작업 이력의 요인 분석을 통한 과학 클라우드 자원 선택 모델

(A Science Cloud Resource Selection Model using Factor Analysis of Job Trace)

김 서 영 † 김 윤 희 †† 황 순 욱 †††
(Seoyoung Kim) (Yoonhee Kim) (Soonwook Hwang)

요 약 클라우드 컴퓨팅의 발전으로 다양한 과학 분야의 연구자들은 연구에 필수적인 슈퍼 컴퓨터를 온-디멘드 서비스로 제공받을 수 있으며, 동적인 자원 확장이 가능해짐에 따라 그들의 연구 환경을 확장해 나가고 있다. 이처럼 사이언스 클라우드는 최근 여러 과학 분야에서 새로운 연구 환경 패러다임으로 주목 받고 있다. 하지만 사용자의 요구에 알맞게 신속하고 동적인 가상 작업 공간을 구성하는 것이 어렵기 때문에 응용의 특성을 고려하여, 적절한 실험 환경을 미리 구성하는 것이 중요하게 여겨진다. 더불어 적정 수준의 성능을 보장하는 가상 머신을 제공하는 스케줄링 메커니즘도 요구된다.

본 논문에서는 작업이력의 통계적 분석을 통한 사이언스 클라우드 프로비저닝 모델을 제안한다. 이 모델은 어플리케이션의 실행에 의해 축적된 작업이력을 분석하여 그 특성을 파악하고, 결과를 자원 프로비저닝 결정에 사용한다. 작업이력 분석을 위해서는 통계적 분석 기법인 주성분 분석을 적용하였으며, 이로써 작업 프로파일들 중 가장 영향력이 큰 요인과 이력을 선별한다. 선별한 요인은 참조할 프로파일을 선택하는 데에 활용하며, 그 중 가장 우수한 성능을 갖는 자원에 가상머신을 배치 후 작업을 스케줄링 한다. 작업 수행 후에는 그 결과를 통해 프로파일의 신뢰성을 평가하고 동적인 환경에 적응하는 알고리즘의 유효성을 보인다. 논문의 후반에 제시된 성능 비교로써, 제안 모델 활용 시에 대기 시간의 감소와 성능 향상을 이끌며 가상화 단점을 극복하고 클라우드의 장점을 극대화 할 수 있음을 보였다. 이로써 클라우드 환경에서의 효과적인 자원 관리를 도우며 관리를 위한 오버헤드 또한 완화시킬 것으로 예상된다.

키워드 : 사이언스 클라우드, 자원 프로비저닝, 주성분 분석, 작업 프로파일

Abstract The advent of cloud computing makes scientists to extend their research environments over supercomputers to on-demand and dynamically scalable resources. Science cloud has become a trend in various scientific domains these days. Depending on user's demands, however, it is difficult to supply optimal environment for executing job rapidly and dynamically. Therefore, it is very important to predict user's requirements and to prepare execution environment in advance. In addition, it needs scheduling mechanisms for virtual machines to offer some level of guaranteed performance of a user application.

This paper proposes a cloud resource provisioning model using statistical analysis of job profiles for science. In this model, we use job profiles which are generated from executions of various applications and identify characteristics of an application by applying statistical analysis. We utilize PCA (Principal Component Analysis) to analyze job profiles and to extract factors which contribute

· 본 연구는 한국과학기술정보연구원의 응용 연구 서비스 개발 과제의 연구 결과로 수행되었음(K-12-L06-C02-S05)

논문접수 : 2012년 1월 20일
심사완료 : 2012년 5월 23일

† 학생회원 : 숙명여자대학교 컴퓨터과학과
sssyyy77@sm.ac.kr
†† 종신회원 : 숙명여자대학교 컴퓨터과학과 교수
yulan@sm.ac.kr
(Corresponding author임)
††† 비 회 원 : 한국과학기술정보연구원 슈퍼컴퓨팅센터 응용연구서비스개발팀 책임연구원
hwang@kisti.re.kr

정보과학회논문지: 컴퓨팅의 실제 및 레터 제18권 제8호(2012.8)

much to execution time or performance. The effective factors are used for selecting reference job profile and deciding a resource site where VM is deployed. An application is executed on the chosen site and its performance result is incorporated into the job profiles with the purpose of evaluating profile's credit. Performance comparisons with other conditions verify that this model can strengthen strong point of cloud computing and make up for its weakness, since it can offer performance improvements as well as reduction of waiting time. As a result, this model can provide efficient management of cloud resource for a service provider and reduce management overheads on Cloud.

Key words : Science Cloud, Resource Provisioning, Principal Component Analysis, Job Profile

