과학 응용 특성에 따른 간섭 방지 스케줄링 실험

안주림 O 김윤희 O*

O숙명여자대학교 컴퓨터과학과

O{julim8990, yulan}@sookmyung.ac.kr

A Case Study of an Anti-interference Scheduling Applying Characteristics of Scientific Applications

Julim Ahn O Yoonhee Kim O*

ODept. of Computer Science, Sookmyung Women's University

요 약

클라우드를 통해 실행되는 응용은 다양해지고 그 요구 또한 증가하고 있다. 클라우드 컴퓨팅 플랫폼은 기존 응용 프로그램의 영역뿐만 아니라 새롭게 등장하는 응용 프로그램의 성능과 자원활용도를 높이기 위해 필요하다. 본 논문에서는 간섭 방지 제어 스케줄링을 제안하여 과학 실험을 위한 통합 컴퓨팅 프레임워크의 설계 및 구현을 제시한다. 실험을 통해 자원 간섭 방지 스케줄링을 자원을 적절히 예약함으로써 성능 저하뿐만 아니라 자원 고갈을 방지 할 수 있는 것을 보였다.

1. 서 론

클라우드 컴퓨팅의 급속한 기술 발전으로 인해 더 많은 응용 프로그램이 다양한 분야의 데이터 센터로 이동되고 있다. 따라서 클라우드를 통해 실행되는 응용은 다양해지고 그 요구도 증가하고 있다. 이러한 변화는 데이터 센터에서의 자원 경쟁을 초래할 수 있으며, 이로 인해 자원 사용률이 낮아지고 데이터 센터가 관리하기에 부담이 될 수 있다. 최적화된 성능과 자원 활용도를 높이려면 컴퓨팅 프레임워크를 설계하는 것이 필수적이다.

통합 프레임워크를 개발하기 위한 관련 연구로는 통합된 다중 레벨 스케줄링과 작업 제출 시스템인 HTCaaS[1]가 있다. 이 시스템은 대규모의 복잡한 과학 계산을 수행하는 것을 목표로 한다. 그러나 세분화된 자원을 할당하는 것보다 대규모 작업 실행에 초점을 맞춘다. 또한 새로운 인프라를 연결하고 확장하기 위해서는 어댑터를 개발해야 한다. 또한 클라우드 환경에 대한 작업 스케줄링 및 자원 할당 연구[2]에서는 응용 프로그램의 현재 작업 부하를 모니터링하고 분석하여 적절한 자원을 할당하는 제어 시스템을 제안했지만, 작업 수행에 초점을 맞추지 않은 자원관리에 초점을 맞췄다.

본 논문에서는 다양한 유형의 자원에 대한 정교한 자원 할당을 제공하는 과학 실험용 통합 컴퓨팅 프레임워크를 설계하고 구현했다.

2. 시스템 아키텍처

본 논문에서는 아키텍처를 소개함으로써 제안한 프레임워크에 대한 구조를 보이고, 실행 시나리오를 예로 들어 프레임워크에서 작업 및 컴퓨팅 자원을 효과적으로 관리하는 방법을 설명한다.

그림 1은 본 논문에서 제안한 프레임워크의 구조[3]를 보여준다.

<그림1 통합 프레임워크의 구조>

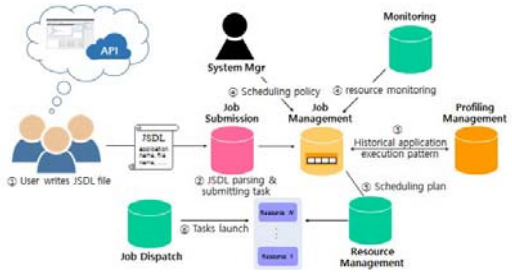
작업 관리와 관련된 모듈로는 작업 제출, 작업 관리 및 작업 발송 모듈이 있다. 작업 제출 모듈은 사용자가 지정한 작업 설명 파일 (JSDL)을 관리하고 구문 분석한다. JSDL[4]은 계산 작업의 요구 사항을 설명하는 확장 가능한 XML 명세이다. 작업 관리 모듈은 주로 작업 스케줄링 및 큐 관리를 담당한다. 워크플로우의 경우 Chronos[5],[6]가 지원하는 종속성을 확인한 후, 동일한 절차가 수행된다. 또한 이 모듈의 주된 역할 중 하나는 작업 스케줄링을 위한 계획을 세우는 것이다. 작업 발송 모듈은 입력 데이터, 실행 파일을 시험하고 할당된 자원으로 작업을 시작한다.

