휴리스틱 메소드를 이용하여 응용 QoS 를 보장하는 복제기반 스케줄링 기법에 관한 연구

안병도*, 김홍수+, 이상근+
*고려대학교 컴퓨터정보통신대학원 미디어공학과
+고려대학교 컴퓨터학과
e-mail : an@korea.ac.kr*, hera@disys.korea.ac.kr+, yalphy@korea.ac.kr+

Study on the Replication based Scheduling for Application QoS with Heuristic Method

Byung-Do An*, Hong-Soo Kim+, Sang-Keun Lee+
*Dept. of Media Science and Engineering, Korea University
+Dept. of Computer Science and Engineering, Korea University

요 약

분산형 데스크탑 그리드 시스템에서 안정적인 연산 수행을 위한 노드 구성 기법과 동적인 환경에 적응적인 스케줄링 기법은 필수 요소이다. 그러나 기존 연구에서는 자원 제공자의 휘발성에 적응적으로 대처하지 못하는 연산 수행 모델을 사용하였기 때문에 데드라인 내에 전체 작업을 완료해야하는 응용에 대처하지 못하는 문제점이 발생한다. 이에 본 논문에서는 자원제공자의 성능과 논리적인 위치기반으로 자가 조직적 연산 오버레이 네트워크(Computation Overlay Network) 구성 기법과 자원제공자의 작업 완료 확률과 신용도 값을 이용하여 응용 QoS 보장을 위한 휴리스틱기반 복제 (Heuristic based Replication) 기법을 제안한다. 성능평가에서는 기존 스케줄링 기법과 자원제공자의 작업 완료 확률과 신용도에 따른 분포를 이용한 복제기반 스케줄링 기법을 비교평가 한다.

1. 서론

데스크탑 그리드 컴퓨팅은 하나의 수행 코드와 각기 다른 입력 값을 가진 여러 개의 작업으로 구성된 병렬 컴퓨팅 응용에 적합하며, SETI@HOME[1], Distributed.net[2] 등의 분산컴퓨팅 프로젝트의 성공과 비즈니스적 관심으로 인해 국내외적으로 주목을 받고 있다. 최근 BOINC[4], Javelin[5], Condor[6] 와 같은 데스크탑 그리드 컴퓨팅에 대한 하부 플랫폼을 제공하기 위한 분산컴퓨팅 기술 연구가 활발히 진행되고 있다.

데스크탑 그리드 컴퓨팅 환경에서의 자원제공자 (작업자 또는 노드로서 연산을 수행하는 주체)는 인터넷으로 연결된 데스크탑을 기반으로 하고 있다. 이러한 자원제공자들은 고장(failure), 휘발성(volatility), 이질성(heterogeneous)과 같은 특성들로 인해 안정적인 연산 수행을 보장할 수 없다. 특히, 자원제공자는 휘발성으로 인해 아무런 제약 없이 자유롭게 연산에 참여할 수 있거나 탈퇴할 수도 있다. 이러한 자원제공자의 특징으로 인해 수행하는 연산이 중간에 중지되는 현상이 발생한다. 따라서, 데스크탑 그리드 컴퓨팅 환경에서 작업의 연속성을 보장해 줄 수 있는 스케줄링 기법이 필요하다.

데스크탑 그리드 컴퓨팅 환경에서의 자원들은 휘발성이 강한 데스크 컴퓨터를 기반으로 연산이 수행된다. 분산형 환경에서 기존 스케줄링 기법들을 사용한다면 성능저하 문제가 발생될 수 있다. 선종결 우선 할당 기법 (eager scheduling)이나 FIFO (First-Come First Serve) 스케줄링 같은 기존 스케줄링 기법들은 작업을 빨리 수행하는 자원제공자에게 먼저 작업을 할당하는 단순 알고리즘만을 적용하여 다음과 같은 문제점이 발생된다. 첫째, 자원제공자의 휘발성과 고장에 스스로 대처하지 못하고 있다. 둘째, 자원제공자의 특성(성능, 가용성, 신뢰도)을 반영하지 못하고 있다. 셋째, 동적으로 변화하는 연산 환경으로 인한 성능이 저하되는 문제가 있다. 넷째, 스케줄링 기법이 중앙관리 서버에 의해 중앙집중식으로 수행되어 확장이 용이하지 못하다. 또한, 데스크탑 그리드 컴퓨팅 환경에서 데드라인 (deadline) 내에 전체 작업을 완료 해야 하는 응용 QoS 를 가정할 경우 기존 스케줄링 기법은 이에 대처할 수 없다.

