Journal of The Korea Society of Computer and Information
Vol. 19, No. 11, November 2014
www.ksci.re.kr
http://dx.doi.org/10.9708/jksci.2014.19.11.001

클라우드 클러스터에서 가상머신 재배치시간을 단축하기 위한 재매핑 기법

김 창 현*, 김 준 상*, 전 창 호*

A Virtual Machine Remapping Scheme for Reducing Relocation Time on a Cloud Cluster

Chang-Hyeon Kim*, Jun-Sang Kim*, Chang-Ho Jeon*

요 약

본 논문에서는 클라우드 클러스터에서 가상머신(VM: Virtual Machine)의 재배치시간을 단축할 수 있는 VM 재매핑 기법을 제안한다. 제안하는 기법은 입력으로 주어진 VM 맵으로부터 순차적으로 이주해야 하는 VM들을 찾고 그 중 일부 VM들의 목적지를 교환함으로써 VM 재배치시간을 단축한다. 목적지가 교환될 VM은 이주 완료시간시간과 물리머신들의 가용 자원량을 근거로 하여 선정된다. 그리고 목적지 교환은 VM 재배치 시간이 더 이상 단축될 수 없을 때까지 반복된다. 시뮬레이션을 통하여 VM 맵을 제안한 기법으로 재매핑 했을 때 재매핑 전에 비해 VM 재배치 시간이 최대 42.7% 단축되었음을 확인한다.

▸ Keywords : 클라우드 컴퓨팅, 가상머신, 재배치, 매핑

Abstract

In this paper, we propose a virtual machine(VM) remapping scheme that reduces VM relocation time on a cloud cluster. The proposed scheme finds VMs that should be migrated in sequence from a given VM map, and exchanges destinations of some VMs among them to reduce the VM relocation time. The VMs, the destinations of which will be exchanged, are chosen based on the amount of physical machine's available resources and migration completion time. The exchange of destinations is repeated until the VM relocation time cannot be shortened any further. Through a simulation, we show that the proposed scheme reduces VM relocation time by 42.7% in maximum.

▸ Keywords : cloud computing, virtual mahcine, relocation, mapping

•제1저자 : 김창현 •교신저자 : 전창호
•투고일 : 2014. 8. 19, 심사일 : 2014. 9. 4, 게재확정일 : 2014. 10. 10.
* 한양대학교 컴퓨터공학과(Dept. of Computer Science & Engineering, Hanyang University ERICA Campus)

I. 서 론

가상화 기술은 클라우드 컴퓨팅의 핵심기술로서 물리적 자원을 논리적으로 구성하여 가상머신(VM: Virtual Mahcine)을 운영할 수 있게 한다. VM은 하나의 독립적인 컴퓨팅 환경으로 물리머신(PM: Physical Machine)의 자원량을 넘어서지 않는 한도에서 자원을 점유하고 사용할 수 있다. 그러므로 여러 개의 VM들이 PM의 자원량 한도 안에서 동시에 운영될 수 있다. 또한 가상화로 인해 VM은 물리자원과 낮은 결합도를 가지기 때문에 VM의 자원이 재할당 될 필요가 있을 때 자원을 제공할 수 있는 다른 PM으로 이주할 수 있다. 이러한 특징은 PM의 자원을 유연하게 사용할 수 있게 한다[1].

클라우드 클러스터는 일반적으로 수백 또는 수천대의 PM들로 이루어진 대규모 클러스터로 에너지 관리가 중요하다. 클러스터의 에너지 소비는 대부분 PM들에 의해 소모된다. 그러므로 운영하는 PM들의 수를 줄일 수 있다면 에너지 소비를 절감할 수 있다. 이러한 관점에 대한 연구의 일환으로 서버통합이 있다. 서버 통합은 VM들을 재배치하여 일부 PM들이 VM을 호스팅하지 않게 하고 그 PM들의 전원을 차단하여 에너지를 절약한다[2].

