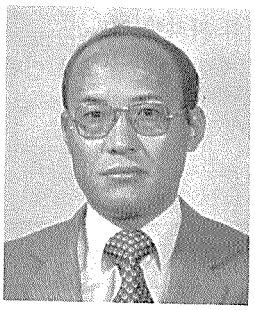

吳 吉 祿
韓國電子通信研究所
컴퓨터연구부장 / 工博

공유 기억장치에 의한 UNIX 다중 프로세서의 성능 및 상용 응용

1. 머리말

다중 프로세서에 의한 컴퓨터 시스템이 학교 및 연구소 등에서 개발되고 실험된지는 오래되었다. 그러나, 실제 IBM이나 기타 대형 컴퓨터 및 중형 컴퓨터를 대신하여 값싸고 더 좋은 성능으로 다중 프로세서를 공급하겠다는 시도는 그렇게 오래되지 않았다. 특히 UNIX 운영 체제가 상업 응용 분야에 적용되기 시작한 것이 최근의 일이며, 이와 같은 UNIX를 사용하여 다중 프로세서를 상용화하기 시작한 것도 최근 2～3년 전의 일이다.

다중 프로세서가 연구나 실험용이 아닌 실제 DBMS 및 상업 응용에 쓰일 수 있도록 하는 시도는 학문적인 논란보다는 실제 고려해야 할 다른 면이 상당히 많다. 즉, 학문적으로는 아키텍쳐상 상당히 고전적인 것이나 실제의 응용을 위한 상용 세계에서는 아직도 초창기의 개척 단계라 할 수 있다.

본 글에서는 이 중 공유 기억장치에 의한 다중 프로세서(Shared Memory based Multiprocessor)가 이와 같은 기존 전통적인 응용분야를 값싸고 고성능으로 도전하기 위한 적합한 구조라는 점, 이에 대한 성능, 그리고 이와 같은 접근을 하고 있는 미국의 회사들을 중심으로 조사하여 검토하고자 한다. 본 주제를 위한 고려는 그 동안의 자체 검토, 1988년 2월 8일부터 13일까지 열린 UNIFORM 및 USENIX, 그리고 그 다음 주에 시행된 미국 회사의 방문으로 결론을 확정짓게 되었다.

본 글에서 대답하고자 하는 의문 사항들은 다음과 같다.

○공유 기억장치에 의한 다중 프로세서의 성능이 UNIX운영 체제하에서 CPU가 10－20개 또는 그 이상일 때 경제적으로 증가하는가?

○공유 기억장치에 의한 다중 프로세서 아키텍쳐가 상업 응용 또는 과학용 응용 등에 적합한가?

○위와 같은 구조를 가진 컴퓨터 회사가 기존 중형, 대형 컴퓨터 회사에 대항하여 성공적으로 자라고 있는가?

○몇 십개 단위의 CPU를 가진 다중 프로세서인 경우 기존의 중대형 컴퓨터를 대치할 경우 공유 기억장치가 유리한가 또는 메시지 교환 방식의 로컬 메모리(local memory) 방식이 유리한가?

이와 같은 질문에 대해 다음의 장에서 차례로 분석, 검토하고자 한다.

2. 다중 프로세서의 분류

다중 프로세서 시스템에는 여러가지 분류가 있으나 현재 상업 응용에 쓰일 수 있는 현실 시스템을 고려하여 여기서는 편의상 단일 버스 형식의 세가지를 언급한다[5]. 그림 1은 그 구조를 간단히 나타내고 있다. 그림중 공유 기억장치에 의한 다중 프로세서(a)는 여러 개의 CPU가 기억장치 및 입출력 장치를 공유한다. 로컬 메모리에 의한 다중 프로세서(b)는 공유 기억장치가 없고 각 CPU가 각각의 로컬 메모리를 가지고 있고 입출력 등은 시스템 버스를 통하여 공유 또는 각 CPU 보드에서 직접 지역적으로 관장한다. 분산처리 시스템(c)은 CPU, 기억장치, 입출력 등을 각각 지역적으로 가지고 있되 소프트웨어적으로 화일 및 입출력 장치를 하나처럼 보이도록 제공한다.