본 논문에서는 분산된 방법으로 응용 수행에 필요한 오버레이 네트워크를 자율적으로 구성하고, 동적인 연산 환경에서 데드라인 내에 연산을 완료 해야 하는 응용 QoS 보장을 위한 복제기반 스케줄링 기법을 제안하고자 한다. 먼저, 본 연구에서는 자원 검색, 작업 할당, 그리고 통신 지연 비용을 줄이기 위하여 자원제공자의 정적인 정보 (자원제공자 성능, 논리적인 거리)에 따라 분산된 방법으로 연산 그룹을 구성하는 연산 오버레이 네트워크를 구성한다. 다음으로, 본 논문에서는 연산 오버레이 네트워크 상에서 연산의 안정성과 신뢰성을 보장하기 위해 자원제공자의 동적인 정보 (작업 완료 확률, 신용도)를 이용한 복제 기법을 통하여 데드라인 내에 전체 연산을 완료할 수 있는 복제기반 스케줄링 기법을 제안한다.

2. 관련연구

2.1 연산 오버레이 네트워크 구성기법

기존의 분산형 데스크탑 그리드 시스템은 트리 형태의 오버레이 네트워크를 구성하는 Organic Grid[7], 그래프 형태의 랜덤 네트워크(Random Network)를 사용하는 Messor[8], 구조적(Structured) 네트워크를 사용하는 CCoF[9] 등이 있다.

Organic Grid[7] 에서는 데이터 분배 및 스케줄링을 위해 성능을 고려한 트리 형태의 오버레이 네트워크를 구성한다. [7]에서 루트노드(클라이언트 노드)에 결함이 발생한다면 모든 작업은 중단된다. 또한, 노드의 가용성을 고려하지 않았기 때문에 오버레이 네트워크의 잦은 재구성이 일어나고, 이로 인해 시스템 전체 성능이 저하되는 문제점이 발생한다.

Messor[8]는 P2P 환경의 이동 에이전트 시스템을 이용한 응용으로, 비구조적으로 이루어진 무작위 네트워크에서 스케줄링 및 자원관리가 이루어진다. [8]에서는 노드 와 노드 간에 작업에 대한 부하균형을 위해서 과도한 메시지를 사용하는 단점이 있다. 또한 오버레이 네트워크에 대한 별도의 구성 및 재구성을 하지 않기 때문에 노드 결함 시 그에 대처 할 수 있는 방법이 부재하다.

2.2 스케줄링 기법

BOINC(Berkeley Open Infrastructure for Network Computing)[4]에서는 자원제공자가 작업 수행을 서버에 요청하면 서버는 작업을 자원제공자에게 할당하는 단순한 스케줄링 기법을 제공한다. 또한, 자원제공자가 수행한 작업의 정확성을 보장하기 위해 하나의 작업을 여러 개 복제하여 수행하는 복제 방법을 이용하고 있다.

Javelin[5] 에서는 트리기반 선종결 우선할당 스케줄링기법을 제시한다. Javelin 에서의 트리 는 자원제공자의 특성을 고려하지 않은채 구성되기 때문에 해당 노드가 고장 날 때마다 계속해서 트리를 다시 구성해야 하는 단점을 가지고 있다. 또한, 자원제공자들의 동적인 특징을 고려하지 않는 선종결 우선할당 스케줄링 기법으로 인해 성능 저하의 문제점이 발생된다.

