 선택되도록 최적경로 결정 알고리즘을 수정해야 한다.

**3. 동적 라우팅 알고리즘과 수정된 경로결정 알고리즘**  
본 논문에서 제안하려는 동적 라우팅 방법은 기존의 포워드/백워드 U자형 선회 라우팅 방법에서 스위치의 트래픽을 고려하도록 한다. 이 라우팅 방법에서는 현 단계의 스위치와 연결된 다음 단계의 두 스위치 중 트래픽이 가벼운 스위치를 선택하여 이동한다. 이 선택과 이동은 선회 단계(FTS/BTS)까지 반복되고 그 단계에서 U자형 선회를 하여 백워드/포워드 라우팅을 한다(표1).

```

- from Source to FTS
if  $S_{l-1}=0$ 
    if Traffic(upper SW)  $\geq$  Traffic(lower SW)
        routing packet into  $S_1 S_2 \dots S_{l-1} S_0$ 
    else
        routing packet into  $S_1 S_2 \dots S_{l-1} S_0$ 
else ( $S_{l-1}=1$ )
    if Traffic(upper SW)  $\geq$  Traffic(lower SW)
        routing packet into  $S_1 S_2 \dots S_{l-1} S_0$ 
    else
        routing packet into  $S_1 S_2 \dots S_{l-1} S_0$ 
- FTS
if  $S_{l-1}=0$  then  $S_{l-1}=1$ 
else ( $S_{l-1}=1$ )  $S_{l-1}=0$ 
- from FTS to Destination
using backward routing
    
```

표 1. 동적 포워드 U자형 선회 라우팅 알고리즘

```

 $d_l = \lfloor \frac{l}{2} \rfloor$ ,  $d_u = \lceil \frac{l}{2} \rceil$ 
IF (source = destination)
THEN 지역 메모리 모듈에 대한 참조요청
ELSE
FTS와 BTS 값을 찾는다.
IF ( FTS = ( l-1- BTS ) = 0 )
THEN 동적 포워드/백워드 U자형 선회 라우팅 선택
ELSE
IF ( FTS  $\leq$   $d_l$  )
THEN 동적 포워드 U자형 선회 라우팅 선택
ELSE
IF ( BTS  $\geq$   $d_u$  ) OR ( BTS =  $d_l$  )
THEN 동적 백워드 U자형 선회 라우팅 선택
ELSE
포워드/백워드 라우팅 선택
    
```

표 2. 수정된 최적경로 결정 알고리즘

최적 경로결정 알고리즘에서는 포워드/백워드 라우팅과 U자형 선회 라우팅이 동일한 개수의 스위치들을 통과하는 경우 포워드/백워드 라우팅을 선택한다. 하지만, 포워드/백워드 라우팅 방법에서는 스위치 트래픽을 고려하여 경로를 변경하지 않기 때문에 동적 라우팅 방법을 사

용했을 때의 효과를 얻을 수 없다. 또한, FTS/BTS에 도달한 후 U자형 선회 또는 계속해서 포워드/백워드 방향으로 라우팅을 할 것인지를 결정할 수도 있다. 하지만, FTS/BTS에 도달할 때까지 동적 라우팅을 한 후 지나온 경로가 포워드/백워드 라우팅시의 경로로 일치하려면, 두 라우팅의 경로가 일치할 확률은 선회할 단계가 t인 경우,  $1/2^t$ 가 되어 지수분포로 줄어, 본 논문에서는 항상 동적 라우팅 방법을 사용하였다(표2).

**4. 시뮬레이션**

분산 공유메모리 환경에서 동적 패킷 스위칭을 사용하는 다단계 버스 네트워크를 대상으로 시뮬레이션을 통하여 정적 라우팅(MBNwFB)과 정적 U자형 선회 라우팅(MBNwFBU), 그리고 본 논문에서 제안하는 동적 U자형 선회 라우팅(MBNwFBDU)의 성능을 비교한다. 비교하는 척도는 평균 응답시간, 스위치의 평균 대기 작업수와 메모리 모듈의 평균 대기 작업수이다. 스위치들의 트래픽 정보는 스위치들 사이에서 주기적으로 갱신되거나 데이터 패킷에 의해 전달되어 사전에 알고 있다는 것을 가정한다.

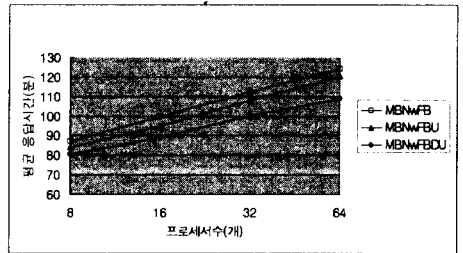


그림 1. 프로세서수 증가에 따른 평균 응답시간 비교

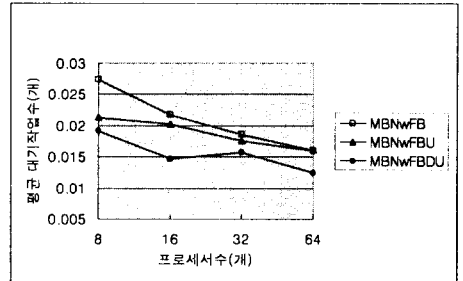


그림 2. 프로세서수 증가에 따른 스위치의 평균 대기 작업수 비교

그림1,2,3은 네트워크 상의 프로세서와 메모리 모듈 수를 증가시킨 시뮬레이션 결과이다. 그림1,2,3으로부터 프로세서 수가 증가할수록 동적 라우팅의 성능이 우수함을 알 수 있고, 특히, 홑수개의 단계를 갖는 경우 포워드 U자형 선회 라우팅과 백워드 U자형 선회 라우팅 방법

을 사용하는 참조 요청들이 정중앙 단계의 스위치들의 서비스를 받기 위해 경쟁함으로써 인접한 단계의 단계를 갖는 경우보다 상대적으로 좋지 않은 성능을 보인다.

