과학 응용 실험을 위한 하이브리드 자원 활용 소프트웨어정의 컴퓨팅(SDC) 플랫폼 설계

안주림[0] 최지은[1] 오유리[2] 김서영[0] 김윤희[0*]
[0]숙명여자대학교 컴퓨터과학과
[1]한국과학기술정보연구원 슈퍼컴퓨팅본부, [2]한국특허정보원
[0]{julim8990, yulan}@sookmyung.ac.kr
[1]jieun1205@kisti.re.kr roh0203@kipi.or.kr [2]yroh0203@kipi.or.kr [0]sssyyy77@gmail.com

A Design of Software-Defined Computing Platform for Scientific Experiments over Hybrid Resources

Julim Ahn[0], Jieun Choi[1], Yoori Oh[2], Seoyoung Kim[0], Yoonhee Kim[0*]
[0]Dept. of Computer Science, Sookmyung Women's University
[1]Supercomputing, KISTI
[2]Korea Institute of Patent Information

요 약

컴퓨팅 기술의 발전으로 다양하고 복잡한 응용의 실행에 대한 지원이 가능해지면서 계산 성능의 최적화와 자원 활용에 대한 중요성이 대두되고 있다. 특히 과학 분야의 응용들의 경우 각기 다른 자원에 대한 요구사항과 작업 실행 특성을 갖고 있다. 따라서 이에 맞는 작업 실행 및 자원 관리 플랫폼이 필요하다. 본 논문에서는 과학 응용을 위한 하이브리드 자원 활용 소프트웨어정의 컴퓨팅 플랫폼인 SD-JMP(Software Defined-Job Management Platform)를 제안하고 그 효용성을 검증하였다.

1. 서 론

컴퓨팅 기술의 끊임없는 발전과 함께 데이터 센터에서는 다양하고 이질적인 작업 실행에 대한 수요가 지속적으로 증가하고 있다. 이에 더 많은 작업이 데이터센터에서 실행되며 제한된 공유 자원에 대한 경쟁이 증가한다. 이로 인해 자원 관리가 어려워지고, 자원 사용률이 낮아질 수 있다. 이러한 데이터센터의 자원 낭비와 성능 하락의 해결책으로 Software-defined Compute(SDC) 기술의 중요성이 강조되고 있는 추세이다[1]. 한편 고성능 컴퓨팅 자원의 수요가 많은 과학 분야의 응용들의 경우, 각 응용마다 자원에 대한 요구 사항이 다르며 그 종류 또한 다양하게 존재한다. 따라서 실제 과학 응용의 특성과 요구사항을 찾아내고, 그에 맞는 실행환경을 도출해야 한다. 이처럼 실제 과학 응용들을 효율적으로 실행하기 위한 작업 실행 및 관리 플랫폼의 설계가 필수적이다.

본 논문에서는 과학 응용을 위한 하이브리드 자원 활용 소프트웨어정의 컴퓨팅 플랫폼인 SD-JMP(Software Defined-Job Management Platform)를 제안한다. 응용 실행의 최적화를 위한 자원관리 소프트웨어 기술 중 하나인 SDC와 같은 소프트웨어로 정의된 구성 요소를 통해 데이터센터의 자원을 효율적으로 사용할 수 있다. 또한 자원을 하이브리드하게 사용하여 실제 과학 응용의 특성과 요구사항을 찾아낸다.

본 논문의 구성은 다음과 같다. 1장의 서론에 이어 2장에서는 관련 연구들을 살펴본다. 3장에서는 시스템의 구조와 작업 제출과 실행 과정을 설명하며 제안하는 플랫폼의 성능 실험을 보이며 4장에 결론과 향후 연구로 정리한다.

2. 관련 연구

과학 응용들을 지원하기 위해 국내에서도 다양한 소프트웨어 정의 플랫폼 연구가 시작되었다[2][3]. High Throughput Computing-as-a-Computing (HTCaaS)[2]은 주로 대용량 계산 처리 작업(High Throughput Computing) 또는 짧고 많은 수의 작업 처리(Many-task Computing)를 지원하는 작업 실행

* 교신저자

관리 시스템이다. 하지만 이 시스템은 실험 당 동일 명세를 가진 자원만 활용하는 등 자원을 세밀하게(fine-grained) 제어하는 데에 한계가 존재하며 여러 자원을 연동할 때, 이를 관리하는 데도 어려움이 따른다.

본 논문에서는 이러한 HTCaaS 기술의 단점을 보완하며, 오픈소스 기반 작업 스케줄링 관리 프레임워크인 ChronOS[4] 및 자원 관리 프레임워크인 Apache Mesos [5]를 활용하여 소프트웨어 정의 컴퓨팅 플랫폼인 SD-JMP를 설계하고 구축한다.

3.SD-JMP(SoftwareDefined-Job Management Platform)

다음은 제안하는 SD-JMP의 구조에 대해서 설명한다. 그리고 제안한 플랫폼 상에서의 성능 실험 및 결과 분석을 통해 작업 실행 및 관리의 효율성을 검증한다.