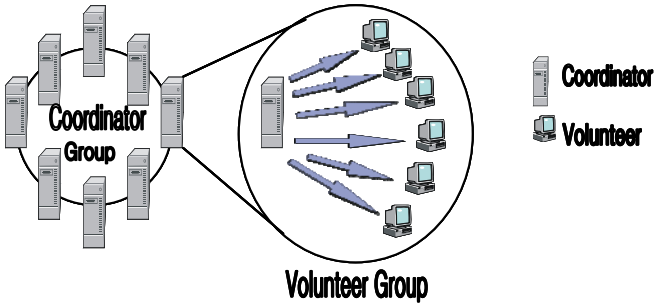
CCOF(Cluster Computing On the Fly)[9]는 CAN 을 이용하여 동일시간대(낮시간 과 밤시간때)를 기반으로 오버레이 네트워크를 구성한 다음 이를 이용한 웨이브 스케줄러(wave scheduler)를 제안한다. 웨이브 스케줄링에서는 작업을 수행하고 있는 자원제공자가 아침시간 때로 접어들면 밤시간 때에 있는 다른 자원 제공자로 작업을 이주시키는 방법을 사용한다.

3. 데스크탑 그리드 시스템 모델

본 논문에서 가정하는 데스크탑 그리드 시스템에서 필요한 노드 구성방식, 작업 분배 방식, 작업 수집 방식 등에 대해 살펴본다. 본 논문에서 가정하는 분산형 데스크탑 그리드 시스템에서는 작업 수행에 참여하는 노드의 역할에 따라 클라이언트 노드, 조정자 노드, 자원 제공자 노드로 분류한다. 클라이언트 노드는 수행하고자 하는 응용을 가진 노드를 의미한다. 그리고 클라이언트는 메시지 전송을 통해 오버레이 네트워크를 구성하게 되고, 오버레이 네트워크가 조직화되면 다른 노드는 가용성에 따라 조정자 노드와 자원 제공자 노드로 분류된다.

그림 1 과 같이 가정하고 있는 분산형 데스크탑 그리드 시스템 모델은 노드 구성 과정을 통해 오버레이 네트워크를 구성한 뒤, 클라이언트가 조정자 노드에 작업묶음을 위탁하면 조정자 노드는 단위 작업을 자원제공자에게 위탁한다.

1) 오버레이 네트워크 구성: 자원제공자의 정적인 정보 (자원제공자 성능, 자원제공 시간, 논리적인 거리)를 기반으로 조정자 노드와 자원제공자 노드로 구분하여 오버레이 네트워크를 구성한다.
2) 작업 묶음 위탁: 구성된 오버레이 네트워크를 통해서, 클라이언트 노드는 조정자 노드에 작업 묶음을 위탁한다.
3) 단위 작업 분배: 조정자 노드는 다수의 자원제공자 노드에게 단위 작업을 분배한다.
4) 작업 수행: 단위 작업을 분배 받은 자원 제공자 노드는 해당 작업을 수행한다.
5) 단위 작업 수집: 단위 작업을 완료한 자원 제공자 노드는 해당 작업의 결과를 조정자 노드에 반환한다.
6) 작업 묶음 수집: 단위 작업들에 대한 수집을 한 조정자 노드는 작업에 대한 결과를 모아 클라이언트에 반환한다.



(그림 1) 링 기반 작업 오버레이 네트워크

4. 연산그룹 구성 기법 및 스케줄링

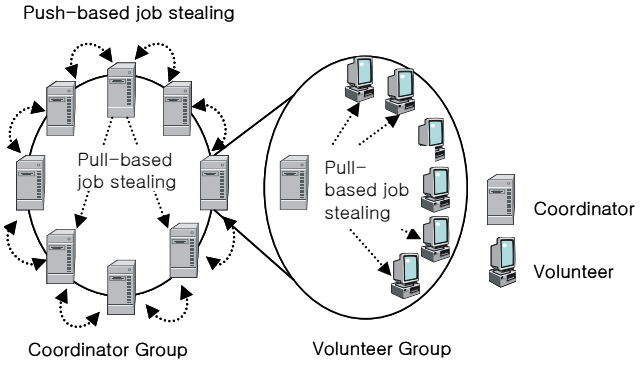
4.1 자원제공자 연산 그룹 구성

본 연구에서는 자원의 정적인 정보(자원제공자 성능, 자원제공 시간, 그리고 논리적인 거리)로 분류된 자원제공자를 기반으로 오버레이 네트워크를 구성한다. 먼저 그림 2 와 같이 구성되는 링 기반 오버레이 네트워크는 자원제공자를 크게 조정자 그룹(coordinator group)과 자원제공자 그룹(volunteer group)으로 나누어진다. 그림 3 은 트리 기반 작업 오버레이 네트워크를 나타낸다. 트리 기반 오버레이 네트워크는 크게 클라이언트 계층 (client level), 조정자 계층(coordinator level), 자원제공자 계층(volunteer level)으로 구성된다.

