# TPC-C 벤치마크를 이용한 SPAX 병렬 컴퓨터의 성능분석

# Performance Evaluation of the SPAX Parallel Architecture based on the TPC-C Benchmark

김희철 · 신정훈 · 이용두

(대구대학교 정보통신공학부)

Hiecheol Kim · Jeonghun Shin · Yong-Doo Lee

(Division of Computer & Communication Engineering, Taegu University)

요 약

일반 병렬 처리 시스템(General Purpose Multiprocessors)과는 달리, 병렬 트랜잭션(Transaction) 처리 시스템의 성능은 메모리의 계층구조와 입출력 시스템의 구조 등에 크게 영향을 받는 특징을 갖는다. 본 논문은 입출력 노드의 성능 분석에 주안점을 두고 전제 시스템에서의 입출력 노드의 개수, 병렬 디스크의 개수 및 상호연결망(Interconnection Network)과의 접속을 제공하는 스위치의 처리 용량 등의 인수들이 SPAX 병렬 트랜잭션 처리 시스템의 미치는 성능의 평가 및 분석에 대한 연구 내용 기술한다. 본 연구에서는 벤치마크로는 병렬 트랜잭션 시스템의 성능 평가에 주로 사용되고 있는 TPC-C 벤치마크를 사용하며 모의 입력(Synthetic workload)을 통한 성능분석을 수행하였다. 본 연구는 입출력 노드에 부하가 많이 걸릴 경우 패킷의 크기에 따라 시스템의 성능에 큰 영향을 미치며, 반면에 입출력 노드내의 상호연결망의 접속(Interface)을 제공하는 XNIF의 데이터 버퍼 개수의 증가는 시스템의 성능 향상에 기여를 하지 않음을 보여준다. 이는 시스템의 성능향상을 위해서는 패킷 전송 경로상의 모든 시스템 요소의 성능 향상이 병행되어야 함을 보여준다. 마지막으로 프로세싱노드와 입출력노드의 처리능력의 균형이 병렬 트랜잭션 시스템의 설계에 있어서 매우 중요함을 보여준다.

## Ⅰ. 서 론

최근에 분산 메모리 병렬 시스템이나 캐쉬 일관성(cache-coherent) 공유 메모리 병렬 시스템 등, 다수의 상용 병렬 컴퓨터들이 등장하고 있다. 이러한 시스템 상에 대용량

OLTP(OnLine Transaction Processing)나 DSS(Decision Support System) 같은 응용 프로그램들이 사용되고 있다[2]. 실제로 최근에 발표된 Sequent사의 STiNG[4]이나 HP사의 SPP-1200은 특히 이러한 데이터베이스 처리 시스템으로 사용되고 있다. 데이터베이스를 기반으로 하는 프로그램들은 주로 트랜잭션(Transaction)의 형태로 요구되고 처리되며 데이터베이스에 저장된 거대한 양의 정보를 사용한다. 전형적인 OLTP 질의들은 간단하며 데이터베이스의 데이터들에 대한 액세스(읽기와 쓰기)를 수반한다. 그러나 데이터베이스는 복잡한 locking 구조를 가지며, 하드 디스크와 같은 I/O 장치로부터 메모리로 데이터 블록을 읽어오는 것을 직접 관리하고, 효율적으로 데이터베이스의 데이터를 관리하기 위해 복잡한 데이터 구조를 사용한다. 그러므로 데이터베이스 벤치마크는 과학기술 벤치마크[10]와는 달리 메모리 계층구조, I/O 구조, 그리고 상호연결망의 구조에 그 성능이 크게 영향을 받는 특징을 갖는다[5].

최근 들어 트랜잭션 시스템의 메모리 계층구조와 상호연결망의 성능 특성을 평가하기 위해 여러 연구가 수행되고 있다[4,8,9]. 국내에서도 병렬 트랜잭션 처리 시스템의 I/O와 상호연결망의 성능 평가 및 분석을 위한 연구가 수행되었다[1,7]. 이 연구들은 모두 고속 병렬 컴퓨터 SPAX(Scalable Parallel Architecture computer based on X-bar network) 시스템에 대하여 병렬 트랜잭션 시스템의 성능 평가를 모두 데이터베이스 벤치마크들 중에 가장 간단한 형태의 하나인 TPC-B를 사용하고 있다[1,7]. 성능 평가의 내용은 시스템 크기, 디스크 개수, 디스크 캐쉬 접근 실패율(Disk Cache Miss), 그리고 IO 노드의 데이터 버퍼의 개수 등이 시스템 성능에 미치는 영향의 분석[7], 그리고 상호연결망의 버퍼 개수와 클락(Clock) 속도 그리고 프로세싱 처리능력과 I/O 처리 능력과의 상관 관계 등에 대한 분석 [1] 등을 포함하고 있다. 이러한 성능 평가 및 분석 내용들은 병렬 트랜잭션 처리 시스템의 성능에 영향을 미치는 요소들을 추출하고 또한 그 I/O와 상호연결망의 설계 상에 고려해야 하는 여러 요소들에 대한 기본 방향을 제시해 주고 있다.

하지만 이러한 연구들의 결과는 병렬 트랜잭션 시스템의 성능 분석에 대한 충분한 이해를 제공하지 못한다. 그 주된 이유는 병렬 트랜잭션 처리 시스템의 성능 평가 시 사용한 TPC-B는 그 알고리즘의 내용이 보편적인 트랜잭션 처리의 내용을 충분히 포함하고 있지 않으며 또한 데이터의 이동이 주로 근접 프로세싱 노드 사이에서만 이루어지는 현상을 가지므로 트랜잭션 시스템에 대한 충분한 성능 평가를 제공하지 못한다. 그러므로 병렬 데이터 처리 시스템(특히 상호연결망)의 정확하고 완벽한 성능 평가를

위해서는 TPC-B 뿐만 아니라 최근의 데이터베이스 시스템용 벤치마크인 TPC-C 또는 TPC-D를 사용한 성능 평과가 수행할 필요가 있다.