이외에도 특별 응용을 위한 여러 종류의 인터코넥션 버스 및 아키텍처가 있을 수 있다. Gordon Bell은 이를 그림 2와 같이 분류했다. 그의 주장은 한 copy의 운영 체제와 여러 개의 CPU가 존재하는 공유 기억장치에 의한 다중 프로세서만이 범용 시스템의 응용에 쓰일 수 있다는 것이다[1].

3. 공유 기억장치 다중 프로세서의 성능

본 글에서 다루려는 주제는 특수 분야의 특수응용에 관한 성능이 아니고 이미 IBM, DEC 등의 기존 컴퓨터들이 가지고 있는 시장(UNIX 계열)의 응용 프로그램 등을 수행할 다중 프로세서의 성능에 관한 내용이다. 그러므로 CPU수가 100개또는 1,000개 등의 단위를 이야기하는 것이 아니고

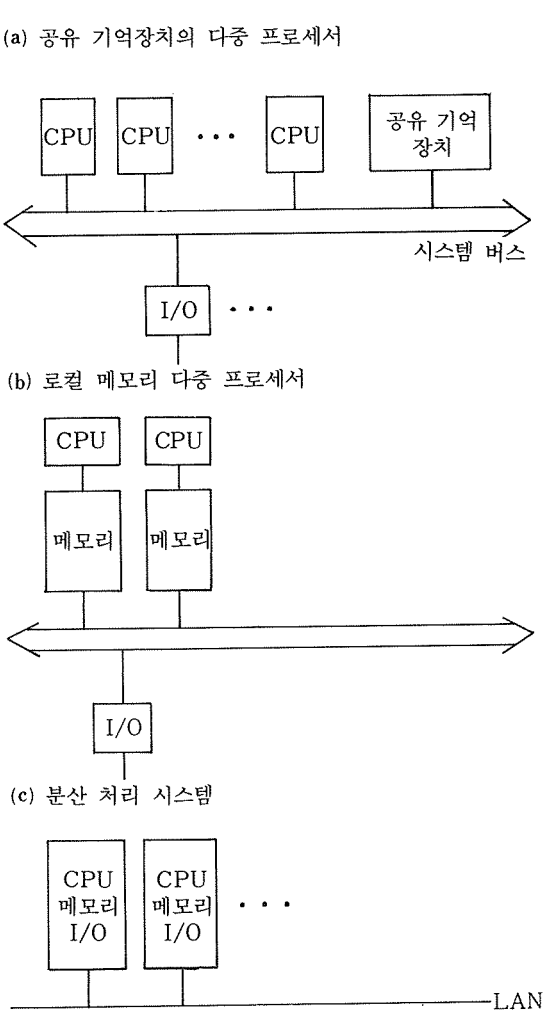

그림 1. 다중 프로세서의 종류

(이 경우 기존 시장의 응용을 수용할 범용성은 상실) 현실적인 10개 정도 또는 20-30개 등의 수십개 단위를 언급한다. 즉 CPU 수가 100개 이상의 경우는 현재의 기술로는 공유 기억장치에 의한 다중 프로세서 방식의 한계가 예견되나, 최근 점차적으로 단일 CPU에 의한 중대형 컴퓨터가 다중 프로세서로 변화됨에 따른 공유 기억장치 다중 프로세서의 적합성을 고려하고자 한다.

