대학교 PC 실습실 기반의 대규모 병렬·분산 처리 시스템

김상선○, 정갑주
건국대학교 정보통신대학 컴퓨터공학과
{ponyboy○, jeongk}@imc.konkuk.ac.kr

Parallel and Distributed System Based on PC Lab.

Sang-Sun Kim○, Karp-Joo Jeong
Department of Computer Science and Engineering, Konkuk University

요 약

최근 응용과학 분야를 연구하는데 많은 컴퓨팅 자원을 필요로 하고 있다. 예로 미생물학 분야에서 분자 모사를 이용한 바이오-나노 연구는 대규모 컴퓨팅 자원을 필요로 한다. 그와 함께 대규모 컴퓨팅 환경을 구축하기 위해서는 많은 자본이 필요하지만, 대부분의 대학에서는 예산 부족 및 관리 능력 부족으로 인해 이러한 장비를 보유하지 못하고 있는 실정이다. 본 논문에서는 이러한 상황에서 컴퓨팅 자원을 제공 하기 위해 기존의 대학 실습실의 컴퓨터들을 이용해서 대규모 병렬-분산 처리 시스템을 구현 모델로 제시하고 실제 직접 구현한 결과를 보여준다. 구현 결과로는 병렬-분산 처리 시스템인 PLinda 시스템과 애플리케이션인 Raytracing 병렬처리 프로그램을 보여준다.

1 서론

최근 들어 응용 과학 분야에서는 연구의 진행을 단축하기 위해 많은 컴퓨팅 자원을 필요로 하고 있다. 항공기 기체 계산을 하거나 물질의 분자 모사를 이용한 바이오-나노 연구, DNA 염기 서열 분석 등 많은 전문 분야에서 대규모 컴퓨팅 자원을 필요로 하고 있지만 그러한 컴퓨팅 자원을 실제로 구축하기란 많은 자본이 필요하다. 더구나 대부분의 대학에서는 그런 환경을 구축하기에는 턱없이 모자라는 예산과 함께 대형 컴퓨터를 유지 및 관리, 보수를 할 만한 인력이 부족하다. 한편으로 대학에서는 방학이나 밤 시간에는 대학 내의 전산 실습실의 컴퓨터들이나 연구실의 컴퓨터들은 대부분 꺼져 있거나 작동을 하고 있지 않다. 각 컴퓨터들은 최소한 펜티엄-III 이상의 기종으로 각각의 컴퓨팅 파워는 미비할지라도 이러한 컴퓨팅 파워를 모아서 쓴다면 슈퍼 컴퓨터의 성능과 비슷한 수준의 컴퓨팅 파워를 얻을 수 있다.

제시된 제안을 가능하게 하기 위해 먼저 대학의 컴퓨팅 자원 환경을 살펴 보면, 대학 내 전산 실습실의 주된 컴퓨터 OS는 Windows 계열이다. 그리고 소규모 Linux 기반의 Cluster와 Linux를 사용하는 실습실의 컴퓨터들과 Windows 컴퓨터들은 대부분 펜티엄 III 이상의 기종으로 되어 있다. 그리고 네트워크 환경은 10~100Mbps로 연결되어 있다.

위의 상황에서 컴퓨팅 자원을 제공하는 병렬-분산 처리 시스템의 구현 모델로 [1] Linda Model을 기반한 [2] Persistant Linda (이하 PLinda)를 이 논문에서는 제안한다. Linda 모델은 매우 단순 하지만 강력한 분산 처리 모델이다. 또한, 이 모델을 기초로 구현한 PLinda는 트랜잭션과 체크포인팅을 포함되어 있어 강건한 시스템이다.

본 논문의 구성은 2장에서 PC 실습실 유휴자원들의 특성 및 병렬-분산 모델에 대해 살펴보고 3장에서 실제 PLinda 시스템 기반한 전체적 구현모습을 제시하고 4장에서 실제 문제를 Raytracing 예제로 통해 보여주며 마지막 5장에 결론을 다루고 있다.

2 PC 실습실과 병렬 · 분산 모델

2장에서는 PC 실습실의 특성에 대해 분석을 하고 그 특성에 특화 시켜 PC 실습실의 병렬-분산 모델을 제시하고자 한다.