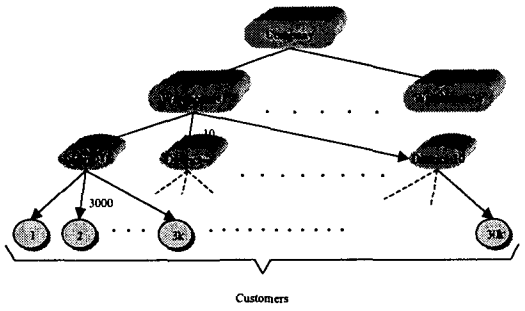
본 연구에서는 TPC-C를 그 벤치마크로 사용하여 SPAX 시스템을 대상으로 시스템 크기, 디스크 개수와 디스크 캐쉬 접근 실패율, 입출력 노드와 상호연결망의 통신 데이터 버퍼의 개수, CPU의 속도, Xcent 스위치의 클럭 속도, 프로세싱 노드와 입출력 노드간의 처리능력의 비율 등의 시스템 성능에 결정적인 영향을 미치는 요소들에 대한 구체적이며 포괄적이고 아울러 정확한 성능 평가 및 분석을 수행하였다. 본 논문에서는 그 연구의 내용중 주로 입출력 노드의 성능 분석 결과를 기술한다. 본고의 2장에서는 데이터베이스 벤치마크에 대한 간략한 설명이 제공되며, 3장에서는 시뮬레이션과 실험 환경이 기술된다. 4장에서는 평가 결과를 설명하며, 마지막으로 5장에서는 결론과 향후 수행될 연구에 대해 기술한다.

## II. TPC-C 벤치마크

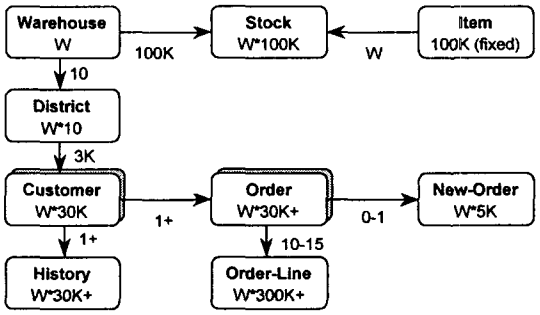
본 절에서는 병렬트랜잭션 시스템의 성능평가를 위해 최근에 주로 사용되고 있는 벤치마크 프로그램이며 본 연구에서 사용되고 있는 TPC-C에 대하여 간략하게 설명한다.

TPC-C(Transaction Processing Council - Benchmark C)는 주어진 컴퓨터 시스템의 트랜잭션 처리 성능을 평가하기 위하여 실제적인 OLTP에 가까운 주문(Order) 처리 시스템을 모델 화한 벤치마크 프로그램이다. TPC-C는 그 처리 조건이나 내용이 TPC-A보다 복잡하고 요구되는 데이터베이스의 용량은 훨씬 크며, 시스템에 대한 작업 부하량은 TPC-A의 3~5배 정도가 된다. TPC-C는 실제적인 도매(Wholesale) 업무를 그 모델로 하고 있으며, 그 데이터베이스는 작업부하(Workload)가 트랜잭션에서 구조상의 변화 없이 분배될 수 있도록 주문 처리에 초점을 되었다. 모델로 사용한 회사는 다수의 지리적으로 분산된 물류창고(Warehouse)를 갖는 대규모 공급업자이다. 각각의 물류창고는 회사에서 취급하는 100,000가지 종류의 제품들을 재고를 보유한다. 각각의 물류창고는 10개의 지점(District)을 갖으며, 각 지점은 3,000명의 고객(Customer)들을 대상으로 독립적으로 물품 주문, 배달, 및 대금 결재를 수행한다. <그림 1>은 TPC-C의 각 개체(Warehouse, District, Customer)들의 계층구조를 보여주고 있다.

TPC-C의 데이터베이스의 구성은 9개의 서로 다른 테이블로 구성된다. 이들 테이블 간의 관계는 <그림 2>와 같이 정의된다. 그림에서 개체 블록 안의 숫자는 레코드의 개




<그림 1> TPC-C 개체의 계층구조



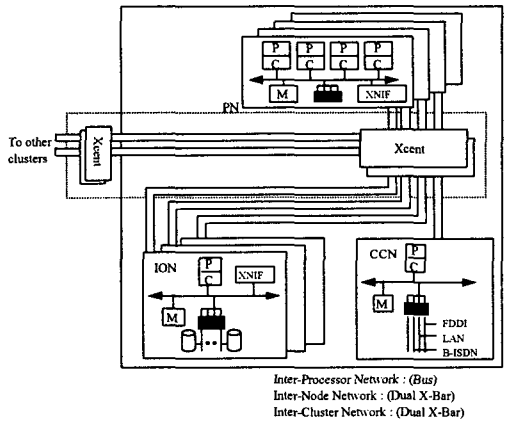

<그림 2> 테이블간의 관계

수를 의미하며, 데이터베이스의 크기를 나타내기 위해 물류창고(Warehouse)의 수에 대한 비율로 표시된다. TPC-C 트랜잭션은 신규주문(New-order), 대금지불(Payment), 주문현황(Order-status), 물품배달(Delivery), 재고현황(Stock-level)등 5가지 종류가 있다. 이들 각각의 종류를 자세히 살펴보면, 신규주문(New-order) 트랜잭션은 고객의 주문을 처리하는 트랜잭션이며, TPC-C 밴치마크의 부하 량(Workload)의 주된 요소가 되는 트랜잭션이다. 대금지불(Payment) 트랜잭션은 고객의 주문 물품에 대한 대금지불을 처리하는 트랜잭션으로 고객의 잔고(Balance)를 수정한다. 주문현황(Order-status) 트랜잭션은 고객이 자신의 마지막 주문 상태를 확인하는 트랜잭션이다. 물품배달(Delivery) 트랜잭션은 10개의 새로운 물품주문 트랜잭션에 의해 주문된 물품을 일괄적으로 배달하는 트랜잭션이며, 큐잉 메커니즘을 통한 지연모드(Deferred mode)로 실행되며 지연된 실행의 결과를 파일에 저장한다. 재고현황(Stock-level) 트랜잭션은 최근에 판매된 제품(Item)들 중에서 지정된 양(Threshold) 보다 작은 재고량(Stock level)을 가지는 제품의 수를 조사하는 트랜잭션이다.

