# 건너뜀 이중링크를 갖는 고확장성 CC-NUMA 시스템

## (A Highly Scalable CC-NUMA System with Skipped Dual Links)

서 효 중 †

(Hyo-Joong Suh)

**요 약** 다중 프로세서 시스템을 구성하기 위하여 점유가 발생하는 버스를 채용한 이래로, 상호연결망의 병목현상을 개선하기 위한 노력은 점대점 연결을 이용한 링 구조까지 발전되어 왔다. 상호연결망의 병목 현상은 다중 프로세서 시스템이 프로세서 수에 따른 선형적 성능 개선을 나타내지 못하게 하는 주요 제한 요소로 작용하였으며, 이러한 병목 현상을 개선하기 위한 상호연결망 구성 방법이 다수 연구되어 왔다. 본 논문은 현재 활발히 채용되고 있는 두 개의 점대점 연결을 이용한 링 구조에서 일정 규칙에 기반한 건너뜀 연결을 갖도록 개선하고 트랜잭션의 전송 경로를 정적 알고리즘으로 구현하여, 이중 링크를 가지는 CC-NUMA 시스템의 부하를 균일하게 분배시킴으로써 시스템의 성능과 확장성을 개선시켰다. 실험 결과에 의하면 단일 링크를 이용한 링 구조의 경우, 프로그램의 부하가 커질수록 프로세서의 추가에 따른 특정 링크의 병목 현상이 발생하였고, 시스템의 성능이 저하되었다. 본 논문에서 제시된 건너뜀 연결 구조의 경우, 동일한 대역폭을 가진 반대 방향 이중 링 구조에 비하여 높은 성능과 균등한 부하 분배에 의한 개선된 확장성을 얻을 수 있었고, 반대 방향 이중 링의 경우에 발견되는 부하 편차를 크게 줄일 수 있었다.

**키워드** : 링 구조 CC-NUMA 시스템, 이중 링크, 확장성

***Abstract*** The multiprocessor system suffers interconnection network contension while exploiting the program's parallelism. A CC-NUMA system based on point-to-point link ring structure is one of the scalable architectures that expand the system bandwidth the number of processors/nodes increases. The dual-ring system is a simple solution to enhance the system performance and scalability by duplicating the links. In ring-based systems, an unbalanced transaction among links makes a hot spot on the interconnection network. In this situation, total system performance and scalability are bound by the hot spot of the links.

In this paper, I propose a dual-link CC-NUMA system, which alleviates the concentration of transactions among the links. By the simulation results, the proposed system significantly outperforms the single-ring and bidirection dual-ring systems. In addition, the proposed system shows better distribution of transactions among the links that achieves an extended scalability.

**Key words** : CC-NUMA system based on ring structure, dual-link, scalability

## 1. 서 론

다중 프로세서 시스템은 단일 시스템에 여러 개의 프로세서를 장착하고 복수 개의 프로세서를 활용하는 프로그램을 수행시킴으로써 전체적인 프로그램 수행 시간의 개선을 얻을 수 있다. 그러나 다중 프로세서 시스템의 병렬 처리 성능은 프로세서의 수에 따라 선형적으로 개선되지 않으며, 이러한 이유는 프로그램에 내재된 병렬성의 한계와 더불어, 여러 프로세서로부터의 접근하는 하드웨어 자원에 대한 경쟁에 의한 것이다[1]. 복수 개의 프로세서가 동시에 동작할 경우, 각 프로세서로부터 발생하는 처리 요구중 가장 많은 것은 메모리에 대한 경쟁인데, 이는 물리적 메모리 소자를 포함해서 프로세서로부터 메모리로 요구되는 모든 전송 경로상의 자원에 대한 경쟁으로 나타난다.

한편 메모리 소자는 DDR SDRAM, RDRAM 등 지속적으로 발전해 왔으며, 그 동작속도 또한 반도체 기술의 발전에 따라서 고속화되어왔다. 그러나 프로세서와 메모리 사이를 연결하는 상호연결망의 경우 물리적인

· 본 연구는 2004년도 가톨릭대학교 교비연구비의 지원으로 이루어졌음

† 정 회 원 : 가톨릭대학교 컴퓨터정보공학부 교수

hjsuh@catholic.ac.kr

논문접수 : 2003년 11월 4일

심사완료 : 2004년 6월 17일

경로의 길이 등으로 고속화에 어려움이 있다. 따라서 프로세서의 메모리 접근 지연은 메모리 소자의 속도에 의한 지연보다, 상호연결망의 동작 속도 및 대역폭에 의한 지연이 큰 비율을 차지하게 되었으며, 다중 프로세서 시스템의 성능 개선에 주요한 한계 요소로 작용해 왔다.

