UltraSPARC(64bit-RISC processor)을 위한 고성능 컴퓨터 리눅스 클러스터링

김 기 영, 조 영 록, 장 종 권
울산대학교 컴퓨터.정보통신공학부
전화 : (052) 259-1637 / 팩스 : (052) 259-1687 / H.P 번호 : 018-570-5196

HPC(High Performance Computer) Linux Clustering for UltraSPARC(64bit-RISC processor)

Ki Young Kim, Young Rok Cho, Jong Kwon Chang
Dept. of Computer Engineering & Information Technology University of Ulsan
E-mail : rldud@hanmail.net

Abstract

We can easily buy network system for high performance micro-processor, Progress computer architecture is caused of high bandwidth and low delay time. Coupling PC-based commodity technology with distributed computing methodologies provides an important advance in the development of single-user dedicated systems.

Lately Network is joined PC or workstation by computers of high performance and low cost. Than it make intensive that Cluster system is resembled supercomputer. Unix, Linux, BSD, NT(Windows series) can use Cluster system OS(operating system). I'm chosen linux gain low cost, high performance and open technical documentation.

This paper is benchmark performance of Beowulf clustering by UltraSPARC-1K(64bit-RISC processor). Benchmark tools use MPI(Message Passing Interface) and NetPIPE. Beowulf is a class of experimental parallel workstations developed to evaluate and characterize the design space of this new operating point in price-performance.

I. 서론

사람들이 컴퓨터를 만든 목적은 계산 능력향상에 있었다. 현대의 컴퓨터의 원형인 폰 노이만이 만들어진 당시에는 탄도계산 등의 군사적 목적에 쓰였었다. 반면, 응용기술이 발전한 현재, 다양한 분야에서 컴퓨터가 사용되는 것은 더욱 중요해졌다. 게다가 화학계의 분자구조 연구, 생물학계의 DNA 연구조사, 복잡한 Chip 설계와 거기에 대한 해석(verification)을 하는데 고속의 Clock을 가지는 CPU를 필요로 하게 되었다.

그런데, 단일 프로세서의 계산 속도는 물리적인 특성에 의하여 한계점을 가지고 있다. 만약 10GHz의 속도를 낼 수 있는 CPU가 있다고 가정해보자. 10GHz란 1초 동안 10의 10승, 즉 100억이라는 엄청난 주파수이다. 1Clock은 하나의 명령을 처리할 수 있는데 1Clock으로 빛이 어느 정도 움직일 수 있는지 계산해 보면,

3000,000,000,000mm(30만Km : 빛의 속도 30만Km/s) ÷ 10,000,000,000(10GHz) = 30mm

전자는 빛의 약 0.6배이므로, 전자의 속도는 1클럭에 18mm 정도 밖에는 진행하지 못하기 때문에 아무리 클럭을 높여도 최고 속도의 CPU Clock은 10GHz 이상은 어렵다고 한다. 이것이 고속 프로세서의 개발을 곤란하게 하고 있는 최대의 이유이다.[1] 따라서, 고성능 단일 컴퓨터를 이용한 계산은 이미 그 한계가 있음이 증명된 상태이다.

이에 대한 대안으로 다수의 프로세서(CPU)가 하나의 문제를 협동적으로 계산하는 병렬 컴퓨팅(Parallel Com puting)이 등장하였다. 기존 몇몇 벤더에 의해 제공되던 병렬컴퓨터 혹은 슈퍼컴퓨터들은 매우 고가이기 때문에 쉽게 접할 수가 없었다. 한편, 최근 마이크로프로세서들은 뛰어난 성능을 보여주고 있으며 고속

네트워크 또한 널리 보급되었다. 이로 인해 단일 컴퓨터들을 네트워크로 연결함으로써 새로운 개념의 병렬 컴퓨터를 만드는 것이 가능하게 되었다. 또한 리눅스라는 공개된 OS(운영체제)는 강력한 네트워크 성능을 제공하며 소스공개로 인한 자유로운 튜닝이 가능하기 때문에 이들을 위한 OS로 널리 사용되고 있다. 따라서, 본 논문에서도 OS로 리눅스를 사용할 것이다. 이러한 개념의 병렬컴퓨터를 통칭하여 '클러스터(Cluster)'라고 부른다.[2, 6]

다음 표1은 CISC(Complex Instruction Set Computer)방식과 RISC(Reduced Instruction Set Computer)방식의 프로세서의 비교를 나타낸 표이다.

