주요기사

저렴한 수퍼컴퓨팅 기술

수퍼컴퓨팅 파워구현에 대한 새로운 대안이 제시되고 있다. 최근에 개발되는 이른바 수퍼컴퓨터들은 초고성능 마이크로 프로세서 몇 개를 활용하는 것이 아니라, 앞에서도 언급한 것과 같이 고성능 마이크로 프로세서 수백 개 또는 수천 개를 연결하여 구성되어 있다. 즉, 이들 수백 개 이상의 마이크로 프로세서들의 연산능력을 동시에 활용하는 병렬처리를 통해서 수퍼컴퓨팅이 이루어지고 있는 것이다.

김 승 조
서울대학교 항공우주공학과 교수

1. 서론

수퍼컴퓨터는 일반적으로 당대에 운용되는 컴퓨터중 가장 빠른 고성능의 컴퓨터를 일컫고 이들 컴퓨터를 이용해서 고성능 연산을 수행하는 일을 수퍼컴퓨팅이라 말할 수 있다. 이러한 일반적인 정의는 컴퓨터, 특히 중앙처리장치(CPU)의 급격한 성능향상으로 인해 약간의 수정이 필요하게 되었다. 예전의 수퍼컴퓨터의 초고성능(당대 다른 컴퓨터에 비해) 전용 중앙처리장치 대신에 현재의 수퍼컴퓨터는 PC(Personal Computer)나 워크스테이션에서 사용하는 마이크로 프로세서를 대량으로 활용하여 병렬처리를 통해 수퍼컴퓨팅을 수행하기 때문이다. 따라서, 적절한 병렬 알고리듬을 쓰지 않으면 수퍼컴퓨터에서도 수퍼컴퓨팅을 이룰 수가 없고, 반대로 여러대의 PC들로도 적절한 병렬연산수단을 가지면 수퍼컴퓨팅이 가능한 시대가 도래한 것이다. 다시 말하면, 조금만 노력을 기울이면 큰 투자없이도 수퍼컴퓨팅을 구현할 수 있는 수퍼컴퓨팅 보편화시대가 된 것이다. 더욱이, 현재의 PC용 CPU, 즉 펜티엄 Ⅲ, Ⅳ 그리고 Athlon Chip들은 내부구조에 맞추어 적절히 Tuning 하여 사용하게 되면 연산속도를 크게 증가 시킬 수 있는 가능성을 가지고 있어 병렬화를 통하지 않더라도 1대의 성능으로써 만족할 만한 계산속도를 얻을 수 있다.

이 글에서는 컴퓨터 환경변화에 대해서 자세히 언급하고, CPU의 Tuning 문제 그리고 이들 기술을 활용한 인터넷 수퍼컴퓨팅시스템, Clustering 수퍼컴퓨팅 시스템 등의 구현 및 성능예시를 통해 그 효과를 살펴본다. 그리고 수퍼컴퓨터 및 수퍼컴퓨팅의 선진국인 미국에서 이루어지고 있는 Grid 기술 및 그 응용으로써의 M&S(Modeling & Simulation) 신드롬도 알아보고, 이제는 수퍼컴퓨팅 자원이 부족한 한국에서도 저렴한 수퍼컴퓨팅기술로써 선진

국에 맞먹는 기술혁신을 가져올 수 있다는 것을 보인다.

2. 컴퓨팅 환경변화 - 수퍼컴퓨팅 패러다임의 변화

1946년 최초의 컴퓨터인 ENIAC이 Mauchly 교수팀에 의해 개발된 이래 초기 컴퓨터 기술은 과학기술 분야의 계산 요구에 의해 발전되어 왔으며, 고속 수치연산에의 요구는 마침내 1976년 CRAY-1이라는 수퍼 컴퓨터가 컴퓨터 설계의 천재 Seymour Cray에 의해 탄생되기에 이르렀다. 그리고 CRAY XMP, YMP, CRAY-2, C90 등이 연이어 개발되었으며 벡터 프로세싱 기법을 이용하여 당내 최고 수준의 계산 속도를 자랑하였다. 그러나 이러한 초기의 수퍼 컴퓨터들은 완전하게 '벡터화'가 가능한 제한된 프로그램들에 대해서만 수퍼컴퓨터라는 이름에 걸맞는 성능을 낼 수 있다는 점뿐만 아니라 엄청나게 비싼 가격 때문에 특수과학기술 분야에서 주로 활용되었다. 그동안의 마이크로 프로세서 기술의 발달로 인한 컴퓨터 성능의 향상과 개인용 컴퓨터(Personal Computer, PC)의 보급 확대로 인해 보다 넓은 분야에 개인용 PC가 활용되게 되었으며, 수퍼컴퓨팅 파워구현에 대한 새로운 대안이 제시되고 있다. 최근에 개발되는 이른바 수퍼컴퓨터들은 초고성능 마이크로 프로세서 몇 개를 활용하는 것이 아니라, 앞에서도 언급한 것과 같이 고성능 마이크로 프로세서 수백 개 또는 수천 개를 연결하여 구성되어 있다. 즉, 이들 수백 개 이상의 마이크로 프로세서들의 연산능력을 동시에 활용하는 병렬처리를 통해서 수퍼컴퓨팅이 이루어지고 있는 것이다. 그리고 인터넷으로 대변되는 네트워크의 급속한 확대는 정보화 시대로의 급속한 변화를 가져왔으며, 기업체나 연구소 등의 사무용 및 업무용 PC 및 워크스테이션들이 서로 간의 데이터를 주고받을 수 있게 되었다. 그래서 이러한 네트워크화된 컴퓨팅 자원들 역시 중요한 계산자원으로 인식되기 시작했다. 특히, 표 1에 나타나 있는 바와 같이, 당대 최고의 수퍼컴퓨터를 구성하고 있는 마이크로 프로세서들과 개인용 PC에 활용되고 있는 마이크로 프로세서들 간의 성능이 거의 차이가 없어지면서(개별성능은 PC가 더 좋을수도 있다) 네트워크로 연결된 개인용 PC들을 이용한 수퍼컴퓨팅 구현이 현실적으로 가능해지기 시작한 것이다.