서버통합은 VM 매핑과 재배치 단계로 나누어진다. VM 매핑 단계에서는 VM들의 자원 요구량에 기반해서 VM들과 PM들의 맵을 생성한다. 그리고 재배치 단계에서는 매핑 단계에서 생성한 VM 맵대로 VM들을 이주시킨다. VM 매핑 단계는 재배치를 계획하는 단계이며 재배치 단계는 계획대로 수행하는 단계로 볼 수 있다. 그러므로 VM 매핑 단계에서 VM들을 PM으로 어떻게 매핑하는가에 따라 PM들의 자원 활용률과 VM 재배치 시간이 달라진다.

최근까지 VM 매핑에 초점을 맞춰 많은 연구들이 진행되어왔으며 공통적으로 VM 매핑을 Bin packing 문제로 접근하고 있다. Bin packing 문제는 다양한 크기의 아이템을 최소의 bin에 담는 문제로 아이템을 이동시키는 비용을 고려하지 않는다. 따라서 VM 매핑을 Bin packing 문제로 접근할 경우 VM들을 재배치하는데 소요되는 시간이 고려되지 않기 때문에 VM재배치 시간이 길어질 수 있다. VM 재배치 시간이 길다면 워크로드가 유동적인 클라우드 서비스 특성상 VM 재배치 도중 VM들의 자원 요구량이 변하여 VM 맵이 무효화 될 수 있다. 그러므로 VM 매핑 과정에서 VM 재배치 시간을 고려할 필요가 있다.

VM 맵대로 VM들을 이주시킬 때 곧 바로 이주할 수 있는 VM과 이주를 대기한 후 이주할 수 있는 VM이 있다. 이주를 대기하는 상황은 PM들의 한정된 자원량 때문에 발생한다. VM이 PM으로 이주하기 위해서는 PM에 VM이 필요한 만큼의 자원이 남아 있어야 한다. 그러나 필요한 만큼의 자원이 없다면 그 PM에서 다른 VM이 이주해나감으로써 자원이 해제될 때까지 기다려야 한다.

VM 재배치시간은 첫 번째로 이주하는 VM이 이주를 시작한 시점부터 마지막으로 이주하는 VM이 이주를 완료하는 시점까지 경과시간과 같다. VM을 재배치할 때 이주를 대기할 필요가 없는 VM들은 VM 재배치가 시작됨과 동시에 모두 이주를 시작한다. 반면에 이주를 대기하는 VM들은 늦게 시작할 수밖에 없다. 그에 따라, VM의 재배치시간이 지연될 수 있다. 따라서 VM 재배치 시간을 단축하기 위해서는 VM들의 이주 완료시간을 앞당길 필요가 있다.

본 논문에서는 VM 맵이 주어질 때 VM들의 목적지를 수정하여 VM 재배치 시간을 단축하는 VM 재맵핑 기법을 제안한다. 제안하는 기법은 우선, VM 맵을 기준으로 순차적으로 이주해야하는 VM들을 찾는다. 그리고 찾은 VM들 중 일부 VM들의 목적지를 수정함으로써 VM들을 재배치하는데 소요되는 시간을 단축한다.

본 논문의 구성은 다음과 같다. 2장에서는 관련연구를 살펴보고 3장에서는 VM 재배치 시간을 단축시키는 VM 재배핑 기법을 제안한다. 4장에서는 시뮬레이션을 통해 제안하는 기법의 성능을 평가한다. 그리고 5장에서 결론을 내린다.

II. 관련 연구

가상화를 통해 자원을 유연하게 사용할 수 있는 클라우드 컴퓨팅의 특성에 기반하여 클러스터의 에너지 소모를 절감하기 위한 다양한 서버통합 기법이 제안되었다[3][4][5]. 이들 연구의 공통점은 Bin packing문제를 휴리스틱 방법으로 접근한 FFD(First Fit Decreasing) 알고리즘에 기반을 두고 VM 매핑을 수행하는 것이다.

FFD 알고리즘은 가변크기를 가지는 정해진 개수의 아이템을 크기를 기준으로 내림차순으로 정렬하고 하나씩 bin에 채워나가는 방법으로 낮은 계산 복잡도와 좋은 성능을 보여준다. 그러나 Bin packing문제가 아이템을 이동시키는 비용을 고려하지 않듯이 FFD 알고리즘 역시 아이템을 이동시키는 비용을 고려하지 않는다. 따라서 FFD 알고리즘으로 VM들을 매핑한다면 PM들의 자원 활용률은 향상시킬 수 있으나 VM 재배치에 많은 시간이 소요될 수 있다.