가. 성능의 종류

대개 한 프로그램에 대한 속도가 얼마나 증가할 수 있는가를 측정하는 병렬 처리(parallel processing)에 따른 성능과 사용자의 증가(multiprocessing, multi-user)에 따른 성능(throughput)으로 나눌 수 있다. 대부분 범용 컴퓨터가 여러 사용

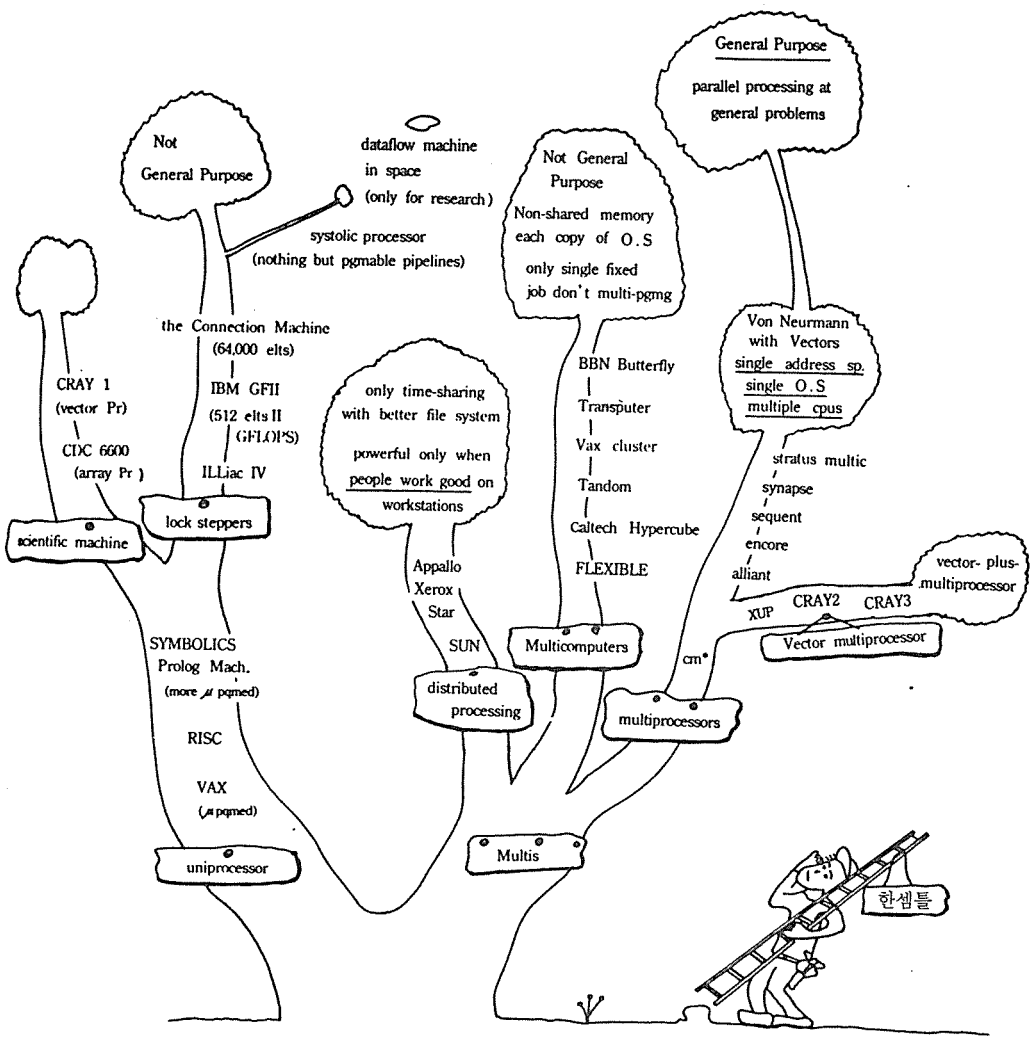

그림 2. Gordon Bell의 다중 프로세서 분류

자가 동시에 쓰고 있으므로 병렬 처리만을 필요로 하는 요구 사항은 일부에 지나지 않는다는 것이 방문한 컴퓨터 회사들의 의견이다. 예로 배치 프로세싱의 경우도 여러 배치가 같이 쓰이거나 다른 사용자의 다른 프로그램과 함께 쓰이지 단독 사용(standalone)으로 쓰이는 경우는 극히 드물다는 것이다. 결국 성능 평가는 여러 사용자에 의한 throughput을 나타내는 것이 대부분이었다.