프로세서 종류 / 항목	CISC 프로세서		RISC 프로세서		
	IBM 370/168	VAX 11/780	IBM 801	RISC I	MIPS
명령어 수	208	303	120	39	55
제어 기억장치	54 KB	61 KB	0 KB	0 KB	0 KB
명령어의 폭	2-6 bytes	2-57 bytes	4 bytes	4 bytes	4 bytes
실행 모델	reg-reg reg-mem mem-mem	reg-reg reg-mem mem-mem	reg-reg	reg-reg	reg-reg

표 1 CISC/RISC 프로세서의 비교[3]

표1에서 보는 바와 같이 RISC 프로세서들은 CISC 프로세서에 비하여 명령어의 수가 크게 감소하였음을 알 수 있다. 또한 명령어를 실행하는 과정에서 마이크로 코드를 사용하지 않고 전적으로 하드웨어에 의하여 실행되므로 제어 기억장치(control memory)가 필요하지 않다. 명령어 형식이 단순화되고 4 바이트 크기로 고정되었음도 특기할 사항이다. 마지막 비교 항목을 보면, RISC 프로세서에서는 모든 연산들의 실행이 레지스터들 사이에서만 일어나고, 연산 과정에서는 기억장치 액세스가 전혀 없도록 함으로써 일반적인 명령어들은 모두 한 주기(cycle) 만에 실행이 완료될 수 있도록 하였다.[3] 이러한 특징들의 이유로 수치모델에서 동일한 clock의 intel Pentium 시스템들에 비해 약 2.5 배 정도 floating 연산이 빠르다. 반면에 Intel社의 ix86계열 프로세서들에 비해 다양한 H/W의 지원을 받기가 어렵다는 단점도 있다.

참고 문헌들에서는 통상 적으로 사용하는 Intel社의 ix86 CPU, AMD의 저가형 CISC형 CPU를 사용한 사례가 많지만, 위에서 기술한 바와 같이 RISC의 장점을 바탕으로 하여 현재 본 저자의 연구실에서 사용하지 않는 Ultra SPARC 1/K 3대로 RISC형 CPU를 사용한 클러스터를 구현 하려고 한다.

II. 여러 가지 Cluster

클러스터의 종류는 그 사용 목적에 따라서, 구현 방법에 따라서, 또 시스템 구조적인 차이에 따라서 여러 가지로 나눌 수 있다. 이 분류는 위의 종류에 따라 명확히 나눌 수 있는 것은 아니지만, 이들을 특징지을 수 있는 방법으로는 알맞은 분류이다.

먼저, 시스템의 구조적인 차이에 따라서 SMP, MPP, NUMA형으로 나뉘어 진다.

- SMP(symmetric multiprocessing) : 다수의 프로세서가 하나의 메모리를 공유 하는 시스템
- MPP(massively parallel processing) : 다수의 프로세서가 각각 독립된 메모리를 가지고 있는 시스템
- NUMA(non-uniform memory access) : 다수의 프로세서의 지역 메모리를 계층적으로 공유 하는 시스템 (MPP를 계선한 시스템)

구현 방법에 따른 분류로는 Active/Standby Cluster, Active/Active Cluster, Shared Nothing Cluster, Shared Everything Cluster, Shared Memory Cluster로 나눌 수 있다. 지면 관계상 모든 설명을 할 수 없으므로 참고문헌을 참조하시길 바란다.[4]

활용 목적에 따른 분류로는 고성능 연산용 클러스터 (HPC : High Performance Cluster), 부하분산 클러스터 (LBC : Loader Balancing Cluster), 고 가용성 클러스터 (HAC : High Available Cluster), 저장 장치클러스터(SAN : Storage Area Network)가 있다.[4] 이 4가지 분류중 본 논문에서 사용하게 될 Cluster는 Beowulf Cluster로 고성능 연산용 클러스터(이하 HPC)에 포함된다. 나머지 클러스터들은 역시 지면 관계상 설명을 생략 하겠다.

