스팟 마켓 기반 클라우드 컴퓨팅에서 VM 이주+

정대용*, 최숙경*, 이정하*, 정광식**, 유헌창*
*고려대학교 대학원 컴퓨터교육학과
**방송통신대학교 정보과학과
e-mail : karat@korea.ac.kr

VM Migration in Spot Market Based Cloud Computing

Daeyong Jung*, SookKyong Choi*, Jungha Lee*, Kwang Sik Chung**, HeonChang Yu*
*Dept of Computer Science and Education, Korea University
**Dept of Computer Science, Korea National Open University

요 약

스팟 인스턴스(Spot instance)는 클라우드 환경에서 사용자가 제시한 입찰 가격으로 클라우드 내의 자원을 이용하여 작업을 수행할 수 있게 하는 새로운 방식이다. 사용자는 자신의 입찰 금액이 클라우드 내의 스팟 인스턴스 가격을 초과하는 한 인스턴스를 실행할 수 있다. 그러나 입찰 가격이 스팟 가격보다 낮다면 작업 실패가 발생하고, 이로 인해 작업 완료 시간은 지연되며 서비스 품질은 저하된다. 이 문제를 해결하기 위해, 본 논문에서는 스팟 인스턴스에서 사용자 비드가 초과되어 작업 수행이 중지된 VM에 대하여 체크포인트 기법과 VM 이주(migration) 기법을 이용함으로써 작업 대기 시간을 줄이는 방법을 제안한다. 이는 작업 중인 스팟 인스턴스에서 작업 실패가 발생할 경우 다른 인스턴스로 이주하여 작업을 재수행하는 기법이다. 실험 결과는 제안하는 VM 이주 기법이 작업을 수행할 수 있는 스팟 인스턴스의 가용성을 증가시킬 수 있음을 보여준다.

1. 서론

최근 클라우드 컴퓨팅에 대한 관심으로 인해 많은 클라우드 프로젝트 및 상업 시스템들이 구현되고 있다. 대다수의 클라우드 컴퓨팅 시스템은 사용자에게 비용-효율적인 방법으로 인스턴스를 제공한다. 일반적으로 인스턴스는 주문형 인스턴스(on-demand instance)와 스팟 인스턴스(spot instance)로 구분된다. 주문형 인스턴스는 사용자의 요구에 따라 고정된 가격으로 클라우드 자원을 가상화시켜 사용자에게 제공하고, 스팟 인스턴스는 사용자 입찰 가격에 근거하여 클라우드 자원을 사용자에게 제공한다[1].

스팟 마켓 기반의 클라우드 환경은 사용자의 수요에 따라 스팟 가격이 변동되고 작업이 수행 또는 실패하게 되는 단점이 있다. 스팟 가격은 사용하고자 하는 스팟 인스턴스에 사용자가 많아지면 가격이 오르고 사용자가 적어지면 가격이 내려가는 마켓 구조를 따르므로, 스팟 인스턴스를 사용중인 사용자의 입찰 가격이 현재의 스팟 가격보다 낮다면 사용 중인 스팟 인스턴스인 VM은 즉각 정지되고 VM 내에서 수행중인 작업은 실패하게 된다. 이후에 스팟 가격이 사용자의 입찰 가격보다 낮아지면 정지되었던 VM이 다시 활성화되고 작업은 재수행된다[2, 3, 4].

