GARS : 그리드 환경을 위한 리소스 매핑 알고리즘

한상렬, 김기형
영남대학교 컴퓨터공학과
srman@yumail.ac.kr, kkim@yu.ac.kr

GARS : Resource Mapping Algorithm for Computational Grid Environment

Sang-Ryoul Han, Kihyung Kim
Dept of Computer Engineering, Yeungnam University

요약

이질적인 계산자원들로 구성된 환경에서 독립적인 작업들을 스케줄링하기 위한 최적의 방법을 찾는 것은 NP-Complete 문제로 알려져 있다[3]. 현재까지 이 문제를 풀기 위한 다양한 휴리스틱 스케줄링 방법이 연구되어 왔다[1][4][5][6]. 본 논문에서는그리드 컴퓨팅 환경을 위한 태스크 매핑 알고리즘을 제안한다. 제안한 알고리즘은 태스크의 완료시간을 계산시간과 통신시간으로 분리하여 노드의 성능과 네트워크의 상태를 감안하여 태스크를 할당하는 네트워크 적응적 매핑 알고리즘이다.

1. 서론

지역적으로 분산되어 있는 이질적인 자원들을 하나로 묶어 거대한 시스템을 구성하고자 하는 연구가 진행되고 있으며, 그 예로서 GRID가 있다. 그리드 환경에서는 구성된 자원들은 특성이 다르기 때문에 같은 작업을 실행할 때 실행 능력이 다르게 된다. 그러므로 작업들을 어떤 자원에 배치할 것인가가 문제가 되고 있으며, 현재 이 문제에 대해 많은 연구가 진행 중에 있다[1][4][5][6].

그리드 구조 중에서 이질적인 환경의 자원들을 하나의 시스템 내의 자원처럼 사용 할 수 있도록 지원해줄 수 있는 시스템을 가장 많이 사용되고 있는 글로버스에서는 정보 서비스 시스템인 MDS(Meta-computing Directory Service)를 이용하고 있다. 그런 단순한 정보제공으로 분산되어져 있는 가용자원의 선택은 불가능하므로, Grid Application Development Software Project에서는 효과적인 자원 검색과 할당 및 모니터링을 위하여 리소스 셀렉션 프레임워크를 제안하였다[7].

이질적인 환경에서의 태스크 할당문제는 NP-Complete 문제로 알려져 있다[3]. 현재까지 이 문제를 풀기 위한 다양한 휴리스틱 알고리즘이 제안되어 있다. 본 논문에서는 동적으로 자원을 검색하고, 할당 할 수 있는 시스템인 리소스 셀렉션 프레임워크의 Mapping Algorithm을 개선하여 자원의 계산 능력과 통신 능력을 고려한 네트워크 적응적 매핑 알고리즘을 제안한다.

본 논문의 구성은 다음과 같다. 2장에서는 그리드 관련 연구를 알아보고, 3장에서는 기존의 알고리즘과 문제점들을 살펴보고, 4장에서는 CPU성능과 네트워크 상태를 적용한 휴리스틱을 제안한다. 5장에서는 실험을 통한 제안한 알고리즘의 성능을 평가하고, 6장에서 결론을 맺는다.

2. 관련연구

Grid Computing

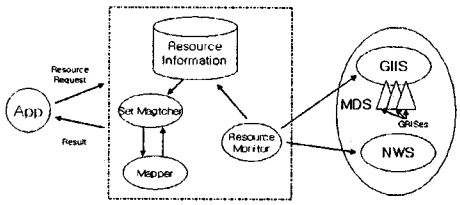
"그리드"라는 용어는 1990년 중반 지리적으로 분산된 고성능 컴퓨터, 대용량 데이터베이스 및 첨단 장비등 정보통신 자원을 네트워크로 연동함으로써 기초 과학과 산업 기술 연구에 필수적인 고속연산,

대량의 데이터 처리, 첨단 장비의 상호 공유 등을 가능하게 할 뿐만 아니라 사이버 공간에서 협업 연구나 작업을 가능하게 해주는 새로운 개념이다[2]. 그리드는 대규모 자원공유, 혁신적인 응용 프로그램, 고기능 등에 초점을 맞추고 있다는 측면에서 전통적인 분산 컴퓨팅 환경과는 구분된다[11].