프로세서로부터 발생한 메모리 접근 시간은 물리적인 경로의 길이에 의한 지연시간과, 경로상에 위치한 자원의 배타적 점유 및 경쟁으로 유발된 대기 시간으로 나누어 볼 수 있다. 다중 프로세서 시스템에서 각 프로세서는 동시에 메모리 접근을 발생시키므로 이러한 메모리 접근 트랜잭션은 경로에 대한 경쟁을 거치게 된다. 결국, 경로의 대역폭과 경쟁의 정도에 따라서 발생되는 대기 시간은 달라지게 되고, 프로세서의 개수를 늘려도 늘어난 프로세서의 경쟁을 완화시키지 않을 경우, 시스템의 전체 성능은 많아진 프로세서 만큼의 이득을 얻을 수 없게 된다[1].

트랜잭션의 전송 경로에 해당하는 상호연결망의 형태에 따라서 경로 점유에 따르는 대기 시간을 줄일 수 있다. 단일 시간에 하나의 트랜잭션만을 전송할 수 있는 공유 버스가 사용될 경우 높은 경쟁을 나타내게 되며, 동시에 여러 트랜잭션을 전송할 수 있는 점대점 연결 구조의 경우 버스에 비해 낮아진 경쟁과 보다 짧은 대기 시간을 나타낸다. 최근 많이 채용되고 있는 SCI (Scalable Coherent Interface)[2]등의 점대점 연결을 이용한 링 구조 CC-NUMA(Cache Coherent Non-Uniform Memory Access) 시스템의 경우, 단일 노드 내에 공유 버스를 통하여 수 개의 프로세서를 연결하고, 각 노드간을 점대점 링크를 통하여 연결하는 구조로 이루어져 있다[3]. 이 구조는 여러 개의 트랜잭션이 동시에 전송될 수 있고, 노드의 추가에 따라서 시스템 전체의 대역폭이 커지며, 동시 전송 가능한 트랜잭션의 수가 많아지므로, 고성능 시스템에 다수 채용된다. 노드 당 점대점 연결을 두 개로 확장하여 반대 방향 이중 링으로 구성한 경우, 복수 링크에 의한 대역폭의 상승과 함께 전송 경로가 단축될 수 있으므로 단일 링에 비하여 더 높은 확장성과 성능을 나타낸다[4].

링 구조가 확장성과 성능에 있어서 많은 장점을 가지고 있는 구조이기는 하나, 특정 노드에 많은 트랜잭션이 발생할 경우, 이 노드를 경유해야 하는 트랜잭션은 높은 경쟁에 의하여 긴 전송지연이 발생하게 된다. 이러한 경우, 특정 링크의 집중된 경쟁으로 인하여 시스템의 전체 대역폭에 비례하는 성능을 얻을 수 없으며, 노드 추가에 따른 성능 향상도 얻을 수 없게 된다.

본 논문은 이중 연결을 가진 링 구조의 CC-NUMA 시스템에서, 특정한 링크에 발생하는 과도한 경쟁으로 인하여 시스템 성능이 저하되며, 시스템의 전체 대역폭을 제대로 활용할 수 없게 되고, 이에 따라 합당한 성능을 얻을 수 있는 노드의 개수가 제한됨에 주목하여, 노드의 두 개의 링크 중 하나는 링 형태로 연결하고, 다른 하나는 건너뜀 연결을 갖는 형태로 연결하여, 링크상에 발생하는 트래픽을 분산함으로써 성능 향상 및 확장성의 개선을 얻고자 하는 것이다.

본 논문의 구성은 다음과 같다. 2장에서 관련 연구에 대하여 살펴보고, 3장에서 반대 방향 이중 링 연결 구조와 건너뜀 연결 구조 및 트랜잭션 전송 경로를 설명하며, 4장에서 시뮬레이션 도구와 환경 및 결과를 제시하고, 5장에서 결론을 맺는다.

## 2. 관련 연구

공유 메모리 다중 프로세서 시스템은 프로세서와 메모리의 배치에 따라 UMA(Uniform Memory Access), CC-NUMA, COMA(Cache Only Memory Architecture)로 나누어 볼 수 있다. UMA 시스템은 모든 프로세서에 대해 메모리의 접근 시간이 일정한 것으로서, 한 물리적 위치에 전체 메모리를 집중시킨 형태이며[1], CC-NUMA는 메모리를 일정 크기로 분할하고, 분할된 각 메모리를 각 프로세서와 가까이 연결하여 배치하여, 전체 메모리 영역을 지역 메모리와 원격 메모리로 분리한 것이다[5]. COMA는 메모리가 아닌 큰 용량의 캐시를 각 프로세서가 갖도록 하여, 자주 접근되는 주소 영역을 프로세서에 가까이 할당할 수 있도록 한 것이다[6]. UMA의 경우 공유 버스 구조의 다중 프로세서 시스템에서 주로 사용되며, 메모리 접근이 집중되는 단점을 갖고 있다. CC-NUMA의 경우 분산된 메모리 접근, 지역 메모리와 원격 메모리의 적절한 분배, 원격 메모리 접근의 효율성을 위한 원격 캐시의 채용 등으로 고성능 상용 시스템에 널리 사용되고 있다. CC-NUMA 시스템은 BBN Advanced Computers의 Butterfly Machine에서 시작되었으며 Stanford의 DASH 로 구현되었다[5]. COMA 시스템의 경우 메모리가 아닌 큰 용량의 캐시를 분산시켜 자주 접근되는 주소 영역을 가까이 배치할 수 있도록 한 구조이나, 운영 체제의 가상 메모리 처리 부분에 대한 변경을 필요로 하며, 대용량의 캐시에 대한 일관성 유지가 필요하여 상용 시스템에 사용되지 못하고 있다.