수퍼컴퓨터에 사용되는 마이크로 프로세서	
ASCI White	375MHz Power3 processor
ASCI Blue-Pacific SST	333MHz PowerPC 604e
IBM SP2(Nighthawk)	375MHz Power3 processor
CRAY T3E 1200	600MHz Alpha processor
HP Superdome	750MHz PA8700 processor
개인용 PC에 사용되는 마이크로 프로세서	
1.0~2.2 GHz Pentium IV processor	
1.5 GHz Athlon processor	
867 MHz PowerPC G4 processor	

표 1. 수퍼컴퓨터와 개인용 PC에 활용되는 마이크로 프로세서들

이러한 컴퓨팅 환경의 변화로 인해, 점점 더 중요성이 부각되고 있는 것은 효율적인 병렬 알고리듬의 개발이라고 말할 수 있을 것이다. 기본적으로 병렬화를 구현하지 않고서는 수퍼컴퓨터를 활용한다고 해도, 결국 고성능 개인용 PC 한 대를 이용하는 것과 거의 동일한 정도의 성능만을 얻을 수 있기 때문이다.

3. PC를 활용한 수치해석에서의 효율성 향상기법

변화된 컴퓨팅 환경하에서 HPC(High Performance

Computing)를 구현하기 위한 가장 중요한 요소는 효율적인 병렬 알고리듬의 구현일 것이다. 풀고자 하는 응용문제의 병렬화 구현, 로드 밸런싱 및 작업 분배 등 효율적인 병렬화를 구현하기 위해서는 많은 문제들을 해결해야 한다. 하지만, 효율적인 병렬화를 구현하기에 앞서, 병렬 컴퓨팅에 참여할 개별 구성 노드들에서 수행되는 계산의 효율화를 먼저 구현할 필요가 있다. 개별 구성 노드들의 계산성능은 병렬 컴퓨팅의 성능을 좌우하는 가장 중요한 인자들 중에 하나이기 때문이다.

본 연구에서는 PC를 활용한 수치해석시의 최적화 기법에 대해서 살펴보고, 최근에 널리 활용되고 있는 Pentium IV 마이크로 프로세서에 대한 LINPACK 벤치마킹[1]을 통해 성능 향상을 살펴본다.

◇ PC 수치해석 최적화 기법

일반적인 수치해석 수행시의 최적화 기법을 살펴보면, 우선 가장 기초적인 것은 최적의 알고리듬을 선택하는 것과 필요하지 않은 연산을 최소로 줄이는 것이다. 그 다음으로 할 수 있는 것은 수치해석에 사용하는 컴파일러의 최적화 기능을 최고로 이용하는 것이다.

최근에 개인용 PC에 널리 활용되고 있는 Pentium IV 프로세서의 메모리 구조와 각 메모리 레벨별 Latency와 Bandwidth가 Fig. 1에 나와 있다[2]. Fig. 1과 같이 Register의 경우에는 CPU에서 필요한 데이터와 명령어를 바로 가져 올 수 있는 반면에 L2 Cache 메모리에서는 정수형 변수 및 실수형 변수 모두 7 CPU 사이클이 지나야만 가져올 수 있다. 시스템의 특성에 따라 다르지만, 메인 메모리의 경우에는 수백 CPU 사이클이 필요하게 된다.

그리고 CPU에서는 다음 명령어 셋을 실행시키

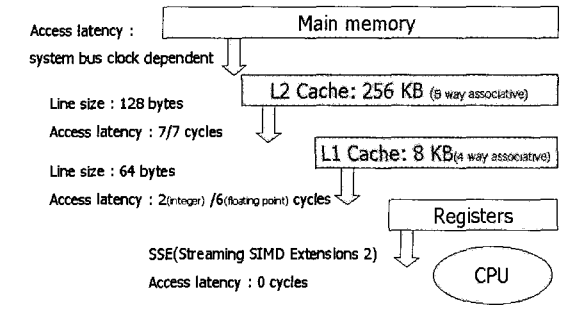

Fig. 1 Pentium 4 Memory hierarchy

기 위해 필요한 데이터를 가장 먼저 Register에서 찾고, 데이터가 없을 경우에는 L1 Cache, L2 Cache, 메인메모리 순서로 데이터를 찾게 된다. L1 Cache의 경우에는 해당 데이터가 없을 때, 즉시 메모리가 업데이트되면서 정보를 갱신하게 된다(Write-through)[3]. L2 Cache의 경우에는 해당 데이터가 없어서 하위 메모리에서 데이터를 Load/Store를 수행해야 할 때, 새로운 데이터를 저장한 공간이 부족할 때에만, 가장 마지막에 사용한 데이터가 저장된 곳에 있는 데이터를 메인 메모리로 보내고, 메인 메모리에서 필요한 데이터를 가져오게 된다(Write-back)[3]. 이러한 L1 Cache, L2 C ache에서의 메모리 업데이트 연산을 최소로 하기 위해서는 수치연산에 동원되는 데이터를 적절히 Block하여, 한번 Load/Store 연산이 발생할 때마다, 메모리 사이즈에 적당한 데이터를 Load/Store하고, 바로 다음 연산에 필요한 데이터들이 최대한 동일한 데이터 Block내에 존재하도록 해 주어야 한다. 결국 효과적인 최적화는 연산회수에 의해 좌우되는 것이 아니라, 레지스터와 L1, L2 Cache 및 메인 메모리들 사이의 주소 참조연산이 얼마나 최소화되었는가에 달려 있는 것이다. 이러한 최적화 기법을 Blocking 기법[4]이라고 한다. 이러한 Blocking 기법을 활용한 수치해석 라이브러리가 많이 개발되었는데, 특히 BLAS(Basic Linear Algebra

Subprograms)[5]가 널리 활용되고 있다. 한편, 최근에 등장한 마이크로 프로세서들은 특수한 기능을 가진 명령어셋을 내장하여 SIMD(Single Input Multiple Data) 기능을 구현하였다. Pentium의 경우에는 MMX(Multimedia Instruction extensions)를 IA-32 아키텍처에 처음으로 도입하였다. MMX는 64 비트의 MMX Register를 이용하여 구현한 것으로, 정수연산을 수행하였다. Pentium III에서는 128 비트의 XMM Register를 이용하여 구현한 SSE (Streaming SIMD Extensions)로 MMX 기술을 확장하여, 단정도 부동소수점 연산을 지원하였다. SSE 기능을 이용할 경우, 이론적으로는 CPU 클럭당 4번의 단정도 부동소수점 연산이 가능해졌다. 인텔에서는 최근에 NetBurst Micro-Architecture에 SSE 기능을 확장하여 SSE2 기능을 모두 144 Instructions으로 구성하여 배정도 부동소수점 연산을 가능하게 하였다[4]. SSE2를 이용할 경우에, CPU 클럭당 2번의 배정도 부동소수점 연산을 수행할 수 있다. 하지만, 이러한 이론 최고성능을 구현하기 위해서는 XMM Register에 다음 명령어를 연속적으로 수행할 수 있도록 데이터를 배치하여 주는 기술(Streaming Technology)이 중요하며, 앞에서 언급한 Blocking 기법과의 연계가 중요하다.