3.1 시스템 구조

그림 1 SD-JMP 시스템 구조도

본 논문에서 제안하는 SD-JMP의 구조는 그림 1과 같다. 주요 모듈들은 크게 작업 관리 및 자원 관리 레이어의 두 카테고리로 분류된다. 작업 관리 레이어는 작업 스케줄링 관리 프레임워크인 ChronOS [4]를 기반으로 하며 이는 '작업 스케줄러 모듈', '작업 큐 모듈', '작업 제출 모듈'로 구성된다. '작업 스케줄러 모듈'은 작업의 타입이나 사용자 선호에 따라 스케줄러를 다양하게 사용 가능하며, '작업 제출 모듈'은 사용자에 의해 제출된 작업 명세 파일을 파싱 하여 태스크들을 생성해 작업 제출을 수행한다. '작업 큐 모듈'은 사용자 별로 '작업 제출 모듈'에서 제출된 태스크들을 큐에 넘겨주며 전체 큐를 관리한다. ChronOS 활용을 통해 워크플로우 특성을 갖는 응용 또한 지원이 가능하며 사용자에 따른 스케줄러 정책 변경 또한 가능하다[4][6].

Mesos 기반의 자원 관리 레이어는 '자원 연동 모듈', '작업 관리 모듈', '자원 관리 모듈'로 구성된다. '자원 연동 모듈'은 클러스터와 클라우드 같은 이기종 컴퓨팅 인프라를 계산 자원으로 서비스 한다. '작업 관리 모듈'과 '자원 관리 모듈'은 각각 작업과 자원의 이력을 DB화하여 관리한다. '데이터 관리 모듈'은 입/출력 데이터와 중간 생성 데이터에 대한 정보뿐만 아니라 과학 워크플로우의 태스크와 데이터의 dependency 정보를 DB화해서 관리한다. Mesos[5]는 여러 대의 컴퓨터를 하나의 컴퓨터처럼 관리가 용이하고 하나의 Mesos 클러스터에서 여러 개의 프레임워크들이 동적으로 실행될 수 있게 도와주는 클러스터 자원 매니저이다. 본 논문에서 제안한 SD-JMP에서는 Mesos의 이러한 장점을 활용하여 태스크 별로 자원 요구량(CPU, Memory, Disk 등)에 따라 세밀(fine-grained)하게 컴퓨팅 자원을 제공할 수 있다.

3.2 SD-JMP에서의 작업 제출과 실행

사용자는 수행하고자 하는 계산 작업에 대한 입/출력 파일 및 실행파일 등의 정보를 명세해야 한다. 이는 JSDL(Job Submission Description Language) 형태로 관리되며, 명세된 JSDL 파일은 GUI(Graphic User Interface), CLI(Command Line Interface) 등을 통해 SD-JMP에 제출한다. '작업 제출 모듈'에서 사용자가 제출한 JDSL 파일을 파싱하여 태스크들이 생성된다. 생성된 태스크는 '작업 큐 모듈'에 의해 작업 큐에 축적된다. 큐의 태스크들은 ChronOS 기반의 '작업 스케줄러 모듈'을 통해 '자원 연동 모듈'로 전달된다. '자원 연동 모듈'은 각 자원에 Mesos-slave를 통해 Mesos와 연동되는데, 클라우드 VM의 경우, Mesos-slave를 VM 내에 설치하여 새로운 VM이 부트 시 자동으로 SD-JMP에 연동되도록 하였다. 또한 SD-JMP에서는 자원들로 하여금 Network File System(NFS) 기반 공유 디렉토리를 사용하며, 작업 별로 워크스페이스가 관리된다.

한편, '작업 관리 모듈'과 '자원 관리 모듈'은 등록되어 있는 자원들의 상태와 이력을 주기적으로 모니터링하고 이 정보를 데이터베이스에 저장한다.

이와 같이 사용자가 SD-JMP를 사용해 작업을 제출하면 자원을 용이하게 추가 및 확장할 수 있고 실험환경을 쉽게 변경할 수 있기 때문에 사용자가 응용의 특성에 맞게 실험환경을 구축할 수 있다.

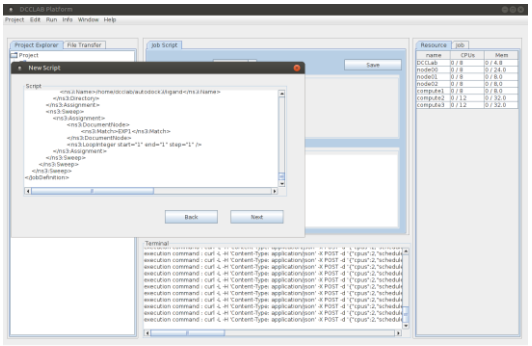

그림 2 SD-JMP GUI 화면

그림 2는 SD-JMP GUI의 일부분인 작업 명세(JSDL) 작성 후 제출을 하는 화면이다. 사전에 등록된 자원들(그림 2의 오른쪽)에 따라 사용자가 원하는 자원 요구량

또는 시스템에서 제공하는 최적의 자원 비율에 따라 작업이 제출된다.