점대점 연결을 이용한 링 구조 CC-NUMA 시스템은 학계 및 산업계의 고성능 상용 시스템에서 다수 채용되고 있다. 이러한 예로, IBM NUMA-Q 시스템과 그 원형인 Sequent STiNG[3], Debois의 Express Ring[7], Data General AViiON 시스템[8], 서울대학교 컴퓨터공학과의 PANDA 시스템[9]등은 점대점 연결을 이용한 링 구조의 시스템이다. 단일 연결의 대역폭을 확장하고,

전송 경로의 단축을 이용하여 두 개의 링크를 이용한 이중 링 구조 또한 고안되었다. 특히 반대 방향 이중 링 구조는 동일 개수의 링크를 이용한 크로스바 구조 등에 비하여 고성능을 나타냄이 보고된 바 있으며[4], Data General AViiON 시스템과 서울대학교의 PANDA-II 시스템이 반대 방향 이중 링 구조로 구현된 경우이다.

## 3. 연결 구조 및 트랜잭션 전송 경로

그림 1은 두 개의 점대점 링크를 이용하여 반대 방향 이중 링 구조로 구현된 시스템이다.노드 내부에 수 개의 프로세서와 지역메모리가 공유 버스를 통하여 연결된 SMP(Symmetric Multiprocessor) 형태로 구성되어 있으며, 각 노드는 서로 점대점 연결을 통하여 링 형태로 연결되어 있고, 두 개의 링크는 서로 반대 방향의 링을 구성한다. 이러한 구조는 Data General AViiON 과 PANDA-II 시스템에서 사용되었으며, 두 개의 연결 링크와 링크 제어기를 가지고 있다.

### 3.1 트랜잭션의 종류 및 전송 경로

링 구조의 시스템에서 적용할 수 있는 캐시 일관성 유지 기법은 트랜잭션의 발생 형태와 밀접한 관련이 있다. chain 디렉토리를 이용한 캐시 일관성을 사용할 경우, 메모리 접근 트랜잭션은 발생 노드로부터 해당되는 주소의 메모리를 가진 홈 노드로 전송되며, 홈 노드의 메모리 연결 정보에 따라 다음 노드로 트랜잭션이 생성된다. full-map 디렉토리에 의한 캐시 일관성 유지를 사용할 경우, 홈 노드에서 일관성 유지에 관련된 모든 노드의 정보를 가지고 있게 된다. 즉 디렉토리에 의한 일관성 유지를 사용할 경우, 트랜잭션은 일대일 전송만이 발생된다. 반면 주로 버스에 사용된 스누핑에 기반한 캐시 일관성 유지 방법은 Express Ring, PANDA 등의 링 구조에서도 사용되고 있으며, 이 경우 방송 트랜잭션과 일대일 전송 트랜잭션이 모두 사용된다.

노드에서 발생하는 메모리 접근 트랜잭션은 요청과 응답으로 분리되며, 스누핑에 기반한 일관성 구조를 사용할 경우, 요청 트랜잭션은 방송 형태로 전송되어 모든 노드가 참여하게 되며, 요청 트랜잭션에 의해 응답을 필요로 하는 노드는 응답 트랜잭션을 일대일로 전송한다. 즉 요청은 방송으로 이루어지고, 응답은 일대일 전송이 된다. 다음 표 1은 발생되는 트랜잭션의 종류와 유형을 나타낸 것이다.

단일 링 구조에서 트랜잭션의 전송은 링크의 전달 방향으로 이루어지며, 모든 트랜잭션은 단일 경로상으로 전달되므로, 트랜잭션 전달을 위한 경로 설정 방법은 별도로 필요하지 않다. 반면, 이중 링 구조의 경우 복수개의 링크 중 한 링크로 트랜잭션이 전송되므로 경로 설정 방법을 필요로 한다. 반대 방향 이중 링 구조의 경우, 트랜잭션 경로 설정에 여러가지 방법을 적용할 수 있으며, 경로 설정 방법으로 접근하는 주소의 최하위 비

표 1 트랜잭션의 종류와 유형

| 트랜잭션 이름 | 트랜잭션 종류 | 데이타 유무 |
|---|---|---|
| 읽기 요청 | 방송 트랜잭션 | 무 |
| 쓰기 요청 | | |
| 무효화 | | |
| 읽기 응답 | 일대일 트랜잭션 | 유 |
| 쓰기 응답 | | |
| 되쓰기 | | |