◇ 수치실험 (Linpack Benchmark)

본 절에서는 이러한 여러 가지 기법들의 실제 성능 차이를 알아보기 위해서 수치실험을 수행해 보자. 수치실험에 사용한 시스템의 사양은 표 2와 같다. Linpack 벤치마킹을 수행하기 위해서 HPL(High-Performance Linpack Benchmark for Distributed-Memory Computers)[6]을 이용하였다. 그리고 Blocking 기법과 SSE2를 이용한 부동소수점 연산기능을 사용하기 위해서 ATLAS (Automatically Tuned Linear Algebra Software)[7]을 이용하였다. ATLAS 설치시에 L2 Cache Blocking Parameter는 별도의 수치실험을 통하여 최고의 성능을 내도록 설정하였으며, 192KB로 설정하였다. Pentium IV L2 Cache는 256 KB이지만 192KB로 한 것은, Cache 메모리 상에는 데이터뿐만 아니라 명령어(Instructions)도 함께 저장되어야 하기 때문이다.

한편, L2 Cache Block 사이즈뿐만 아니라 해당

CPU : Pentium 4 1.5 GHz
Main Memory : RDRAM 1GB
HDD : 40GB(7200 rpm)
Mother Board : Intel i845 Chipset FSB 400MHz
OS : Linux (2.4.2 Kernel)
C/C++ Compiler : gcc-2.96
ATLAS : 3.3.1 version(SSE2 ON) 3.2.1 version(SSE2 OFF)

표 2 수치해석에 사용된 시스템 사양

데이터를 적절히 Block화하여야 한다. 데이터의 블록화라는 것은 LU Decomposition을 수행하기 위한 연산을 주어진 행렬의 각각의 요소별로 수행하는 것이 아니라, 주어진 행렬을 Block화하여 크기가 작은 여러 개의 행렬로 분해한 다음, 모든 연산을 행렬대 행렬연산으로 바꾸어서 연산을 수행하는 것을 말한다[4]. 이 때, 하나의 정방행렬인 데이터 블록의 크기를 NB라고 정의하였다. NB의 값을 조절하여 성능에 미치는 영향도 살펴보았다.

우선, 계산방법에 따른 성능을 차이를 알아보았다. Linpack 벤치마킹을 수행하기 위한 선형연리방정식 해법을 가장 널리 활용되는 Gauss 소거법[8] 사용했을 경우, LU Decomposition[8]을 이용했을 경우, LU Decomposition 방법에서 ATLAS 라이브러리를 이용한 Cache Blocking 기법을 이용한 경우

(Ver. 3.2.1), SSE2 기능까지 이용한 경우(Ver. 3.3.1)로 나누어서 성능을 비교해 보았다.

각각의 경우에서 최고의 성능을 낼 때의 데이터를 비교한 결과가 Fig. 2에 나와 있다. 모든 경우에서 앞에서 언급한 컴파일러를 이용한 기본적인 최적화는 모두 수행한 상태에서 비교하였다. 가장 널리 활용되는 Gauss 소거법을 이용한 경우를 살펴보면, 이론 최고성능치(1.5GHz · 2 = 3.0 Gflops) 대비 3% 미만의 성능을 얻을 수 있었다(Linpack Benchmark 행렬크기 1,000일 때). LU Decomposition의 경우에는 5.5% 정도임을 알 수 있다(Linpack Benchmark 행렬크기 1,000일 때). 그러나 Gauss 소거법에 비해서는 1.84배 정도 빨라진 것을 알 수

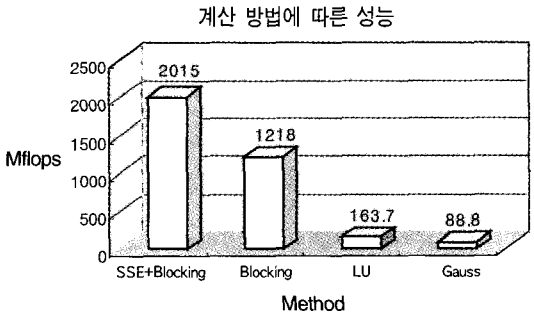

Fig. 2 계산 방법에 따른 성능비교

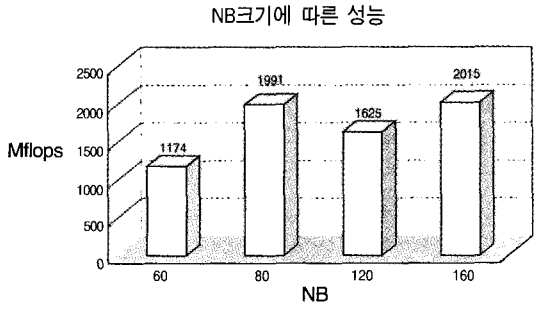

Fig. 3 데이터 Block 사이즈(NB)에 따른 성능차이
Linpack Benchmark 행렬크기 11,000일 때

있다. 그러나, Blocking 기법 사용하여 얻은 최고의 성능은(NB=160, Linpack Benchmark 행렬크기 11,000일 때) 1.2 Gflops로 최고성능대비 40% 정도의 성능을 얻을 수 있었다. 마지막으로 SSE2를 이용한 경우에는 67% 정도인 2.0 Gflops를 달성하였다(NB=160, Linpack Benchmark 행렬크기 11,000일 때) 이것은 Gauss 소거법을 이용했을 때와 비교하면 22배가 넘는 성능향상을 가져 온 것이다. 첫 번째 실험에서 최고의 성능을 기록한 경우인 Blocking기법과 SSE2를 동시에 이용한 경우에 대해서 데이터 Block 사이즈에 의한 성능향상 효과를 살펴보았다. Fig. 3의 결과를 보면, NB가 성능에 결정적인 영향을 준다는 것을 알 수 있다. 가장 낮은 성능을 보인 경우는 NB가 60일 때이며 1.174 Gflops의 성능을 보였다. 최고의 성능은 NB가 160일 때, 2.015 Gflops의 성능을 보였다.