그리드 구조는 범용적인 구성 요소들을 위한 요구 사항을 나타낸다. 많은 기관들이 Grid 응용 서비스를 개발하고 있으며 그 중에서 대부분이 글로버스라는 미들웨어를 사용한다. 글로버스는 작고 대부분이 기존의 표준을 기반으로 하는 프로토콜들이 채택되었다.

리소스 셀렉션 프레임워크

각 어플리케이션별로 상이한 가용자원 검색과 할당을 단일 플랫폼으로 통합하고자 제안 된 것이 리소스 셀렉션 프레임워크이다[7]. 리소스 셀렉션 프레임워크는 세 가지 구성요소로 이루어져 있는데, MDS에서의 GRIS와 같은 역할을 하는 resource monitor와, 어플리케이션에서 요구하는 자원과 가용자원 가운데 가장 적합한 자원을 찾는 set match, 그리고 마지막으로 선택되어진 자원을 효율적으로 할당하는 mapper로 구성되어있다. <그림 1>은 Resource Selector의 구조를 설명하고 있다.



<그림 1> Resource Selector Architecture

리소스 셀렉션 프레임워크의 mapper에서 사용되는 매핑 알고리즘의 메커니즘은 다음과 같다.

1. 받은 리스트에서 CPU 속도가 가장 빠른 노드의 노드를 선택한다.
2. 선택된 노드에서 가장 빠른 네트워크 속도를 가진 노드를 선택한다.
3. 2번의 과정을 필요로 하는 노드의 수만큼 반복한다.

이 알고리즘은 적어도 WAN 영역에서 같은 클러스터나 도메인이 인접한 위치에 있는 자원을 선택함을 원칙으로 하고 있다.

3. Heuristic 알고리즘

이질적인 자원으로 구성된 환경에서 태스크의 스케줄링 방식에는 정적 스케줄링과 동적 스케줄링으로 분류된다[8]. 정적 스케줄링에는 패치모드 알고리즘이 있는데, 패치모드 알고리즘은 일정시간(일반적인 시간 간격이나, 고정된 회수에 의한 간격)후에 할당되지 않은 태스크를 각 노드로 할당하는 방식이다[1].

알고리즘의 성능 측정을 위해서 사용되어지는 것이 makespan[11]이다. makespan은 전체 작업 중 마지막 작업이 스케줄링 되었을 때 자원들의 이용 가능시간 중 가장 큰 값이다. 각각의 알고리즘에 대해 간단히 알아본다.

Max-min : 태스크가 각 노드에서 실행되었을 경우의 완료시간을 구해서 그 완료 시간들 중 가장 긴 시간을 가지는 태스크를 선택하여 다시 해당 태스크가 가장 짧은 완료시간을 가지는 노드를 선택하여 할당한다. 할당된 노드는 집합 U에서 제거한다. 이 작업을 할당되지 않은 태스크 집합 U의 모든 태스크가 할당 될 때 까지 반복하게 된다[9][3][10].

Relative Cost Algorithm : RC 알고리즘은 위에서 설명한 Max-min, Fast Greedy에서 고려하지 않은 부하 분산 개념을 추가시킨 알고리즘이다.

이질적인 환경에서 태스크들을 할당할 때 부하분산의 개념을 추가시킴으로서 태스크의 작업 실행시간을 줄일 수 있다. 가장 이상적인 알고리즘은 태스크 실행시간과 부하 분산을 동시에 만족하는 것이다[5].

