사이언스 클라우드 기반 과학 응용 스케줄링 적응성 실험

송내영 김서영 박선아 김윤희[1] 김종암
숙명여자대학교 컴퓨터과학과, 서울대학교 기계항공공학부
{nai0315, sssyyy77, gloliana, yulan}@sookmyung.ac.kr,
chongam@snu.ac.kr

Scheduling Adaptability Experiments for Scientific Applications on Science Cloud

Naeyoung Song Seoyoung Kim Sunah Park Yoonhee Kim[1]
Chongam Kim
Dept. of Computer Science, Sookmyung Women's University
Mechanical and Aerospace Engineering, Seoul National University

요 약

클라우드 컴퓨팅 환경에서 가상 머신을 기반으로 스케줄링 하는 것은 IaaS(Infrastructure-as-a-service)로써 사용자가 원하는 가상 환경을 만들어 필요에 따라 서비스로 제공해 줄 수 있다는 이점을 가진다. 하지만 자원을 필요로 하는 여러 가지 작업이 동시에 이루어져야 할 때에 스케줄링의 불편과 자원의 비효율적 사용을 초래할 수 있다. 따라서 작업을 포함한 가상 머신들을 사용자의 요청에 따라 적절한 시간에 스케줄링 해 주는 것이 필요하다. 본 논문에서는 사이언스 클라우드 환경을 구축할 수 있는 오픈네블라(OpenNebula)[1]와 함께 Haizea 스케줄러[2]를 사용하여 사용자들의 작업 요청을 스케줄링 하는 것을 보인다. Haizea 스케줄러는 작업을 포함한 가상머신들의 우선순위와 자원의 활용도를 기반으로 가상머신을 보류/재시작(suspend/resume)하여 스케줄링 한다. 이는 전체 실행 시간을 줄이고 높은 우선순위를 가지는 작업을 포함하는 가상머신을 먼저 실행할 수 있도록 한다. 대규모 컴퓨팅자원을 필요로 하고 실행시간이 상대적으로 긴 과학 계산 어플리케이션을 대상으로 우선순위에 따라 가상머신을 실행하여 과학계산 클라우드 환경의 적응성을 실험하였다.

1. 서 론

클라우드 컴퓨팅 환경에서 가상 머신을 이용하여 작업을 하는 것은 사용자가 원하는 가상 환경을 만들어 필요에 따라 서비스로 제공해 줄 수 있다는 이점을 가진다. 가상 머신을 사용하는 또 다른 장점은 한정된 자원을 효율적으로 사용할 수 있다는 점이다. 구성된 환경을 이미지화해서 관리하는 어플라이언스의 형태이기 때문에 동일한 환경의 복사와 이동이 쉽다는 점 또한 장점이다.

현대 사회에서 막대한 자원을 필요로 하는 과학적 응용 연산의 수행이 많아지고 그러한 수행들을 클라우드 컴퓨팅 환경에서 실행시키는 일이 빈번해 졌다. 그 이유는 과학 응용 연산이 가지는 그들만의 특성 때문에 클라우드 컴퓨팅 환경에서 수행시키는 것이 가지고 오는 유리함 때문이다. 이로써 편리함을 얻고 정확성을 보장 받는다는 이익은 있지만 아직도 수행 시간이 길어서 오는 불편함과 적절한 자원의 선택과 사용에 있어서 많은 연구가 필요하다.

클라우드 컴퓨팅 환경에서 작업 단위로 스케줄링

1) 교신저자 (Corresponding Author): 김윤희

* 이 논문은 2009년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No.2009-0084669).

하는 것 또한 문제를 제기한다. 자원을 필요로 하는 여러 가지 작업이 동시에 이루어져야 할 때에 스케줄링의 불편과 자원의 비효율적 사용을 초래할 수 있다는 것이다. 따라서 작업이 아닌 가상 머신을 기반으로 스케줄링하고, 작업을 포함한 가상 머신들을 우선순위에 따라 적절한 시간에 스케줄링 해 주는 것이 필요하다.