1. 서 론

최근 e-사이언스 연구 환경의 장점을 용이하게 하는 사이언스 클라우드 기술이 새로운 연구 환경 패러다임으로 주목 받고 있다. 이는 가상화 기술을 핵심으로 하는 클라우드 컴퓨팅 기술과 e-사이언스의 장점을 접목시켜 확장성과 유연성이 강화된 가상 작업 공간을 제공한다. 아울러, 과학 또는 교육 중심 프로젝트가 IaaS (Infrastructure as a Service) 기반의 클라우드 컴퓨팅을 활용하여 손쉽게 실험을 수행할 수 있고, 과학기술 분야 커뮤니티의 인프라 활용 시 요구 사항을 잘 파악하여 이를 충족하는 것을 목적으로 한다. 즉, 연구자가 수행하는 연구에 맞는 자원을 원하는 시간만큼 빌려 사용할 수 있는 기능을 제공한다. 따라서 다분야의 연구 특성에 맞는 인프라를 동적으로 제공 가능한 사이언스 클라우드만의 서비스로 연구의 규모뿐만 아니라 효율성과 유연성을 크게 제고시킬 수 있다. 이러한 점에서 클라우드 컴퓨팅 기술은 e-사이언스 기술을 이어가는 차세대 연구 환경으로써 활용도가 매우 높다. 하지만 이러한 클라우드 컴퓨팅 환경에서 사용자들이 요청한 작업에 대해 그들이 명시한 QoS 및 SLA(Service- Level Agreement)를 준수하여 자원을 제공하기 위해서는 보다 클라우드 환경에 적합한 자원 관리 및 제어 기술이 요구된다. 즉 클라우드는 모든 사용자의 어플리케이션이 동작할 수 있도록 유연하고 이용이 쉬운 실행 환경을 제공해야 하며, 사용자가 이용하고 있는 어플리케이션의 성능을 보장해 줄 수 있는 메커니즘을 제공해야 한다. 과학 분야의 어플리케이션의 경우, 세부적인 도메인에 따라 그 특성이 다르고 요구되는 컴퓨팅 환경이 다르므로 이를 위해서는 어플리케이션의 특성을 세밀하게 파악하는 일이 필요하다. 기존 클러스터 또는 그리드와 같은 컴퓨팅 패러다임에서는 어플리케이션의 특성 파악을 위해 컴퓨팅 사용 이력을 다각도에서 분석하여 자원 관리 연구나 운용 및 컴퓨팅 환경 설계, 성능 평가 등 다양한 분야에 활용하였다. 이처럼 사용 이력들을 다양한 통계적 기술에 적용해 분석하고, 어플리케이션의 특성과 클라우드 컴퓨팅 환경 속성들 사이의 상호 작용을 파악할 필요가 있다. 한편, 사용 이력은 주로 거대한 양으로 존재하며, 다양한 요인들을 포함한다. 따라서 이력을 활용하여 그 특성을 찾기 위해서는 방대한 양의 데이터 중 그것들을 대표하고 의미 있는 데이터를 찾는 것이 중요하다. 이를 위해 주성분 분석(Principal Component Analysis)[1] 기법을 사용해 요인들 사이의 상관 구조를 파악하며 영향력 있는 요인을 중심으로 데이터들의 차원을 축소하여 적절히 해석하고 활용한다. 요인의 영향력을 주성분 분석을 통해 수치화하고, 수치화된 정보를 다음 가상 환경을 생성할 자원 선택에 활용하며 과거에 이용되었던 정보를 반영하고 있으므로 더욱 안정적인 자원 선택을 가능케 하고 사용자에게 빠른 프로비저닝 서비스를 제공할 수 있으며 어플리케이션 수행에 최적화된 가상 환경을 제공할 수 있다.

본 논문의 구성은 다음과 같다. 2장에서 가상화 기반 환경에서의 최적 성능을 보장하는 실행 환경 연구를 알아보고, 3장에서 제안하는 프로비저닝 모델을 자세히 다루고 알고리즘을 보인다. 4장에서는 제안한 모델을 기반으로 클라우드심(Cloudsim)[2]을 이용한 실험과 그 결과를 분석하여 제안한 방법의 성능을 검증한 후, 마지막으로 결론을 정리한다.

2. 관련 연구

가상화된 컴퓨팅 환경은 사용자가 하나의 물리적 컴퓨터에 여러 컴퓨팅 환경을 운영할 수 있도록 돕는다. 이러한 운영은 하나의 물리적 서버를 조금 더 효과적으로 활용하기 위해 사용되며, 특히 각 가상 환경에 정밀한 자원할당이 가능하기 때문에 이를 고려한 전체 가상 머신 형태 결정과 운영 방식은 차세대 컴퓨팅을 위한 핵심 기술로 떠오르고 있다. 가상화된 컴퓨팅 환경에서 수행할 어플리케이션은 다양할 수 있으므로 어플리케이션의 자원 요구량 또는 성능 수준에 알맞은 정확한 자원을 할당하는 것은 사용자의 QoS요구를 만족시키고 자체 컴퓨팅 자원의 운영 비용을 낮추기 위해 중요하게 여겨진다. 따라서 현재의 자원 할당 상태와 자원 요구량을 분석해 적절히 자원을 프로비저닝하기 위한 연구들이 최근 활발하게 진행되고 있다[3,4].

[3]는 비선형(Non-linear)예측 기법을 활용하여 현재

의 어플리케이션 작업량 분석에 따른 가상 머신 프로비저닝 제어 구조를 제안하였다. 실시간으로 수행 환경 관찰을 통해 현재의 어플리케이션 요구량과 자원 사용량을 분석하여 미래의 워크로드 변화를 예측하여 그 정보와 SLA를 기반한 마이그레이션 기반 가상 머신 프로비저닝 을 수행한다. 이러한 제어 과정을 통해서 물리 자원 상태 또는 현 상태에 알맞은 최적 상태를 유지한다. 한편, [4]는 가상 서버의 작업량을 조절하며 동시에 정확한 응용의 사용량을 예측할 수 있는 클라우드 관리 시스템을 제안하였다. 이 연구는 고정된 수의 서버를 갖는 환경에서 가상 풀 서버 기반의 프로비저닝을 제공하며 어플리케이션의 요구를 논리적 회기(logistic regression) 모델로 예측하여 다음 요청에 대한 대기 가상 머신을 프로비저닝한다. 풀 서버 기반의 프로비저닝 과정에 소요되는 시간을 얼마나 단축할 수 있는지를 기반으로 프로비저닝될 대기 가상 머신을 결정하고 이로써 전반적 성능 향상을 이끌 수 있다는 제안을 하였다.

하지만 [3,4] 연구 모두 어플리케이션 수행 성능, 처리량 또는 어플리케이션의 특성에 대해서는 고려치 않고 단지 서비스 요청 량을 중심으로 시스템을 관리하고 프로비저닝을 제공한다. 또한 시스템 관리를 위해 이미 배치된 가상 머신에 대해 마이그레이션 기반 재배치를 요구하여 가상 머신 프로비저닝을 제공한다. 하지만 잦은 마이그레이션 요구는 오히려 성능 저하의 원인이 될 수 있다[5,6]. 이 같이 발생하는 오버헤드는 본 연구에서 대상으로 하는 과학 어플리케이션의 클라우드 환경에서의 수행에 있어서 최악의 경우, 작업 수행 동안 막대한 시간 낭비를 일으킨다. 따라서 사용자가 가상 머신을 요청 시 최적의 물리 자원에 적절한 가상 머신 형태를 예측하여 제공하며 균형적인 가상 머신 배치를 제공할 필요가 있다. 더불어 과거 수행 기록을 기반으로 어플리케이션 특성을 고려한 효율적인 자원 프로비저닝 및 관리는 최근 자원 상태와 사용자들의 요구를 파악하는 데에 효과적으로 활용될 수 있다.