지금까지 살펴본 기존의 알고리즘은 태스크 특정 노드에서 실행 했을 때 걸리는 완료시간을 기준으로 하거나, 네트워크 상태만을 고려하여 태스크를 할당하는 것을 알 수 있다. 이럴 경우 가용자원의 성능 차이가 많을 경우 예상 완료시간과 차이를 보일 수 있다. 다음 장에서 제안한 알고리즘을 설명한다.

4. GARS(Grid Adaptable Resource Selection) 알고리즘

본 논문에서 제시한 네트워크 적응적 매핑 알고리즘은 이질적인 컴퓨팅 환경에서 태스크가 할당될 가장 이상적인 노드를 찾는 알고리즘이다. 일반적으로 태스크 할당 알고리즘들은 태스크들의 완료시간,

GARS(Grid Adaptable Resource Selection) Algorithm

1) 할당 되지 않은 태스크 i, 노드 j라고 할때, i가 j에서 완료시간 = $ETC(i, j)$
2) 노드 집합 Nm에서 CPU성능이 가장 좋은 노드를 선택(selected_node=1)
3) For i=1 to task_number do
3.1) 선택되어진 노드 S에서 β값이 가장 큰 노드 n을 찾는다

$$\beta(S,n) = \max_{1<n<m}(CS(S)/CS_{avg})^{\alpha}/(NL(S)/NL_{avg})^{1-\alpha} = (rCS)^{\alpha}/(rNL)^{1-\alpha}$$

3.2) 선택된 노드는 노드 집합 N_m에서 n을 삭제하고, 선택된 노드 집합 CandidateSet에 n을 추가한다.
3.3) 선택된 노드 집합 CandidateSet의 모든 노드에서 노드 집합 N_m으로의 β값이 가장 큰 노드를 찾는다.

$$\beta(S(i),n) = \max_{1<i<task\_vm}(\max_{1<n<m}(CS(S(i))/CS_{avg})^{\alpha}/(NL(S(i))/NL_{avg})^{1-\alpha} = (rCS)^{\alpha}/(rNL)^{1-\alpha})$$

3.4) 선택된 노드는 노드 집합 Nm에서 n을 삭제하고, 선택된 노드 집합 CandidateSet에 n을 추가한다.
3.5) 모든 태스크가 할당 될 때 까지 3)을 반복한다.

<그림 2> GARS Algorithm

실행시간등 만을 고려한 방식을 사용하고 있다[1][4][5][6]. 리소스 셀렉션 프레임워크의 매핑 알고리즘은 최초 노드 선택 시에만 CPU성능을 고려한다.[7].

본 논문에서 제안한 알고리즘은 그리드 환경에서 태스크의 완료시간을 계산시간과 통신시간으로 나누어서 태스크의 작업 성향에 따라 비율을 조정하여, 태스크의 작업성향에 따라 할당되는 노드가 달라지게 함으로써, 결과론적으로는 보다 빠른 태스크 완료시간을 가질 수 있게 된다.

<그림 2>는 알고리즘의 pseudo code이다. 알고리즘을 살펴보면, 우선 스케줄링될 노드와, 그 노드에서 작업될 태스크가 주어진다. 태스크는 특정 노드에 의존적이지 않고, 노드의 성능에 따라 실행시간이 변한다고 가정한다.

여기서 사용되어지는 β 값은 기준이 되는 노드에서 선택될 노드로의 상대적인 성능 비를 구하는 것이다. 상대적인 CPU 성능비(rCS)와 상대적인 네트워크비(rNL)를, 주어진 태스크의 작업 성향값(α)을 적용시킴으로서 β 값을 계산한다.

$$rCS(i) = CS(i) \;/\; (\sum_{1<n<m} CS(n)/m)$$

$rCS_{(i)}$는 기준 노드에 연결되어진 모든 노드(m)의 CPU 평균치에서 기준 노드에서 선택한 노드($CS_{(i)}$)의 CPU성능비를 나누어준 값이다.

$$rNL(i) = NL(i) \;/\; (\sum_{1<n<m} NL(n)/m)$$

$rNL_{(i)}$는 기준 노드에서 연결되어진 모든 노드(m)의 네트워크 평균치에서 선택된 노드($NL_{(i)}$)의 네트워크 상태를 나누어준 값이다.