본 논문에서는 과학 응용 연산이 클라우드 컴퓨팅 환경에서 스케줄링 될 때 그 적응성을 실험하였다. 클라우드 컴퓨팅 환경을 구축할 때 미들웨어로서 오픈네블라(OpenNebula)를 사용하였고 가상 머신을 기준으로 스케줄링 하는 Haizea 스케줄러를 사용하였다. 본 논문의 첫 번째 실험에서는 과학 응용 연산을 단일 머신 상에서 순차적으로 수행하는 것보다 가상 머신을 사용하여 수행했을 때, 작업을 병렬적으로 동시에 처리할 수 있기 때문에 시간적인 측면에서 더 유리함을 보였다. 두 번째 실험에서는 우선순위에 따라 스케줄링을 할 때, 작업을 기준으로 스케줄링 하는 것과 가상 머신을 기준으로 스케줄링 하는 것의 비교를 통해 시간과 자원의 효율성을 보였다.

본 논문의 나머지 부분의 구성은 다음과 같다. 2장에서는 사이언스 클라우드에 대해 알아보고 3장은 오픈네블라의 구조에 대해 설명한다. 또한 4장에서는 구현과 실험한 내용을 기술하고, 끝으로 5장에서 결론과 향후 연구 방향에 대해 제시한다.

2. 사이언스 클라우드

클라우드 컴퓨팅 환경의 구축에 대한 연구가 많이 진행 중이다. 그 중에서 클라우드 컴퓨팅 환경의 미들웨어로는 오픈네블라(OpenNebula)와 Nimbus 등이 있다. 이와 같은 미들웨어를 사용하여 구축한 클라우드 컴퓨팅 환경에서 사용할 수 있는 스케줄러로는 가상 머신을 기반으로 스케줄링 하는 Haizea 등이 있다. Reservoir[3]는 미들웨어로 오픈네블라(OpenNebula)를 사용하는 클라우드 환경을 가지는 프로젝트이다. 이러한 클라우드 환경에서 실행하는 작업들은 과학적인 응용 연산이 많은데 다음 절에서 그러한 작업들의 특성에 대해 알아본다. 그 후에는 클라우드 컴퓨팅 환경에 대한 연구들에 대해서 알아본다.

2.1 과학 응용의 특징

과학적 응용 연산의 속성과 특성들을 이해하는 것은 중요하다. 예를 들어 전산 유체 역학(Computational Fluid Dynamic, CFD)은 여러 매개변수 집합들로 다수의 실험들을 컴퓨터를 이용해서 수치해석적인 방법으로 계산한다. 이 때 연구자는 동일한 해석 대상에 다양한 파라미터를 적용시킴으로써 조건 변화에 따른 다양한 실험 결과 데이터를 얻기를 원한다. 이 실험들은 계산 집중적이고 독립적으로 실행될 수 있기 때문에, 고 효율성을 필요로 하고 막대한 계산을 하는 자원들을 많이 필요로 한다. 이에 따라 계산 작업 수행 시간도 길어지기 때문에 작업 할당에 있어 효율적인 작업 수행이 이루어지도록 작업을 스케줄링 해야 한다.

2.2 오픈네블라(OpenNebula)

오픈네블라(OpenNebula)[1]는 클라우드 컴퓨팅 환경을 구성할 수 있는 오픈 소스 툴킷으로서 여러 대의 물리적 머신들을 네트워크로 묶어 가상 머신 상에 작업을 분산하여 처리할 수 있는 클라우드 환경을 만들어 준다. 또한 오픈네블라(OpenNebula)는 가상 머신을 만들고, 만들어진 가상 머신들을 모니터링 하고 제어하는 기능을 제공한다.

2.3 Nimbus

Nimbus[4]는 오픈네블라(OpenNebula)와 같은 레벨의 미들웨어로써, 클라우드 컴퓨팅을 실현하기 위해 가상 머신의 실행환경을 동적으로 설정할 때, 특히 소프트웨어의 동적 설정을 가능하게 하는 기술이다. 가상화를 기반으로 하는 클라우드 컴퓨팅을 실현하려는 경우 가상 머신의 실행 환경을 동적으로 설정하는 것이 중요하다. 여기서 가상 머신의 실행환경이란 실제 물리적 컴퓨터로부터 할당 받은 자원의 량(CPU, Memory 할당량)과 소프트웨어 설정 상태를 말한다. Nimbus는 가상 머신에 동적으로 소프트웨어를 배치하고 설정하는 것에 관심을 두고 연구를 진행하고 있는 프로젝트 및 시스템 이름이다. Nimbus가 제안하는 이러한 가상 머신 관리 기술은 현재 오픈 프로젝트로써 누구나가 이를 테스트해 보고 관련 방법을 수정해 볼 수 있다. 최근 들어 Nimbus의 연구는 다양한 수의 소프트웨어 환경을 대상으로 동적인 설치 및 환경 설정 방법을 연구 중에 있으며 더불어 아마존 EC2 서비스와 같은 기존의 가상화 기반 클라우드 컴퓨팅 서비스와의 호환성을 연구하고 있다.