나. 공유 기억장치의 다중 프로세서에서 CPU의 수에 따른 성능 저해 요소를 계층별로 보면 다음과 같이 구분할 수 있다.

1) Amdahl의 법칙

한 프로그램 입장에서만 보았을 경우 CPU의 증가에 따라 단일 프로세서의 경우보다 몇 배나 빠른가를 나타낸다. 프로그램내의 serialization해야 되는 부분의 비율을 X라 할때 아무리 CPU 수가 많아도 1/x이상 빠를 수 없다는 것이다. 즉, 다중 프로세서를 single 사용자의 single 프로그램용으로 사용할 경우 이 문제가 적용된다. 그러나 UNIX 다중 프로세서는 앞서 언급하였듯이 여러 사용자 및 여러 타스크(multi-tasking)의 일을 하게 되므로 이와 같은 경우에 의한 성능 저하를 고려해야 할 경우는 극히 저조할 것으로 예측된다.

2) 하드웨어 요소

하드웨어의 요소로는 버스에 걸리는 성능 한계를 들 수 있다. 이것은 버스의 속도와 캐쉬의 구현에 관계되는데, 지금까지의 상용 다중 프로세서에서는 약 30개까지의 상용 마이크로 프로세서를 이용한 CPU를 허용하고 있다. 예로 Intel 80386 CPU를 사용하고 있는 SEQUENT사의 시스템 버스 속도는 80 Mbyte/sec이며 기타 제어를 위한 보조 버스로 SLIC이라는 5 MB/sec를 따로 쓰고 있다. 이와 같은 구성으로 30개까지의 CPU를 구성할때 실제 버스 사용률이 60-70%라고 하고 있다. Encore사의 Multimax 시스템의 경우도 20개까지의 CPU(NS32332 또는 32532 계획)를 허용하는 100 MByte/sec의 시스템 버스를 사용하고 있고 버스 사용도는 30%이하라 주장하고 있다. 즉, 버스의 contention을 줄이기 위해 더 빠른 버스 및 더 좋은 캐쉬 알고리즘을 쓰고 있는데 write-back을 쓰는 SEQUENT의 경우 hit ratio가 95-98%를 기록하고 있다.

즉, 공유기억장치에 의한 다중 프로세서의 경우 사용할 수 있는 프로세서의 수는 시스템 버스의 가속화 및 좋은 캐쉬 설계에 좌우된다고 할 수 있는데 현재까지의 미국 등의 추세로는 10-30개 까지는 별 문제가 없이 출하되고 있다.

3) 소프트웨어 요소

아무리 하드웨어가 여러 개의 프로세서 수를 문제없이 제공한다해도 운영체제의 커널 부분에서 여러 개의 프로세서들이 공유 기억장치의 critical region을 서로 접근하기 위해 발생하는 serialization이 크다면 소프트웨어 병목현상이 발생할 수 있다. UNIX는 원래 다중 프로세서를 염두에 두고 만들지를 않았으므로 커널 부분을 잘 설계하지 않으면 쉽게 병목 현상이 있을 것이라는 것이 일반적인 생각이다. 이 부분에 대한 자세한 기술적인 언급은 다음 장에서 한다. 다만 공유 기억장치에 의한 UNIX의 다중 프로세서 시스템을 제작 공급하고 있는 많은 회사가 커널 부분의 데이타 구조 등을 변경함으로써 소프트웨어 병목 부분을 5% 정도로 낮추었고, 실험 및 벤치 마크에서도 이를 입증하고 있다.

4. UNIX운영 체제 커널의 다중 프로세서 운영 체제화

앞 장에서 언급한 바와 같이 공유 기억장치 내의 critical region을 여러 프로세서들이 접근하려할 때는 그 충돌을 방지하기 위해 spin lock 또는 P, V세마포(semaphore)를 이용하여 한 프로세서만이 사용할 수 있게 한다. 이에 의한 오버 헤드(overhead)가 얼마나 되는가 하는 것이 소프트웨어의 병목현상이 될 수가 있다.