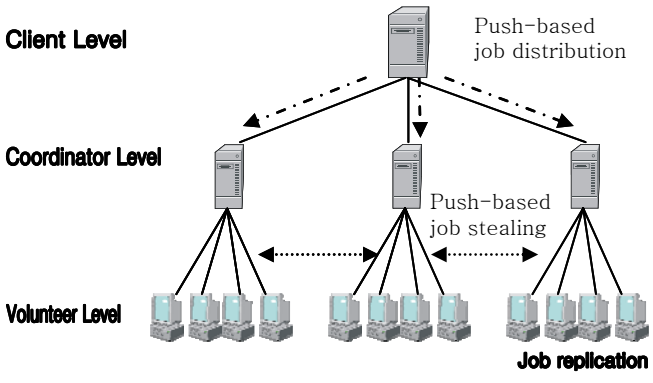
여기서는 링 기반과 트리 기반 작업 오버레이 네트워크에 동작하는 연산 수행 모델을 기술한다. 조정자 그룹에서는 클라이언트로부터 주어진 작업 묶음을 조정자 그룹의 조정자들 에게 작업을 분배하는 역할을 한다. 이때 클라이언트에서는 Push 기반 작업 분배 방식을 사용하여 조정자들 에게 작업을 분배한다. 또한, 조정자 그룹에서는 조정자들의 작업부하가 불균등하게 된 경우에는 작업부하가 없는 조정자는 작업부하가 높은 조정자 에게 작업을 가져오는 작업 가로채기(jop stealing) 방식을 사용하게 된다. 자원제공자 그룹에서는 자원제공자 그룹의 조정자가 각 자원제공자에게 단위 작업을 분배하게 된다. 자원제공자는 자신의 작업을 수행한 후 연산 결과를 조정자 에게 반환하게 된다. 이때 조정자 는 자원제공자에게 다른 작업을 더 주게 된다. 즉, 자원제공자 그룹에서는 Pull 기반 작업 분배 방식을 사용하여 단위 작업을 분배하게 된다. 만약 자원제공자의 가용성, 신용도가 낮은 경우에는 안정적인 연산 수행을 위해 작업을 복제하여 다른 자원제공자에게 분배하게 된다.

트리 기반 작업 오버레이 네트워크에서 연산 수행 모델은

그림 3 과 같이 구성된다. 먼저 클라이언트 계층에 클라이언트 는 작업묶음을 Push 방식으로 분배하게 된다. 작업묶음을 받은 조정자들은 자원제공자 계층의 자원제공자들에게 단위 작업을 분배하게 된다. 만약 자원제공자가 단위작업을 수행한 경우에는 자신의 부모 노드인 조정자 계층의 조정자 에게 작업을 재요청하게 된다. 만약 자원제공자 계층의 자원제공자가 단위작업을 수행한 경우에는 자신의 부모 노드인 조정자 계층의 조정자에게 작업을 요청하게 된다. 요청을 받은 조정자 는 단위작업을 다시 자원제공자에게 할당하게 된다. 이와 같이 Pull 방식으로 작업요청과 수행이 이루어지게 된다. 또한, 자원제공자의 연산 완료 확률, 신용도, 자원제공자시간이 낮은 경우에는 자원제공자의 휘발성으로 인한 연산 지연 및 실패현상이 발생한다. 이러한 현상을 극복하기 위해 본 논문에서는 자원제공자 계층에서 복제 기법을 사용하게 된다. 자원제공자의 동적인 정보를 바탕으로 데드라인 내에 결과를 되돌려줄 확률 값과 자원제공자의 신용도에 따라서 복제의 대상 및 수를 결정함으로써 기존의 복제 기법에서 발생하는 정적 복제수에 의한 자원의 낭비 문제와 그로 인해 발생하는 성능저하 문제를 해결하고자 한다.