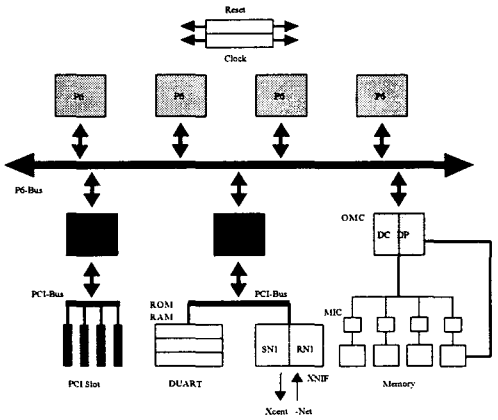
## III. 실험 환경

### 3.1 SPAX 시스템

고속 병렬 컴퓨터 SPAX(Scalable Parallel Architecture computer based on X-bar Network)는 한국전자통신연구소(ETRI)에서 개발되고 있는 고속 병렬 머신이다[1]. SPAX 시스템은 최소 1개, 최대 16개의 클러스터(Cluster)들로 구성된다. <그림 3>에 보여지는 것과 같이 각 클러스터는 한 개 이상의 프로세싱 노드(PN), 1개의 입출력 노




<그림 3> 클러스터의 구조

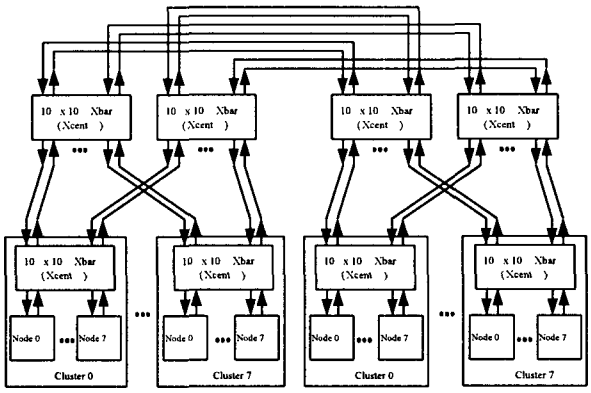



<그림 4> 프로세싱 노드의 구조

드(ION) 및 1개의 통신접속 노드(CCN)로 구성된 구조를 갖는다.

각각의 노드에 대하여 살펴보면, 프로세싱 노드(PN)는 마이크로프로세서와 공유 메모리를 내장한 한 공유 메모리 병렬 구조를 갖는다 <그림 4>. 한 개의 프로세싱 노드(PN)에는 4개의 Intel P6 마이크로프로세서가 탑재되어 있으며, 1개의 프로세서 당 512KB 이상의 캐쉬를 지원하며, 노드 당 최대 1GB의 메모리 용량을 갖는다. 버스 대역 폭은 최대 500MB/초 이상이며, 크로스바 연결망에 대한 접속회로(XNIF: Xcent-Net Interface)를 두어서 상호연결망에의 고속 접속을 지원한다. 입출력 노드는 1개의 Intel P6 마이크로프로세서 탑재, 대용량 데이터 버퍼는 전원 back-up 기능, 고속 DMA 제어기, 4개의 fast SCSI-II 버스 인터페이스 제어기 등으로 구성되어 있다. 통신접속 노드(CCN)는 상용 표준버스인 PCI 버스를 제공함으로써, 상용의 LAN, FDDI, B-ISDN, ATM등의 고속 통신 장치들을 접속할 수 있게 한다. 내부구성은 Intel P6 마이크로프로세서가 탑재되어있으며, 내부 RAM과 ROM을 가지고 있다. 그리고 버퍼 메모리, 상호연결망 인터페이스인 XNIF 및 VME64 버스 인터페이스 등으로 구성되어 있다.

Xcent-Net이라고 불리는 SPAX의 상호연결망은 클러스터 내의 노드 상호간 (Intra-cluster) 또는 서로 다른 클러스터간의 (Inter-cluster) 메시지 전송을 위한 통신 채널을 제공한다. Xcent-Net은 10×10 크로스바 스위치(Crossbar switch) 구조를 갖는 Xcent 스위치들로 구성된 계층적 이중 크로스바 연결망 구조를 갖는다. 그 구성은 <그림 5>에서 보여진다.




<그림 5> 시스템의 상호연결

SPAX시스템은 기본적인 사양으로 각 클러스터 당 4개의 프로세싱 노드(PN), 즉 16개의 프로세서가 장착되며, 클러스터 및 노드 단위로 확장이 가능하다. 시스템은 최대 16개의 클러스터까지 확장 가능하며, 각 클러스터는 최대 64개의 프로세싱 노드 및 최대 64개의 입출력 노드와 통신 접속노드를 갖는다.

## 3.2 성능 평가 방법

본 연구에서는 성능 평가 방법으로 SPAX 시스템의 하드웨어의 시뮬레이션을 이용하며 그 입력은 TPC-C 사양에 준하여 모의(Synthetic) 트랜잭션을 생성시켜 사용한다. 시뮬레이터는 시뮬레이터 개발 도구인 CSIM과 C언어를 사용하여 SUN사의 Enterprise상에서 개발하였으며 TPC-C의 구현 부분과 시뮬레이션 통계를 위한 부분으로 구성된다. TPC-C 사양에 따라 각 프로세서는 1개의 물품창고(Warehouse)에 매핑이 되며 그 물품창고(Warehouse)에 할당된 고객(Customer)에 대하여 트랜잭션을 수행하게된다. TPC-C의 5가지의 트랜잭션 중에 신규주문(New-order) 트랜잭션의 경우 전체 제품의 1%는 다른 프로세서에 할당되고, 대금지불(Payment) 트랜잭션의 경우 전체 고객(Customer)의 15%가 다른 웨어하우스에 할당되며 본 고에서는 이러한 트랜잭션을 원격(Remote) 트랙잭션이라 한다. 다음은 원격 트랜잭션의 구현내용을 보여주고 있다.