2.1 PC 실습실 특성 분석

대학 PC 실습실의 잠재적 컴퓨팅 파워는 매우 매력적이다. [2] 그 이유는 첫 번째로 각 대학의 PC 실습실들은 수백 대에서 수천 대의 PC(Personal Computers)를 가지고 있고 이러한 자원을 모으면 슈퍼 컴퓨터 성능을 충분히 달성할 수 있다. 두 번째로는 이러한 PC들은 대부분 idle 한 상태이다. 그리고 방학기간에는 완벽히 idle 한 상태로 있다. 세 번째로는 특정 사용자가 사용하는 PC와는 달리 개인 자료들을 저장하고 있지 않아 다른 자료를 쉽게 저장할 수 있고 저장 공간을 쉽게 확보할 수 있다. [3] 마지막으로 학생들의 대학 PC 실습실의 PC 사용 시간이 규칙성을 보여주고 있다. 학기 중 PC사용량이 많은 낮시간에는 빈번히 PC 상태가 변하지만 규칙성을 가지고 있다. 이런 규칙성은 PC의 사용시간대를 예측할 수 있고 예측된 이외의 시간에 PC 실습실의 컴퓨팅 자원을 자유롭게 사용할 수 있다는 사실을 말해준다. PC 상태 사용 규칙성을 잘 이용하면 효율적인 PC 컴퓨팅 자원를 얻을 수 있다. 이와 같이 얻은 PC의 컴퓨팅 자원을 한 곳에 모으면 슈퍼 컴퓨터의 성능을 충분히 얻을 수 있다. 따라서, 위와 같은 이유로, 대학의 PC 실습실은 대규모 병렬-분산 컴퓨팅 자원으로 매우 적합하다는 사실을 알 수 있다.

2.2 PC 실습실 병렬 · 분산 모델

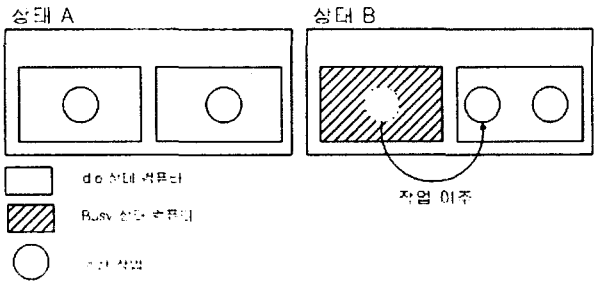
대학 내의 PC들은 학기 중이나 방학 중이라도 학생들

이 사용한다. PC 실습실의 요구사항은 PC 사용자가 불편을 느끼지 못하도록 하면서 PC 실습실의 PC의 자원을 사용하는 것이다. 2.2절은 이런 PC 실습실의 상황을 반영하는 병렬-분산 모델을 제시한다.

불편을 못느끼게 하는 방법은 몇 가지가 있다. PC의 사용률을 사용자가 느끼지 못할 정도로 점유하거나 사용자가 사용하지 않는 시간을 택해 PC를 사용하는 것이다. 하지만 사용자가 이용하는 동안에 PC의 자원을 조금이라도 점유를 한다면 사용자는 조금이라도 영향을 받는다. 따라서, 사용자가 컴퓨터를 사용하는 이외의 시간에 컴퓨터의 자원을 사용해야 한다.

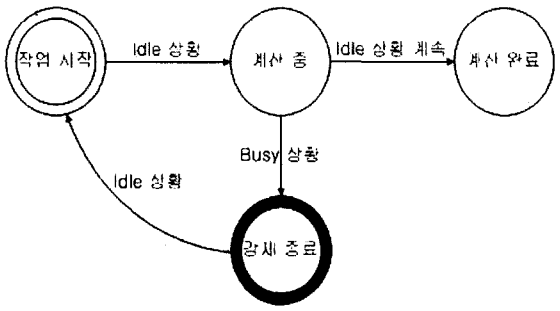
병렬-분산 모델을 제시하기 앞서 사용자가 사용하는 컴퓨터를 Busy 상태의 컴퓨터로 정의하고 사용하지 않는 컴퓨터를 Idle 상태의 컴퓨터로 정의한다. 다음은 PC 실습실의 병렬-분산 모델을 기본 개념이다. 이 개념은 다음과 같이 상태가 변하며 동작된다. 사용자가 Idle 상태의 컴퓨터를 사용하면 바로 컴퓨터 상태는 Busy상태가 돼고 컴퓨터에서 할당받는 컴퓨팅 자원은 바로 반납을 한다. 중단된 작업은 바로 다른 Idle한 상태의 컴퓨터로 이주하여 다시 동작한다.[그림 1] 그리고 사용자가 일정시간 사용을 하고 있지 않는 컴퓨터는 Busy 상태에서 Idle 상태로 바뀐다. [그림 2]은 병렬-분산 모델의 기본 개념을 나타내고 있다.