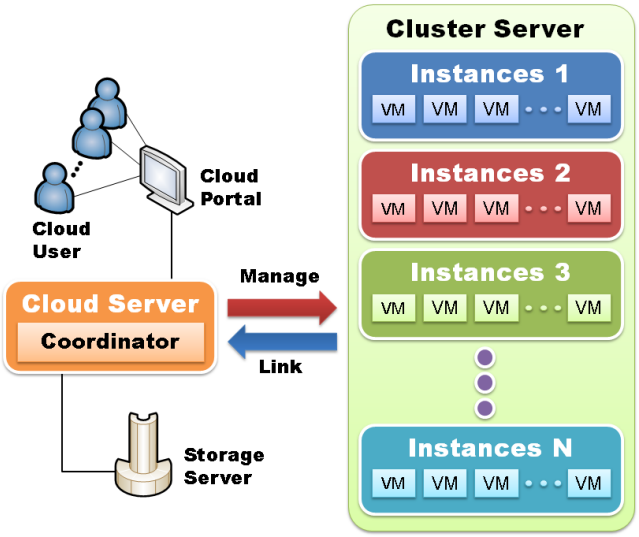
따라서 현재의 스팟 가격에 의해 수행중인 작업이 실패하는 문제를 해결하고자, 본 논문에서는 기존 연구[5]에서의 체크포인트 기법을 이용하여 작업 수행 시간을 줄일 수 있는 VM 이주(migration) 기법을 제안한다. 기존의 체크포인트 기법은 작업의 평균 수행 시간을 이용하여 VM의 실패 시점을 예상하여 체크포인트를 수행함으로써 자원의 가격 변동에 따른 작업 실패 시 작업의 손실을 최소화하고 VM의 회복 시간을 줄여 전체 작업 완료 시간을 줄일 수 있었다. 하지만 스팟 가격이 사용자의 입찰 가격보다 높을 경우 VM에서의 작업이 중단되므로 작업이 재수행할 때까지 작업 대기 시간이 발생하였다. 이에, 본 논문에서는 체크포인트 기법을 이용하여 스팟 인스턴스에서 사용자 비드가 초과되어 VM이 중지되어 작업 수행하지 못하는 작업 대기 시간을 줄일 수 있는 VM 이주 기법을 제안한다. 단, VM을 이주시키기 위한 시간이 추가적으로 발생한다.

2. 시스템 구성

스팟 인스턴스를 이용한 클라우드 컴퓨팅 환경은 (그림 1)과 같다. 클라우드 컴퓨팅 환경은 클라우드 사용자, 클라우드 포탈, 클라우드 서버, 클러스터 서버, 스토리지 서버로 이루어진다. 클러스터 서버는 여러 인스턴스 타입

+ 본 논문은 2012년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. 2012-046684)

을 가진 노드들로 구성된다. 클러스터 서버를 구성하는 각 노드는 각 인스턴스 타입에 맞는 VM을 생성하여 관리한다. 클라우드 사용자는 클라우드 포탈에 접속하여 클러스터 서버 내의 스팟 인스턴스 유형을 선택하고, 스팟 인스턴스인 VM을 이용한다. 클라우드 사용자의 요구사항에 맞는 VM을 수행시키기 위해 클라우드 서버 내의 코디네이터는 스팟 인스턴스들의 히스토리를 관리하고, VM이 여러 인스턴스들 사이를 이주할 수 있도록 각 인스턴스를 모니터링한다. 또, 각각의 VM 노드가 스팟 인스턴스에 대한 체크포인트를 수행하고, 클라우드 서버가 VM 이주를 결정한다.



(그림 1) 스팟 인스턴스를 이용한 클라우드 환경

3. 작업 시간을 줄이기 위한 VM 이주 방법

본 논문에서는 사용자가 하나의 스팟 인스턴스 타입을 선택하여 VM에서 작업을 수행할 때 현재의 스팟 가격이 사용자 입찰가격(비드)을 초과하여 사용자 작업이 계속 수행되지 못하고 정지되었을 경우, 다른 스팟 인스턴스로 VM을 이주함으로써 작업을 계속 수행할 수 있는 방법을 제안한다.