4. Clustering 시스템을 이용한 수퍼컴퓨팅

네트워크로 연결된 워크스테이션이나 개인용 PC들의 컴퓨팅 자원을 활용하기 위한 기술인 Clustering 기술은 1994년 Donald Becker에 의해 소개되고 가능성이 입증된 후, 자체적인 수퍼컴퓨팅 시스템을 구축하고자 하는 연구소나 대학 등에서 널리 활용되고 있다. 특히, POSIX 운영체계 규격을 따르며, UNIX System V, BSD 확장을 덧붙여 만들어진 공개 UNIX Clone 운영체제인 Linux 시스템을 설치한 개인용 PC들을 이용하여 구축한 Clustering 시스템은 과학기술분야뿐만 아니라, 웹서비스 등 다양한 분야에서 사용되어지고 있다. 본 연구에서는 서울대학교 항공우주 구조연구실에서 구축하여 운영 중인 SSC(*S*eoul National University *S*uper *C*luster) 시스템과 수치예제를 통해 Clustering 시스템을 통한 수퍼컴퓨팅 기술을 살펴보고자 한다.

◇ SSC 시스템 구성

SSC 시스템은 기본적으로 32 노드로 구성되어 있다. 각 구성노드는 기본적으로 Pentium Ⅲ 1GHz Dual CPU 시스템으로 구성되어 있다. 메인보드는 VIA 694x SMP Chipset(MSD DP694E)을 사용하였으며, 256MB의 PC-133 메인메모리와 20GB HDD, AGP 그래픽 카드가 각각 설치되었다. Pentium Ⅲ 1GHz CPU는 시스템 설계 당시(2001년 7월), 가격대 성능비가 가장 좋았고 Dual CPU 시스템을 구성하기에도 적합했다. 그리고 SSE(SIMD Streaming Extensions)[1-26]기능을 이용하여 뛰어난 성능을 발휘할 수 있기 때문이다. 메인보드는 Pentium Ⅲ 1GHz Dual 시스템을 지원하는 Chipset이 VIA Chipset 뿐이었다. 그리고 한 대의 노드(Master)에는 모니터 한 대와 키보드, 마우스 등을 설치하여 시스템 모니터링 및 병렬 작업 모니터링 등의 기능을 수행할 수 있도록 하였다.

모든 노드들에는 리눅스 시스템을 기본 OS로 선택하였으며, gcc-2.96 compiler를 기준으로하여 모든 응용프로그램들을 개발하였다. Message-Passing Library로는 LAM을 사용하였다.

SSC 시스템의 네트워크 구성을 살펴보면, 기본적으로 24-port switching 허브 2대(Bandwidth 3.2 Gbit/s)로 32대의 노드들이 연결되어 있다. 각 구성노드들에는 기본적으로 하나씩의 Fast Ethernet(100Mbps) 랜카드 하나씩을 장착하였으며, Master 노드에는 2개의 랜카드를 설치하여, 하나는 Cluster 시스템 내의 로컬 네트워크 구성을 위해 사용하고, 다른 하나는 외부 네트워크와의 연결을 위해 사용하였다. 그리고 랜카드는 가격대비 성능비가 뛰어나고, 리눅스에서 쉽게 사용할 수 있는 RTL8139 Chipset의 랜카드를 사용하였다.

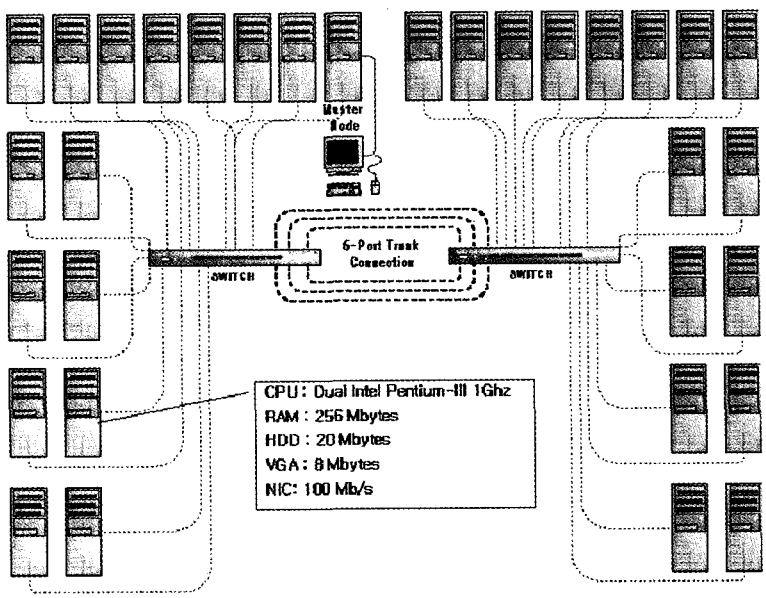

Fig. 4 SSC 구성 및 네트워크 구성

이러한 네트워크 구성 및 각 시스템 구성은 Fig. 4에 잘 설명되어 있다. SSC시스템을 구성하기 위한 모든 부품들은 일반 PC에도 널리 활용되는 부품들로서, 모든 부품들이 용산전자상가를 통해 구입한 것들이다. SSC 시스템을 구성하기 위해 필요했던 비용은 네트워크 캐이블 가격 등까지 모두 포함하여 32,347,800원이었다. 구성된 SSC 시스템을 앞에서 언급한 blocking기법과 SSE기능을 활용할 수있게끔 최대한으로 튜닝한후 응용프로그램을 가동했다. 이 SSC 시스템을 활용하여 항공우주 구조물에 가해진 충격위치를 찾는 응용문제를 풀었다[9]. 가해진 충격위치와 시스템 반응사이의 관계를 Neural Network을 이용해서 찾아보았으며, 충격에 의한 시스템 반응은 유한요소기법과 중앙차분법을 이용한 시간적분법으로 구하였다. 그리고 구해진 충격에 의한 구조물의 시간응답은 Wavelet Packet 변환[10]을 이용하여 변환된 후, Neural Network의 입력데이터로 활용되었다. 수치실험 결과, 가장 오래 동안 수행되었던 수치실험은 47시간 23분의 시간이 소요되었으며, 총 17.25 Petaflops(10^{15} floating-point operations)의 단정도 부동소수점 연산이 이루어졌다. 따라서 대략 47시간 가까운 동안 101.12 Gflops/s라는 시스템 성능이 지속적으로 유지되었고, 시스템 구성에 소요된 비용, 32,347,800원(10% 부가가치세 포함)을 성능으로 나누면 319.3원/Mflops/s 정도의 Price/performance 결과를 얻을 수 있다.