- 신규주문(New-order) 트랜잭션
  - 임의의 웨어하우스(프로세서) Pi에서 원격 트랜잭션이 발생하였을 경우, 그 해당

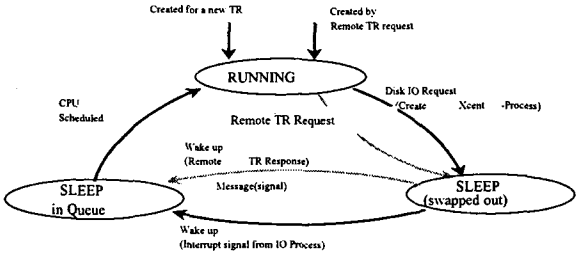
하는 물품(item)을 소유하고 있는 웨어하우스(프로세서) Pj에게 원격 트랜잭션을 위임한다. Pj는 트랜잭션을 처리하고 Pi에게 그 결과를 돌려준다.

- District_next_o_id의 갱신은 트랜잭션이 생성된 웨어하우스, 즉 $P_i$에서 일어난다.
- 주문(order)에 대한 기록은 트랜잭션이 생성된 웨어하우스, 즉 $P_i$의 New-order와 Order 테이블에 기록된다.
- Stock_quantity의 갱신은 트랜잭션을 실제로 처리한 웨어하우스, 즉 $P_j$에서 일어나며,
- 주문된 item에 대한 기록은 트랜잭션이 생성된 웨어하우스, 즉 $P_i$의 Order-line 테이블에 기록된다.

- 대금지불(Payment) 트랜잭션
  - 임의의 웨어하우스(프로세서) $P_i$에서 원격 트랜잭션이 발생하였을 경우, 그 해당하는 고객을 소유하고 있는 웨어하우스(프로세서) $P_j$에게 원격 트랜잭션을 위임한다. $P_j$는 트랜잭션을 처리하고 $P_i$에게 그 결과를 돌려준다.
  - 웨어하우스 테이블의 District_YTD의 갱신은 트랜잭션이 생성된 웨어하우스, 즉 $P_i$에서 일어난다.
  - Customer_balance의 갱신은 트랜잭션을 실제로 처리한 웨어하우스, 즉 $P_j$에서 일어난다.
  - History는 트랜잭션이 생성된 웨어하우스, 즉 $P_i$에 기록된다.

TPC-C의 시뮬레이션 모델은 5개의 트랜잭션 프로세스, 입출력 프로세스, 그리고 Xcent 프로세스로 구성되며 그 간략한 설명은 아래와 같다.

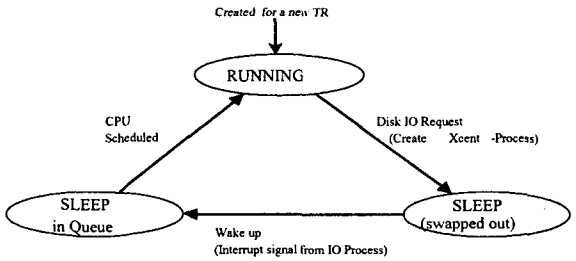
- 트랜잭션 프로세스들로는 신규주문(New-Order) 프로세스, 대금지불(Payment) 프로세스, 주문현황(Order-status) 프로세스, 배달(Delivery) 프로세스, 재고현황(Stock-level) 프로세스가 있으며 각 프로세스는 TPC-C의 해당되는 트랜잭션에 대한 시스템 상에서의 처리 과정을 시뮬레이션 한다. <그림 6>은 신규주문과 대금지불 프로세스의 상태 다이어그램을 보여준다. 이 두 프로세스는 구동(Running), 대기(Wait-in-queue), 휴지(Sleep-being-swapped-out)의 세 가지 다른 상태를 갖는다. 이 프로세스들은 새로운 트랜잭션을 시작하거나 다른 프로세서로부터 트랜잭션 요구를 받았을 때 생성되어 구동 상태에 놓이게 되며 트랜잭션의 수행 중에 디스크의 입출력이 필요하거나, 다른 프로세서에게 트랜잭션을 위임할 경우 휴지




<그림 6> 신규주문과 대금지불 프로세스

상태가 된다. 휴지 상태의 프로세스는 이후에 적절한 응답(원격 트랜잭션의 결과 메시지나 입출력이 끝났음을 알리는 인터럽트 신호)에 의해 깨어나며, 대기 상태가 된다. 큐에 있는 프로세스가 스케쥴되면, 그것은 구동 상태가 되어 트랜잭션의 나머지 부분의 처리를 계속하게 된다. <그림 7>은 주문현황, 배달, 재고현황 프로세스의 상태 다이어그램을 보여준다. 이 프로세스들도 구동, 대기, 그리고 휴지의 세 가지 다른 상태를 갖는다. 이 프로세스들은 기본적으로 앞의 두 종류의 프로세서와는 달리 다른 프로세서와의 상호 관련을 갖지 않는다. 그러므로 트랜잭션을 시작할 때 생성되어 구동 상태에 놓이게 되며 트랜잭션의 수행 중에 디스크의 입출력이 필요할 경우에만 휴지상태가 된다. 휴지 상태의 프로세스는 이후에 입출력이 끝났음을 알리는 인터럽트 신호에 의해 깨어나며, 대기 상태가 된다. 큐에 있는 프로세스가 CPU에 할당(스케쥴) 되면, 그것은 구동 상태로 되어 트랜잭션의 나머지 부분의 처리를 계속하게 된다.