HPC는 고성능의 계산능력을 제공하기 위한 목적으로 제작된다. 주로 연구개발(R&D)이나 과학 계산용으로 활용가치가 높으며, 그 종류로는 Beowulf, Score, Enfusion등이 있다.

- Beowulf : 여러 대의 리눅스 시스템을 사용하여 클러스터를 구축하는 방법
- Score : Beowulf Project와 거의 동시에 일본의 기술 연구조합 신정보처리 개발기구에서 1992년부터 10년 계획으로 통산성에서 프로젝트를 위탁 받아 진행. Beowulf보다 더 나은 성능과 편리함을 추구한 클러스터
- Enfusion : 터보 리눅스 사에서 패키지화된 클러스터링 소프트웨어로서 병렬처리가 아닌 평행처리가 특징

이들 중 본 논문에서는 Beowulf를 사용하여 HPC를 구현한다. Beowulf는 PC를 Ethernet과 같은 LAN으로 연결하여 만든 PC 클러스터에서 병렬화한 프로그램을 실행시켜서 슈퍼컴퓨터를 구현한 것이다. 소프트웨어의 이름이나 커널의 이름이 아닌 여러 대의 리눅스를 이용해 클러스터를 구축한 것을 말한다. 보통 대학의 전산과나 소규모 연구소에서 많이 시도하는 방법으로

서 GFLOPS(컴퓨터의 계산 능력: FLOPS = FLoating -point Operation Per Second)급의 슈퍼컴퓨터를 구현할 수 있다. 비용도 절감하고 특정 밴더에 종속되지 않기 위해 베어울프 개발자들은 주로 리눅스를 운영체계로 채택하고 클러스터 내의 메시지 패싱은 표준 프로토콜을 이용한다. 병렬컴퓨팅에서 Beowulf는 MPP와 NOW(Network Of Workstation)보다 조금 낮은 등급으로 분류한다.[2, 4]

Beowulf와 같은 HPC의 개념을 간략한 그림으로 설명을 하자면 그림1과 같다.

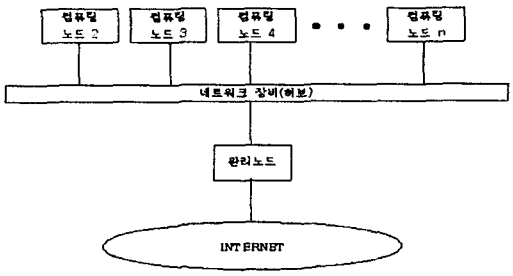

그림 1 일반적인 HPC의 개념도

하나의 관리노드가 프로그램을 병렬로 분리시킨 다음 다수의 n개의 컴퓨팅노드들에게 계산 명령을 주면 컴퓨팅노드들은 그 계산된 결과들을 관리노드에게 돌려주고, 관리노드가 이 계산 값들을 종합하여 최종의 완성된 결과를 얻는 것이다. 이 노드간의 통신에는 MPI(Message Passing Interface) 등의 병렬처리용 라이브러리와 PVM(Parallel Virtual Machine) 같은 병렬화를 위한 패키지를 사용하게 된다.[4] MPI와 PVM에 관해서는 3절에서 자세히 설명하겠다. 본 논문에서 구현되어진 부분은 클러스터를 구성하여 MPI를 이용한 벤치마크 프로그램으로 네트워크 퍼포먼스 측정을 하는 것이다.

III. NetPIPE(Network Protocol Independent Performence Evaluator)에 의한 밴치마크

병렬 라이브러리에 사용되는 Middle ware는 클러스터 시스템일 경우 크게 PVM(Parallel Virtual Machine)과 MPI(Message Passing Interface)로 구분할 수 있다. 양쪽 모두 관련문서가 상당히 많이 있으며 한 가지 차이점을 꼽으면 PVM은 이기종간의 시스템에서 돌아간다는 점과 MPI는 동기종간의 클러스터에서 고려된 라이브러리라는 점이다. PVM vs. MPI 에 관련된 많은 논쟁거리가 꽤 오래전부터 진행되어 왔고 아직도 논쟁이 벌어지는 곳도 있다. 여기서는 동기종 간의 클러스터 시스템으로 꾸밀 것이므로 MPI를 설치할 것이다. MPI는 병렬 라이브러리의 사실상의 표준으로 자리 잡았다. 대표적인 MPI 라이브러리에는 LAM-MPI와 MPICH가 많이 사용된다. 기본적으로 MPI의 통신은 rsh을 이용하며 병렬 프로세서 간 점대점(Point-to-Point) 통신을 한다. MPI를 이용하여 실행할 병렬 프로그램은 논리적, 혹은 물리적으로 클러스터 각 노드 간 동일한 디렉토리에 존재해야 한다.[5]