2.4 Reservoir

Reservoir(Resources and Services Virtualization without Barrier)[3]는 IT서비스를 유틸리티로서 효율적이고 안정적인 배포 및 운영하기 위한 차세대 컴퓨팅 클라우드 프로젝트이다. 특히 단일 클라우드 컴퓨팅의 제한된 확장성과 클라우드 컴퓨팅 서비스

간 인터오퍼러블리티의 부재를 핵심 쟁점으로 지적하고, 이를 해결하기 위해 Open Federated Cloud Computing 플랫폼을 연구한다. 기업 비즈니스 요구에 맞추기 위해서 SLA(Service Level Agreement) 관리 및 동적 자원제어 알고리즘을 개발하고, 국가적, 지역적으로 분산되고 서로 다른 정책, 제도 하에 관리되는 클라우드 컴퓨팅 테스트베드들 사이의 인터도메인 상호연동을 가능케 하기 위한 기반 연구에 중점을 둔다. 기본적으로 그리드 기술과 가상화 기술을 기반으로 구축될 예정이다.

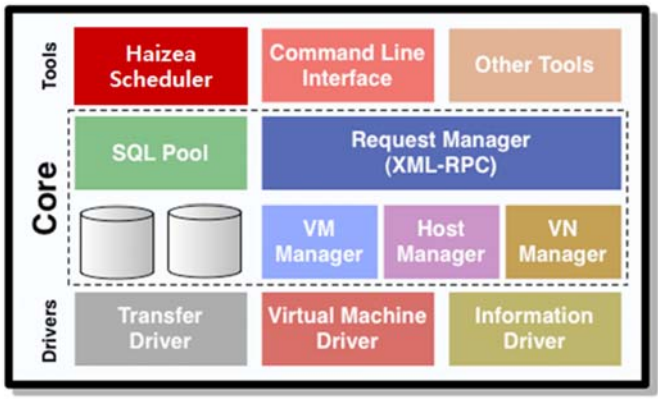
2.5 Haizea

Haizea 스케줄러[2]는 advanced_reservation(AR)과 best_effort, 그리고 immediate의 3가지 방식의 스케줄링을 제공한다. AR은 특정한 시간에 자원들을 필요로 하며, best_effort는 자원들이 가능한 대로 사용한다. 마지막으로 immediate는 요청 될 당시에 자원들이 있으면 사용하고 없다면 거절된다.

Haizea 스케줄러는 그 자체로서는 시뮬레이터로써의 역할만 한다. Haizea 스케줄러가 물리적인 자원들에 대해서 실행되게 하려면 오픈네블라(OpenNebula)와 같이 실행되어야 한다. 따라서 오픈네블라(OpenNebula)는 자체적인 스케줄러 대신에 Haizea 스케줄러를 사용하여 가상 머신을 스케줄링 한다.

3. 사이언스 클라우드에서의 효율적인 작업 스케줄링 모델

오픈네블라(OpenNebula) 오픈 소스 툴킷은 그림1과 같은 구성을 가진다.



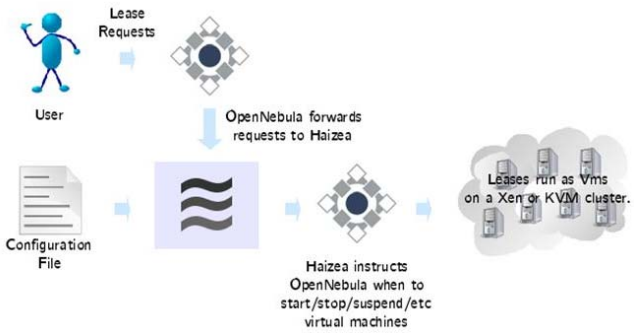
(그림 1) 오픈네블라(OpenNebula)의 구조

오픈네블라(OpenNebula)는 드라이버로서 transfer driver, virtual machine driver, information driver를 가진다. Transfer driver는 가상 머신을 만드는데 사용하는 디스크 이미지를 가상 머신을 실행시킬 로컬 머신에 전송하는 데 사용된다. Virtual Machine driver는 가상 머신을 띄울 하이퍼바이저를 말하며, information driver는 가상 머신을 모니터링 할 때 각 가상 머신에 대한 정보를 주고받을 수 있는 드라이버를 말한다.