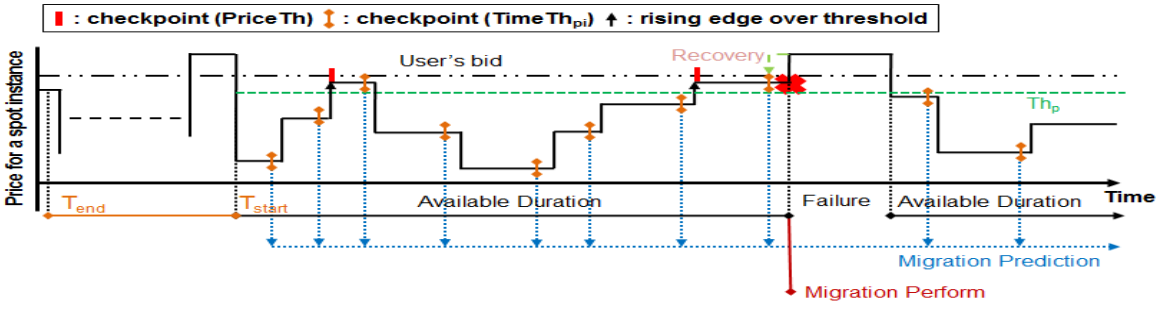
VM 이주를 위해 기존의 연구[5]에서 제안한 가격 히스토리 기반의 체크포인트 및 체크포인트 시점 예측 기법을 이용한다. 기존 연구에서 체크포인트는 다음의 경우에 수행된다. 첫째, 사용자가 제시한 가격과 가격 임계값 사이에 상승 엣지가 있을 경우에 체크포인트를 한다. 둘째, 스팟 로그 히스토리 정보를 이용하여 각 비드의 결함 확률과 평균 작업 시간을 계산하여 결함 이전의 예상 시간에서 체크포인트를 한다. 첫 번째의 경우, 작업 실행 중에 스팟 인스턴스의 가격이 사용자 비드를 초과한다면 많은 데이터 손실이 발생할 수 있다. 따라서 본 논문에서는 두 번째 방법을 이용하여 각 비드에서 사용할 수 있는 평균 작업 시간을 이용한 체크포인트 정보를 전체 인스턴스의 이용 시간을 최소화하도록 한다.

VM 이주를 판단하기 위해서 체크포인트 시점에 다른 인스턴스의 스팟 가격과 이용 가능 시간을 확인한다. VM 이주 시 발생하는 비용을 고려하여 VM 이주 시간이 수행 시간보다 작고 연속적으로 수행이 가능할 경우에 VM 이주를 해야 한다. 또, 사용 중인 인스턴스에서 작업이 정지되었을 경우에는 이미 계산한 최적의 인스턴스로 VM을 이주한다. 인스턴스 타입, 스팟 로그 히스토리 정보 등의 상태에 따라 이주를 판단하기 위한 최적의 임계값을 설정할 수 있다.

(그림 2)는 가격 히스토리 기반 체크포인트 기법 및 VM 이주 기법을 보여준다. 체크포인트 시점을 결정하기 위해 스팟 로그 히스토리 정보를 이용하여 가격 임계값과 시간 임계값을 결정한다. 결정된 두 임계값을 통하여 인스턴스가 중지될 경우, 두 임계값에서 체크포인트한 시점에서 작업을 재수행된다. 그리고 VM 이주 기법을 통해 다른 인스턴스로 이동을 하게 되는데, 기존 인스턴스에서 다른 인스턴스로 이동하는 위치를 판단하는 시점은 시간 임계값 시점에 체크포인트되는 시점에 가격과 이동 가능한 시간을 고려한 최적의 인스턴스를 찾는다. 그리고 결함이 발생했을 경우, 미리 검색한 인스턴스로 VM을 이주하고 마지막 체크포인트 시점부터 작업 수행을 재시작한다.

그림에서 표현한 용어에 대한 의미는 다음과 같다.

- Checkpoint(체크포인트) : 가격 임계값(PriceTh)에 따른 체크포인트와 시간 임계값(TimeTh)에 따른 체크포인트가 있음. 전자의 경우, 설정한 임계값을 초과했을 경우에 체크포인트 함. 후자의 경우, 각 비드에서 설정된 수행시간에 체크포인트 함.
- Migration Prediction(이주 예측) : 시간에 따른 체크포인트 위치인 TimeTh에서 결함 시 이주 가능한 인스턴



(그림 2) 체크포인트 및 이주 방법

스를 계산함.

- Migration Perform(이주 수행) : 작업 수행이 실패했을 경우 다른 인스턴스로 이주하는 시점.
- T_{start}, T_{end} : 체크포인트 임계값을 계산하기 위해 가격 히스토리를 분석하는 시작 구간과 끝 구간.