(그림 2) 링 기반 작업 오버레이 네트워크 의 연산수행 모델.



(그림 3) 트리 기반 작업 오버레이 네트워크 의 연산수행 모델

4.2 자원제공자 특성에 기반한 복제 기법

데스크탑 그리드 환경에서 데드라인 내에 전체 연산을 완료 해야 하는 응용들의 요구사항을 만족시키기 위해서는 자원제공자의 휘발성 같은 연산 결함을 극복하여야 한다. 본 논문에서는 동적인 데스크탑 그리드 컴퓨팅 환경에서 자원 제공자의 휘발성으로 인한 연산 중단 및 실패를 복제 기법을 이용하여 해결하고자 한다. 복제 기반 스케줄링 기법에서는 자원제공자의 동적인 정보로써 가용성과 성능을 바탕으로 데드라인 내에 결과를 되돌려줄 확률 값과 자원제공자의 신용도에 따라서 복제의 대상 및 수를 결정함으로써 기존의 복제 기법에서 발생하는 정적 복제수 에 의한 부하 문제와 그로 인해 발생하는 성능저하 문제를 해결하고자 한다. 먼저 자원제공자가 데드라인 내에 연산을 완료할 확률을 정의한다.

[정의] 연산 완료 확률 Γ : *자원제공자의 연산 완료 확률은 연산 결함 환경에서 데드라인 내에 연산을 완료할 확률이다.*

자원제공자 i 가 임의의 작업을 완료할 때까지 평균 λ_i 를 갖는 지수분포를 따른다고 할 때 자원제공자 i 가 작업 j 의 데드라인 d_j 내에 작업을 완료할 확률인 Γ_i 는 다음과 같이 계산된다.

$$\Gamma_i(D \le d_j) = \int_0^d \lambda_i e^{-\lambda_i d_j} = 1 - e^{-\lambda_i d_j} \quad \text{수식 (1)}$$

여기에서, D 는 자원제공자의 연산 수행 시간을 나타낸다. λ_i 는 자원제공자 i 의 평균 결함율(failure rate)로서 자원제공자 i 의 과거 수행한 이력을 바탕으로 데드라인 내에 평균 결함이 발생한 횟수로서 계산된다.

[정의] 신용도 Ψ : *신용도는 자원제공자가 정상적으로 연산을 수행할 수 있는 지를 판단하는 요소이다.*

자원제공자 신용도는 자원제공자가 조정자로부터 작업들을 할당 받아 데드라인 내에 몇 개의 작업을 수행했는지에 따라 결정되는 값으로 다음과 같이 계산된다.

$$\Psi_i = \frac{k}{n} \quad \text{수식 (2)}$$

수식 (2)에서 n 은 자원제공자 i 가 연산 수행을 위해 할당받은 작업의 개수를 나타낸다. 그리고 k 는 과거에 자원제공자 i 가 데드라인 내에 연산을 수행한 평균 개수를 나타낸다.

수식 (2)에서 n 은 과거에 자원제공자 i 가 데드라인 내에 연산을 수행한 평균 개수를 나타낸다.

다음으로 우리는 자원제공자의 연산 완료 확률과 신용도에 따른 복제 기법에 대해 기술한다. 본 논문에서 복제 기법은 하나의 자원제공자가 데드라인 내에 연산 완료 확률이 응용이 요구하는 임계값 Γ_θ 을 넘지 못할 때 자원제공자 그룹에서 선택 정책(selection policy)에 따라 자원제공자를 선택하여 연산 그룹 (computation group)을 구성한다. 구성된 연산 그룹에 대한 연산 완료 확률 Γ_g 이 Γ_θ 를 만족하는지 조사하여, 만약 만족한다면 현재 연산 그룹에 작업 j 를 분배한다. 그렇지 않다면 또 다른 자원제공자를 선택하여 Γ_g 를 계산하게 된다. Γ_g 는 연산 그룹에서 하나 이상이 작업을 완료할 수 있는 확률로서 다음과 같이 계산된다.