가상 머신들에 대한 정보는 데이터베이스에서 sql을 사용해 관리하고 가상 머신, 가상 머신을 실행하는 로컬 머신, 가상 네트워크의 상태를 주기적으로 체크하여 사용자에게 알려준다. 오픈네블라(OpenNebula)의 API는 XML-RPC를 통해서 제공된다. 가상 머신의 관리와 물리적 머신, 가상 네트워크는 이 인터페이스를 통해서 제어된다.

Command line으로 가상 머신을 만들라는 명령어를 주면 Haziea 스케줄러는 오픈네블라로 엮인 클라우드 컴퓨팅 환경에서 가상 머신을 스케줄링하여 적절한 node를 결정한다. 그러면 head node는 그 node에게 디스크 이미지를 transfer driver를 이용해 전송하고 전송 받은 node는 virtual machine driver를 이용해 가상 머신을 실행시킨다.

Haizea 스케줄러는 그림 2와 같은 방식으로 가상 머신을 스케줄링 한다.



(그림 2) 오픈네블라(OpenNebula)와 Haziea의 상호작용

사용자가 가상 머신을 만들 자원을 요청하면 오픈네블라(OpenNebula)가 Haizea로 스케줄링을 요청한다. Haizea는 가상 머신을 보류(suspend)하고 재시작(resume) 등의 명령을 오픈네블라(OpenNebula)에게 전달하고, 오픈네블라(OpenNebula)는 그 명령어들을 통해 가상 머신을 제어할 수 있다.

4. 구현 및 실험

본 논문에서 구축한 클라우드 환경에서 실험한 과학 응용 연산은 항공 역학에 관한 작업들로써, 작업들은 작은 크기의 입력 데이터를 받아들여 많은 자원들을 이용해 거대한 크기의 계산을 요구한다. 작업들 사이에는 의존성이 없기 때문에 여러 개의 가

상 머신에 작업들을 분산하여 실행할 수 있다. 첫 번째 실험에서는 과학 응용 연산을 가상 머신을 활용한 환경에서 수행한 결과를 살펴본다. 두 번째 실험에서는 우선순위에 따라 가상 머신을 스케줄링 하는 것을 보여 과학 응용 연산의 클라우드 환경에서의 적응성을 실험하였다.

오픈네블라(OpenNebula)와 Haizea 스케줄러를 이용하여 구축한 클라우드 환경은 표 1과 같다.

(표 1) 실험 자원의 정보

Host name	CPU arch.	Total memory
palas	i686	4109892
flora	i686	4051192
iris	x86_64	4051196

총 3대의 머신들은 모두 Intel Core 2.67GHz의 프로세서를 가진다. palas머신은 클라우드 환경의 head node로써 오픈네블라 1.4.버전(OpenNebula 1.4)과 Haizea 스케줄러를 설치했다. Virtual Machine Driver인 가상화 하이퍼바이저로는 KVM[5]을 사용했다. 실험에서 사용되는 가상 머신들은 head node에 저장되어 있는 약 0.3GBytes의 디스크 이미지를 이용해 만들어진다. 가상 머신을 보류(suspend)하거나 재시작(resume)할 때에 오픈네블라(OpenNebula)는 디스크 이미지와 파일들을 가상 머신을 실행시켰던 로컬 머신에 저장한다.

4.1 실험 #1

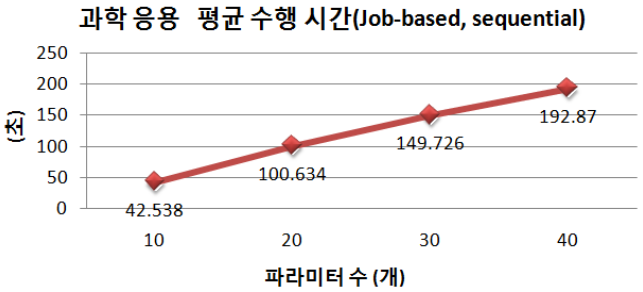
첫 번째 실험에서는 항공 역학에 관한 작업들을 단일 머신에서 실행한 결과와 가상 머신들에 분산해 실행한 결과를 비교한다.