4. 실험

본 연구에서의 실험은 아마존에서 제공하는 스팟 인스턴스의 가격 히스토리[6] 및 인스턴스의 타입을 이용하였다. 이들 데이터를 이용하여 VM 이주를 할 경우 특정 기간 동안에 작업을 수행할 수 있는 스팟 인스턴스의 가용성을 확인해보고자 하였다.

4.1 실험 데이터

아마존에서 제공하는 스팟 인스턴스의 타입은 <표 1>과 같다. m1.xlarge은 기본 인스턴스(standard instances) 중 하나이고, c1.xlarge는 고성능 인스턴스(high-CPU instances) 중 하나이다.

<표 1> 인스턴스 타입에 따른 속성

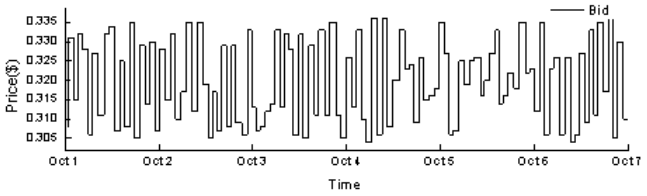
종류 / 속성	m1.xlarge	c1.xlarge
Compute unit	8 EC2	20 EC2
Virtual cores	4 cores (2 EC2)	8 Cores (2.5 EC2)
Memory	15 GB	7 GB
Storage	1690 GB	1690 GB

(그림 3)은 아마존에서 제공한 두 스팟 인스턴스의 가격 변화 추이를 보여준다(2010년 10월 1일 ~ 2010년 10월 7일). (a)는 m1.xlarge의 스팟 가격 변화를, (b)는 c1.xlarge의 스팟 가격 변화를 보여준다.

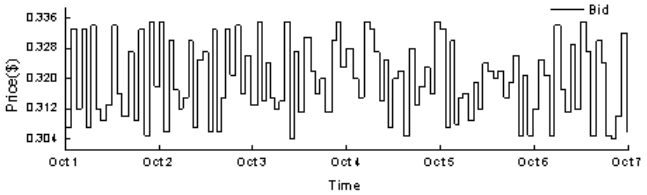
4.2 실험 결과

(그림 3)에서의 기간 동안 사용자 비드가 $0.32일 경우의 스팟 인스턴스의 가용성을 (그림 4)에 나타내었다. 가용성은 실험 기간 동안 각 스팟 인스턴스가 작업을 수행할 수 있는 정도를 나타낸다. (a)는 m1.xlarge 인스턴스를 사용할 경우의 가용성(55.22%)을, (b)는 c1.xlarge 인스턴스를 사용할 경우의 가용성(54.04%)을, (c)는 VM 이주를 사용하여 m1.xlarge와 c1.xlarge의 두 인스턴스를 사용할 경우의 가용성(85.94%)을 보여준다. 하나의 인스턴스를 사용했을 때보다 두 인스턴스를 사용하여 VM 이주를 했을 경우, 인스턴스의 가용성이 높아짐을 알 수 있다.

또, VM 이주 기법을 이용했을 때의 가용성이 m1.xlarge 인스턴스를 단독으로 사용했을 때보다 30.72%, c1.xlarge 인스턴스를 단독으로 사용했을 때보다 35.9% 증가하였다.

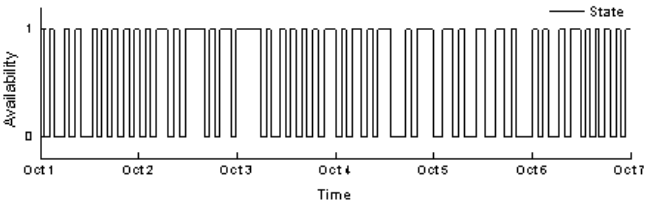


(a) eu-west-1.linux.m1.xlarge



(b) eu-west-1.linux.c1.xlarge

(그림 3) 스팟 가격 변화 추이



(a) eu-west-1.linux.m1.xlarge

(b) eu-west-1.linux.c1.xlarge

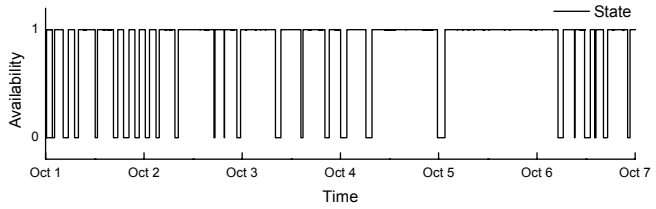
(c) Combine (a) with (b)