본 연구에서 제안한 클라우드 프로비저닝 모델은 지역적으로 분포되어 있는 클러스터에 가상 머신을 배치할 때 작업이력 분석 결과에 따라 배치될 자원을 선택한다. 어플리케이션 특성을 결정하는 파라미터들을 대상으로 그 영향력을 분석 후 실행 시간과의 관계를 통해 가상 머신을 할당할 클러스터 자원을 선택한다. 또한 작업 프로파일의 신뢰도를 평가하여 오류 상황 또는 작업 부하(장시간 대기) 같은 클러스터 자원들의 상황을 반영하여 안정적인 클러스터 자원 선택을 보장한다.

3. 클라우드 프로비저닝 모델

본 절에서는 제안하는 프로비저닝 모델이 적용된 자원 관리 시스템을 소개하고, 대상 응용의 특징 및 모델에 적용된 기법에 사용되는 분석법을 자세히 설명한다. 그 다음, 가상 머신을 배치할 자원 선택 방식을 표현한 알고리즘을 소개한다.

3.1 사이언스 클라우드 자원 관리 시스템

그림 1은 사이언스 클라우드 자원 관리를 위한 시스템 구조를 보여준다. 본 연구에서는 가상 머신 이미지 형태로 미리 구성된 실행 환경을 작업 실행 때 배치하여 제공하는 클라우드 모델을 기반으로 한다. 사용자는 지리적인 위치에 관계없이 시스템에 어플리케이션 수행을 위한 계산 서비스 요청을 하며, 자원 관리 계층(Resource Mgmt. Layer)에서는 사용자가 요청한 계산에 사용될 어플리케이션의 특성을 요인 추출 서비스(Factor Extract Service)를 기반으로 생성할 가상 머신 이미지를 선택하고, 이를 배치할 특정 물리 자원 사이트를 결정한다. 결정된 자원에 가상 머신을 배치하고 작업을 스케줄링 하며, 수행 후에는 그 이력을 프로파일 저장소에 포함시킨다. 또한 수행 결과를 기반하여 프로파일을 평가(Profile evaluator)한다. 수행되는 가상 머신은 동일한 실제 물리 머신 내에서 자원을 분할하여 사용하는 형태로 제공되며, 하나의 동일한 실제 자원 내에서 독립적이고 분별된 형태로 제공될 수 있어 사용을 증대시켜준다. 물리 머신들은 사용자의 요구 사항을 만족시키기 위한 실제 자원인 다양한 서버와 네트워크로 구성된다. 본 논문에서 연구한 부분은 클라우드 자원 관리 계층에 포함된다.

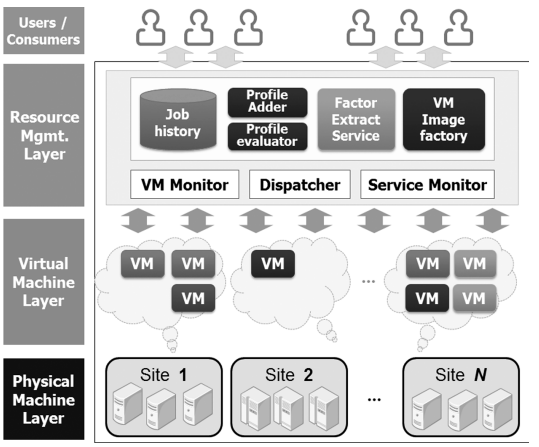

그림 1 클라우드 자원 관리 구조

3.2 대상 응용 특징 및 요인 정의

본 연구에서는 다양한 종류의 과학 어플리케이션 중에서도 특히 HTC(High Throughput Computing) 구조를 갖는 유체해석(CFD, Computational Fluid Dynamics

표 1 작업 프로파일 요인 정의

		요인 이름	요인 분류
f_1	V1	AMACH	어플리케이션 요인
f_2	V2	RE	
f_3	V3	AOA	
f_4	V4	ITMAX	
f_5	V5	CPU clock /1 node	자원 요인
f_6	V6	안정도	
f_7	V7	대역폭	
f_8	V8	거리	

[7]) 어플리케이션을 대상으로 하였다. 이 경우, 각 실행 작업 간 의존성이 없는 계산 실험들에 파라미터 값을 변화 시키고 그 결과를 관찰하는 형태로 수행된다. 따라서 응용에 사용되는 파라미터 값의 변화에 따라 수행 시간의 변화가 유동적이며 작업 수행 성능에 큰 영향을 미치는 특징을 갖는다. 이러한 응용 특징을 반영하여 적절히 요인을 정의할 필요가 있다. 요인(Factor)은 수행하는 작업 속성 또는 축적된 프로파일 형태에 따라 다양하게 정의될 수 있지만, 작업 수행 시간에 영향을 미치는 요소로써 기본적으로 어플리케이션과 관련된 요인(파라미터, 작업 크기, 길이 등)과 컴퓨팅 자원과 관련된 요인(CPU성능, 메모리 크기, 네트워크 대역폭 등)들로 정의될 수 있다. 예를 들어, e-AIRS2.0[8]에서 축적된 이력들을 대상으로 분석에 사용될 요인을 표 1과 같이 정의할 수 있다. e-AIRS2.0는 항공우주 통합 연구 환경으로 주로 유체해석 응용을 사용하므로 이 HTC 기반 응용에서 주로 사용되는 파라미터인 AMACH, RE, AOA, ITMAX를 어플리케이션 요인으로 정하였다. 또한 현재 계산 자원의 상태에 따라 그 수행 시간이 좌우될 수 있으므로 자원 속성을 반영하기 위해 이 시스템에서 활용하는 환경인 PRAGMA 그리드[9] 자원의 모니터링 정보를 참고해 자원 요인을 정의하였다. 표 1에 보이는 것처럼, 모니터링 정보에 따라 노드 당 CPU 클락, 각 클러스터의 안정도(성공 횟수/전체시도횟수), 대역폭, 클러스터 위치로 선정하였다.