항공 역학에 관한 작업들은 실행할 때, 매개 변수들을 필요로 한다. 여러 매개 변수 집합들을 입력 데이터로 받아 들여 같은 작업을 매개변수만 바꾸어 실행한 후 결과 값을 반환한다. 매개변수의 값이 커지면 커질수록 계산해야 하는 양이 많아진다. 단일 머신에 작업을 실행시킬 때, 파라미터의 개수를 10개 단위로 늘려가며 작업을 수행할 때에 실행 시간을 측정한 것을 표 2로 나타내었다. 5번 실행의 실행 시간을 측정 후 평균 시간을 계산하였다.

(표 2) 반복 횟수에 따른 평균 수행 시간

파라미터 수(개)	수행 시간 평균(초)
10	42.538
20	100.634
30	149.726
40	192.87

단일 머신에서 작업을 수행할 때는 여러 파라미터 집합들을 하나의 프로세서에서 수행해야 하기 때문에 파라미터의 수가 늘어갈수록 실행 시간이 늘어갈 수 밖에 없다. 수행 횟수가 늘어갈수록 수행 시간이 정확히 비례하지 않는 이유는 파라미터 개수뿐만 아니라 값에 따라서 수행 시간이 영향을 받기 때문이다. 표 2를 그래프로 나타낸 것이 그림 3이다.



(그림 3) 파라미터 개수 당 평균 수행 시간의 그래프

같은 작업을 가상 머신에서 수행했을 때는 수행 시간을 효과적으로 줄일 수 있었다. 가상 머신은 동시에 여러 개가 실행 될 수 있기 때문에 파라미터의 집합을 적절히 나누어 작업의 입력 데이터로 주어 실행 시킬 수 있다. 파라미터를 40개를 입력 데이터로 하여 같은 작업을 40회 반복하여 수행하는 것을 가상 머신에서 수행한다고 가정하여 보자. 40개의 파라미터를 10개씩 나누어서 4대의 가상 머신 상의 작업에게 파라미터를 10개씩 주어서 동시에 수행시켰다. 가상 머신 상의 작업이 시작되는 시간은 가상 머신이 activate되는 시간으로 가정하고, 작업이 끝나는 시간은 가상 머신에서 작업을 끝마치고 결과 파일을 head node로 전송하는 시간으로 가정한다. 평균 시간을 계산하기 위해 5번 수행한 결과가 표 3과 같다.

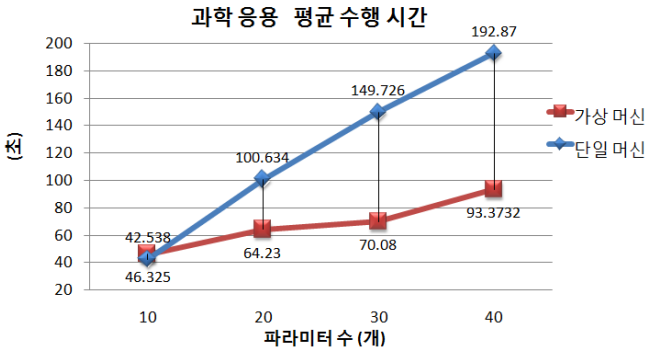
(표 3) 가상 머신 당 수행 시간

	가상머신 1(1-10)	가상머신 2(11-20)	가상머신 3(21-30)	가상머신 4(31-40)
1번째	77.5	93.096	84.091	77.775
2번째	77.7	92.17	85.23	78.9
3번째	75.8	94.55	84.01	76.95
4번째	76.3	93.8	82.95	77.028
5번째	77.5	93.25	83.77	76.88
평균(초)	76.96	93.3732	84.0102	77.4346

가상 머신 4개를 동시에 실행 시킨 후에 결과 값을 받아 오는 것으로써 파라미터를 변경시키며 40

번 반복하는 작업이 끝난다. 모든 가상 머신이 끝나야 작업이 끝나는 것이기 때문에 총 수행 시간은 제일 수행 시간이 긴 가상 머신2의 수행 시간을 따른다. 따라서 파라미터가 40개인 작업을 가상머신 4개에서 돌린 수행 시간은 약 93.37초가 된다.