본 논문에서 사용한 H/W적인 재원은

관리노드(1대), 컴퓨팅노드(2대) <Ultra SPARC-1K(64 bit RISC Process) : CPU-143MHz, Memory-64MB>와 3Com社의 5 port 10/100Mbps Switching Hub 1대를 사용하였다.

NetPIPE는 네트웍 퍼포먼스 측정 벤치마크 프로그램이다. Ethernet 카드, 혹은 네트웍 인터페이스 간에 간단한 Ping-Pong 테스트를 하여 메시지 Size를 변경해가면서 다양한 측정을 한다. 여러종류의 하드웨어를 지원하며, PVM, MPI , MPI-2 등 병렬 소프트웨어와 네트웍 퍼포먼스의 병합측정도 가능하다.[5] NetPIPE의 설치와 사용법에 대해서는 참고문헌의 관련 문서를 참조하기 바란다.

다음은 본 논문의 구현 시스템 상에서 NetPIPE의 측정 결과를 정리한 것이다.

NetPIPE의 양방향 tcp상의 네트워크 측정 퍼포먼스이다. 10/100Mbps의 스위칭 허브를 사용했지만, Ultra SPARC 기기자체가 오래된 재원이라서, 자체의 네트워크 카드(NIC)가 10Mbps로 동작하기 때문에 Band-Width는 10Mbps이하인 Mbps상에서 동작이 된다.

```
[cadlab@wycluster1 NetPIPE_3.3]$ ./NPtcp -r &
[cadlab@wycluster1 NetPIPE_3.3]$ rsh node01
[cadlab@node01 NetPIPE_3.3]$ ./NPtcp -t -h node00
Send and receive buffers are 131070 and 131070 bytes
(A bug in Linux doubles the requested buffer sizes)
Now starting the main loop
  0:    1 bytes    500 times -->    0.03 Mbps in     262.63 usec
  1:    2 bytes    380 times -->    0.06 Mbps in     262.86 usec
  2:    3 bytes    380 times -->    0.09 Mbps in     262.72 usec
  3:    4 bytes    253 times -->    0.12 Mbps in     257.08 usec
  4:    6 bytes    291 times -->    0.15 Mbps in     299.03 usec
 ---------------------------------------------------------------
107: 1572864 bytes  3 times -->    8.35 Mbps in 1437052.17 usec
108: 1572867 bytes  3 times -->    8.42 Mbps in 1425543.51 usec
[cadlab@node01 NetPIPE_3.3]$
```

그림 2 NetPIPE의 양방향 ping-pong테스트 결과

본 논문을 위한 시스템에서의 Peak결과 값은 다음과 같다.

Message Size	Bandwidth	Latency
1.5Mbytes	8.4Mbps	1425543 μ sec

표 2 NetPIPE의 양방향 ping-pong 결과의 최대값

다음은 MPI를 이용한 네트워크 성능을 측정한 결과이다. 여기서는 lam-mpi의 mpi를 측정한 것이다.

```
[cadlab@wycluster1 NetPIPE_3.3]$ mpirun -O -np 2 ./NPmpi
0: wycluster1
1: node01
Now starting the main loop
  0:    1 bytes    500 times -->    0.02 Mbps in     376.99 usec
  1:    2 bytes    265 times -->    0.05 Mbps in     331.93 usec
  2:    3 bytes    301 times -->    0.07 Mbps in     335.20 usec
  3:    4 bytes    198 times -->    0.09 Mbps in     335.05 usec
  4:    6 bytes    223 times -->    0.14 Mbps in     335.66 usec
 ---------------------------------------------
107: 1572864 bytes  3 times -->    8.39 Mbps in 1430233.50 usec
108: 1572867 bytes  3 times -->    8.39 Mbps in 1430256.33 usec
[cadlab@wycluster1 NetPIPE_3.3]$
```

그림 3 NetPIPE에서 MPI를 이용한 네트워크 성능 테스트 결과

본 논문을 위한 시스템에서의 mpich의 mpi 네트워크 최대 성능은 다음과 같다.