몇몇 연구에서는 서버통합에 소요되는 시간을 줄이기 위해

VM들의 이주시간을 고려하여 VM 매핑을 수행하였다. Ferreto 등은 VM의 이주시간으로 인해 서버통합의 완료시점이 지연될 수 있음을 지적하고 주어진 시간 안에 서버통합을 완료할 수 있는 VM 매핑 기법을 제안하였다[6]. Ho 등은 이주하는 VM의 수를 줄임으로써 VM 재배치시간을 단축시키는 VM 매핑 기법을 제안하였다[7]. Beloglazov 등은 주어진 PM의 자원 활용률 상한을 초과시키는 VM들만 자원 활용률이 낮은 PM으로 매핑시킴으로써 이주가 필요한 VM들의 수를 줄이는 기법을 제안하였다[8]. 이들 연구들은 VM 재배치에 소요되는 시간을 줄이기 위해 이주시킬 VM들의 수를 줄이는 방법으로 접근한다. 그러므로 이들 기법은 이주시킬 VM들의 수에 대한 제약 때문에 자원 활용률 향상측면에서만 본다면 FFD 알고리즘 보다 성능이 낮다.

자원 활용률을 향상과 긴 VM재배치 시간으로 인한 문제점들을 동시에 고려할 때 높은 자원 활용을 보장하면서 VM 재배치 시간을 단축할 필요가 있다.

III. 가상머신 재매핑 기법

본 장에서는 VM 재배치시간을 단축시키기 위한 VM 재매핑기법을 제안한다. 제안하는 기법은 VM 맵으로부터 순차적으로 이주해야하는 VM들을 찾고 그 VM들 중 일부 VM들의 목적지를 수정함으로써 VM들을 재배치하는데 소요되는 시간을 단축한다.

VM 맵은 VM들의 자원 요구량과 PM의 보유 자원량을 바탕으로 생성된 PM들과 VM들 간의 호스팅, 피호스팅 관계를 나타낸다. 그러므로 VM 맵을 참고하면 VM들의 이주 목적지를 알 수 있다. 그리고 한 VM이 이주해갈 PM에 자원 여유분이 부족하다면 그 PM에서 이주해나갈 예정인 VM들의 자원량을 비교하여 어떤 VM이 이주해나갈 때까지 대기해야하는지 알 수 있다.

그림 1은 FFD 알고리즘으로 VM 맵을 생성하고 VM들을 재배치할 때 빈번히 나타나는 VM들의 순차적인 이주의 한 예이다. 실제 환경에서는 CPU, 메모리, 네트워크 대역폭 등 여러 종류의 자원이 고려될 수 있으나 이 예에서는 설명의 간편화를 위하여 PM의 자원 종류를 CPU와 메모리로 제한하였고 PM이 보유한 각 CPU와 메모리의 최대량은 1이다. 그림 1에서 직각 사각형은 PM을 나타내며 라운드 사각형은 VM을 나타낸다. 그리고 각 VM안에 두 값의 튜플은 두 종류의 자원에 대한 각 요구량을 나타낸다. 예를 들어 VM1의 자원 요구량 (0.5, 0.7)은 VM1이 0.5의 연산능력과 0.7의 메모리 량을 요구한다는 의미이다. 화살표는 VM이 이주해야하는 것을 의미한다.