3.3 작업 프로파일 분석

작업 이력은 앞 절과 같이 정의된 요인 정보에 따라 표 2처럼 축적될 수 있으며, 어플리케이션/자원 요인 각각에 대해 수행 시간과의 비례 관계에 따라 내림/ 오름차 순으로 정렬 뒤 전체 개수의 상위 20, 40, 60, 80%에 해당 하는 값, 그리고 최소와 최댓값을 경계로 하여 5 단계로 나눠 점수화한다. 점수화한 프로 파일은 표 3와 같이 표현되는데, 자원 요인의 경우에는, 그림 2를 5단계로 점수화한 후 반영한다. 가령, 표 2에서 *id* 110번의 이력은 sakura클러스터에서 수행됐으므로 이의 cpu 클락 수치 의 해당 범위를 계산하여 f_5 를 표현한다. 모든 파라미터를 표 3과 같이 점수화 후, PCA 기법을 통해 표 4와 같이 분석한다. PCA 기법의 수행 과정은 다음과 같다. n개의 작업 이력과 d개의 요인을 갖는 경우, $n \times d$ 행렬 J를 구성하여 상관 행렬 Cov(J)를 식 (1)를 통해 계산한다. Cov(J)에 대응하는 고유 벡터/ 값 쌍을 계산하면 요인 개수만큼의 쌍이 생성 되며, 각 요인에 대응되는 d 개의 생성된 고유 값 중 가장 그 값이 큰 요인이 영향력이 큰 요인이 된다. 또한 값이 가장 큰 고유 값에 대응되는 고유 벡터를 이용하여 작업 이력들에 포함된 모든 작업들의 주성분 점수를 계산 한다[1]. 이 때, 주성분 점수가 가장 높은 작업 이력에 해당하는 작업이 대표 작업이 된다.

$$\mathrm{Cov}\ (\mathrm{J}) = \frac{1}{n-1}\left(\mathrm{J} \cdot \mathrm{J}^{T}\right) \quad (1)$$

표 2 작업 프로파일의 예

id	j_{id}				r	s	*VM (id)*	T_{id}
	f_1	f_2	f_3	f_4				
110	0.8	500000	8	10000	sakura	Done	vm4	43310
111	0.7	5	6	10000	luke	Done	vm1	3275

표 3 점수화 한 작업 프로파일의 예

id	j_{id}				r				s	*VM (id)*	T_{id}
	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8			
110	4	3	4	3	2	5	2	1	Done	vm4	43310
111	3	1	4	3	5	2	5	1	Done	vm1	3275

Resource	CPU clock per node	Stability	Bandwidth	Distance
aurora	1313.882	100	750	2
f32	8672.291	92	763	0
komolongma	5120.5	100	800	13
luke	6000	95	1000	0
nucleus	2193	100	800	1
rocks-52	2388.2	100	623	14
rocks-153	3200	97	827	14
sakura	1815.647	98.9	742	0
sirius	1400	80	863	2

그림 2 PRAGMA 자원 정보

표 4 작업 이력의 주성분 분석

id	j_{id}				r				T_{id}
	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	
1	2	1	4	3	5	2	5	1	1483
2	4	1	4	3	2	5	4	1	1014
⋮	<-- 생 략 -->								
78	3	1	4	3	2	5	2	1	1933273
79	3	1	4	3	2	5	2	1	1919399
Eig_V	0.18	0	0	0	**2.33**	0.46	1.16	0.85	
순위	**1**	-	-	-	**1**	4	2	3	

$Eigen\_Vector_1$= (−0.04, 0, 0, 0, 0.93, 0.25, 0.85, 0.80) (2)

$PC\_j_1 = -0.04*[(2 - M(f_1))/s(f_1)] + ... + 0.8*[(1-M(f_8))/s(f_8)] = 1.716$ ($M(f_i)$: f_i의 평균, $s(f_i)$: f_i의 표준편차) (3)

표 4는 79개의 표본 작업이력들에 대해 PCA 수행 후 그 영향력을 수치화 한 결과의 일부를 보인다. 분석 결과, 표본 이력들 중에서는 어플리케이션 요인 중, f_2 ~ f_4의 값의 변화가 없고 'AMACH'에 해당하는 f_1만 값에 변화가 있는 경우여서 f_2 ~f_4에 0이라는 결과가 초래되었다. 따라서 고유 값이 가장 큰 f_1의 영향력이 높다고 평가되며, 자원과 관련된 요인 중에서는 CPU 클락에 해당되는 f_5가 높은 고유 값을 가지므로 영향력이 큰 요인이 된다. 각 작업에 대하여 계산된 고유 벡터를 이용해 주성분 점수를 구한다. 가령 id가 1인 작업의 프로파일에 대한 주성분 점수는 식 (3)와 같이 계산된다 [1]. 가장 큰 고유 값에 대응되는 고유 벡터가 식 (2)와 같다고 했을 때, 고유 벡터가 포함한 값들은 순서대로 각 요인들의 가중치(weight)를 의미하게 된다. 따라서 작업 id 1의 주성분 점수는 요인마다 그 가중치를 나타내는 고유 벡터 값과 평균 및 표준점수를 이용하여 표준화시킨 값을 곱하여 식 (3)과 같이 계산한다. 이 작업들의 가상 머신 환경 정보는 새로운 작업을 위한 가상 머신 환경을 준비하는 데에 사용된다. 만약 사용자로부터 <AMACH: 0.7, RE: 5, AOA: 7, ITMAX: 10000>의 파라미터를 갖는 작업 수행을 요청 받는 경우, 먼저 어플리케이션 요인 중 가장 영향력이 큰 AMACH의 값이 동일한 프로파일을 작업 이력 집합에서 찾아 후보 프로파일에 포함시킨다. 표 5는 후보 프로파일을 나타낸 것이다. 이들 중 수행 시간이 가장 적게 소요되며, 자원 대표 요인인, CPU 클락 값이 높은 작업 이력을 참조하게 된다. 수행 시간이 작은 72번, 9번, 73번 작업으로부터 후보자원 "rocks-52, rocks-153"을 얻을 수 있다. 그림 2에서 대표 자원 요인인 자원 별 CPU 클락 정도를 참고하면, 두 가지 후보 자원 중 "rocks-153"자원이 그 수치가 더 높으므로 최종적으로 작업을 수행할 자원으로 선택한다. 작업 수행을 위해 준비해야 하는 가상 머신은 대표 작업으로 측정됐던 작업 5, 18, 19, 20, 26, 27번 중에 후보 프로파일 리스트(표 5)에 포함된 작업 5번의 VM 사양인 vm6 형태를 사용한다. 따라서 "rocks-153"클러스터에 vm6인 CPU 2개, memory 1024MB(표 6)의 가상 머신을 배치시킨 후, 작업을 그 가상 머신에 스케줄링 한다.