α값은 0<α<1의 값을 가지며, 1에 가까울수록 태스크의 작업 성향은 계산적(computation)을 띈다는 의미이며, 값이 0이라는 의미는 태스크의 작업이 단순히 통신만을 한다는 것이고, 1이라는 의미는 서로간의 통신은 없고 단순히 계산만 한다는 의미이다.

위에서 주어진 rCS값과 rNL값, α 값을 가지고, β 값을 구할 수 있는데, 노드 k에서 노드 i로의 β 값은 다음과 같이 정의 한다.

$$\beta = ((rCS_{(i)})^{\alpha} / ((rNL_{(i)}))^{(1-\alpha)}$$

β 값이 크다는 의미는 CPU의 계산 성능이 다른 노드보다 상대적으로 뛰어나고, 네트워크 지연율이 상대적으로 빠르다는 의미를 나타낸다.

제안한 알고리즘은 β 값이 큰 노드를 선택함으로서, 태스크의 작업성향에 따라 이상적인 노드를 선택하게 된다.

5. 성능평가

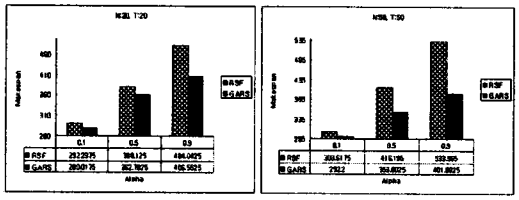
본 연구에서 제안한 알고리즘의 성능을 비교하기 위하여 기존의 그리드 상에서의 리소스 매핑 알고리즘인 RSF 알고리즘과 비교 실험하였다.

실험하기 위해 필요한 값들인 태스크 정보, 노드 정보등을 임의적으로 생성하거나, 직접 입력하는 방식을 택하였으며, 각 노드는 작업이 할당될 시 바로 작업 가능하다고 가정하였다.

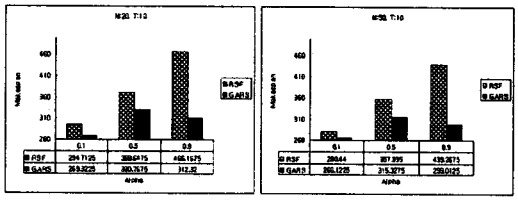
실제 자원의 실행시간은 작업의 프로파일 등 여러 가지 방법을 통하여 구할 수 있다[1][2]. 실험은 α값의 변화에 따른 최고 완료시간을 관찰하였다. 각 실험은 여러 번(50~100회)의 실험 후 평균값을 가지

고 그림으로 나타내었다.

<그림 3>은 할당할 태스크와 노드 수가 동일하고, 모든 노드의 CPU성능과 모든 태스크의 실행시간이 유사할 때 α값의 변화를 주면서 성능평가를 하였다.



<그림 3> 태스크수와 노드수 동일



<그림 4> 태스크수와 노드수 상이

<그림 4>는 노드의 수가 할당할 태스크의 수 보다 많고, 할당할 태스크의 완료시간이 모두 유사할 때 태스크의 작업 성향에 따라 성능평가를 하였다. 전체적으로 볼 때 네트워크 상태가 일정하면, 제안한 알고리즘은 기존의 알고리즘들 보다 좋은 성능을 나타낸다. RSF의 매핑 알고리즘은 네트워크 성능만을 고려하여 자원을 할당하므로 α값이 1에 가까울수록 제안한 알고리즘보다 성능이 저하 되는 것을 볼 수 있다. 그리고 할당될 노드가 많을 경우 제안한 알고리즘의 성능은 더 좋은 것으로 나타난다. 제안한 알고리즘의 결과는 네트워크와 노드의 성능을 고려함으로서 α값이 0.5를 기준으로 보다 좋은 결과를 나타내고 있다. 실험을 통하여 본 논문에서 제안한 알고리즘은 태스크의 성향에 따라 기존의 알고리즘보다, 변화가 크게 있지 않으며, 일정한 성능을 내는 것으로 나타났다.