전통적인 옛 버젼의 UNIX 시스템은 일반적으로 커널 모드의 비율이 40-50%인 것으로 알려져 있다[2]. 즉, 한 프로세서가 커널을 접근할때 다른 프로세서는 기다려야 했다. 그러므로 다중 프로세서 시스템을 master/slave양식으로 실현할 경우 시스템 모드의 비율이 프로세서 증가에 따라 훨씬 더 늘어나 실제적인 다중 프로세서의 효과는 별로 늘지 않아왔다. 즉 master/slave양식에서는 커널 전체를 한 덩어리로 locking, unlocking의 단위로 함으로써 적게 배정된 사용자 모드 부분만 여러 프로세서가 동시에 할 수 있었다.

현재 제품화되어 나오고 있는 symmetric 다중 프로세서 시스템은 master/slave 형식과는 전혀 다르다. 즉, 커널 부분도 여러 개의 프로세서가 접근을 하되 critical region만 세마포를 통해 조절하고 있기 때문에 커널 대 사용자 모드의 비율은 소프트웨어 병목을 측정하는데는 관계가 없게 된다. 즉, 커널내의 프로세서 테이블 등의 공유 구조를 여러 프로세서가 접근할때 얼마나 병렬적으로 액세스할 수 있게 해줄지를 결정하는 알맹이의 크기(granularity)를 정하는 것이 가장 큰 문제가 된다. 기존 UNIX의 커널은 다중 프로세서용이 아니기 때문에 커널 데이타 구조의 변경 등을 통해 symmetric 다중 프로세서가 가능한 최대로 동시에 접근할 수 있도록 설계를 하고 있다.

다중 프로세서를 제작하는 회사마다 구현의 방법이 조금씩 차이가 있으나 커널의 이와 같은 세마포에 의한 직렬성(serialization) 때문에 생기는 오버헤드는 5% 미만으로 주장하고 있었다. 즉, 공유 기억장치 때문에 생기는 소프트웨어 병목 현상은 상당히 적다는 것이다.

이번 USENIX에 발표된 DEC에서의 실험에서

는 VAX8300의 여러 CPU를 이용해 만든 공유 기억장치에 의한 symmetric 다중 프로세서의 경우 커널의 다중 프로세서화에 의한 오버 헤드는 3% 이하라고 하고 있다[3]. 참고로 그들이 구현한 spin lock들의 contention rate를 보면 표-1과 같다.

5. 벤치마크 실험

다중 프로세서의 성능을 측정하기 위한 적합한 벤치 마크 프로그램이 없다는 것과 벤치 마크 자체로써 성능 평가가 어렵다는 점 등이 그동안 문제로 지적되어 왔다. 최근에는 새로운 벤치 마크 회사들이 등장하여 이러한 점들을 극복하려 하고 있는데, 비지네스 응용 분야의 벤치마크를 위한 Neal Nelson & Associates의 Business Benchmarks와 AIM Technology사의 Benchmark Suite Ⅲ 등이 대표적인 예이다[4].

표-1

Lock	grabs (1000s)	conflict rate (%)	Average spins
process stats	630	0.03	1.0
time-outs	460	0.38	1.1
time	230	0.01	1.0
fs buffer pool	290	0.94	3.5
free page list	220	1.48	1.5
CPU scheduler	180	1.37	1.4
vm segment	130	0.04	3.0
copymap	100	0.62	2.3
TCP chains	70	0.04	2.4
biodone	70	0.07	2.7
disk	60	2.61	13.5
bus map	60	0.00	0.0
mbuf ppool	40	0.13	4.4
gnode pool	36	0.26	1.9
interface queue	29	0.00	0.0
kmemall	26	0.05	1.0
namei cache	25	0.17	1.3
swapmap	24	0.16	2.8
file table	19	0.00	0.0
socket	17	0.66	8.2
file system struct	14	0.11	5.0
proc table	14	0.00	0.0
credential pool	13	0.00	0.0