표 5 후보 프로파일의 예

id	j_{id}				r	s	VM (id)	T_{id}
	f_1	f_2	f_3	f_4				
5	**0.7**	5	7	10000	**rocks-52**	Done	vm6	9897
9	**0.7**	5	7	10000	**rocks-52**	Done	vm3	1261
72	**0.7**	5	7	10000	**rocks-52**	Done	vm5	**907**
73	**0.7**	5	7	10000	**rocks-153**	Done	vm3	**1482**
76	**0.7**	5	7	10000	sakura	Done	vm2	1934276

표 6 가상 머신 종류

가상 머신 종류	CPU 개수(개)	Memory 크기(MB)	가상 머신 종류	CPU 개수(개)	Memory 크기(MB)
vm1	1	128	vm4	1	1024
vm2	1	256	vm5	2	512
vm3	1	512	vm6	2	1024
			vm7	4	1024

3.4 작업 이력 분석 기반 가상 머신 배치 알고리즘

활용되는 작업 이력과 대표 작업이력은 다음과 같이 표현된다.

$$P = < j, vm(j), r, T_{exec}, s > \quad (4)$$

$$vm(j) = (\text{cpu\#, mem\_size}) \quad (5)$$

$$PC = < f_{prin\_a}, f_{prin\_r}, j_{prin} > \quad (6)$$

식 (4)는 각 작업 프로파일을 나타내며, 첫 번째 요소인 j는 작업 벡터를 나타내며 각 요인 개수와 동일한 요소를 갖는다. 두 번째 요소인 $vm(j)$는 가상 머신 속성을 표현하며, 가상 머신 속성에는 식 (5)와 같이 가상 CPU개수와 메모리 사이즈가 포함된다. 식 (4)의 세 번째 요소인 r은 가상 머신이 배치된 물리 자원, 그리고 T_{exec}는 작업 실행 시간을 나타낸다. 마지막 s는 작업의 수행 상태 값을 나타내며 'Done'과 'Failed'이 포함된다. 또한 PCA분석 후 선별된 대표작업 정보(PC)는 식 (6)과 같이 세 가지 요소를 포함하여 표현된다. f_{prin_a}는 어플리케이션 측면의 대표 요인에 해당하는 명칭(예: AMACH 또는 MA)이 대응되며, f_{prin_r}은 대표 자원 요인에 해당한다. j_{prin}는 각 작업에 대하여 식 (3)을 수행하였을 때 가장 높은 수치를 갖는 작업 집합과 후보 작업들(표 5)의 교집합에 해당하는 작업들이 해당된다. $P_{selected}$에는 새롭게 수행될 작업과 가장 유사한 형태를 갖고 동시에 성능이 좋게 평가된 작업이력이 포함되며 이 프로 파일의 수행자원 정보를 참조하여 가상 머신 배치를 준비시킨다. 배치 될 가상 머신 종류는 j_{prin}에 해당하는 프로파일의 가상 머신 속성을 참조하며, 작업이 수행되면 그 프로파일을 P_{new}로 표기하고 작업이력 집합에 새롭게 포함시킨다.

전반적인 프로비저닝 과정은 다음과 같이 요약된다. 1) 먼저 축적된 작업이력을 대상으로 PCA을 이용한 프로파일 분석을 수행한다. 2) 분석을 통해 추출된 결과를 활용하여 작업수행을 위한 가상 머신이 배치될 물리자원이 결정되고 가상 머신 속성이 결정된다. 3) 수행이

Given JobHistory $J_{recent} = \{ j_1, j_2, \dots , j_n \}$
Each j_i is a d-dimensional vector
Factor F= $\{ f_1, f_2, \dots , f_d \}$
And PC means Principal Components.
1) PC=<$f_{prin\_a}, f_{prin\_r}, j_{prin}$>, f_{prin_a}= *null*, f_{prin_a}= *null*, j_{prin}= *null*
2) $P_{selected}$ = < j, $vm(j)$, r , T_{exec}, s >= ∅
3) $r_{selected}$ = *null*
4) If **PC** not exists then
5) PC= PCA(**F**, $\mathbf{J}_{recent}$)
6) EndIf
7) $P_{selected}$ = ProfileSelect(PC.f_{prin_a} , PC.f_{prin_r}, j_{new})
8) $r_{selected}$ = $P_{selected}.r$
9) $vm(j_{new})$ = vm(PC.j_{prin})
10) Deploy $vm(j_{new})$ on $r_{selected}$
11) Dispatch j_{new} on $vm(j_{new})$

그림 3 가상 머신 배치 알고리즘

ProfileSelect(f_{prin_a} , f_{prin_r} , j_{new})
1) $P_{candidate}$ =*null* , $P_{selected}$ = *null*
2) **F**={$f_1, \dots f_a, \dots, f_d$ && $1 \le a \le d$}, $f_{prin\_a} \in \{f_1, \dots f_a\}$, $f_{prin\_r} \in \{f_{a+1}, \dots f_d\}$
3) P ∈ JobHistory
4) Repeat each P In JobHistory
5) Find P where P.$j(f_{any|1 \le any \le a})$ == $j(f_{prin\_a})$
6) $P_{candidate}$ = $P_{candidate}$ ∪ P
7) until $P_{uncheck}$ == empty
8) $P_{selected}$ = MIN(P.T_{exec}) && MAX(P.$r(f_{prin\_r})$)
9) return $P_{selected}$

그림 4 참조 프로파일 선택 알고리즘

끝나면 참조했던 이력과의 비교를 통해 참조한 이력을 평가하며, 이력 분석의 재수행 여부를 결정한다. 위 과정은 다음 알고리즘들로 구체화된다.