5. 인터넷 수퍼컴퓨팅

최근에 등장한 PC에 활용되는 고성능 마이크로프로세서의 경우에 수퍼컴퓨터에 활용되는 마이크로 프로세서에 비해서 결코 성능이 떨어지지 않으며, 마이크로 프로세서의 메모리 구조를 잘 활용한 최적화를 수행하면 충분히 수퍼컴퓨팅을 수행할 수 있음을 이미 살펴보았으며, Clustering 시스템을 통해 네트워크로 연결된 개인용 PC들도 효율적인 병렬처리 알고리듬과 최적화 기법을 이용할 경우에 수퍼컴퓨터에서나 달성 가능했던 성능을 달성할 수 있음을 확인할 수 있었다. 이러한 수퍼컴퓨팅 구현에 관한 새로운 기술들을 바탕한 인터넷 수퍼컴퓨팅 기술에 대해 살펴보고, 그 타당성 및 가능성을 살펴보았다.

◇인터넷 수퍼컴퓨팅 기술의 기본개념

인터넷 수퍼컴퓨팅 기술이라는 것은 기존의 초고속 네트워크를 이용한 분산병렬처리 기술과는 다르게, 인터넷으로 연결되어 있는 PC들을 소프트웨어적으로 서로 연결해서, 인터넷상에 존재하는 가상적인 수퍼컴퓨터를 구현하여, 이를 통해서 수퍼컴퓨팅을 수행하는 기술을 말한다. 기존의 네트워크를 이용한 수퍼컴퓨팅 기술과 비교하면 하드웨어적으로 우선 가장 큰 차이점은 인터넷을 사용한다는 점과 PC를 사용한다는 점이다. 그래서 ASCI White 수퍼컴퓨터처럼 각각의 컴퓨터들을 연결하는 별도의 초고속 네트워크가 필요 없게 되고, 기존의 PC를 사용함으로써 획기적으로 계산비용을 낮출 수 있게 된 것이다. 그리고 더욱 좋은 점은 기존의 수퍼컴퓨터는 초기 구입가격뿐만이 아니라, 시스템 유지보수비도 필요하지만, 인터넷 수퍼컴퓨팅에 동원되는 PC들은 별도의 시스템 유지보수비가 필요 없다는 점이다. 이미 각각의 유저가 자기의 업무를 수행하기 위해서라도 유지보수를 하기 때문이다. 그리고 PC 클러스터링 시스템과 비교한다면, PC를 이용한다는 점은 같지만, 시스템의 사용과 운영상에 근본적인 차이가 있다. 클러스터링 시스템의 경우, 과학기술계산용이나, 웹서비스 등과 같이 특정목적을 위해서 설계되고

개발되어진다. 그래서 시스템을 구축하기 위해서는 별도의 PC를 구입해야만 한다. 그리고 지속적으로 클러스터링 시스템을 이용할 연구계획이나 시스템의 운영계획이 없으면, 상당한 시간동안을 그냥 전원만 켜 놓은 채, 아무 일도 수행하지 않고 있게 된다. 또한 클러스터링 시스템을 구성하는 개별 컴퓨터 각각은 독자적인 아무런 의미도 가지고 있지 않다. 단지 전체 클러스터링 시스템의 한 부분으로만 존재할 뿐이다. 하지만, 인터넷 수퍼컴퓨팅의 경우는 이미 사무용이나, 웹서버 등의 특정목적을 위해 존재하고 있는 PC들을 활용하여, 필요할 경우에 일시적으로 필요한 수만큼의 PC를 Fig. 5와 같은 하나의 가상 수퍼컴퓨터로 구현하기 때문에, 계산을 수행하지 않을 때에는 각각의 PC들은 원래의 목적으로 활용될 수 있다. 그리고 클러스터링의 경우에 아무리 분산병렬 환경으로 시스템을 구현한다고 해도, 노드 하나를 교체하거나, 노드를 증설해야 할 경우에는 상당한 어려움이 있다. 컴퓨터를 더 구입해야 하고, 필요에 따라서는 전체 시스템의 설정을 다시 해야할 경우도 많다. 하지만 인터넷 수퍼컴퓨팅에서는 이러한 확장성이나 노드 교체가 상당히 간단하다. 실제로 시스템을 구현하는 것이 아니라, 소프트웨어적으로 가상적으로 구현하기 때문에 언제든지 특정 컴퓨터를 다른 컴퓨터로 교체할 수 있고, 전체 컴퓨터 수도 언제든지 손쉽게 변경할 수 있다. 그리고 역시 클러스터링 시스템에서도 별도의 고속 네트워크를 필요로 한다. 그리고 일반적으로 클러스터링의 경우, 각각의 노드들의 하드웨어 사양이 모두 같은 것이 일반적이지만, 인터넷 수퍼컴퓨팅의 경우에는 각각의 컴퓨터들의 하드웨어 사양은 일정하지 않다.