$$\Gamma_g = 1 - \prod_{i=1}^{n} \overline{\Gamma_i} \quad \text{수식 (3)}$$

수식 (3)에서 $\overline{\Gamma_i}$ 는 자원제공자 i 가 데드라인 내에 연산

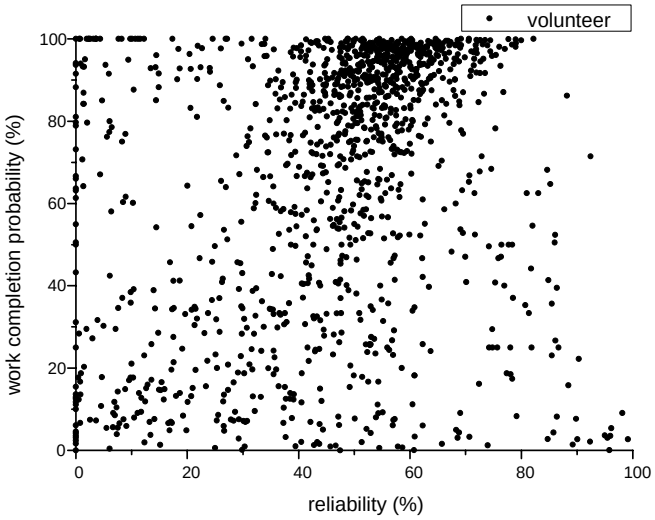
을 완료하지 못할 확률로서 $1-\Gamma_i$ 를 나타낸다.

연산 그룹에 대한 연산 완료 확률은 자원제공자 그룹에서 자원 선택 알고리즘을 통해서 자원을 선택하게 된다. 본 논문에서는 자원제공자 그룹에서 가장 연산 완료 확률이 높고 신용도 값이 좋은 자원제공자를 선택하기 위해서 $MAX(\Gamma_i \times \Psi_i)$ 에 해당하는 자원제공자를 선택하여 연산 그룹을 구성하게 된다.

5. 성능평가

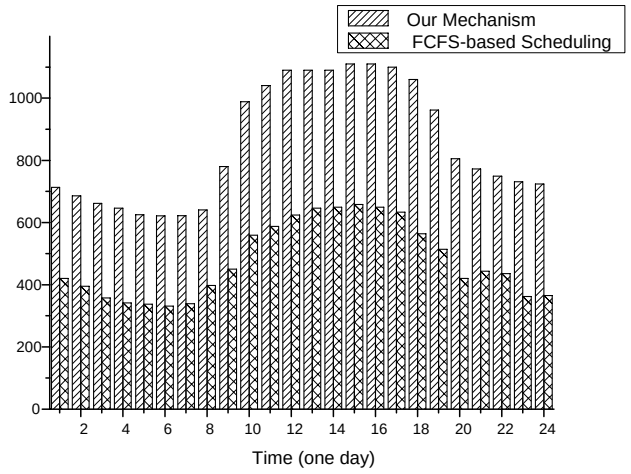
본 논문에서 제안하는 복제기반 스케줄링 기법의 성능을 평가하기 위해서 Korea@Home 데스크탑 그리드 시스템[3] 테스트 베드를 이용하여 실험적으로 구현하였다. Korea@Home 은 인터넷에 연결된 데스크탑 컴퓨터들의 유휴 컴퓨팅 시간을 이용하여 그리드 응용을 수행하는 데스크탑 그리드 시스템이다. 성능 평가를 위해 사용된 응용은 생명공학 분야에서 가상 탐색 기술을 기반으로 신약 후보 물질을 찾아내는 응용이다. 이 응용에서 하나의 작업은 결함이 없는 자원제공자에서 평균적으로 약 16 분이 소요되었다.