그림 3은 제안하는 클라우드 프로비저닝 모델을 나타낸 알고리즘이다. 작업들의 특징을 나타내는 요인 종류를 정의한 후에 작업이력을 d 개의 요인으로 나타내어 축적한다. 만약 대표하는 요인이 없다면 주성분 분석을 통해 대표 작업 이력(j_{prin})을 선출하고 작업의 특징을 잘 나타내는 대표 요인들(어플리케이션 요인 f_{prin_a} 와 자원 요인 f_{prin_r})을 추출한다(그림 3, line 5). 새롭게 작업 요청이 들어오면, 작업 특징을 나타내는 요인을 바탕으로 동일한 값을 갖는 작업이력을 선택해야 하는데, 이는 그림 4와 같다. 새로 요청된 작업 정보와 대표 요인들의 정보에 따라 모든 축적한 작업 이력들 중에서 먼저 어플리케이션 요인(f_{prin_a})에 따라 동일한 값을 갖는 프로파일들을 후보로 지정한다(그림 4, line 4-7). 후보 프로파일($P_{candidate}$)중, 실행 시간 면에서 작은 값을 갖는 범위에 속하고, 자원 대표요인의 점수 면에서 가장 큰 점수를 갖는 자원에서 수행된 프로파일을 선택한다(그림 4, line 8). 가령 표 5의 예에서는 후보 프로파일 중 9번, 72번, 73번에서 사용된 자원인 rocks-52와 rocks-153이 선출되고, 이들 중 앞서 설명하였듯이 대표 자원 요인의 점수가 더 높은 rocks-153에서 수행된 73번 프로파일이 최종적으로 참조할 작업 이력($P_{selected}$)으로 결정된다. 이로부터 가상 머신을 배치할 자원(r)을 참조한다.

3.5 평가 알고리즘

선택된 자원에서 가상 머신의 수행이 성공적 일수도 있고, 실패의 가능성도 존재한다. 따라서 가상 머신 상에 스케줄링 된 작업 수행을 끝마친 작업에 대한 평가를 통해 자원의 현재 상태를 반영하도록 한다. 만일 작업 수행이 성공일 경우 가상 머신이 수행된 자원이 안정적일 확률이 높고, 실패 때에는 그 자원의 안정도가 감소하였 다고 평가할 수 있다. 그림 5는 설명한 평가 알고리즘을 순서도로 표현한 것이다. 수행 결과가 성공인 경우, 참조된 작업 프로파일의 수행 시간과 현재 수행된 가상 머신 수행 시간을 비교하여 그 오차를 계산하고, 오차의 크기가 참조된 프로파일의 수행 시간의 절반 이상을 Threshold로 정하고 그 오차와 Threshold값의 비교를 통해 요인들의 신뢰도 점수를 조절한다. 만일 오차가 특정 한계치(Threshold)보다 크다면 PCA를 재수행하여 대표 이력과 요인을 재 추출 한다. 본 논문에서는 Threshold에 대하여 참조한 프로파일의 수행 시간의 절반으로 가정하고 이 값에 대한 결정은 향후 연구로 남겨두겠다.

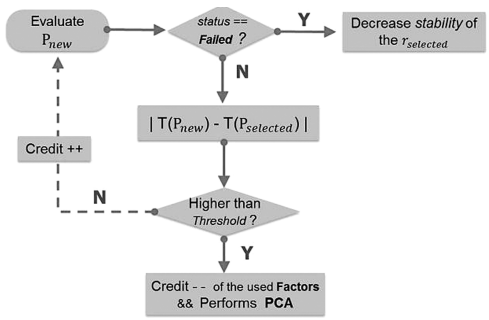

그림 5 평가 알고리즘 순서도

4. 실험 및 결과 분석

4.1 실험 환경

본 연구에서 대상으로 하고 있는 클라우드 환경의 경우, 현재 공개되어 제공되는 작업 수행이력이 거의 전무하다. 따라서 동일한 응용을 대상으로 대규모 그리드 환경에서 수행된 작업들의 평균 실행 시간과 클라우드 환경에서의 평균 실행 시간의 차를 통해 클라우드에서의 평균오버 헤드 시간을 계산 하였으며[10], 이를 활용해 클라우드 기반의 작업 이력 집합을 시뮬레이터 도구를 활용해 재생성하였다. 이때의 시뮬레이션 도구로는 Java

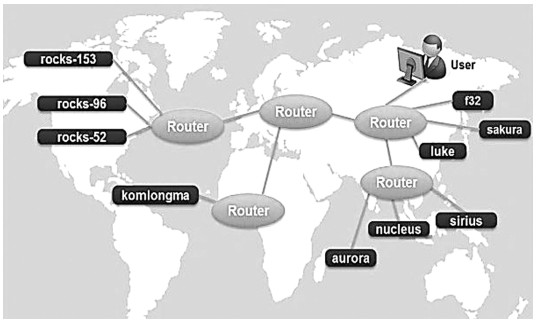

그림 6 클라우드 시뮬레이션 토폴로지

기반의 클라우드 환경 중심 시뮬레이션 도구인, 클라우드 심(Cloudsim)[2]을 사용하였으며, 새로운 작업이력 생성 에 활용된 그리드 작업 이력은 앞서 예로 들은 국내 항공우주 수치해석 포탈인 e-AIRS 2.0에서 2008년 8월부터 2009년 4월에 걸쳐 축적된 이력이다. 수행된 응용은 3.2절에 설명한 특성을 갖고 있으며, 시뮬레이션에 사용된 자원 토폴로지로는 그림 6에 보인 바와 같이 PRAGMA 그리드[9] 자원 토폴로지를 기반으로 하여 각 클러스터를 클라우드 사이트라 가정 하고 가상 클러스터 기반의 자원 토폴로지로 재 구성하여 이를 기반으로 본 시뮬레이션을 수행하였다.