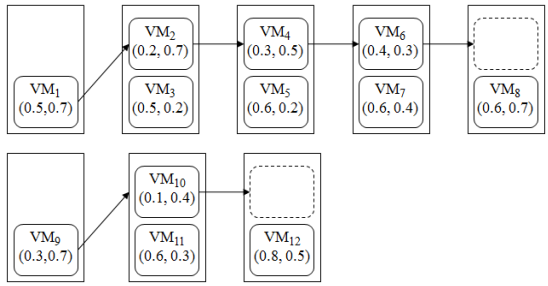

그림 1. VM 이주 계획
Fig. 1. VM migration plan

VM들이 그림 1과 같이 이주해야한다면 VM_1은 VM_2의 이주가 완료를 기다려야 한다. 그리고 VM_2는 VM_4의 이주완료를, VM_4는 VM_6의 이주완료를 기다려야 한다. 따라서 이 VM들은 VM_6, VM_4, VM_2, VM_1 순서로 이주할 수밖에 없다. 그리고 VM_9, VM_{10} 역시 VM_{10}, VM_9의 순서로 이주할 수밖에 없다. 앞으로 이들처럼 순차적으로 이주해야할 VM들을 이주열이라고 한다. 그림 1에서는 2개의 이주열 (VM_1, VM_2, VM_4, VM_6)과 (VM_9, VM_{10})이 존재한다.

VM 재배치시간은 가장 수행시간이 긴 이주열에 의해 결정된다. 이주열의 수행시간은 이주열에 속한 모든 VM들이 이주를 완료하는데 소요된 시간을 말한다. 모든 VM들의 이주시간이 같다면 VM 재배치 완료 시점은 가장 많은 VM이 속한 이주열에서 마지막으로 이주하는 VM의 이주완료시점과 같다. 그러나 실제 클라우드 클러스터에서는 VM의 자원 요구량에 따라 할당되는 메모리가 달라지며 그에 따라 이주시간 역시 달라진다. 따라서 이주열에 속한 VM개수만으로 이주열의 수행시간을 평가할 수 없다.

이주열의 수행시간을 평가하기 위해서는 VM의 이주시간을 알 필요가 있다. VM의 이주는 VM에 할당된 메모리를 목적지 PM으로 전송하는 작업을 통해 수행된다. 그러므로 VM의 이주시간 m은 식 1과 같다.

$$m = \frac{memory\ \ size + dirty page\ size}{transmission\ rate} \qquad (1)$$

이주열의 수행시간은 모든 VM들이 이주를 완료하는데 소요된 시간이다. 그러므로 이주열의 수행시간은 이주열에 속한 모든 VM들의 이주시간을 합한 것과 같다.

만약 수행시간이 가장 긴 이주열의 수행시간을 단축시킬 수 있다면 VM 재배치시간도 단축시킬 수 있다. 이주열의 수행시간을 단축시키기 위해서는 이주열을 구성하는 VM들이 달라져야 한다. 우리는 그러기 위해서 서로 다른 이주열에 속한 VM들의 목적지를 교환한다.

서로 다른 이주열에 속한 두 VM들의 목적지를 교환하기 위해서는 교환될 목적지 PM에 각 VM들이 점유할 수 있는 충분한 자원이 있어야 한다. 서로 다른 이주열에 속하는 어떤 VM_i와 어떤 VM_j가 있을 때 VM_i의 자원 요구량을 d_i, VM_j의 자원 요구량을 d_j라 하고 VM_i가 목적지 PM로 이주완료 했을때 그 PM의 자원 여유분을 r_i, VM_j가 목적지 PM으로 이주완료 했을 때 그 PM의 자원 여유분을 r_j라고 하자. 그렇다면 식 2를 만족하면 VM_i와 VM_j간의 목적지는 교환될 수 있다.

$$d_i \leq d_j + d_j \wedge d_j \leq d_i + r_i \quad (2)$$

식 2에서 튜플 간의 연산은 튜플을 구성하는 값들 각각에 대해 독립적으로 수행되며 튜플 간의 대소는 튜플을 구성하는 모든 값에 대해 크거나 작음으로 결정된다. 그림 1에서 예를 들면, 두 이주열 (VM_1, VM_2, VM_4, VM_6)과 (VM_9, VM_{10})에서 VM_2과 VM_9에 대해 살펴보면 d_2와 r_2는 각각 (0.2, 0.7), (0.2, 0.1)이고 d_9와 r_9는 각각 (0.3, 0.7), (0.1, 0)이다. 그리고 $d_2 + r_2$는 (0.4, 0.8), $d_9 + r_9$는 (0.4, 0.7)이므로 식 2를 만족한다. 따라서 VM_2와 VM_9간에 목적지 교환이 가능하다. 두 VM간의 목적지를 교환하면 이주열은 그림 2와 같이 (VM_1, VM_2, VM_{10})과 (VM_9, VM_4, VM_6)이 된다.