지금까지 설명한 것처럼, 인터넷에 연결된 기존의 PC들을 활용하기 때문에 인터넷 수퍼컴퓨팅기술이 기존의 방법들에 비해서 많은 장점을 가지게

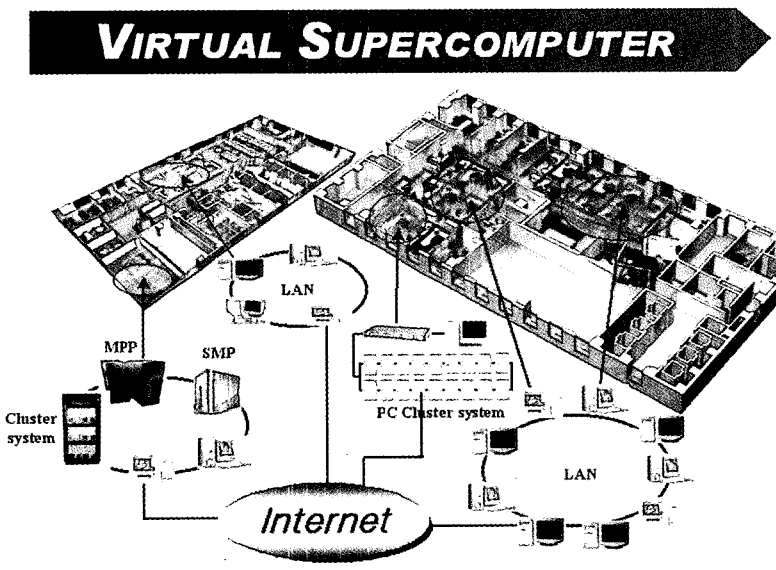

Fig. 5 Internet Supercomputing의 개념도

되지만, 반드시 장점만이 있는 것은 아니다. 우선, 가장 큰 단점은 역시 별도의 고속 네트워크를 구축하지 않고, 인터넷을 통해서 컴퓨터들간의 통신이 이루어지므로 전송속도가 기존의 수퍼컴퓨팅 전용 서버나 클러스터링 시스템에 비해서 상당히 느리다는 것이다. 그래서 인터넷 수퍼컴퓨팅 기술을 활용하여 분산병렬처리 프로그램을 개발하려고 할 경우에는 컴퓨터들간의 통신을 최대한으로 줄이는 방향으로 프로그램을 작성해야만 할 것이다. 이러한 이유로 인해서 실제 풀고자 하는 문제를 가장 잘 파악하고 있는 것이 중요하다. 바꾸어서 이야기하면 인터넷 수퍼컴퓨팅의 경우, 상대적으로 느린 시스템의 통신속도에서의 발생하는 단점을 효율적인 분산병렬 알고리듬의 개발로 극복해야 하는 어려움이 있는 것이다.

◇ 가상수퍼컴퓨팅 시스템, InterSup II를 이용한 수퍼컴퓨팅

인터넷 수퍼컴퓨팅의 수행하기 위한 가상 수퍼컴퓨터 시스템인 InterSup II를 128 CPU Node로 구성하였다. InterSup II는 서울대학교 내의 4개의 연구실에 있는 연구용 PC들과 앞에서 살펴 본 64 CPU Node 시스템인 SSC 시스템으로 크게 구성되어 있다. InterSup II 시스템을 이용하여 해석한 구조모델은 cutouts을 가지고 있는 3D 복합재료 구조물로써, 모두 681,552개의 3D solid 20 node-element가 사용되었으며, 전체 자유도는 10,022,004이다. 상세 응력해석은 인장력을 z 방향으로 가해지는 경계조건으로 계산되였다. 이러한 초대형 1000만 자유도 이상문제를 InterSup II를 이용해서 해석하는데, 걸린 시간은 12,720초로서 3시간 30분 정도의 시간이 소요되었으며, 17.4GB의 메모리가 사용되었다

6. Piggyback 수퍼컴퓨팅

Piggyback 수퍼컴퓨팅 기술은 인터넷 수퍼컴퓨팅 기술을 기반으로 하며, 클러스터 컴퓨팅의 장점을 접목시킨 기술이라고 할 수 있다. 우선 기존의 네트워크에 연결된 컴퓨터의 CPU 파워향상을 위해 Dual CPU 시스템화하고 또한 메모리를 추가 설치한다. 이렇게 함으로 해서 기존 유저들이 PC를 이용하는 일반 업무를 원활하게 하면서도 많은 CPU파워를 요구하는 병렬 수퍼컴퓨팅도 가능케한다. 그러나 대부분의 컴퓨터 기능은 기존의 PC성능에 의존하기 때문에 Piggyback이라는 이름을 붙인 것이다. 각각의 PC에는 리눅스를 기본 OS로 설치하며, 별도의 소프트웨어를 이용하여 다른 OS를 리눅스 기반 위에서 사용할 수 있다(이러한 PC 시스템을 SePC(Server-embedded PC)라고 부른다). 이상을 개념적으로 설명한 것이 Fig. 6이다.