(그림 4) 인스턴스의 상태(사용자 비드 : $0.32)

(그림 5)는 사용자 비드를 $0.325로 설정했을 경우의 가용성을 보여준다. 사용자 비드가 $0.325일 경우, m1.xlarge 인스턴스의 가용성은 67.36%, c1.xlarge 인스턴스의 가용성은 62.03%, VM 이주를 사용하여 두 인스턴스를 사용할 경우의 가용성은 99.53%이다.

또, VM 이주 기법을 이용했을 때의 가용성이 m1.xlarge 인스턴스를 단독으로 사용했을 때보다 32.17%,

c1.xlarge 인스턴스를 단독으로 사용했을 때보다 37.5% 향상되었다.



(그림 5) VM 이주 했을 경우의 가용성
(사용자 비드 : $0.325)

이상의 결과로, 사용자 비드가 증가할수록, VM 이주 기법을 이용할수록 각 인스턴스의 가용성이 증가하는 것을 알 수 있다. 또, VM 이주할 수 있는 인스턴스 수를 증가시킬수록 인스턴스의 가용성은 증가한다.

5. 결론

본 논문에서는 스팟 인스턴스 기반의 클라우드 컴퓨팅 환경에서 작업 수행의 안정성을 높이고 작업 수행 시간을 최소화하기 위한 클라우드 자원의 가격 변동을 고려하는 VM 이주 기법을 제안하였다. 스팟 로그 기반 체크포인트 기법을 이용하여 체크포인트 시점을 효율적으로 결정하고 각 사용자 비드에서의 이용 시간을 고려한 체크포인트 기법을 VM 이주에 적용함으로써, 사용 중인 인스턴스에서 VM 결함이 발생하더라도 다른 인스턴스로 VM을 이주하여 작업을 계속 수행할 수 있도록 하였다.

제안하는 VM 이주 기법을 검증하기 위해 아마존에서 제공하는 인스턴스인 m1.xlarge와 c1.xlarge를 이용하여 실험을 하였고, 이를 통해 하나의 인스턴스를 사용했을 때보다 VM 이주 기법을 이용하는 경우 인스턴스의 가용성이 높아짐을 확인하였다. 향후 연구로 VM 이주 기법을 적용한 환경과 기존 환경을 비교 분석하여 성능의 차이를 알아보고자 한다.

참고문헌

[1] I. Foster, Z. Yong, I. Raicu, and S. Lu, “Cloud Computing and Grid Computing 360-Degree Compared”, Proc. of 2008 Grid Computing Environments Workshop, pp. 1-10, 2008.

[2] Amazon EC2 spot Instances, http://aws.amazon.com/ec2/spot-instances.

[3] S. Yi, D. Kondo, and A. Andrzejak, “Reducing Costs of Spot Instances via Checkpointing in the Amazon Elastic Compute Cloud” Proc. of the 2010 IEEE 3rd Int. Conf. on Cloud Computing, pp. 236-243, 2010.

[4] S. Yi, J. Heo, Y. Cho, and J. Hong “Taking Point Decision Mechanism for Page-level Incremental Checkpointing based on Cost Analysis of Process Execution Time,” J. of Information Science and Engineering, vol. 23, no. 5, pp. 1325 - 1337, 2007.

[5] D. Jung, S. Chin, K. Chung, H. Yu and J. Gil., “An Efficient Checkpointing Scheme Using Price History of Spot Instances in Cloud Computing Environment”, In: Proceeding NPC'11 Proc. of the 8th IFIP Int. Conf. on Network and parallel computing. pp. 185-200, 2011.

[6] Cloud exchange, http://cloudexchange.org.