만일 작업의 특성이 다른 긴작업 계산에는 사용상태가 변하지 않는 컴퓨터 확보가 필요하다. 왜냐하면 긴작업을 수행하는 동안 컴퓨터의 상태가 수시로 바뀌어 계속해서 롤백되는 상황이라면 긴작업은 완료할 수가 없기 때문이다. 따라서, 안정적으로 Idle한 상태의 컴퓨팅 자원을 얻기 위해선 Idle 시간을 예측하여 그 자원을 미리 확보해야만 한다. [2]의 논문에서는 그러한 예측 모델을 두가지를 제시하고 있다. 경험적 모델과 분석적 모델이다. 경험적 모델은 과거의 Idle 상태 상황 데이터를 가지고 예측하는 방법이고 분석적 모델은 HMM(Hidden Markov Model)을 근간한 예측 모델이다. 주목할 점은 두 모델을 비교하면 경험적 모델의 예측 성공률이 전체적으로 높다는 사실이다. 따라서, 경험적 모델을 기초로 해서 스케줄링을 하면 안정적인 컴퓨팅 자원을 얻을 수 있다.



[그림 1] 작업 이주 모습

사용자가 불편을 느끼지 못하도록 하면서 예측 가능한 안정적 컴퓨팅 자원을 확보한 PC 실습실의 컴퓨팅 자원을 모아 슈퍼컴퓨터와 비슷한 성능을 가지는 병렬-분산 컴퓨터를 구축하는 것이 이번장에 제시한 모델의 목표다.



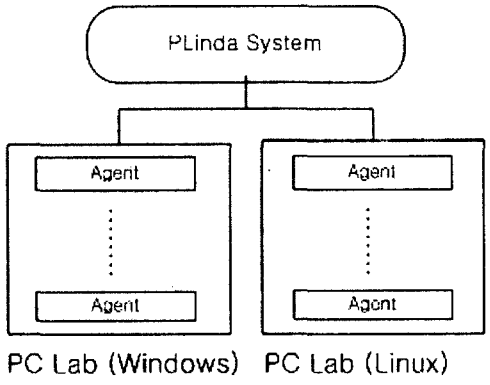
[그림 2] PC 실습실 컴퓨터의 작업 전이도

3 PLinda 시스템 기반 구현

[4] PLinda 시스템은 Linda 모델을 근간으로 트랜잭션과 체크포인팅이 구현된 시스템이다. 또한 PLinda 시스템은 작업 이주와 장애처리를 할 수 있다. 이러한 시스템을 PC 실습실의 환경에 맞게 병렬-분산 모델을 적용과 함께 Linux와 Windows 등 이질적 OS의 자원을 동시에 사용할 수 있도록 설계를 변경하고 구현했다.

3.1 PLinda 시스템 구조

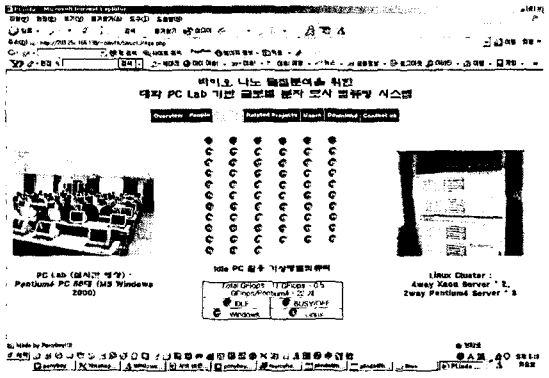
[그림 3] PLinda 전체 구성도를 보면 PLinda 시스템은 agent와 밀접한 관계를 가지고 있다. PLinda 시스템은 크게 PLindaServer, PLindaAdmin으로 나눌 수 있다. 그리고 Agent는 PLindaDaemon으로 구성 되어 있다. 우선, PLindaAdmin은 PLinda 시스템을 제어하는 관리 인터페이스다. PLindaServer는 실제 TupleSpace를 관리하고 Tuple작업들을 스케줄링하는 기능이 있는 핵심 부분이다. 마지막으로 Agent인 PLindaDaemon은 각 시스템의 자원을 실제로 공유하는 역할을 대신 해준다.



[그림 3] PLinda 전체 구성도

Agent는 작업을 Idle한 컴퓨터에 동작시킨다. 만일 Agent가 장애 유발 되거나 Idle한 컴퓨터가 Busy 상태로 변하면, 즉시, 작업은 중단되고 다른 Idle한 컴퓨터를 대

리하는 Agent로 작업은 이주하게 된다. ([그림 1][그림 2] 참조)

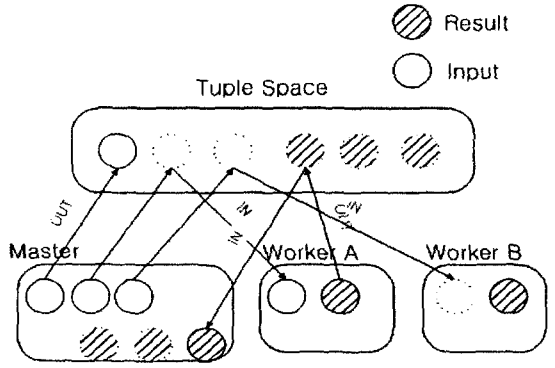


[그림 4] PC 실습실의 실제 Idle/Busy 한 컴퓨터 모니터링 모습 - 가운데 붉은 색 버튼은 현재 busy 하거나 동작 되지 않는 컴퓨터를 표현하며 파란색은 현재 Idle한 컴퓨터를 표시한다.