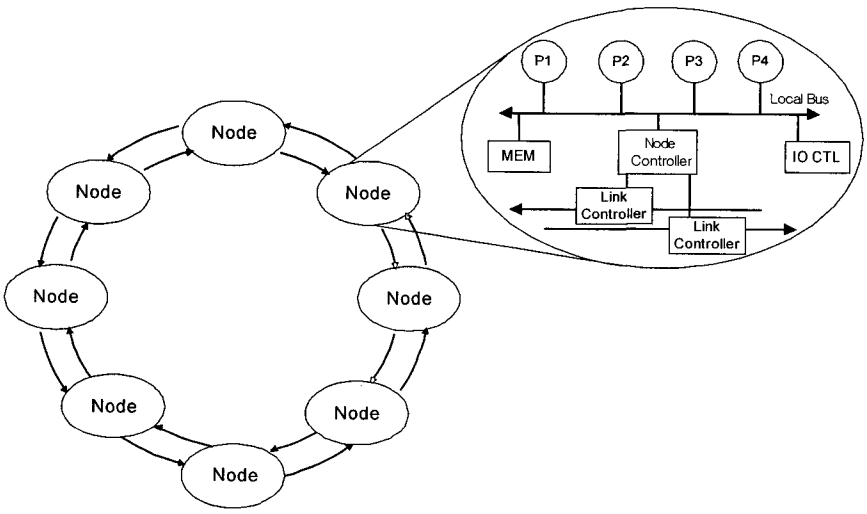



그림 1 반대 방향 이중 링 구조의 시스템

트에 따라 연결 경로를 선택하는 방법, 요청 및 응답에 따라 전송 경로를 달리하는 방법, 해당되는 주소의 메모리를 가지는 노드와의 경로길이에 따라 설정하는 방법, 데이타 전송을 일으키는 노드간의 물리적 거리에 따라 전송 경로를 설정하는 방법, 해당되는 홉 노드쪽 방향으로 전송하는 방법 등이 있다.

### 3.2 건너뜀을 갖는 이중 연결 구조

본 논문에서 제안하는 구조는 그림 2와 같이 두 개의 링크를 갖는 노드 중 하나를 링 형태로 연결하고, 다른 하나를 건너뜀을 갖는 노드로 연결한 것이다.

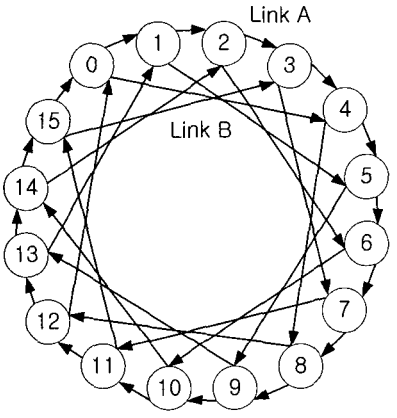


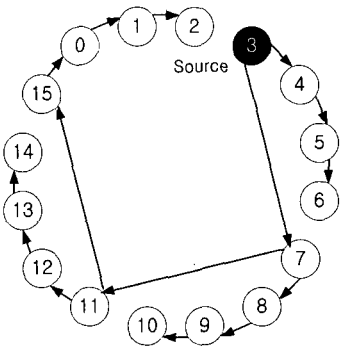

그림 2 건너뜀 경로를 갖는 이중 링크 구성, 4 건너뜀, 16 노드

각 노드는 A와 B 두 종류의 링크를 가지며, A 링크의 경우, 단일 링 구조와 동일하게 구성되고, B 링크는 A와 동일한 방향으로 일정 건너뜀 수를 갖는다. 그림 2는 4 건너뜀을 갖는 형태로 연결된 것이다. 건너뜀 수는 일정 값으로 제한되어 원활한 부하 분배를 이룰 수 있도록 설정되며, 다음과 같이 제한된다.

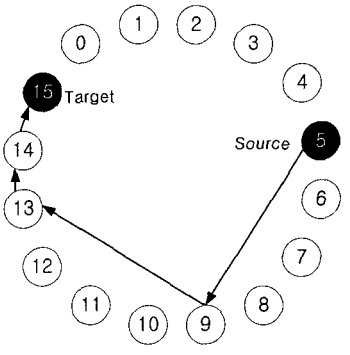
- $N$ : 총 노드의 개수
- $k$ : 링크 B 연결 건너뜀 수
- $N = k_i$, $N \geq 2k$ 이며, $k$, $i$는 각각 1보다 큰 정수

이러한 제한이 제대로 적용되는 $k$가 존재하기 위해서는, 총 노드의 개수는 소수가 되지 말아야 한다. 그러나 다중 프로세서 시스템에서 노드의 개수 및 프로세서의 개수 등은 대부분 2의 승수로 구성되므로, 별다른 제한으로 작용하지 않는다.

그림 2와 같은 연결 구조에서 방송 및 일대일 트랜잭션의 전송은 트랜잭션을 생성시키는 노드와 받을 노드의 물리적 위치에 따라서 결정된다. 방송 트랜잭션의 경우 모든 노드에 전달되어야 하므로, 건너뜀 링크를 가지는 이중 연결 구조 시스템에서 다음과 그림 3(a)와 같이 전달된다. 일대일 트랜잭션의 경우, 그림 3(b)와 같




(a) 방송 트랜잭션 전송




(b) 일대일 트랜잭션 전송

그림 3 건너뜀 링크를 갖는 구조에서 방송 및 일대일 트랜잭션의 전송

이 트랜잭션을 생성한 노드로부터 목적 노드로 전달되는 최단 경로를 이용한다.