앞의 회사들은 벤치 마크의 타입을 18가지 정도로 나누어 (Nelson사의 경우) 경우마다 측정을 하고 또한 job이 증가함에 따라 성능이 얼마나 달라지는가를 결과로 보여주고 있다. 공유 기억장치에 의한 다중 프로세서에 대해서도 벤치 마크 테스트를 수행하고 있는데 이번 출장에서 얻은 데이타는 (Nelson사의 경우) 각 테스트마다 job이 증가할 때에 따른 수행속도와 같은 상태에서 CPU 수가 늘어갈 때의 상태를 나타낸 것이다. 이번의 데이타가 모든 경우의 수를 다 나타낸 것이 아니나, 그 중 다음의 세가지에 대해 각각 CPU의 증가에 따른 성능비를 알아 보았다.

Test 2 : 16,32 bit 및 64 bit 계산 function call, 메모리 할당 등을 포함한 15,000 cycle의 loop 테스트.

Test 3 : sequential 및 random으로 길고 짧은 레코드를 디스크 I/O를 통해 읽고 쓰는 기능을 포함한 250 cycle loop.

Test 7 : double precision floating-point 계산의 4연산(add, subtract, multiply, and divide)를 포함한 25,000 cycle loop.

Test 3의 경우는 디스크 I/O이므로 메모리를 늘리거나 디스크 콘트롤러를 늘임으로써 성능이 급격 향상될 수 있으므로 CPU의 증가 수와는 조금 다른 면을 보일 수 있으나, 그 개괄적인 비교표는 a, b, c, d 및 e와 같다. 편의상 job수 증가에 따른 세부표는 생략하고 2개 CPU의 몇 배인지만 평균으로 계산하였다.

Neal Nelson사가 UNIFORM에서 제공한 데이타에 의한 개괄이 표-2인 것에 비해 AIM Technology사의 방문으로 들어본 그들의 경험도 거의 비슷한 의견이었다. 예로서 Sequent사의 벤치 마크 시험 경우를 보면 CPU 증가에 따른 성능의 증가 결과에 대해 그들도 놀랐다고 한다.

6. 다중 프로세서 사례

다음은 UNIFORM 및 각 회사 방문에 따른 자료 정리이다.

접촉한 모든 회사는 공유 기억장치에 의한 다중 프로세서 구조의 성능에 아무 문제가 없다고 대답

했다. Sequoia사는 공유 기억장치에 의해 현재 16개 CPU를 사용하고 있는데 앞으로 64개까지 늘릴 예정이라 한다(UNIX 커널을 거의 다시씀). Arete사는 현재 4개 CPU를 가지고 있는데 앞으로 8개로 늘릴 예정이라 한다. 이 경우 UNIX 커널은 별로 바꾸지 않았다고 한다. Masscomp사도 8개까지의 CPU를 제공할 예정이며, 성능을 8배까지 하는데 문제없다고 한다. Alliant사도 8개의 CPU를 가지고 있으나, 일반 용도와는 다른 특수 구조를 가지고 있었다.

표-2

(a) Encore의 Multimax

	2 CPUs	10CPUs (2개 CPU의)	16CPUs (2개 CPU의)
Test 2	기준	4-5배	5-6배
Test 3	기준	1.5-2.5배	4-7배
Test 7	기준	4-6배	6 배

(b) Sequent

	2 CPUs	4 CPUs (2개 CPU의)	10CPUs (2개 CPU의)	16CPUs (2개 CPU의)
Test 2	기준	2 배	5-7배	5-7배
Test 3	기준	2 배	4 배	배
Test 7	기준	2 배	5-7배	6-10배