3.3 성능 실험 및 분석

과학응용은 서로 다른 자원에 대한 요구사항과 작업 실행 특성을 가지고 있다. 플랫폼에서 제공하는 작업 분배 성능 및 효율성 검증을 위해서 클러스터와 클라우드의 하이브리드 환경 상에서 성능 실험을 수행하였다. 실험에는 신약 개발 분야의 단백질과 지질 결합모드를 알아내는데 사용되는 분자 도킹 응용 중 하나인 Autodock[7] 응용이 활용되었다. 클러스터와 클라우드(vm1, vm2)에서 작업 100개의 평균 시간을 측정하였다. 이 경우 플랫폼에서 제공하는 클러스터와 클라우드로의 작업 분배 비율을 [10:30 - 90:10]까지로 정의해 작업을 분배하여 각각 수행시간을 측정하였다. 클라우드의 vm1(vCPU:6, Mem(GB):4, Disk(GB):40)은 4개, vm2(vCPU:4, Mem(GB):4, Disk(GB):20)는 6개씩 사용하여 클러스터와 유사한 자원 환경을 구성하였으며, VM별로 vCPU, Disk 자원의 사이즈를 다르게 설정하였다.

Name	vCPU	Mem(GB)	Disk(GB)
Cluster	24	70	6500
Cloud	24	62.6	376

표1 실험 자원 환경

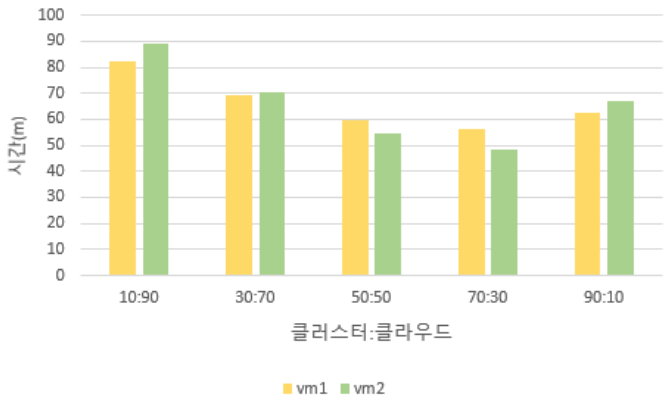

그림 3 실험 결과

그림3은 클러스터와 클라우드의 작업을 비율 별로 분배하여 산출된 5가지 결과를 보여준다. 각 비율 별 평균 수행 시간은 10:90은 85.75(m), 30:70은 69.95(m), 50:50은 57.1(m), 70:30은 52.2(m), 90:10은 64.85(m)이다. 실행 시간이 가장 적은 70:30은 10:90보다 39%, 30:70보다 25%, 50:50보다 8%, 90:10보다 19% 더 줄어든 것을 알 수 있다.

실험결과에서 작업을 다르게 분배하여 응용을 실행하면 실행 시간이 달라지는 것을 알 수 있다 따라서 작업의 자원 요구에 따라 작업을 다르게 분배하는 것이 의미 있다는 결과를 도출 할 수 있다.

4. 결론 및 향후 연구

본 논문에서는 과학 응용을 위한 하이브리드 자원을 지원하는 새로운 실행 플랫폼인 SD-JMP에 대해 소개하고 그 효용성을 검증하였다. 과학 응용들의 경우 효과적인 리소스 관리를 위해 올바른 리소스를 작업의 다양한 특성에 따라 할당하여 실행시키는 플랫폼이 매우 중요한 요소임을 알 수 있다. 소프트웨어 기술을 활용하여 기존 하드웨어에서 직접 제어하던 영역들의 유연성을 높이고 상호운용성을 증대시킬 수 있을 것으로 기대된다.

향후 구축된 SD-JMP 환경에서 데이터 집약적인 과학 워크플로우를 위한 스케줄러 연구를 진행하고자 한다.

Acknowledgement

이 논문은 2015년도 정부(미래창조과학부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(2015M3C4A7065646)

참 고 문 헌

[1] Nam, Yoonsung, et al. "Workload-aware resource management for software-defined compute." Cluster Computing 19.3: 1555-1570 (2016).

[2] Rho, Seungwoo, et al. "HTCaaS: a large-scale high-throughput computing by leveraging grids, supercomputers and cloud." High Performance Computing, Networking, Storage and Analysis (SCC), 2012 SC Companion:. IEEE, 2012.

[3] 유정록 외. "다분야 계산과학 시뮬레이션을 위한 EDISON 플랫폼 연구." 인터넷정보학회지 13.3:23-33(2012).

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

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

[6] Garyali, Piyush, Matthew Dellinger, and Binoy Ravindran. "On best-effort utility accrual real-time scheduling on multiprocessors." International Conference On Principles Of Distributed Systems. Springer Berlin Heidelberg, 2010.

[7] Goodsell, David S., Garrett M. Morris, and Arthur J. Olson. "Automated docking of flexible ligands: applications of AutoDock." Journal of Molecular Recognition 9.1 : 1-5(1996).