## 4. 시뮬레이션 방법 및 결과

다중 프로세서 시스템의 시뮬레이션을 하기 위한 도구로, 본 논문에서는 Augmint를 사용하였다[10]. Augmint 시뮬레이터는 실행 구동형 시뮬레이터로서 전단부와 후단부로 분리된다. 전단부는 실행할 병렬프로그램이 사용하는 프로세서의 수에 따라서 각 프로세서로부터 발생하는 메모리 접근을 추적하여 메모리 접근의 종류에 따른 함수를 호출하며, 이 함수는 후단부에서 구현하게 된다. 시뮬레이터의 후단부는 시뮬레이션할 대상 시스템의 프로세서 이외의 모든 부분을 모사하게 되며, 캐시, 일관성 프로토콜, 상호연결망에 대한 트랜잭션의 경쟁 등을 구현하여야 한다. 본 실험의 성능 평가를 위하여 프로세서로부터 발생한 메모리 접근에 대하여 즉시 처리 완료시키는 이상적인 모델과, 단일 링 구조, 반대 방향 이중 링 구조, 건너뜀 연결을 갖는 이중 연결 링 구조를 모델링하였다. 이상적인 모델의 경우, 캐시와 상호연결망 등 모든 메모리 관련 요소가 무한한 크기와

무한히 짧은 시간 안에 완료될 수 있도록 하였으므로, 프로그램의 병렬성이 완벽히 구현될 수 있는 형태이다.

캐시 일관성 프로토콜은 방송과 일대일 트랜잭션을 모두 발생시키는 스누핑을 설정하였다. 반대 방향 이중링 구조의 경우 트랜잭션을 전송하는 경로 설정 방법이 별도로 필요하며, 방송 트랜잭션은 홈 노드가 가까운 링크쪽으로 전송하도록 설정하였고, 일대일 전송에서는 가까운 거리를 갖는 링크쪽으로 전송하도록 하였다.

성능 평가에 사용한 프로그램은 SPLASH-2 벤치마크[11] 프로그램 중 FFT와 LU, RADIX 세 가지를 사용하였으며, 각 프로그램의 부하는 다음 표 2와 같다.

시뮬레이션을 위하여 사용한 프로세서의 개수는 동일 부하의 프로그램에 대하여, 각각 4, 8, 16, 32 개의 프로세서를 적용하였으며, 노드 당 프로세서의 수는 1개와 2개인 경우에 대하여 각각 적용하였다. 따라서 노드 당 1 프로세서인 경우는 4 노드, 8 노드, 16 노드, 32 노드인 경우가 되고, 노드 당 2 프로세서인 경우는 4 노드, 8 노드, 16 노드에 해당된다.

건너뜀 경로를 갖는 구조에서 프로세서와 노드 수에 따른 건너뜀 수는 표 3과 같이 적용되었다.

건너뜀 수의 값은 노드 수에 대비하여 적절한 크기를 갖는 것이 효율적이다. 건너뜀 수는 노드 수에 대하여 $N = ki$, $N \geq 2k$ 이며, $k$, $i$는 각각 1보다 큰 정수를 만족하는 수이면 가능하나, 방송 및 일대일 트랜잭션의 전달 경로상 $k$, $i$의 차가 적을수록 유리하며, 위의 건너뜀 수는 이러한 값 중 작은 수를 선택한 것이다.

프로세서의 속도 및 링크의 속도 등은 다음 표 4와 같이 최근의 프로세서와 SCI 링크의 전송 값에 근사하도록 설정하였다.

표 2 시뮬레이션에 이용된 프로그램 및 부하

| 프로그램 | 부하 |
|---|---|
| FFT | -m16 -p*프로세서수* -n2048 -l5 |
| LU | -n128 -p*프로세서수* -b16 |
| RADIX | -p*프로세서수* -n131072 -r1024 -m2097152 |

표 3 건너뜀 경로를 갖는 이중 연결 구조에서 건너뜀 수

| 노드 수 | 건너뜀 수 |
|---|---|
| 2 | - |
| 4 | 2 |
| 8 | 2 |
| 16 | 4 |
| 32 | 4 |

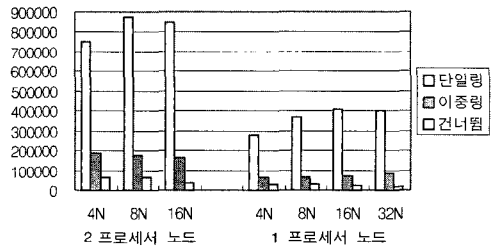
표 4 실험 대상 시스템 환경

| 항목 | 값 |
|---|---|
| 프로세서 클럭 속도 | 1 GHz |
| 시스템의 프로세서 수 | 4, 8, 16, 32 개 |
| 프로세서 당 캐시 크기 | 64 Kbyte |
| 프로세서 캐시 연관 | 4 way |
| 노드 당 프로세서 수 | 1개, 2개 |
| 노드 내 버스 속도 | 266 MHz |
| 링크 전송 대역폭 | 1Gbyte/s |
| 캐시 교체 정책 | Least Recently Used |