4.2 성능 분석

본 실험에서는 자원 오류 없이 10000개의 작업에 대해 시뮬레이션 총 6회를 수행하였고, 동일한 시뮬레이션 자원 토폴로지에 제안한 모델과 무작위(random) 자원 선택법, 라운드 로빈 자원 선택법에 대해 시뮬레이션을 수행하여 그 결과 분석을 통해 평균 작업 실행 시간 및 대기 시간을 비교한다. 대기 시간 비교를 위해, 3.4절에서 제시한 프로파일의 파라미터에 작업 대기 시간을 나타 내는 파라미터를 추가하였으며, 또한 표준 성능을 대표 하는 비교 치를 위해 **e-AIRS2.0** 에서 실제 사용하고 있는 자원 선택법을 통해 수행된 작업들을 분석하여 평균(작업) 실행 시간 및 평균 대기시간을 계산 후 함께 비교하였다. **평균 작업실행 시간**은 클라우드 환경일 경우, 가상 머신이 배치된 시점부터 다시 회수된 시점까지의 시간을 의미하며, 그리드 환경의 경우는 작업 제출 후 계산 자원에서 의 실행 명령 후 모든 수행이 끝난 시점까지를 측정 하였다. 작업이력 분석과 결과 분석을 위해서는 오픈 소스 통계 툴인 R 통계[11]를 사용하여 분석하였다. 실험 결과는 그림 7과 같다.

먼저 그림 7의 상단 그래프를 참고하면 작업 실행 시간 만을 비교해 보았을 때 클라우드 환경에서의 작업 수행(B, C, D)이 그리드 환경 (A)에서 보다 작업 실행 시간이 더 소요됨을 알 수 있다. 이는 프로비저닝 시간

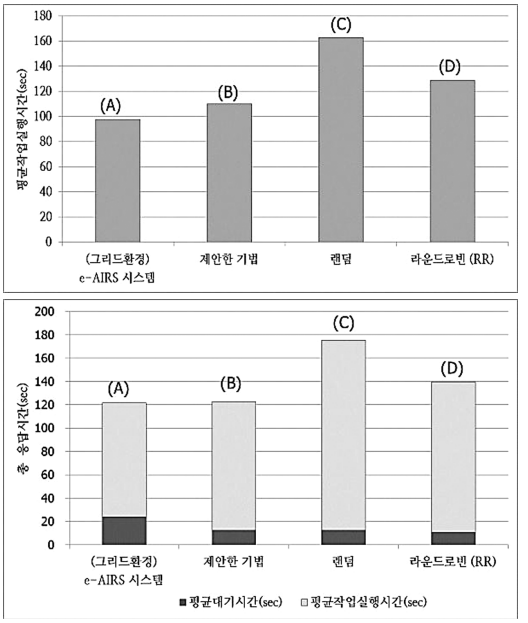

그림 7 자원선택 기법 별 작업 실행시간비교(상) 및 총 응답시간 비교(하)

이외에 가상화 기반 수행 환경에서의 오버헤드 때문이다[10]. 또한 클라우드 환경에서 다양한 프로비저닝 기법과 비교하였을 때, 제안한 프로비저닝 모델 적용 시의 작업 실행 시간이 적게 걸린 것으로 보인다. 이는 C, D 방식의 경우, 가상 클러스터들 사이에서 특정 위치에 가상 머신 이 집중되어 수행되므로 물리 자원을 최대 한계치까지 나누어 쓰기 때문에 성능 저하를 발생시켜 수행 시간의 증가를 일으키기 때문이다. 또한 가상 클러스터의 상태 즉 각 사이트의 상태에 상관 없이 가상 머신을 배치하기 때문에 특정 위치에 가상 머신이 치우치게 되어 그만큼 수행 시간의 지체를 발생한 것으로 보인다. 그림 7의 하단 그래프는 전체 응답 시간을 비교한 그래프이다. 전체 응답 시간은 사용자의 대기 시간을 포함한 수치이다. 먼저 (A)와 (B)의 응답 시간을 비교해 보면, 전체 응답시간 면에서는 앞선 결과와 달리 유사하게 시간이 소요되는 결과가 발생하였다. 이는 제안한 기법에서의 평균 작업 수행 시간이 더 길었음에도 불구하고, 큐에서 대기하는 시간이 비교적 짧았기 때문이다. 클라우드 환경에서는 다수의 물리 자원에 대해 동적 자원 할당이 가능하기 때문에 큐에서 대기 시간이 다른 환경보다 상대적으로 적다. 게다가 제안한 방식을 적용하는 경우, 균형적인 가상 머신 배치 및 작업 스케줄링이 제공되기 때문에 대기 시간의 따라서 총 응답 시간으로 비교하였을 때, 본 논문에서 제안한 가상 머신 할당을 위한 자원 선택 모델이 성능 면에서 그리드 환경에서의

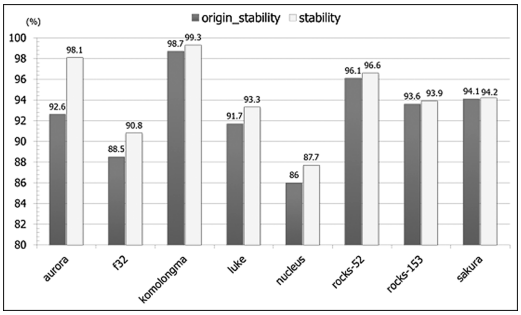

그림 8 실험 전/후 자원의 안정도 비교

작업 수행 성능을 제공하며 클라우드 환경의 동적인 자원 활용의 장점을 제공받을 수 있다.