- 입출력 프로세스는 입출력 노드의 디스크 입출력의 처리 과정을, Xcent-Net 프로세스는 Xcent-Net 상의 데이터 패킷의 전송 과정을 시뮬레이션 하기 위한 프로세스이다. 자세한 내용은 본 논문에서는 소개하지 않으며[2]에 자세히 설명되어 있다.




<그림 7> 주문현황, 물품배달, 재고현황 프로세스

# Ⅳ. 성능 평가 결과

본 절에서는 시뮬레이션을 통해 측정한 성능과 그 분석 결과를 기술한다.

## 4.1 시험 시스템 벤치마킹 환경설정

앞 절에서 살펴본 바와 같이 SPAX 시스템은 16 개까지의 클러스터를 가질 수 있다. 한편, 시뮬레이션의 입력 부하 량은 시뮬레이션의 시간을 결정하므로 적정한 SPAX 시스템의 크기 및 시뮬레이션의 입력 부하 량을 조사할 필요가 있다. 본 절에서는 대상 시스템의 성능 분석에 필요한 기본 시스템 환경을 조사하기 위해, 시뮬레이션 입력 량을 결정하는 터미널 당 트랜잭션의 수와 시스템의 크기가 시스템의 성능에 미치는 영향에 대한 실험을 수행하였다. 실험 시스템의 기본 사양은 SPAX 시스템의 초기 설계 값에 따라 XNIF의 패킷 송수신을 위한 SNI와 RNI에 각각 1개의 4-패킷 데이터 버퍼(Packet data buffer)를 갖으며, 패킷의 크기는 64 바이트로 정의되었다. 또한 각 입출력 노드의 디스크 드라이브의 개수를 2개로 설정하였다. 이러한 사양 하에서는 입출력 요구의 양이 많거나 디스크 시스템이 매우 빠른 속도로 데이터를 전송할 경우, 입출력 노드에서 병목현상(Bottleneck)을 발생한다. 이를 위하여 디스크 캐쉬 실패율을 90%로 설정하여 트랜잭션 당 입출력 요구의 확률이 비교적 높은 값을 갖도록 하였다. 먼저 시스템의 크기가 디스크 입출력과 TPC-C를 벤치마크로 사용할 경우 트랜잭션 처리 성능 값을 나타내 주는 TPS(Transaction Per Second)를 살펴본다. 실험 결과는 시스템의 크기가 변하더라도 디스크의 평균 큐 길이와 응답 시간은 거의 변화가 없으며, 아울러 TPS의 값도 거의 동일함을 알 수 있었다.

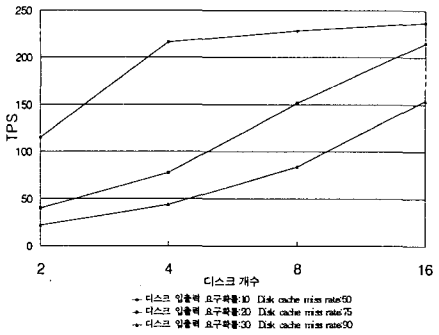
한편 시뮬레이션을 수행하기 위한 입력 량을 결정하는 터미널 당 적정한 트랜잭션의 수를 조사하기 위해 다양한 입출력 요구의 확률을 가진 시스템을 구성하여 터미널 당 트랜잭션의 수가 50, 100, 200, 400인 경우에 대해 실험을 수행하였다. 그 결과 값을 분석하면 터미널 당 트랜잭션의 수가 50이상일 경우 시스템의 성능은 터미널 당 트랜잭션에 거의 영향을 받지 않으며 단지 시뮬레이션 시간만 길어진다는 결과를 얻을 수 있다.

시스템의 크기는 시스템 각 요소의 성능에 거의 영향을 미치지 않는다는 실험 결과를 바탕으로 본고의 나머지 부문의 실험에서는 시스템의 크기를 1개의 클러스터를 갖

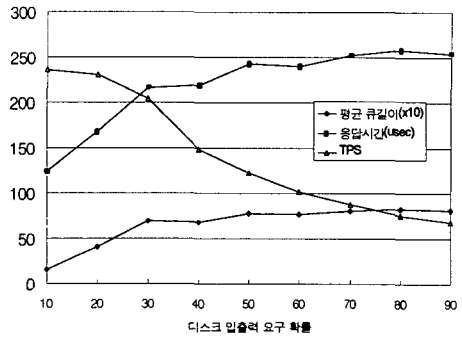
는 시스템으로 한정시키며, 불필요하게 긴 시뮬레이션 시간을 피하기 위해서 터미널 당 트랜잭션 수가 100으로 설정하여 성능 평가를 수행한다. 한편, 각 트랜잭션의 입출력 요구의 확률 값은 TPS 값과 디스크의 성능에 심각한 영향을 미치며 다른 과학적 벤치마크(Scientific Benchmark)와는 달리 데이터베이스 어플리케이션들은 지역성(locality)을 거의 갖지 않으므로 각 트랜잭션 당 평균 입출력 요구의 확률을 비교적 높은 값인 30%로 설정하여 실험을 수행하였다.