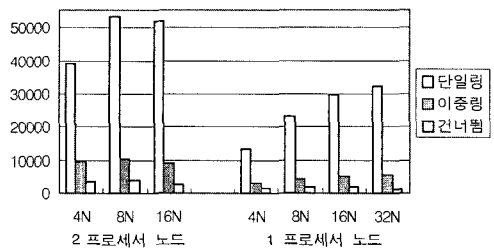
### 4.1 링크에 대한 경쟁

다음 그림 4는 각 링크에 대한 재시도 횟수의 평균을 나타낸 것이고, 그림 5는 재시도 횟수의 표준 편차를 나타낸 것이다. 그림의 왼쪽 세 종류의 것은 노드 당 2 프로세서인 경우 이고, 오른쪽 네 가지는 노드 당 1 프로세서로 구성한 경우이다.

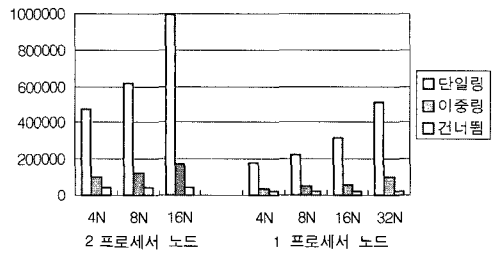
그림 4의 각 링크에 대한 재시도 횟수는 프로세서의 증가에 따라 커졌으며, 이는 평균적으로 링크에 대한 더 높은 경쟁을 나타냈음을 의미한다. 노드 당 프로세서의 수를 두 개로 한 경우에 비하여 노드 당 프로세서의 수를 한 개로 줄인 경우, 높아진 대역폭에 의한 이득을 나




(a) FFT

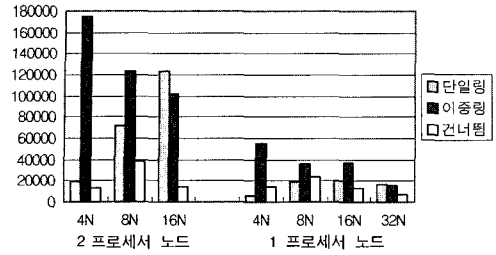



(b) LU

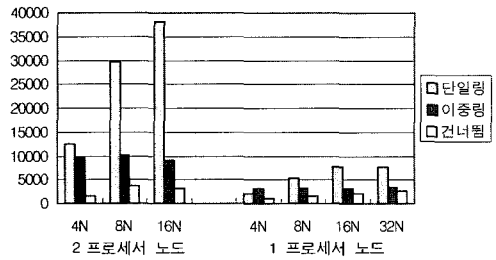



(c) RADIX

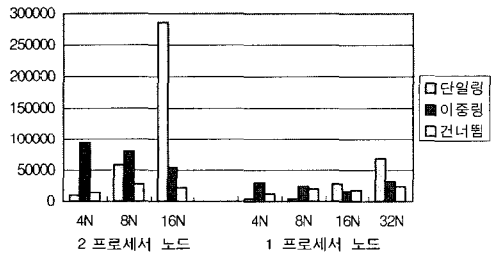
그림 4 노드에 발생한 재시도 횟수의 평균값




(a) FFT




(b) LU




(c) RADIX

그림 5 노드에 발생한 재시도 횟수의 표준편차

타냈으며, 결과적으로 더 적은 재시도 횟수를 나타냈다. 평균적인 재시도 횟수가 적은 경우, 더 효율적인 전송이 이루어졌음을 의미한다. 반대 방향 이중 링은 본 논문에서 제안한 건너뜀 링크 구조에 비하여 높은 재시도 횟수를 나타냈으며, 이는 동일한 대역폭을 가짐에도 불구하고 건너뜀 링크를 갖는 구조가 더 효율적임을 의미한다.

그림 5는 각 링크에 발생한 재시도 횟수의 표준 편차를 구한 것이다. 표준 편차가 크게 나타날 경우, 특정 링크에 대한 재시도가 다른 링크에 비하여 높게 나타났음을 의미하며, 이 경우 경쟁이 높게 나타난 링크가 시스템 성능을 저하시키고 확장성을 제한하게 된다. 각 프로그램에 나타난 표준 편차를 보면, 단일 링 구조의 경우 노드의 수 증가에 따라서 편차가 증가되고 있으며, 노드 당 1 프로세서인 경우에 비하여 노드 당 2 프로세서를 적용한 경우 훨씬 높은 편차를 나타냈다. 따라서 단일 링의 경우에 노드에 프로세서의 수를 늘림에 따라서 병목 링크가 발생함을 알 수 있다. 반대 방향 이중 링의 경우, FFT 프로그램에서 단일 링에 비하여 높은 편차를 나타내는 경우가 발견되며, 특히 노드에 여러 프로세서를 적용하였을 경우, 링크에 대한 불균일한 경쟁이 심하게 나타남을 알 수 있다. 반면, 건너뜀 링크를 갖는 구조는 대부분의 경우 이중 링에 비하여 낮은 편차를 나타내고 있으며, 이는 전체 링크에 대한 부하가 비교적 균일하게 이루어지고 있음을 의미한다. 특히, 노드의 증가나 노드 당 프로세서를 2개 적용하였을 경우도 적은 편차를 보이고 있다. 이는 건너뜀 링크에 의하여 시스템 전체에 존재하는 링크에 대한 경쟁이 비교적 균일하게 발생하고 있고, 시스템에 노드를 추가하거나, 프로세서의 수를 증가함에 따라서 발생할 수 있는 링크에 대한 불균일한 경쟁을 감소시킬 수 있음을 의미한다.