자원 관리와 관련된 그룹은 자원 관리 및 모니터링 모듈로 구성된다. 자원 관리 모듈은 주로 여러 유형의 자원 할당 및 가상 컴퓨터 관리를 담당한다. 이 모듈은 시스템이 여러 컴퓨터를 하나의 컴퓨터로 제어하고 동

* 교신저자

적 방법으로 자원을 할당하는데 도움이 되는 Mesos[7], [8]를 기반으로 한다. 모니터링 모듈은 주기적으로 CPU, 메모리, 디스크, 가상 자원 및 작업 상태를 수집하여 다른 모듈과 공유한다. 프로파일링 관리 모듈은 제출 된 작업을 기반으로 시뮬레이션 기록을 관리한다. 각 작업이 완료 될 때마다 의미 있는 정보가 프로파일 스키마 형태로 추출되고 프로파일 정보가 저장된다.

3. 실행 시나리오



<그림2 실행 시나리오>

본 논문에서 제안한 프레임워크에서 작업 실행 및 프로세스의 전체 단계는 그림 2[3]에 묘사되어 있으며, 전체 실행 과정은 아래와 같다.

1) 작업 제출 모듈은 JSDL 파일을 구문 분석하고 필요한 정보를 추출한다. 파일에 지정된 작업을 수행한 다음 해당 작업 정보와 함께 작업 관리 모듈의 큐로 보낸다.

2) 작업 관리 모듈은 프로파일 관리 모듈에 작업 특성을 묻는다.

3) 모니터링 모듈을 통해 사용 가능한 자원의 정량적 정보를 수집한 후 작업 관리 모듈은 스케줄링 정책을 기반으로 작업에 대한 스케줄링 계획을 결정하고 자원 관리 모듈로 보낸다.

4) 자원 관리 모듈은 사전에 설정한 정책으로 가상 시스템을 시작한다.

5) 작업 발송 모듈은 입력 및 실행 파일을 검사하여 작업을 시작하고, 최종적으로 작업을 VM으로 보낸다.

4. 간섭 방지 제어 스케줄링

데이터 센터에서 실행되는 많은 컴퓨터 작업은 CPU, 메모리 집약 또는 I / O 집약적인[9],[10] 특성을 포함하고 있다. 이러한 종류의 작업은 실제 요구하는 자원 대신 부적절한 자원이 할당될 수 있는데, 이는 전반적인 성능 저하를 야기한다. 제한된 자원으로 성능을 보장하려면 전체 응용이 간섭 방지 방식으로 실행될 수 있도록 해야한다.

[3]의 연구에서는 이러한 문제를 해결하기 위해 응용에 실제로 필요한 특성을 파악함으로써 정교하고 균형 잡힌 스케줄링을 제안한다. 응용 프로그램 인식 간섭 방지 스케줄러는 다음과 같은 세 가지 기능을 제공한다.

첫째, 자원 요구 사항과 그 정도(예 : CPU / Mem / Disk / 포트 가중치)를 프레임워크의 프로파일링 관리 모듈에서 가져온다.

둘째, 여러 종류의 자원과 가중치에 대한 필요성에 따라 작업을 스케줄링하여 균형 잡힌 자원 활용을 유도하고 작업이 간섭을 감지하지 못하도록 하는 간섭 방지 기반 스케줄링 서비스를 제공한다.

마지막으로는 최대 예상 자원 요구 사항인 Maximum Estimated Resource requirements(MER)을 사용하여 자원 예약을 제공하는 워크플로우 지원 스케줄링이다. MER은 각 작업의 자원 요구 사항 및 워크플로우의 최대 동시 도수로부터 유도된다.

4. 성능 실험

AutoDock3[11]은 약물 재위치 영역에 주로 사용되는 분자 모델링 시뮬레이션 소프트웨어이다. Autodock3는 표 1과 같이 메모리 사용량이 적고, 100% 에 가까운 CPU 사용률을 유지한다.