6. Conclusion

이질적 자원으로 구성된 환경은 어떤 작업을 실행할 때 그 작업이 요구하는 특성에 따라 다양한 실행시간의 차이를 보인다는 점에서 균질의 자원으로 구성된 환경과 차이가 난다. 각 자원에서의 실행시간이 이질적 자원으로 구성된 환경에서의 독립적인 작업들을 스케줄링 하는 알고리즘을 설계하는데 중요한 부분을 차지한다는 것을 실험을 통하여 확인하였다. 본 논문에서 제안한 네트워크 적응적 알고리즘은 태스크의 작업 성향을 고려하여 설계되었다.

그리드 환경에서 계산그리드의 작업은 항상 네트워크 성향을 띄고 있기 때문에, 기존의 RSF 매핑 알고리즘보다는 우수한 성능을 나타낸다.

참고문헌

[1] M.Maheswaran, S.Ali, H.J.Siegel, D.Hensgen, and R.F.Freund, "Dynamic Matching and Scheduling of a Class of Indenpendent Tasks onto Heterogeneous Computing System," Proc, of the 8th Heterogeneous Computing Workshop, pp. 30-44, April, 1999

[2] Czajkowski,K, Fitzgerald,S., Foster,I. "Grid Information Services for Distributed Resource Sharing," 2001.

[3] O.Ibarra, C.Kim. "Heuristic algorithms for scheduling independent tasks on nonidentical processors." Journal of the ACM, 77(2):280-289, Apr, 1977.

[4] H.Barada, S.M.Sait, and N.Baig, "Task Matching and Scheduling in Heterogeneous Systems using Simulated Evolution," Proc, of the 15th Parallel and Distributed Processing Symposium, pp. 875-882, 2001

[5] M.Wu, W.Shu, "A High-Performance Mapping Algorithm for Heterogeneous Computing Systems" Parallel and Distributed Processing Symposium., Proceedings 15th International, Apr 2001

[6] B.Hamidzadeh, L.Y.Kit, D.J.Lilja, "Dynamic Task Scheduling using Online Optimization" Journal of Parallel and Distributed Systems, vol. 11, pp. 1151-1153, 2000.

[7] C.Liu, L.Yang, I.Foster, D.Angulo, "Design and Evaluation of a Resource Selection Framework for Grid Applications" in proceeding of th 61th IEEE symposium on High-Performance Distributed Computing, July 2002.

[8] M.Maheswaran, T.D.Braun, H.J.Siegel, "Heterogeneous Distributed Computing," Encyclopedia of Electrical and Electronics Engineering, J. G. Wdbster, editor, John Wiley & Sons, vol. 8, pp. 679-690, 1990

[9] R.Armstrong, D.Hensgen, T.Kidd, "The relative performance of various mapping algorithms is independent of sizable variances in run-time predictions" 7th IEEE Heterogeneous Computing Workshop(HCW '98), Mar. 1998, pp. 79-87

[10] R.F.Freund, M.Gherrity, S.Ambrosius, "Scheduling resources in multi-user, heterogeneous, computing environments with SmartNet" 7th IEEE Heterogeneous Computing Workshop (HCW '98), Mar. 1998, pp. 13-17

[11] I.Foster, C.Kesselman, and S.Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organization." Jounal of High-Performance Computing Applications, vol. 15, no. 3, pp. 200-222, 2001.