### 4.2 실행 시간

다음 그림 6은 캐시 및 상호연결망에 대한 경쟁이 존재하지 않는 이상적인 시스템에서 4 프로세서 시스템에 대하여 프로세서의 증가에 따라서 얻을 수 있는 수행 시간의 비율을 나타낸 것이다. 이 비율은 프로그램에 내재된 병렬성 정도를 의미한다.

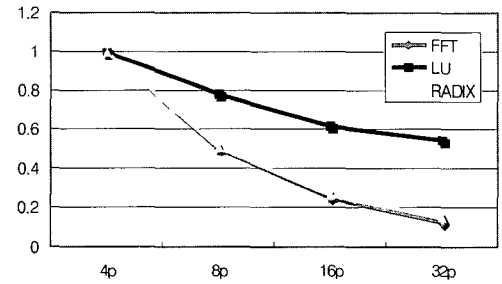



그림 6 이상적인 시스템의 프로세서 증가에 따르는 수행 시간 비율

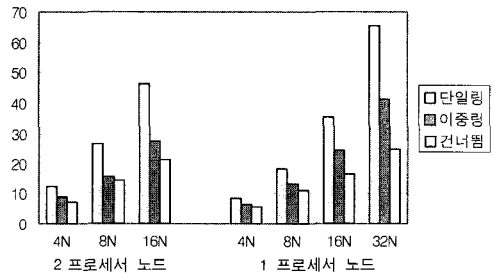
FFT와 RADIX의 경우 이상적인 시스템에서는 프로세서의 증가에 따라서 선형적인 수행 시간의 개선을 나타내고 있으며, LU의 경우 프로세서의 개수가 두 배로 증가함에 따라서 20% 정도의 수행 시간을 줄일 수 있음을 알 수 있다. 다음 표 5는 이상적인 경우에 각 프로그램이 수행하는 데 걸린 사이클이다.

표 5 이상적인 경우 각 프로그램의 수행 사이클

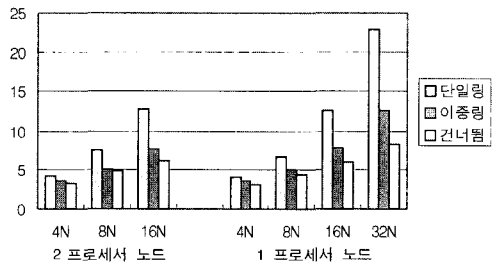
| 프로세서 수 | FFT | LU | RADIX |
|---|---|---|---|
| 4 | 11,726,152 | 4,440,507 | 26,707,295 |
| 8 | 5,875,480 | 3,462,192 | 13,508,517 |
| 16 | 2,950,984 | 2,732,183 | 7,020,250 |
| 32 | 1,490,416 | 2,428,085 | 3,998,275 |

다음 그림 7은 단일 링과 반대 방향 이중 링, 건너뜀 링크를 갖는 링에 대하여 위의 그림 6에서 제시한 이상적인 실행 시간에 대한 비율을 나타낸 것이다.

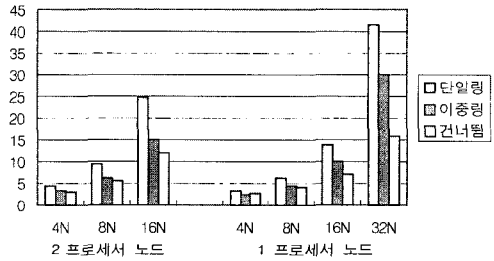
세 개의 프로그램에 대하여 이상적인 경우에 대한 성능 저하 정도는 표 5에 나타난 각 프로그램의 부하 크기와 비례하였다. 모든 경우에, 건너뜀 구조가 가장 이상적인 시스템과 적은 차이를 나타냈다. 이상적인 시스템에 대한 실행 시간의 비율는 노드의 증가에 따라 점차 높아졌다. 노드의 증가에 따라 비율이 급격히 증가한 경우, 시스템상 노드의 추가에 따르는 성능 향상의 정도가 적어짐을 의미하며, 그림 7에서 건너뜀 링크를 갖는 구조의 성능비율이 급격히 증가하지 않는 것으로, 시스템에 보다 많은 프로세서를 장착할 수 있는 확장성을 갖게 됨을 알 수 있다.