Application	Avg. use of CPU(%)	Avg use of Mem(%)
Autodock	99.2	1.1
Montage	87.3	10.5

<표1. 응용의 자원 사용량 측정표>

Montage[12]는 다중 천문 영상을 이용하여 복잡한 FITS를 제작하는 천문 영상 모자이크 엔진이다. Montage는 CPU 사용률은 높지만 메모리 사용률은 10.5%로 상대적으로 낮다.

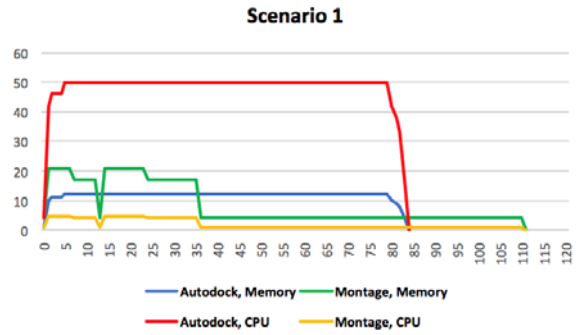
프레임워크의 효율성을 입증하기 위해 서로 다른 스케줄링 및 할당 정책을 갖는 세 가지 시나리오를 가정한다.

1. 응용 프로그램 프로파일 링을 지원하지 않고 전체 사용률을 고려하지 않은 스케줄링
2. 간섭 방지 제어 없이 응용 프로파일링을 사용하는 스케줄링
3. 응용 프로파일링 및 간섭 방지 제어를 이용한 스케줄링

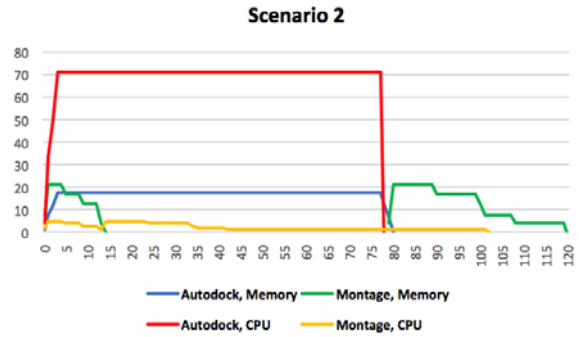
첫 번째 시나리오는 자원을 할당하는 기본 정책이 비율 R/n%(R: 사용 가능한 총 자원의 비율, n: 실행할 응용 프로그램의 수)로 자원을 배포한다고 가정한다.

그림 3은 기본 스케줄링 정책이 사용되는 첫 번째 시나리오의 결과 그래프다. 두 개의 응용 프로그램이 수행되고, 사용 가능한 자원의 초기 비율은 R/n에서 100% 이므로 각 응용 프로그램의 작업에 총 자원의 50%를 할당한다.

각 응용 프로그램 작업이 초기 정책에 의해 할당된 자원에서 수행되기 때문에 Autodock3 응용의 CPU 사용량을 제외하고는 Autodock3의 Memory 사용량 11.6%, Montage CPU 사용량 2%, Montage Memory 사용량 9%로 자원 사용률이 낮다.

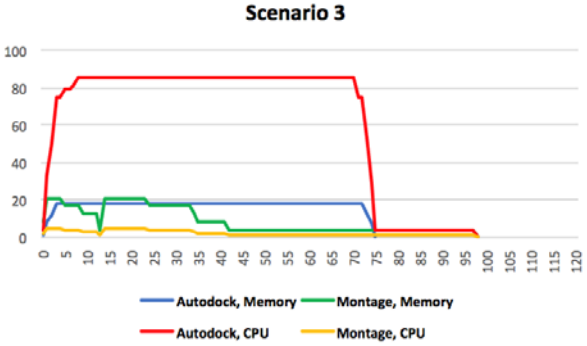


<그림3 시나리오1>



<그림4 시나리오2>

두 번째 시나리오는 응용 프로그램의 각 자원이 프로파일링 모듈을 사용하여 원하는 자원을 제공하는 적절한 가상 자원을 받았지만, 실행중인 워크플로우 응용(Montage)에 대한 자원 부족이 발생했다. 프로파일링 모듈에서 Autodock3은 CPU intensive 응용임을 알 수 있고, 따라서 워크플로우 응용인 Montage의 MER만큼을 제외한 CPU자원을 제공받아 실행된다. 하지만 Montage응용은 Memory 자원 부족이 발생해 63분의 대기 시간이 발생한다. 이러한 결과는 특정 응용에 자원 집중과 각 서버의 전반적인 사용이 고려되지 않기 때문에 발생한다.