추가적으로, 작업 수행의 결과가 실패(Failed)라고 여겨지는 기준에 가상 머신의 배치 명령 후, 큐잉 시간이 평균 큐잉 시간의 일정 배수 이상인 경우를 포함하였으며, 여기에서는 2배로 가정하고 실험을 하였다. 작업이 실패가 되는 경우, 그 작업은 다른 클러스터 자원으로 재할당이 되어야 되므로 전체 시스템의 처리량 (Through-put)을 감소시키는 원인이 될 수 있다. 따라서 실패 횟수를 줄여 전체 처리량에 영향을 주지 않는 것이 중요하다. 실패 횟수를 줄이려면, 자원의 상태를 잘 반영하여 작업 스케줄링을 수행하여야 한다. 본 논문에서 제안한 방법이 자원의 상태를 잘 반영하여 작업 스케줄링을 하는지를 확인하기 위해 시뮬레이션 전/후의 자원 안정도를 비교했다. 안정도는 전체 수행 횟수에 대한 성공률이다. 결과는 그림 8과 같다. 그림 8에서 붉은 색(좌측) 막대 그래프는 실험 전의 안정도를 나타내며, 초록 색(우측) 막대 그래프는 실험 후의 자원 안정도를 나타낸다. 결과에서 볼 수 있듯이 전반적으로 클러스터 자원들의 안정도가 증가함을 보였으며 최대 5% 이상의 향상을 보였다. 따라서 제안한 프로비저닝 모델을 사용하였을 때 더욱 실패율을 줄일 수 있으며 안정적인 자원에의 프로비저닝을 통해 전반적인 자원 안정도와 작업 처리량의 증가를 이끌었음을 알 수 있다.

5. 결론 및 향후 연구

현재 클라우드 데이터 센터는 물리적인 서버 상에 하나의 컴퓨터 시스템뿐만 아니라 여러 개의 시스템이 동작하고 있다. 이러한 물리적 서버의 효율적인 관리를 위해 가상화 기술이 활용되고 있다. 하지만 가상화 기술을 통해 효율적으로 자원 활용이 가능한 반면, 자원 관리 비용과 복잡도가 증가한다. 따라서 클라우드 시스템의 운용을 위한 효율적이고 향상된 자동 자원 관리 메커니즘이 요구된다. 본 논문에서는 사이언스 클라우드 환경에서의 작업 수행 이력의 요인 분석을 통해 그 영향력에 따라 클라우드 자원을 프로비저닝 함으로써 효율적인 클라우드 자원 관리를 이끄는 모델을 제안하였다. 특히 본 연구에서는 과학 분야의 어플리케이션에 초점을 맞추어 과학 어플리케이션의 수행으로 축적된 작업 이력을 기반으로 시뮬레이션 실험을 하였으며 적응적 자원 프로비저닝을 제공함을 보였다.

향후 연구로 다양한 작업 이력을 바탕으로 여러 측면의 요인을 분석해 보고 요인 별 반영 정도에 따른 성능 분석을 추가적으로 연구할 것이며, 사이언스 클라우드 환경과 계산 그리드 환경을 동시에 활용할 수 있는 하이브리드(hybrid) 환경에서의 작업 스케줄링을 본 논문에서 제안한 알고리즘을 보완하여 적용할 것이다.

참 고 문 헌

[1] Jolliffe, I. T., "Principal Component Analysis," Springer Verlag, 2002.

[2] Rodrigo N. Calheiros, Rajiv Ranjan, Anton Beloglazov, Cesar A. F. De Rose, and Rajkumar Buyya, "CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms, Software: Practice and Experience (SPE)," vol.41, no.1, pp.23-50, ISSN: 0038-0644, Wiley Press, New York, USA, Jan. 2011.

[3] XiaoYing Wang, et al., "Appliance-Based Autonomic Provisioning Framework for Virtualized Outsourcing Data Center," *Autonomic Computing, 2007. ICAC '07.* Fourth International Conference on, p.29, 11-15 Jun. 2007.

[4] Machida, F., Kawato, M., Maeno, Y., "Just-in-Time Server Provisioning Using Virtual Machine Standby and Request Prediction," *Autonomic Computing, 2008. ICAC '08. International Conference on*, pp.163-171, 2-6 Jun. 2008.

[5] Nikolas Huber, Marcel von Quast, Fabian Brosig, Samuel Kounev, "Analysis of the Performance-Influencing Factors of Virtualization Platforms," *OTM 2010, Part II, LNCS 6427*, pp.811-828, 2010.

[6] Akoush, S., Sohan, R., Rice, A., Moore, A.W., Hopper, A., "Predicting the Performance of Virtual Machine Migration," *Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS), 2010 IEEE International Symposium on*, pp.37-46, 17-19 Aug. 2010.

[7] CFD, http://www.cfd-online.com/

[8] e-AIRS 2.0, http://eairs.kisti.re.kr/gridsphere/

[9] PRAGMA (Pacific Rim Application and Grid Middleware Assembly), http://www.pragma-grid.net/about

[10] Seoyoung Kim, Yoonhee Kim, Naeyoung Song, Chongam Kim, "Adaptable scheduling schemes for scientific applications on science cloud," *Cluster*

Computing Workshops and Posters (CLUSTER WORKSHOPS), 2010 IEEE International Conference on, pp.1-3, 20-24 Sept. 2010.

[11] R Development Core Team (2011), "R: A language and environment for statistical computing," R Foundation for Statistical Computing, Vienna, Austria, http://www.R-project.org/

김 서 영

2010년 숙명여자대학교 컴퓨터과학과 졸업(학사). 2012년 숙명여자대학교 대학원 컴퓨터 과학부 졸업(석사). 관심분야는 그리드/클라우드 관리, 메타 스케줄링, 워크플로우 제어 등

김 윤 희

1991년 숙명여자대학교 전산학과(학사) 1996년 Syracuse Univ. 전산학과(석사) 2000년 Syracuse Univ. 전산학과(박사) 1991년~1994년 한국전자통신연구원. 2000년~2001년 Rochester Institute of Technology 컴퓨터공학과 조교수. 2001년~2004년 숙명여자대학교 컴퓨터과학과 조교수. 2004년~2009년 숙명여자대학교 컴퓨터과학과 부교수. 2009년~현재 숙명여자대학교 컴퓨터과학부 교수. 관심분야는 그리드 컴퓨팅 환경(PSE), 워크플로우 제어, 그리드/클라우드 관리 등

황 순 욱

1990년 서울대학교 수학과(학사). 1995년 서울대학교 계산통계학과(석사). 2003년 미국 남가주대학교 전산과학과(박사). 2003년~2006년 일본 국립 정보학 연구소 연구원. 2006년~현재 한국과학기술정보연구원 책임연구원. 관심분야는 그리드 컴퓨팅, 클라우드 컴퓨팅, 분산 컴퓨팅 등