(a) FFT




(b) LU




(c) RADIX

그림 7 이상적인 시스템의 수행 시간에 대한 각 구조의 수행 시간 비율

## 5. 결 론

최근 고속의 마이크로프로세서가 다수 등장하면서, 시스템 성능은 상호연결망 및 메모리 접근 속도에 대해 더욱 큰 영향을 받게 되었다. 프로세서 기술은 상호연결망의 고속화에 비하여 더욱 빠른 속도로 발전해 나가고 있으며, 상호연결망의 병목 현상은 상대적으로 주요한 문제로 대두되고 있다.

상호연결망의 대역폭 및 지연을 개선하기 위하여 복수개의 링크를 이용한 링 구조는 SCI와 같은 고속의 점대점 연결을 이용함으로써 고성능 시스템으로 다수 사용되고 있으나, 시스템의 여러 링크에 부하가 균일하게 배분되지 않을 경우, 특정 링크에 대한 경쟁으로 과도한 지연이 발생하여 시스템 성능이 저하된다.

단일 링 구조의 경우 트랜잭션의 전달 경로가 유일하게 구성되므로, 링크에 대한 부하 분배를 하드웨어적으로 해결할 수 없으나, 이중 연결을 이용할 경우, 링크의 적절한 연결형태로 부하 배분을 보다 개선시킬 수 있다.

반대 방향 이중 링 구조의 경우, 단일 링 구조에 비하여 어느정도 개선된 부하 배분과 성능을 나타내나, 특정 링크에 대한 과부하 현상이 여전히 나타나며, 이로 인한 성능 저하와 시스템에 추가할 수 있는 프로세서의 확장성이 제한된다.

본 논문은 이중 연결 중 하나를 건너뜀을 갖도록 하고, 이러한 구조에 대해 적절한 방송 트랜잭션과 일대일 트랜잭션 전달 경로를 설정함으로써 여러 링크에 대한 부하 배분을 개선하였다. 이러한 결과 반대 방향 이중 링 구조와 동일한 시스템 대역폭과 하드웨어 자원을 이용하면서, 더 높은 프로그램 수행 성능과 보다 많은 수의 프로세서를 수용할 수 있도록 확장성이 개선되었다.

차후 과제로서 SCI에서 이용된 chain 디렉토리 일관성 유지 기법이 사용되었을 경우, 트랜잭션 전송 경로 개선 방법의 연구가 진행중이다.

## 참 고 문 헌

[1] John L. Hennessy, David A. Patterson, David Goldberg, Computer Architecture: A Quantitative Approach, 2nd Ed., Morgan Kaufmann, 15 May, 2002.

[2] IEEE Computer Society, IEEE Standard for Scalable Coherent Interface(SCI), Institute of Electrical and Electronics Engineers, Aug. 1993.

[3] Tom Lovett and Russel Clapp, "STiNG : A CC-NUMA Computer System for the Commercial Marketplace," Proc. of the 23th Int. Symp. on Computer Architecture, pp. 308-317, May 1996.

[4] H. Oi and N. Ranganathan, "Performance Analysis of the Bidirectional Ring-Based Multiprocessor,"

Proc. of ISCA 10th Int. Conf. on Parallel and Distributed Computing Systems, pp. 397-400, October 1997.

[5] Daniel Lenoski, James Laudon, Kourosh Gharachorloo, Wolf-Dietrich Weber, Anoop Gupta, John Hennessy, Mark Horowitz, and Monica S. Lam, "The Stanford Dash multiprocessor," Computer, Vol. 25 No.3, pp. 63-79, Mar. 1992.

[6] A. Saulsbury, T. Wilkinson , J. B. Carter, and A. Landin, "An Argument for Simple COMA," Proc. of the 1st IEEE Symp. on High-Performance Computer Architecture, pp. 276-285, 1995.

[7] L. Barroso and M. Dubois, "The Performance of Cache-Coherent Ring-based Multiprocessors," Proc. of the 20th Int. Symp. on Computer Architecture, pp. 268-277, May 1993.

[8] http://www.dg.com/

[9] http://panda.snu.ac.kr/nrl/

[10] A-T. Nguyen, M. Michael, A. Sharma, and J. Torrellaz, "The Augmint multiprocessor simulation toolkit for Intel x86 architecture," Proc. of the IEEE Int. Conf. on Computer Design, Oct. 1996.

[11] S.C.Woo, M.Ohara, E.Torrie, J.P.Singh, and A.Gupta. "Methodological considerations and characterization of the SPLASH-2 parallel application suite," Proc. of the 22th Int. Symp. on Computer Architecture, pp. 24-36, 1995.

서 효 중

1991년 서울대학교 이학사. 1994년 서울대학교 공학석사(컴퓨터공학). 2000년 서울대학교 공학박사(컴퓨터공학). 2002년 지씨티 리서치 선임연구원. 2003년~현재 서울대학교 컴퓨터연구소 객원연구원. 2003년~현재 가톨릭대학교 컴퓨터정보공학부 전임강사. 관심분야 컴퓨터 구조, 병렬처리 시스템, 내장형시스템, 클러스터 시스템