[그림 4]는 실제 대학 PC 실습실의 모니터링 모습이다. 좌측 그림은 실시간 PC 실습실 동영상을 보여주며 중간의 여러 아이콘들은 그 PC 실습실의 컴퓨터 상황이다. 아이콘 위에 마우스를 올려놓으면 아이콘이 나타내는 PC의 작업상황을 보여준다. 이 모니터링을 통해 Idle 상태에서 Busy 상태로 전이해 작업이 다른 Idle상태의 컴퓨터로 이주하는 모습을 볼 수 있다.

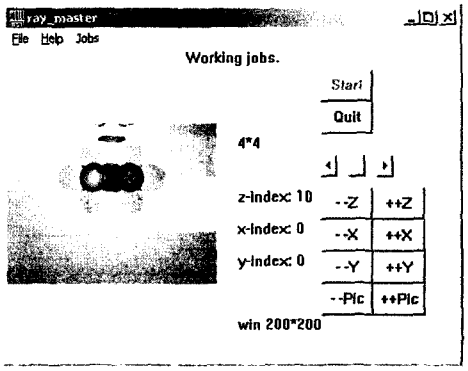
4 예제 구현 : Raytracing

PLinda에 동작 되는 Raytracing을 보여준다. [그림 5]는 Raytracing 과 같은 응용 프로그램들의 핵심 구조인 Master/Worker Model 이다. Master는 작업할 Tuple을 TupleSpace에 올려놓은 다음 Worker에 의해 계산된 결과 Tuple을 TupleSpace로부터 수집하여 최종 결과를 산출하는 역할을 하고 Woker는 Master에 의해 생성된 Tuple을 가져와 계산을 한 다음 그에 대한 결과를 다시 TupleSpace에 올려놓는다.



[그림 5] Master와 Worker 모델의 동작 모형

[그림 6]는 Master, Woker모델을 기초로 만든 실제 Raytracing 실행 중의 모습을 나타내고 있다. Ray_master와 Ray_woker 라는 두 프로그램이 있는데 Ray_master는 실제 물체의 Raytracing할 전체 그림을 각각 나눠 Tuple로 만들어 TupleSpace에 올리면 각 Ray_woker는 TupleSpace로부터 Tuple을 가져와 계산을 한 후 결과 값을 TupleSpace에 다시 올려준다. 최종적으로 Ray_master는 계산될 결과 Tuple을 수집하여 최종 그림을 완성하는 역할을 한다.



[그림 6] 동작 중인 Raytracing 모습

5 결론 및 향후 개선점

지금까지 PC 실습실 특성 분석 및 병렬-분산 처리 모델과 PLinda시스템에 기반한 구현 구조에 대해 설명했다. 그리고 예제로 Raytracing을 제시했다. 이러한 PLinda 시스템을 통한 병렬-분산 처리 시스템은 각 대학의 대규모 컴퓨팅 자원을 지원하는데 매우 효과적이다.

향후 개선점은 WebService를 통한 PLinda시스템을 확장과 함께 트리 구조의 여러 PLinda 시스템이 구성되어 보다 규모가 큰 시스템 구축 및 원격지의 PLinda간의 TupleSpace가 공유되어 대학간의 대규모 컴퓨팅자원을 쉽게 공유 할 수 있도록 구현할 예정이다.

참고 문헌

[1] http://www.cs.yale.edu/HTML/YALE/CS/Linda/linda.html

[2] Suntae Hwang, Eun-Jin Im, Karpjoo Jeong, and Jysoo Lee, An Idle Compute Cycle Prediction Service for Computational Grids, 2003

[3] Suntae Hwang, Karpjoo Jeong, Eun-Jin Im, Chongwoo Woo, Kwang-Soo Hahn, Moonhae Kim, and Sangsan Lee, *An Analysis of Idle CPU Cycles at University Computer Labs. Lecture Notes in Computer Science, 2667 (1)* : 733 -741, 2003

[4] K. Jeong, et all, "PLinda 2.0: A Transactional/Checkpointing Approach to Fault Tolerant Linda", Proc. the 13th International Symposium on Reliable Distributed Systems, 1994