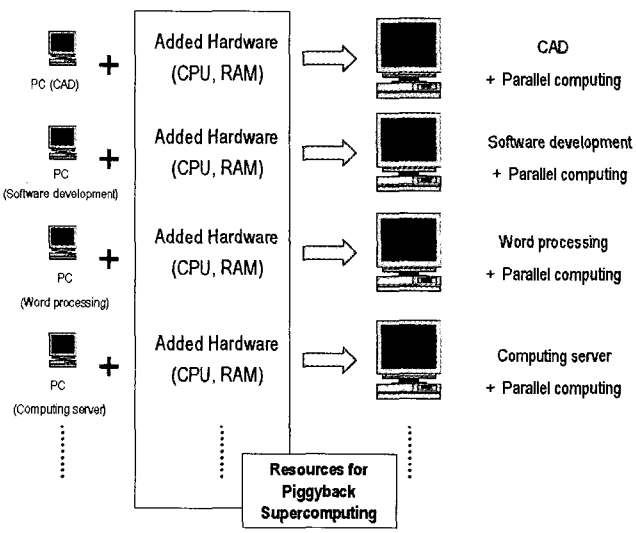

Fig. 6 Piggyback 수퍼컴퓨팅의 개념도

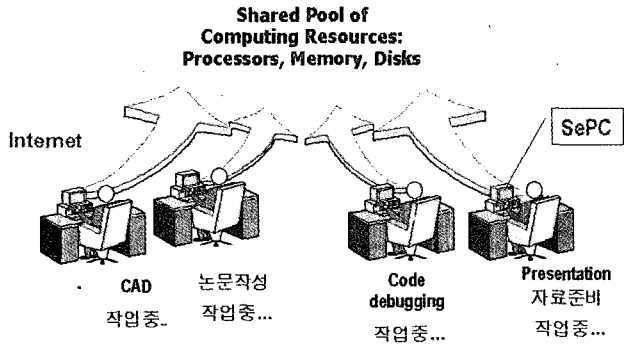

Fig. 7 Piggyback 수퍼컴퓨팅

이러한 수퍼컴퓨팅 기술의 장점은 Fig. 7에 설명된 것처럼, 각 PC의 유저들이 자신의 업무나 연구를 하는 동안에도, 대규모의 컴퓨팅 자원을 요구하는 특정 유저가 각 PC들의 자원을 활용하여 병렬 계산을 수행할 수 있다는 것이다. 다시 말해서, 인터넷 수퍼컴퓨팅 기술의 경제성 및 전문 병렬 클러스터 시스템의 사용 편의성을 모두 갖춘 것이 Piggyback 수퍼컴퓨팅이라고 볼 수 있으며, 인터넷 수퍼컴퓨팅의 범용성 및 확장성을 크게 넓힐 수 있는 응용기술이다.

7. 결론 - 수퍼컴퓨팅의 미래

지금까지 살펴 본 것과 같이, 변화된 컴퓨팅 환경에 적합한 수퍼컴퓨팅 구현기술은 분산병렬처리 기술이며 인터넷으로 연결된 개인용 PC들도 효율적인 분산병렬처리 알고리듬과 인터넷 수퍼컴퓨팅 기술을 이용한다면 충분히 경쟁력 있는 수퍼컴퓨팅을 수행할 수 있다.

특히, 최근에 등장한 PC들의 연산처리 능력은 이 글에서 살펴 본 것처럼 메모리 구조를 반영한 효율적인 최적화를 수행한다면 ASCI White와 같은 당대 최고의 수퍼컴퓨터를 구성하는 개별 CPU의 성능에 결코 뒤지지 않다는 사실은 인터넷 수퍼컴퓨팅 기술 및 PC 기반 수퍼컴퓨팅의 타당성을 뒷받침하고 있다. 그런데 최근에 미국을 비롯한 선진국에서 경쟁적으로 개발하고 있는 기술이 바로 그리드(Grid) 컴퓨팅 기술이다. 그리드라는 것은 첨단 네트워크 기술과 고성능 컴퓨팅 기술의 결합을 통한 4A (Advanced Computers, Advanced Networks, Advanced Applications, Advanced Experts)의 유기적 결합체라고 설명할 수 있다. 다시 말해서, 정보 및 과학기술 분야의 연구에 필요한 4A 자원을 누구나 원하는 규모로 적시에 활용할 수 있도록 유기적으로 결합한 첨단 지식 인프라를 의미한다. 그래서 그리드라는 인프라 내에서 분산되어 있는 컴퓨팅 자원들뿐만 아니라, 데이터 자원들, 실험장비들, 인적 자원들까지 가능한 모든 자원들이 네트웍으로 통해 유기적으로 결합되어 있는 것이다. Fig. 8는 미국에서 개발 중인 IPG(Information Power Grid)를 활용한 항공기 설계의 개념도이다.

여기서 중요한 것은 이러한 그리드 컴퓨팅에서 가장 핵심이 되는 기술이, 분산되어 있는 컴퓨팅 자원의 유기적 결합을 통한 분산 어플리케이션 수행기술이며, 바로 인터넷 컴퓨팅 기술의 핵심이라는 점이다. 다시 말해서 개념적으로 볼 때, 인터넷 컴퓨팅이 그리드 컴퓨팅에 포함되는 개념이라고 볼 수 있다. 다만, 분산되어 있는 컴퓨팅 환경을 결합하는 네트워크를 구현하는 데 있어서, 그리드 컴퓨팅의 경우 별도의 고속 네트워크 기술을 개발하고 적용하는 것을 모두 포함하는 개념이며, 인터넷 컴퓨팅의 경우에는 인터넷을 그대로 활용한다는 차이점이 있다. 그리고, 인터넷 컴퓨팅의 경우에는

인터넷에 연결된 개인용 PC의 자원을 최대한 이용하는 방향으로 연구가 추진되고 있는 반면, 현재 추진되고 있는 그리드 컴퓨팅 프로젝트의 경우에는 기존의 수퍼컴퓨팅 인프라와 대규모 컴퓨팅 파워들을 체계적으로 통합하는 것에 중점을 두고 있다는 차이점이 있다.

이러한 그리드 컴퓨팅 기술의 개발 방향은 이미 존재하고 있는 상당한 규모의 수퍼컴퓨팅 자원이 있기 때문이기도 하다. 그래서 이러한 개발방향은 우리 나라의 현실에는 상당한 무리가 있다고 볼 수 있다. 그러나, 그리드 컴퓨팅 개념을 우리 나라 현실을 고려하여 개발하는 것은 국가경쟁력 차원에서 반드시 추진해야 하는 과제이다. 따라서 인터넷 수퍼컴퓨팅 기술을 이용한 수퍼컴퓨팅기술의 개발은 우리 나라의 실정에 적합한 독자적인 수퍼컴퓨팅 파워를 구현할 수 있다는 점 이외에도 선진국과 대등한 수준의 수퍼컴퓨팅 기반 연구를 수행해서 다양한 과학기술분야에서 국가경쟁력 강화에 기여할 수 있다는 점에서 상당한 의미를 가진다고 볼 수 있다.

그런데 여기서 우리가 생각해 보아야 할 것은 미국과 같은 선진국에서 그리드 컴퓨팅 기술고 같은 새로운 개념의 수퍼컴퓨팅 기술을 개발하고 구현하고자 하는 궁극적인 목적이다.