### 4.2 입출력 노드 디스크 개수

입출력 노드의 성능에 영향을 미치는 주된 요소는 입출력 노드의 프로세서 개수, 병렬 디스크의 개수와 디스크 입출력 요구 확률 및 디스크 캐쉬 접근 실패율 등이다. 입출력 노드의 프로세서 개수는 병렬 디스크의 개수는 각각 입출력 노드의 처리 용량과 디스크 입출력의 용량(Bandwidth)을 결정하며 시스템 구성 시 가변적으로 설정할 수 있는 사양이다. 한편, 디스크 입출력 확률 및 디스크 캐쉬 접근 실패율은 주어진 프로세싱 및 입출력 노드 및 주기억장치의 크기에 대하여 주로 응용프로그램의 특성에 의해 결정된다. 본 절에서는 시스템 구성 사양인 디스크 드라이브의 개수가 주어진 응용프로그램의 특성에 의해 결정되는 사양인 입출력 요구 확률 및 디스크 캐쉬 접근 실패율 하에서 시스템의 성능에 미치는 영향을 평가한다. <그림 8>에서 볼 수 있듯이, TPS 값이 최대 임계치(Threshold)를 넘기 전에는 디스크 드라이브의 수가 증가함에 따라 디스크 드라이브의 평균 큐의 길이와 평균 응답 시간은 선형적으로 감소하며 TPS의 값은 증가한다. 디스크 입출력 요구 확률이 10%, 디스크 캐쉬 접근 실패율이 50%일 경우 즉, 입출력 노드에 부하가 적게 걸릴 경우에는 디스크 드라이브 개수가 4 이상이면 TPS의 값이 거의 변화가 없으므로 TPC-C 벤치마크의 입출력 요구를 충분히 처리할 수 있다고 볼 수 있다. 디스크 입출력 요구 확률이 30%, 캐쉬 접근 실패율이 90%일 경우 즉, 입출력 노드에 부하가 많이 걸릴 경우에는 디스크 드라이브의 개수가 16일 때까지 큐의 길이와 TPS값이 급격히 변화한다. 이 경우에는 각 입출력 노드당 16개의 디스크 드라이브를 가지는 시스템만이 빈번한 입출력 요구를 충분히 처리할 수 있다고 볼 수 있다. 나머지 세 번째 경우에는 디스크 드라이브의 수가 증가함에 따라 평균 큐의 길이는 감소하며 TPS는 증가하지만 입출력 노드 당 8개의 디스크 드라이브를 가지는 시스템이면 충분히 입출력 요구를 처리할 수 있다고 볼 수 있다. 이러한




<그림 8> 디스크 개수와 Disk cache miss rate의 영향




<그림 9> 입출력 노드의 데이터 버퍼의 성능

성능 분석 결과는 SPAX 시스템이 최대의 성능을 발휘하기 위해서는 시스템 설계 시 시스템의 디스크의 입출력 요구 확률 및 디스크의 캐쉬 접근 실패율 등 응용 프로그램의 특성에 적합하도록 입출력 서브시스템의 디스크 드라이브의 개수를 결정하여야 한다는 점을 알려준다.

### 4.3 입출력 노드의 XNIF의 버퍼 개수

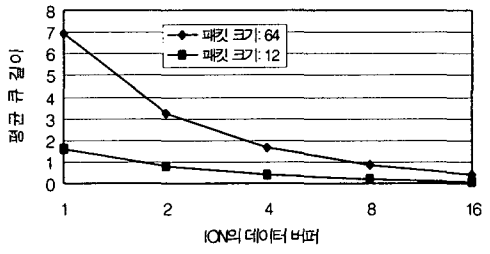
입출력 노드의 상호연결망과의 접속에 관련된 XNIF의 성능 평가를 위해 먼저 SPAX 시스템의 기본 설계 사양에 따라 각 입출력 노드에 1개의 4-패킷버퍼를 갖는 시스템을 구성하여 성능 평가를 수행하였다. 입출력 노드의 XNIF 및 상호연결망의 여러 요소의 성능을 조사는 디스크 접근에 따른 병목현상의 영향을 배제하기 위해 처리될 데이터가 항상 디스크 캐쉬에 존재하여 디스크 액세스가 필요하지 않으며(디스크 캐쉬 접근 실패율 = 0%), 입출력 노드 당 디스크 드라이브의 수는 무한하다고 가정한다. 이러한 설정 하에서 디스크 입출력 요구 율을 변화시켜 얻는 다양한 입출력 요구량에 대하여 수행한 시스템의 성능 평가 결과가 <그림 9>에 보여진다.

TPS의 값은 디스크 입출력 요구 확률이 30%에서 40%로 증가함에 따라 204.1에서 148.6으로 급격히 감소함을 볼 수 있다. 이러한 성능 저하원인을 분석하기 위하여 <그림 9>에 보여지는 바와 같이 입출력 노드내의 XNIF의 데이터 버퍼의 평균 큐 길이와 평균 응답 시간을 조사한 결과, 디스크 입출력 요구 확률이 10%에서 70%로 증가함에 따라 데이터 버퍼의 큐의 길이와 응답 시간은 선형적으로 증가하며 80%이상일 경우에

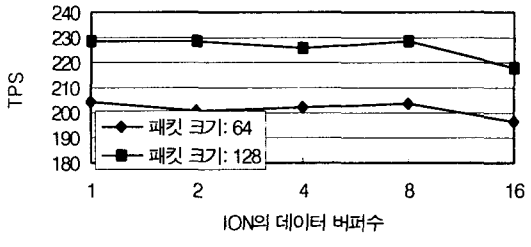
는 거의 변하지 않는다. 이 결과는 입출력 요구가 증가할 경우에는 SPAX 시스템의 기본 사양으로 정해진 1개의 4-패킷 데이터 버퍼에 병목 현상 초래되어 전체 시스템의 성능 저하의 주된 원인이 될 수 있음을 시사한다. 그러므로 이러한 환경에서는 입출력 노드의 데이터 버퍼수의 증가가 요구된다.

입출력 노드의 데이터 버퍼의 개수를 증가시킬 때 얻을 수 있는 시스템의 성능향상을 측정하기 위하여 다양한 패킷 크기에 대하여 데이터 버퍼의 수가 2, 4, 8, 16개의 각 경우의 데이터 버퍼의 평균 큐의 길이와 평균 응답 시간, 및 TPS의 값을 조사하였다. 각 경우 데이터 버퍼의 크기는 패킷의 크기와 동일하다고 가정하며, 또한 앞 절에서 논의한 바와 같이 디스크 접근(Disk access)이 성능에 미치는 영향을 배제하기 위해 처리될 데이터는 항상 디스크 캐쉬에 존재하며(디스크 캐쉬 접근 실패율 = 0%), 디스크 드라이브의 개수는 무한하다고 가정한다. 디스크 드라이브의 개수가 증가함에 따라 입출력 노드의 데이터 버퍼의 평균 큐의 길이는 짧아지지만, 예상한 것과는 달리 시스템 성능인 TPS 값과 트랜잭션의 평균 응답 시간은 거의 변하지 않았다 (<그림 10, 11>). 이러한 이유를 살펴보기 위해 패킷이 전송되는 경로상의 다른 시스템 요소들의 평균 큐의 길이를 조사하였다. (<그림 12>)