Message Size	Bandwidth	Latency
1.5Mbytes	8.3Mbps	1430256 μsec

표 3 NetPIPE에서 MPI를 이용한 네트워크 성능 테스트 결과의 최대값

각 노드의 네트워크 카드가 100Mbps의 성능을 가질 수 있다면 더 좋은 성능이 나올 수도 있겠지만 본 논문을 위한 시스템에서 그러한 것은 지나친 욕심인 것 같아서 위의 결과 값만으로 만족한다. 네트워크 카드의 성능이 10Mbps가 최대이므로, 대역폭은 8Mbps정도로 나타나며, 메시지 크기도 대역폭의 크기에 맞춰 작은 값으로 측정되어졌다.

VI. 결론 및 추후 연구

본 논문에서는 네트워크 성능을 위주로 테스트가 진행 되었다. 현재 사용되는 Gbit-ethernet과는 차이가 나지만 나름대로 적절한 결과가 나왔다. 현재 구현된 시스템(wycluster1)으로 SCALAPACK이라는 전세계 슈퍼컴퓨터 랭킹(http://www.top500.org)을 측정할 때 쓰여지는 벤치마크를 사용하여 CPU의 Computing 성능 테스트를 해보아야겠다. 당연히 RISC쪽 성능이 좋지만, 비슷한 클럭속도를 갖는 CISC프로세서의 Cluster System과 비교를 해서 더 성능이 좋다는 것을 확인 해 볼 수 있을 것이다.

본 논문에서는 3대의 node로 클러스터를 구성했지만, 동일 기종의 하드웨어(본 논문의 재원과 같은 Ultra SPARC 또는 그 이상 성능의 기기)를 구비할 수 있다면 최초의 Beowulf Cluster[1]와 같이 UltraSPARC으로도 더 많은 node로 슈퍼컴퓨터와 대등한 성능의 클러스터를 구현할 수 있을 것이다.

참고문헌 & Reference Site

[1] Satoshi Kawamura, Home Cluster Server by two PC, Linux Magazine, pp. 70-97, Jan. 2001

[2] Chance Reschke & Thomas Sterling & Daniel Ridge, A Design Study of Alternative Network Topologies for the Beowulf Parallel Workstation, IEEE International Symposium on High Performance Distributed Computing, Vol. 15, 1996

[3] 김종현, 병렬 컴퓨터 구조론, 생능출판사, Jul. 1994

[4] 정낙수, Linux for Network, 한컴리눅스, Oct. 2001

[5] 이진호, HPC(High Performance Computer) Linux Cluster HowTo, http://doc.kldp.org/wiki.php/DocbookSgml/HPC-KLDP

[6] 이상문 & 정병수, Linux Clustering, Linux@Work, 신영미디어, pp. 138-168, Oct. 2000

Reference sites

- System Tuning for Linux http://people.redhat.com/alikins/system_tuning.html
- Linux Performance tuning http://linuxperf.nl.linux.org/
- LAM-MPI http://www.lam-mpi.org
- MPICH http://www-unix.mcs.anl.gov/mpi/mpich/
- Linux Kernel Guide http://kldp.org/KoreanDoc/html/Kernel-KLDP/
- NetPIPE Benchmark http://www.scl.ameslab.gov/
- Top500 SuperComputer http://www.top500.org
- Beowolf Project http://www.beowulf.org
- Cluster References http://www.cluster.or.kr

1) 최초의 Beowulf Cluster : 1994년 NASA(미항공우주국)의 고다드 우주항공센터에서 스털링과 베커에 의해 16대의 486컴퓨터를 이더넷과 리눅스 운영체제로 묶어서 당대의 수준급 슈퍼컴퓨터에 필적하는 클러스터를 만들었다.