그러한 목적을 잘 파악할 수 있는 한 예가 바로 미국 국방성이 추진하고 있는 M&S (Modeling & Simulation) Master Plan이다. 미국은 1991년에 DMSO(Defense Modeling & Simulation Office)라는 국방부 소속 전담 조직을 구성하여 M&S기술의 개발을 주도하고, 국방의 여러 분야에 활용할 수

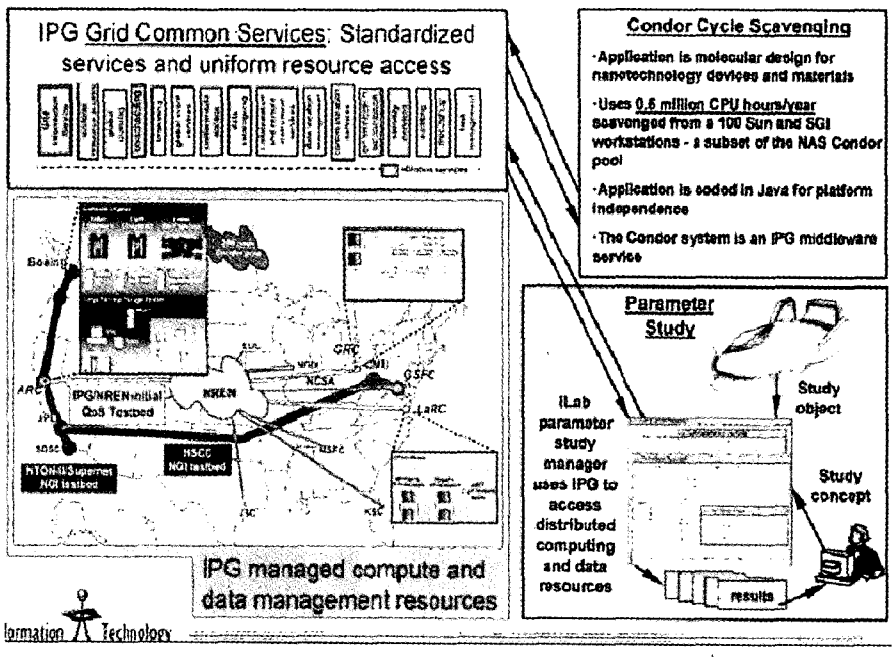

Fig. 8 IPG(Information Power Grid)를 활용한 항공기 설계

있도록 하고 있다. M&S 기술은 국방전략 수립, 각종 무기체계의 개발 및 획득 체제, 병참 분야, 가상모의 전투를 통한 군사훈련 및 교육 등의 다양한 국방분야에서 활용되고 있다. 특히, 무기체계의 개발 및 획득체제에서 M&S 기술은 SBA (Simulation-Based Acquisition) 기술로 특화되어 집중적으로 개발되고 있는데, SBA기술은 지금까지 다양한 분야에서 수행되어 왔던 시뮬레이션과는 다르게 PBS(Physics-based Simulation)을 구현하는 기술이며 고정밀 모델의 개발 및 실시간 해석 능력의 확보가 필수적이다.

이러한 고정밀 모델의 개발 및 실시간 해석 능력의 확보를 위해서는 수퍼컴퓨팅 기술에 기반한 초정밀 해석 능력의 확보 및 개발이 선행되어야 하기 때문에 미국은 바로 그리드 컴퓨팅 개발과 같은 수퍼컴퓨팅 기술의 개발에 엄청난 투자를 하고 있는 것이다. 그리고 이러한 초정밀 해석 능력에 기반한 M&S 기술은 국방분야뿐만 아니라, 모든 제품의 설계 및 개발에 활용될 수 있기에 더욱 중요한 것이다.

결국 수퍼컴퓨팅 기술에 기반한 초정밀 해석능력을 통한 시뮬레이션 기술의 개발과 여러 분야에의 응용기술이 국가 과학기술 경쟁력의 척도가 될 것이라고 볼 때, 우리 나라의 현실에 적합한 개발 방향의 설정 및 추진이 시급하다고 볼 수 있다. 우리 나라의 경우에 선진국에 비해 상대적으로 해석 능력 보다는 설계 및 개발 경험의 부족이 큰 단점인 과학기술분야가 많다고 볼 수 있다.

항공우주 비행체의 설계 및 개발 과정 역시 개념설계단계에서부터 최종제작에 이르는 과정동안 설계 변경 및 그에 따른 각종 실험 및 수치해석 등의 반복과정을 거치게 되므로 많은 비용과 시간 인력이 필요하므로 설계 및 개발 경험이 많이 부족하며, 전통적인 방식으로는 선진국과의 기술격차를 좁힌다는 것은 거의 불가능한 실정이다. 따라서 인터넷 수퍼컴퓨팅 기술의 실용화 및 보급을 통해 보다 경제적으로 수퍼컴퓨팅을 구현할 수 있다면 가상 공간상에서 다양한 설계 및 개발을 통해 새로운 기술혁신을 보다 더 빠른 시간에, 보다 적은 비용으로 수행할 수 있을 것이다.

8. 참고문헌

[1] J. Dongarra, *Performance of Various Computers Using Standard Linear Equations Software*, http://www.netlib.org/benchmark/performance.ps

[2] Intel Corp., *"Intel Pentium 4 and Intel Xeon™ Processor optimization Reference Manual,"* http://developer.intel.com

[3] S. Goedecker, and A. Hoisie, *Performance Optimization of Numerically Intensive Codes*, SIAM, Philadelphia, PA, 2000

[4] J. Dongarra, I. Duff, D. Sorensen, and H. van der Vorst, *Numerical Linear Algebra for High-Performance Computers*, SIAM, Philadelphia, PA, 1998

[5] J. J. Dongarra, J. Du Croz, S. Hammarling, and R. J. Hanson, "An extended set of FORTRAN Basic Linear Algebra Subprograms," ACM Trans. Math. Soft., 14 (1988), pp. 1-17.

[6] A. Petitet, R. C. Whaley, J. Dongarra, and A. Cleary, *HPL - A Portable Implementation of the High-Performance Linpack Benchmark for Distributed-Memory Computers*, http://www.netlib.org/benchmark/hpl/

[7] R. Whaley, A. Petitet, and J. Dongarra, "Automated Empirical Optimizations of Software and ATLAS Project," http://www.netlib.org/atlas/index.html

[8] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipies in C, 2nd Edition*, Cambridge University Press, 1992

[9] Kumar, V., Shekhar, S., Amin, M., A Scalable Parallel Formulation of the Backpropagation Algorithms for Hybercubs and Related Architectures, IEEE transactions on Parallel and Distributed Systems, Vol. 5, No. 10, pp 1073-1090, October 1994.

[10] Raghuveer, M.R. and Bopardikar, A.S., *Wavelet Transforms Introduction to Theory and Applications, Addison-Wesley*, 1998