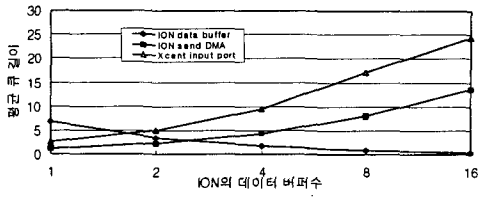
입출력 노드의 데이터 버퍼의 개수가 증가함에 따라 데이터 버퍼의 평균 큐의 길이는 감소하였지만, 반대로 디스크 캐쉬로부터 데이터 버퍼로 데이터를 전송하는 역할을 수행하는 시스템 요소인 DMA의 평균 큐의 길이가 증가함을 볼 수 있다. 그리고 한번에 한 개의 패킷을 저장할 수 있는 Xcent 스위치의 입력 포트는 증가된 다수의 입출력 노드의 데이터 버퍼에 의해 공유되므로 평균 큐의 길이가 급격히 증가하였다. 입출력 노드에서 데이터 버퍼가 이용 불가능 할 경우에는 데이터의 전송이 발생하지 않으므로 PCI 버스는 그 동안 사용되지 않는다. 따라서 PCI 버스의 평균 큐의 길이는 거의 일정함을 볼 수 있다. 그러므로, 입출력 노드의 Xcent 스위치의 병목현상은 데이터버퍼에서



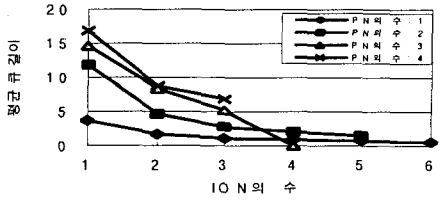

<그림 10> 입출력 노드의 데이터 버퍼의 영향(I)




<그림 11> 입출력 노드의 데이터 버퍼의 영향(2)




<그림 12> 입출력 노드의 데이터 버퍼의 영향(II)

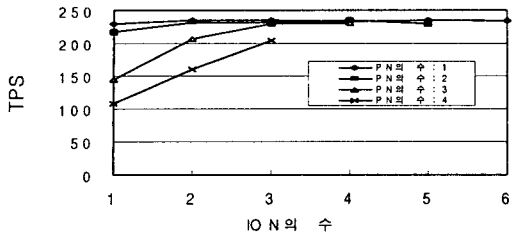



<그림 13> 프로세싱 노드와 입출력 노드의 처리 능력의 영향(I)

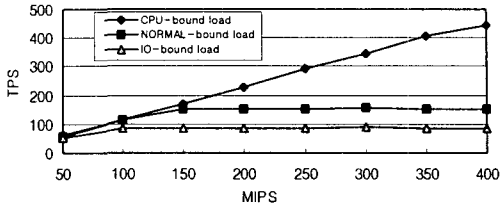
DMA로 그 위치를 변경하였을 뿐 계속 존재하기 때문에 전체 시스템의 성능은 향상되지 않음을 알 수 있다. 이 실험 환경 하에서는 결과를 통해 패킷의 전송 경로인 프로세싱 노드의 수신 데이터 버퍼(Receive data buffer), DMA, PCI 버스에는 병목 현상이 존재하지 않음도 확인할 수 있었지만, 입출력 노드의 DMA와 Xcent 스위치의 입력 포트의 병목현상을 해결한 상황에서는 패킷 전송 경로상의 다음 연결 요소에 병목현상이 발생할 것임을 예상할 수 있다. 본 절의 실험 결과는 입출력 노드의 데이터 버퍼의 처리능력을 증가시킨다 할지라도, 패킷 전송 경로상의 관련된 그 외의 요소들의 처리 능력의 증가시키지 않고서는 시스템 전체의 성능 향상을 기대할 수 없음을 보여준다.

### 4.4 입출력 노드의 개수

본 절에서는 트랜잭션 처리를 수행하는 프로세싱 노드의 성능을 분석하기 위해 프로세싱 노드의 개수와 CPU의 속도가 시스템에 미치는 영향에 대한 성능 분석 결과를 기술한다. 프로세싱 노드와 입출력 노드간의 처리 능력의 관계를 조사하기 위해 프로세싱 노드와 입출력 노드의 수가 다양한 시스템을 구성하여 성능평가를 수행하였다. <그림 13>, <그림 14>은 입출력 노드의 데이터 버퍼의 평균 큐의 길이와 응답 시간을 나타낸다. 프로세싱 노드의 수가 2일 경우까지는 입출력 노드의 수에 관계없이 TPS의 값이 거의 일정하며, 프로세싱 노드의 수가 3개 이상일 경우에는 입출력 노드의 수에 따라 TPS의 값이 급격히 변함을 볼 수 있다. 예를 들면, 프로세싱 노드의 개수가 4일 경우, 입출력 노드의 개수가 1일 때는 TPS의 값이 107.417인 반면, 입출력 노드의 개수가 3일 때는 204.122로 TPS의 값이 거의 2배정도 증가함을 볼 수 있다. 이것은 프로세싱 노드의 처리능력이 충분하지 않을 경우에는 시스템의 전체 성능이 입출력 노드의 처리능력에 거의 영향을 받지 않으나, 프로세싱 노드의 처리능력이 충분할 경우에는 시




<그림 14> 프로세싱 노드와 입출력 노드의 처리 능력의 영향(II)




<그림 15> CPU 속도의 영향

스템의 전체 성능이 입출력 노드의 성능에 의해 좌우됨을 의미한다. 그러므로 병렬 트랜잭션 처리 시스템의 설계 시 CPU당 최대의 성능을 얻기 위해서는 프로세싱 노드와 입출력 노드간의 처리능력이 균형을 이루도록 신중히 고려해야 함을 알 수 있다.