(c) Pyramid

	2 CPUs	3 CPUs (2개 CPU의)	4 CPUs (2개 CPU의)
Test 2	기준	2-3배	4 배
Test 3	기준	2 배	3 배
Test 7	기준	2-3배	3-7배

(d) Masscomp

	2 CPUs	3 CPUs (2개 CPU의)	4 CPUs (2개 CPU의)
Test 2	기준	1.5배	1.5-2배
Test 3	기준	0.8배	0.8-1배
Test 7	기준	1.5-2배	1.5배

(e) Arrete

	2 CPUs	4 CPUs (2개 CPU의)
Test 2	기준	2-3배
Test 3	기준	1-1.5배
Test 7	기준	2-2.5배

Pyramid사도 현재는 4개의 CPU를 제공하고 있으나 앞으로 16개까지 제공하는 시스템을 개발중이라 한다. 이곳에서도 공유 기억장치에 의한 다중 프로세서 구조가 앞으로 미니와 메인프레임 시장을 궁극적으론 이겨나갈 것으로 믿고 있다. 이것은 공유 커널을 symmetric 프로세서들이 spin lock으로만 구현된 코드를 접근할때 그 오버 헤드는 대부분의 경우 0.5% 이하이고(30 CPU 해당 프로그램 경우 실험), 메모리 관리 등의 응용을 섞을 때도 최대 경우 5% 이하라고 한다. 이 다중 프로세서 버젼의 커널이 단일 프로세서 버젼보다 8-10% 정도 많다고 한다. 그들 의견으로는 CPU 20개 이상이라도 커널을 다중 프로세서화한 것에 대한 오버 헤드는 대개 1% 이하로 별 문제가 없다고 본다는데, 대개 resource의 80%가 idle 하기 때문이라 한다. Pyramid의 경우는 공유 기억장치에 의한 UNIX 다중 프로세서 시장이 앞으로 20년은 유효할 것으로 보고 있다.

Sequent사도 다중 프로세서화를 위한 커널 성능 분석에는 거의 같은 데이타를 가지고 있다. 이 곳에서는 spin lock 및 P, V 세머포를 모두 이용하였다고 한다. 이 경우 커널의 serialization 요인은 5% 이하라고 하고 있다. Pyramid사가 각 CPU의 성능을 높이는 노력과 CPU의 수를 늘리려는 노력을 동시에 하고 있는데 비해, Sequent사에서는 CPU는 상용 마이크로 프로세서를 사용하고 더 많은 CPU의 증가를 성취하는데 노력하고 있다.

이번 회사 방문에서 발견한 또 다른 사실은 이와 같은 공유 기억장치에 의한 다중 프로세서 시스템들이 대부분 데이타 베이스 등을 사용하는 상용 응용 시스템으로 판매되고 있다는 점이다. 즉, 지금까지는 이러한 아키텍쳐가 학교나 연구소에서 병렬처리(자연 과학 응용의) 실험에만 쓰이는 것으로 생각하던 기존 관념을 모두 벗어나 명실공히 기존의 미니 및 메인 프레임이 차지하던 일반 상용 응용 분야에 강력히 진출하고 있다는 점이다. Pyramid나 Sequent사들의 각 1,000 시스템 이상의 판매중 70% 이상이 DB를 사용하는 응용, OA용, 증권 시장용, 트랜잭션 처리용 등의 범용으로 공급되고 있었다. 그들의 이론으로는 주기억 용량을 많이

필요로하는 이와 같은 상용 응용 시스템은 메시지를 기반으로하는 구현으로는 성능 및 프로그래밍상 거의 불가능하다는 것이다. 이와 같은 논점은 다음 장에서 다루기로 한다.

7. 공유 기억 다중 프로세서와 메시지 기반 다중 프로세서의 비교

메시지 기반 다중 프로세서는 프로세서마다 자기의 기억장치를 각각 가지고 공유 기억장치가 없다. 그러므로 이와 같은 구조에서는 프로그래머가 사전에 수행되려는 일의 성질을 알고 (1)미리 각 프로세서에 해당하는 업무를 잘 분담하도록 프로그램을 하고, (2)수행 도중 업무 부담 등이 변하지 않고 지속되어야 그 효율을 최대로 할 수 있다. 즉, 응용이 일반의 기존 범용용의 프로그램을 수용한다든가 또는 시시각각 다른 종류의 응용 프로그램이 수행되거나 그 업무량이 변한다면 효율이 너무 떨어지게 된다. 그러므로 특수 연구 분야인 수치계산 및 인공지능 등의 병렬 알고리즘 등이 개발되어 100개 이상의 프로세서를 사용하는 연구 분야에서는 위에서 언급한 (1)과 (2)사항에 관한 사전 연구를 통해 병렬 알고리즘 등이 개발되어 메시지 기반 다중 프로세서를 효율적으로 사용할 수 있으나 일반 범용의 컴퓨터 대용으로는 대부분 시도를 하고 있지 않다.