같은 방식으로 파라미터가 10개, 20개, 30개인 환경에서도 가상 머신 4개를 동시에 돌린다는 가정으로 실험한 후 수행 시간을 측정하였다. 이 수행 시간 측정 결과와 단일 머신에서 실행한 작업의 수행 시간 측정 결과를 하나의 그래프로 그려본 것이 그림 4이다.



(그림 4) 파라미터 수에 따른 평균 수행 시간 비교

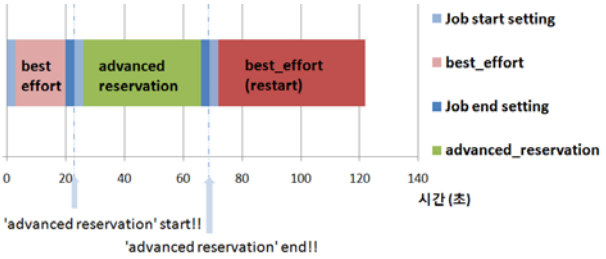
그림 4를 보면 파라미터의 개수가 많아질수록 가상 머신을 사용할 때에 수행 시간 측면에서 이점이 많아짐을 볼 수 있다.

이상의 결과는 단일 머신에서 수행시키는 작업을 가상 머신 4개에 분산하여 수행시킨 다는 조건 하의 결과이다. 결과를 살펴보면 가상 머신에서 작업을 돌릴 때의 수행 시간이 단일 머신에서 작업을 돌릴 때 보다 (1/4 + a)정도로 줄어듦을 알 수 있다. 여기서 a는 가상 머신을 실행 시킬 때 세팅 하는 시간이다. 따라서 가상 머신 n개 기반의 작업을 실행 시킬 때는 그렇지 않을 때 보다 (1/n + a) 만큼의 시간이 걸림을 알 수 있다.

4.2 실험 #2

두 번째 실험에서는 특정한 시간에 실행되어야 하는 작업을 포함하는 가상 머신과 그렇지 않은 가상 머신을 우선순위에 따라 스케줄링 하는 것을 알아본다.

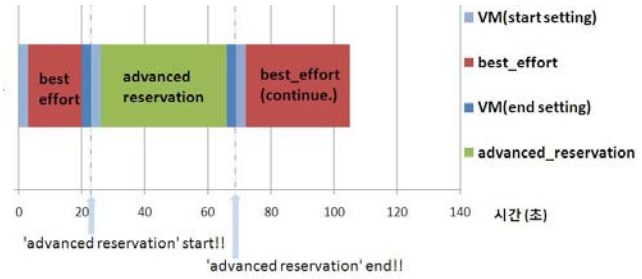
작업을 기준으로 스케줄링 할 때에는 특정한 시간에 요청되어야 하는 작업(Advanced_Reservation) 때문에 당장 시작될 수 있는 작업(Best_Effort)들이 낭비될 수 있다. 그 예가 그림 5와 같다.



(그림 5) 작업 기반 스케줄링에 따른 수행 시간

Best_effort의 작업이 0부터 시작되었다 하더라도 특정 시간에 요청되어야 하는 AR(advanced_reservation)이 시작되기 전에 best_effort는 자원을 선점 당하기 때문에 반드시 중지되거나 끝나야 한다. AR이 끝난 후에 다시 best_effort를 실행 할 수 있지만 전에 실행했던 부분을 보상 받을 수 없고 다시 처음부터 실행시켜야 한다. 이것은 자원의 낭비와 시간의 낭비를 초래한다.

스케줄링의 단위를 가상 머신으로 바꾼다면 이 같은 낭비를 줄일 수 있다. Haizea 스케줄러를 이용한 가상 머신의 스케줄링에서는 가상 머신의 보류(suspend)와 재시작(resume)을 제공하기 때문에 자원이 선점되기 전에 작업을 포함한 가상 머신을 보류(suspend)함으로써 그 때 까지 실행한 결과를 유지할 수 있다. 앞선 실험을 가상 머신 스케줄링 단위로 바꿔 실행한 결과가 그림 6이다.