## 4.5 CPU 클락 속도

CPU의 클락 속도가 시스템의 성능에 미치는 영향에 대해 분석하기 위해 시스템의 구성을 다음과 같이 CPU-bound load, IO-bound load, Normal-bound load로 구분하여 성능 평가를 수행하였다.

- CPU-bound load: 디스크 입출력 요구 확률 10%, 디스크 캐쉬 접근 실패율 50%
- Normal-bound load: 디스크 입출력 요구 확률 20%, 디스크 캐쉬 접근 실패율 75%
- IO-bound load: 디스크 입출력 요구 확률 30%, 디스크 캐쉬 접근 실패율 90%

<그림 15>는 CPU의 속도가 증가함에 따라 TPS의 값이 증가함을 보여준다. CPU-bound load일 경우의 TPS 값의 증가율은 IO-bound load일 경우의 증가율 보다 훨씬 큰 값을 가지는 것을 볼 수 있다. 이는 CPU의 속도가 증가함에 따라 CPU-bound load일 경우에는 시스템의 성능에 큰 영향을 미치지만, IO-bound load일 경우에는 시스템의 성능에 작은 영향을 미친다는 것을 의미한다. 따라서 시스템의 처리 능력을 향상시키기 위해 CPU의 속도를 증가시키고자 할 경우에는, 먼저 사용하는 프로그램의 특성을 CPU-bound load나 IO-bound load의 관점에서 분석한 후 시스템의 CPU 속도를 결정하여야 함을 의미한다.

# V. 결 론

본 연구에서는 병렬 트랜잭션 시스템 중 TPC-C를 사용하여 한국전자통신연구소(ETRI)에서 개발된 고속 병렬 컴퓨터 SPAX(Scalable Parallel Architecture computer based on X-bar network) 시스템에 대한 성능 분석을 수행하였다. 성능 분석은 주로 입출력의 성능에 주안점을 두고 수행되었으며, 주어진 병렬 디스크의 개수에 대하여 디스크 입출력 요구의 확률이 증가하여 시스템의 입출력 노드에 부하가 많이 걸릴 경우 시스템의 성능이 급격히 감소하기 때문에 병렬 디스크 개수를 증가시켜 디스크 데이터 전달 용량(Bandwidth)을 증가시켜야 함을 보여준다. 시스템의 성능향상을 위한 입출력 노드의 데이터 버퍼의 수와 Xcent 스위치의 입력 포트의 버퍼수의 증가는 데이터 버퍼 처리 용량은 증가하였지만 패킷 전송 경로상의 다른 시스템 요소의 처리 능력이 불충분해짐으로서 전체 시스템의 성능에는 거의 영향을 미치지 못하였다. 따라서 전체 시스템의 성능향상을 위해서는 패킷 전송 경로상의 모든 요소들의 성능 향상이 동시에 일어나야 함을 제시한다. 또한 본 연구에서는 시스템의 성능향상을 위해 CPU 속도를 증가시키고자 할 경우에는, 먼저 시스템의 용도를 CPU-bound load나 IO-bound load의 관점에서 분석한 후 시스템의 CPU 속도를 결정해야 함을 보여주며, SPAX 시스템과 같은 병렬 트랜잭션 시스템의 설계에 있어서 프로세싱 노드와 입출력 노드 사이의 처리 능력의 균형은 CPU당 최대의 성능을 얻기 위해서 필수적임을 보여준다. 본 연구에서는 TPC-C 벤치마크를 사용하여 대상 시스템의 성능을 분석함에 있어서 모의 입력을 통한 성능분석 방식을 채택하고 있다. 그 분석 결과는 병렬 트랜잭션의 설계에 있어 귀중한 많은 정보를 제공해 주고 있지만, 향후 보다 정확한 성능분석을 위해서는 실제 데이터베이스 시스템을 구성하여 보다 성능평가가 수행되어야 할 것이다.

# 참 고 문 헌

[1] 한종석, 박경석, 한우종, 심원세, "SPAX 시스템의 Xcent 크로스바 라우팅 스위치 설계", 한국정보과학회 추계학술논문집, Vol. 23, No. 2, pp. 1649-1655, 1996년 10월

[2] 정원교, 김희철, 이용두, "병렬 컴퓨터의 데이터베이스 벤치마크 성능분석", 한국처

리학회 추계학술논문집, Vol. 4, No. 2, pp. 1424-1429, 1997년 10월

[3] Computer Architecture Research Lab. of ETRI, "SPAX Hardware Subsystem Design Specification (V. 1.0)"

[4] D. DeWitt and J. Gray. Parallel Database Systems: The Future of High Performance Database Systems. Communications of the ACM, 35(6):85-98, June 1992.

[5] J. Gray, "The Benchmark handbook for Database and Transaction Processing Systems." Second Edition, Morgan Kaufmann, 1993.

[6] T. Lovett and R. Clapp. STiNG: A CC-NUMA Computer System for the Commercial Marketplace. In Proceedings of the 23rd Annual International Symposium on Computer Architecture, pages 308-317, May 1996.

[7] K. Lee, M. Dubois, Performance Evaluation of Parallel Transaction Processing in SPAX, Technical Report #1, University of Southern California

[8] R. Marek, E. Rahm, "Performance Evaluation of Parallel Transaction Processing in Shared Nothing Database Systems," Proc. of PARLE, May 1992, pp. 295-310.

[9] S. Tharkar and M. Sweiger. Performance of an OLTP Application on Symmetry Multiprocessor System. In Proceedings of the 17th Annual ISCA, pages 228-238, May 1990.

[10] S. Woo, M. Ohara, E. Torrie, J. P. Singh, and A. Gupta. The SPLASH-2 Programs: Characterization and Methodological Considerations. In Proceedings of the 22nd Annual ISCA, pages 24-36, July 1995.