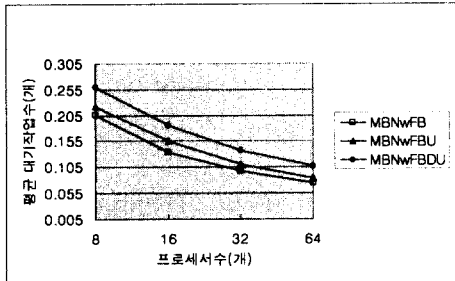


그림 3. 프로세서수 증가에 따른 메모리 모듈의 평균 대기 작업수 비교

그림 4, 5, 6에 보인 대로 스위치 크기를 변화시켜 본 시뮬레이션에서는 8x8 스위치 사용시 스위치 서비스 요구량이 버스의 처리용량을 과도하게 초과하여 스위치의 평균 대기 작업수가 큰폭으로 증가하고 이 때문에 4x4 스위치를 사용하는 경우의 성능이 더 좋게 나타난다.

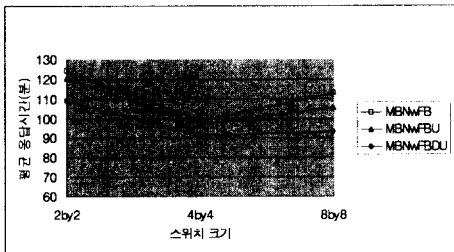


그림 4. 스위치 크기 증가에 따른 평균 응답시간 비교

### 5. 결론

본 논문에서는 버디특성으로 생기는 잉여 경로를 활용하고 라우팅 중에 스위치의 트래픽 정보를 바탕으로 경로를 선택, 이동하는 동적 라우팅 알고리즘을 제안하고 알고리즘 활용도를 높이기 위해 최적경로 결정 알고리즘을 수정하였다. 시뮬레이션을 통해 스위치들의 트래픽이 분산되어 평균 대기 작업수와 메모리 참조에 대한 평균 응답시간 면에서 제안된 알고리즘이 가장 좋은 성능을 나타냄을 알 수 있었다. 프로세서수를 증가시킨 시뮬레이션에서는 시스템이 홀수개의 단계들을 갖는 경우, 포워드 U자형 선회 라우팅과 백워드 U자형 선회 라우팅 방법을 사용하는 참조요청들이 정중앙 단계의 스위치들의 서비스를 받기 위해 경쟁함으로써 인접한 단계들을 갖는 경우보다 상대적으로 좋지 않은 성능을 보인다. 스위치 크기를 증가시킨 시뮬레이션 결과에서는 4x4 스위

치를 사용한 시스템의 성능이 좋았지만 프로세서 수에 대한 최적의 스위치 크기 결정을 위해서는 프로세서 수와 스위치 크기를 변수로 한 시뮬레이션을 더 할 필요가 있을 것이다.

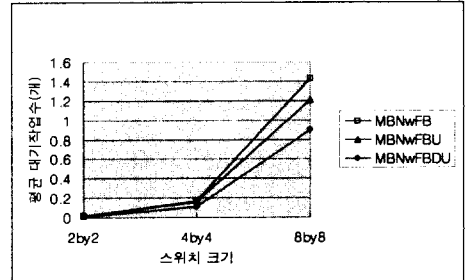


그림 5. 스위치 크기 증가에 따른 스위치의 평균 대기 작업수 비교

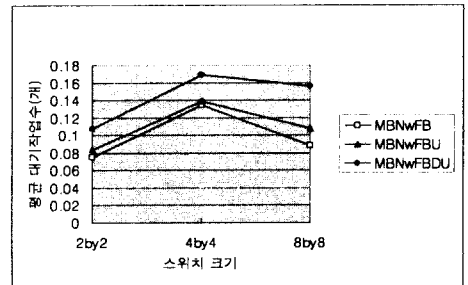


그림 6. 스위치 크기 증가에 따른 메모리 모듈의 평균 대기 작업수 비교

### 6. 참고 문헌

- [1] L. N. Bhuyan, R. R. Iyer, T. Askar, A. K. Nanda, and M. Kumar, "Performance of multistage bus networks for a distributed shared memory multiprocessor", IEEE trans. on parallel and distributed systems, vol. 8, no. 1, pp. 82-95, Jan. 1997
- [2] Dharma P. Agrawal, "Graph theoretical analysis and design of multistage interconnection networks", IEEE trans. on computers, vol. C-32, no. 7, pp.637-648, Jul. 1983.
- [3] L. N. Bhuyan, A. K. Nanda, and T. Askar, "Performance and reliability of the multistage bus networks", proceedings of the Int. conf. on parallel processing, pp. 26-33, Aug. 1994.
- [4] S. M. Mahmud, "Performance Analysis Of Multilevel Bus Networks For Hierarchical Multiprocessor", IEEE trans. computers, vol. 43, no. 7, Jul. 1994.