Korea@Home 데스크탑 그리드 시스템에서 일주일 동안 각 자원제공자들에 의해 수행된 로그 정보를 분석하여 그림 4 와 같은 분포를 얻었다. 그림에서, x 축에서 신뢰도 값은 작업을 받아간 것에서 실제 결과를 반환한 비율로서 수식(2)와 같이 계산되었다. y 축은 결과 완료 확률로서 각 자원제공자의 과거 수행 이력을 바탕으로 데드라인 내에 결과를 되돌려줄 확률을 수식(1)을 기반으로 계산된 값을 나타낸다. 자원제공자 분포는 스케줄링 시에 $MAX(\Gamma_i \times \Psi_i)$ 값에 따라서 자원제공자를 선택하기 위해 사용되었다.



(그림 4) 작업 완료확률과 신용도에 따른 자원제공자들의 분포

그림 (5)에서 성능 평가는 각 자원제공자가 조정자에게 보내온 작업 결과로서 측정하였다. 조정자는 한 시간 단위로 Linpack benchmark [10]에 의해 데스크탑 그리드 시스템의 성능을 평가한다. 기존연구에서 사용된 FCFS 기반 스케줄링 기법에서는 자원제공자가 작업 풀에 작업을 요청하면 자원제공자 풀에서 랜덤으로 자원제공자를 선택하여 작업을 수행한 결과에 대해 성능을 측정하였다. 성능 측정 결과, 기존의 FCFS 를 사용한 기법보다 본 논문의 복제기반 스케줄링 기법이 모든 시간대에서 약 1.6 배 정도의 성능 향상을 보였다. 이는 본 논문에서 제안하는 기법이 기존 기법에 비해 스케줄링 시에 자원제공자의 동적인 특성으로 인해 발생될 수 있는 성능저하를 복제기반 스케줄링 기법을 통해서 해결하였기 때문이다.



(그림 5) 복제기반 스케줄링 기법과 FCFS 기반 스케줄링 기법과의 성능 비교

6. 결론

본 논문에서는 데스크탑 그리드 컴퓨팅 환경에서 자원제공자의 동적인 정보를 이용한 복제 기법을 통하여 데드라인 내에 전체 연산을 완료할 수 있는 복제기반 스케줄링 기법을 제안 했다. 성능평가 결과 데드라인을 지원하는 응용에서 기존의 FCFS 기법을 이용한 스케줄링 보다 본 논문에서 제안한 복제기반 스케줄링 기법이 자원제공자의 작업 완료 확률과 신뢰도를 이용한 분포를 통하여 성능향상을 보일 수 있었다.

참고문헌

[1] SETI@home, “http://setiathome.ssl.berkeley.edu”
[2] Distributed.net, “http://distributed.net”
[3] Korea@Home homepage. http://www.koreaathome.co.kr
[4] David P. Anderson, “BOINC: A System for Public-Resource Computing and Storage”. GRID,2004
[5] M. O. Neary and P. Cappello, “Advanced eager scheduling for Java-based adaptive parallel computing,” Concurrency and Computation: Practice and Experience, Vol. 17, Iss. 7-8, pp. 797-819, July 2005.
[6] D. Thain, T. Tannenbaum, and M. Livny, “Distributed Computing in Practice: The Condor Experience,” Concurrency and Computation: Practice and Experience, Vol. 17, Iss. 2-4, pp. 323-356, Feb. 2005.
[7] Arjav J. Chakravarti, Gerald Baumgartner and Mario Lauria, “Self-Organizing Scheduling on the Organic Grid”, International Journal of High Performance Computing Appolications, Vol. 20, pp. 115-130, 2006
[8] Alberto Montresor, Hein Meling and Ozalp Babaoglu, “Messor: Load-Balancing through a Swarm of Autonomous Agents”, Agents and Peer-to-Peer Computing, LNAI 2530, pp. 125-137, 2003
[9] V. Lo, D. Zhou, D. Zappala, Y. Liu, and S. Zhao, “Cluster Computing on the Fly: P2P Scheduling of Idle Cycles in the Internet,” 3rd Int. Workshop on Peer-to-Peer Systems, LNCS 3279, pp. 227-236, 2004.
[10] J. Dongarra, "Performance of various computers using standard linear equations software," *ACM SIGARCH Computer Architecture News*, Vol. 20 , pp. 22-44, June 1992.