(그림 6) 가상머신 기반 스케줄링에 따른 수행 시간

그림 6에서는 AR이 시작되기 전에 실행했던 best_effort가 AR이 끝나고 나서도 낭비되지 않는다. 따라서 best_effort는 자신의 작업의 남은 부분만 실행 해 주면 된다. 결과적으로 수행 시간을 작업을 기준을 스케줄링 했을 때 보다 줄일 수 있다.

4. 결론 및 향후 방향

본 논문은 작업 단위로 스케줄링 하는 것 보다 가상 머신 단위로 스케줄링 하는 것이 가상 머신의 보류(suspend)와 재시작(resume)을 이용하여 사용자의 작업 요청을 우선순위에 따라 스케줄링 할 수 있

기 때문에 실행시간 측면에서나 자원의 활용도 측면에서 더 우수함을 보였다. 이러한 스케줄링 모델은 특정한 시간에 시작 되어야 하는 작업이나 클라우드 컴퓨팅 환경의 모든 자원을 사용해야 하는 작업을 스케줄링 할 때 자원의 비효율적 사용과 스케줄링의 불편을 줄이는데 도움을 줄 수 있다.

또한 단일 머신에서 작업을 수행할 때와 가상 머신 기반으로 작업을 수행할 때의 수행 시간을 비교하여, 가상 머신을 기반으로 하는 작업의 수행 시간이 감소함을 보였다. 이는 파라미터 집합을 입력 데이터로 하는 과학 응용 연산의 수행 시간을 효과적으로 감소시키도록 한다.

향후의 연구에서 유연한 방식의 스케줄링을 위해 작업 이력에 맞는 스케줄링 알고리즘을 개발할 것이다. 그에 따라서 작업의 특성에 맞는 스케줄링이 가능하게 되어 효율적인 자원의 활용이 가능해 질 것이다. 또한 현재 그리드 기반의 실험 환경을 확장하면서 그리드와 클라우드 기반을 모두 지원해서 하이브리드 인프라 구축을 목표로 하고 있다.

참 고 문 헌

[1] OpenNebula – The open source toolkit for cloud computing http://www.opennebula.org

[2] Haizea – An open source VM-based Lease manager http://haizea.cs.uchicago.edu

[3] RESERVOIR – Resources and Services Virtualization without Barriers http://62.149.240.97/

[4] NIMBUS – Nimbus is cloud computing for science http://www.nimbusproject.org/

[5] KVM – Kernel Based Virtual Machine http://www.linux-kvm.org

[6] Warren Smith, Ian Foster, Valerie Taylor, "Scheduling with Advanced Reservations," ipdps, pp.127, 14th International Parallel and Distributed Processing Symposium (IPDPS' 00), 2000.

[7] B. Sotomayor, R. S. Montero, I. M. Llorente, and I. Foster, "Capacity Leasing in Cloud Systems using the OpenNebula Engine." Cloud Computing and Applications 2008 (CCA08), 2009.

[8] 조정현, 김병상, 송은혜, 김윤희, 김종암, 정민중, "e-AIRS:협업 및 동적 파라메터 실험을 위한 항공 우주 포탈", 한국정보과학회 가을 학술발표논문집 Vol.33, No.2, 2006.

[9] B. Sotomayor, "A Resource Management Model for VM-Based Virtual Workspaces", Masters paper, University of Chicago, Febuary 2007.

[10] Keahey K., T. Freeman, "Science Clouds: Early Experiences in Cloud Computing for Scientific Applications." Cloud Computing and Its Applications 2008 (CCA-08), Chicago, IL. October 2008.

[11] Sotomayor, B., K. Keahey, I. Foster, T. Freeman, "Enabling Cost-Effective Resource Leases with Virtual Machines", HPDC 2007 Hot Topics session, Monterey Bay, CA. June 2007.

[12] Borja Sotomayor, Rubén Santiago Montero, Ignacio Martín Llorente, Ian Foster, "Resource Leasing and the Art of Suspending Virtual Machines," hpcc, pp.59-68, 2009 11th IEEE International Conference on High Performance Computing and Communications, 2009

[13] 송혜원, 김회영, 윤찬현, "컴퓨팅 클라우드 자원 제어 기술", 정보처리학회지 제 16권 제2호 (2009. 3)