<그림5 시나리오3>

세 번째 시나리오는 프로파일링 기능을 사용하고, 워크플로우가 있는 경우 스케줄러가 최대 MER만큼 리소스를 예약하는 간섭 방지 제어를 사용한다. 스케줄러는 자원을 적절히 예약함으로써 성능 저하뿐만 아니라 자원 고갈을 방지 할 수 있다. 또한, 각 서버의 활용을 고려한 간섭 방지 스케줄링은 전체 서버의 활용도 향상에 큰 영향을 미친다. 자원 활용 측면에서 13.4%의 향상을 보였고, 정책을 포함하지 않은 조건에 비해 평균 유지 시간을 35% 향상시켰다.

5. 결론 및 향후 연구

과학 응용 프로그램의 특성으로 인해 가상화된 컴퓨팅 환경에서 이러한 응용 프로그램을 실행하는 데는 여전히 어려움이 있다. 이러한 문제를 해결하기 위해 본 논문에서는 다양한 응용 프로그램의 속성을 고려하여 여러 유형의 자원에 대한 정교한 할당을 제공하는 과학 실험용 통합 컴퓨팅 프레임워크를 제안했다. 본 연구에서 제안한 프레임워크의 성능 평가 실험 결과에 따르면 자원 활용 측면에서의 향상과 평균 유지 시간 감소를 알 수 있다.

향후 연구로는 사용자가 중복된 시뮬레이션을 수행하지 않도록 데이터 관리를 담당하는 모듈을 설계, 개발할 예정이다.

Acknowledgement

이 논문은 2015년도 정부(미래창조과학부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(2015M3C4 A7065646)

참 고 문 헌

[1] Rho, S., Kim, S., Kim, S., Kim, S., Kim, J. S., Hwang, S. “ HTCaaS: a large-scale high-throughput computing by leveraging grids, supercomputers and cloud.” In High Performance Computing, Networking, Storage and Analysis (SCC), 2012 SC Companion: (pp. 1341-1342). IEEE.

[2] Wang, XiaoYing, et al. "Appliance-based autonomic provisioning framework for virtualized outsourcing data center." Autonomic Computing, 2007. ICAC'07. Fourth International Conference on. IEEE, 2007.

[3] Kim, S., Ahn, J., Kim, H., Kim, Y., & Choi, J. (2017, September). iSDF: An integrated software-defined computing framework for scientific experiments. In Network Operations and Management Symposium (APNOMS), 2017 19th Asia-Pacific (pp. 157-162). IEEE.

[4] Anjomshoaa, Ali, et al., “Job sumbission description language (jsdl) specification, version 1.0” Open Grid Forum, GFD. Vol. 56. 2005

[5] Chronos, https://mesos.github.io/chronos/

[6] Dellinger, Matthew, Piyush Garyali, Binoy Ravindran., "ChronOS Linux: a best-effort real-time multiprocessor Linux kernel." *Design Automation Conference (DAC), 2011 48th ACM/EDAC/IEEE*. IEEE, 2011.

[7] Mesos, http://mesos.apache.org/

[8] Hindman, Benjamin, et al., "Mesos: A Platform for Fine-Grained Resource Sharing in the Data Center." NSDI. Vol. 11. No. 2011. 2011.

[9] Bikash Sharma, Ramya Prabhakar, Seung-Hwan Lim, Mahmut T. Kandemir, and Chita R. Das. “Mrorchestrator: A fine-grained resource orchestration framework for mapreduce clusters” In IEEE CLOUD, pages 1–8, 2012.

[10] R. Boutaba, L. Cheng, and Q. Zhang. “On cloud computational models and the heterogeneity challenge” J. Internet Services and Applications, 3(1):77–86, 2012

[11] Autodock, http://autodock.scripps.edu/

[12] Montage, http://montage.ipac.caltech.edu/