즉, 기존의 미니 및 메인 프레임 컴퓨터 응용 분야인 범용 시스템으로 다중 프로세서가 대용되려면 공유 기억장치에 의한 다중 프로세서를 구성할 수 밖에 없다는 것이 방문한 회사 및 기존 학자들의 주장이다[1]. 이러한 범용 응용에서는 응용 프로그램의 성격 및 부하(load)가 시시각각 변하고, 특히 DBMS의 경우에는 그 크기가 커서 큰 공유 기억장치의 한 copy가 공용으로 쓰이는 것이 효율적이라는 측면에서 공유 기억장치에 의한 다중 프로세서가 적합하다는 것이다. 실제 기존 중대형 컴퓨터 시스템의 대용을 시도한다는 것은 그 분야의 응용 프로그램을 그대로 수용한다는 것으로, 메시지 기반 시스템에서는 load balance상 프로그램을 재작성하는 등 프로그래머의 개입이 다시 필요한 경우가 생긴다는 것을 가장 심각한 단점으로 보고 있다. 즉, 메시지를 기반으로한 분산 다중 프로세서를 기존 범용으로 쓸 경우 (1)성능의 급격한 저하 및 (2)프로그래밍이 매우 어렵다는 것이다. 기존의 단일 프로세서용 범용 프로그램을 분산 처리하여 높은 성능을 내기 위하여 사용자 입장에서 필요로 하는 변경 여부는 그림 3 과 같다.

공유 기억장치에 의한 다중 프로세서는 fork 에 의한 프로세스 발생만 사용자가 만들면 그 이후의 다중 처리는 운영 체제가 하게 된다. 이에 반해 메시지 기반은 아직도 그 이상의 많은 부분(프로세스 이주, 경우에 따라 라우팅 등등)을 사용자가 책임을 져야 하도록 되어 있다.

8. 결론

기존 중대형 컴퓨터를 위한 소프트웨어는 대부분 단일 프로세스를 위해 쓰여진 프로그램들이다. 이와 같은 중대형급 시장을 위한 고성능의 다중 프로세서 구조의 컴퓨터 개발을 위해서는 범용성을 가진 공유 기억장치에 의한 다중 프로세서가 적합하다. 현재 외국의 사례로도 범용의 다중 프로세서로는 공유 기억장치에 의한 방식을 취하고 있다. 이 분야의 시장은 이제 초창기이며, 더 값싸고 성능이 좋으면서 기존 응용 프로그램을 얼마나 잘 수용하는가에 그 성패가 달려있다. 기존의 범용 중대형 컴퓨터 회사들도 경쟁사들의 출현으로 인해 이와 같은 다중 프로세서에 의한 성능 향상의 시도를 시작하고 있다.

참고문헌

1. Gordon Bell, "Interview with Gordon Bell", UNIX Review, Feb. 1986.
2. M. Bach and S. J. Buroff, "Multiprocessor UNIX Operating Systems", AT&T Bell Lab. Technical Journal, Vol. 63, No. 8, Oct. 1984.
3. G. Hamilton and D. S. Conde, "An Experimental Symmetric Multiprocessor Ultrix Kernel", USENIX Winter Conf., Feb. 1988.
4. H. J. Hindin and M. Bloom, "When choosing a system, how do you balance vendor claims...", UNIX World, Jan. 1988.
5. 박 승규, "다중 프로세서 컴퓨터 동향", 전자 통신 동향 분석, 1987. 2.