VM간의 목적지를 교환하면 이주열이 수정됨에 따라 이주열의 수행시간도 변하게 된다. VM_i의 이주시간을 m_i로 표기한다면 VM_2와 VM_9의 목적지 교환 전 두 이주열의 수행시간은 $m_1 + m_2 + m_4 + m_6$와 $m_9 + m_{10}$이다. 그러나 교환 후에는 $m_1 + m_2 + m_{10}$와 $m_9 + m_4 + m_6$이 된다. 따라서 VM간 목적지 교환 전과 후의 가장 긴 이주열 수행시간을 비교하여 교환 후가 짧다면 두 이주열을 동시에 수행했을 때 마지막으로 이주하는 VM의 이주완료 시점을 앞당길 수 있다.

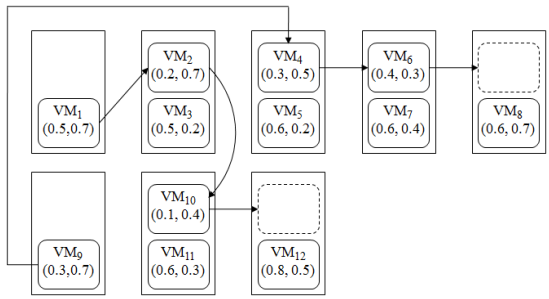

그림 2. VM_2와 VM_9의 목적지 교환 후 VM 이주계획
Fig. 2. VM migration plan after exchanging destinations between VM_2 and VM_9

어떤 VM_i와 어떤 VM_j를 각각 포함하는 두 이주열이 있을 때 두 이주열 중 수행시간이 긴 이주열의 수행시간을 $l_{\max}$로 표기하자. 그리고 VM_i와 VM_j의 목적지를 교환한 후의 두 이주열의 수행시간을 각각 l_i와 l_j로 표기하자. VM_i와 VM_j의 목적지를 교환하여 $l_{\max}$가 단축되려면 l_i와 l_j는 $l_{\max}$보다 작아야 한다. 따라서 VM_i와 VM_j의 목적지를 교환하여 두 이주열 중 최장 수행시간이 단축되는 조건은 식 3과 같다.

$$l_i < l_{\max} \quad \wedge \quad l_j < l_{\max} \quad (3)$$

제안하는 기법은 식 2와 식 3을 이용하여 라운드를 반복하며 VM간의 목적지를 교환한다. 각 라운드에는 다음 과정을 수행한다. 우선 이주열들 중 가장 수행시간이 긴 이주열을 찾는다. 그리고 나머지 이주열을 대상으로 식 2와 식 3을 사용하여 VM들간의 목적지 교환을 통해 최장 이주열의 수행시간을 단축시킬 수 있는 이주열을 찾는다. 찾은 이주열이 하나라면 그 이주열과 VM의 목적지를 교환하고 그 이상이라면 그 중 수행시간을 가장 짧게 단축시킬 수 있는 이주열과 VM의 목적지를 교환하고 라운드를 종료한다. 반면에, 찾은 이주열이 없다면 반복을 종료하고 수정된 VM들의 목적지를 반영하여 VM 맵을 갱신한다.

IV. 성능평가

본 장에서는 시뮬레이션을 통해 제안하는 기법의 성능을 평가한다. 시뮬레이션은 FFD를 통해 생성한 VM 맵만으로 VM들을 재배치하는 경우와 FFD를 통해 생성한 VM 맵을

제안한 기법을 통해 재 매핑 한 후 VM들을 재배치하는 경우에 대해 각각 수행되며 두 경우 대한 이주열의 수행시간 변화와 그에 따른 VM 재배치 소요시간을 비교 분석한다.

시뮬레이터로는 C++ 기반의 클라우드 컴퓨팅 시뮬레이터인 SimCloudIS[9]를 사용하였다. 시뮬레이션의 환경 변수는 표 1과 같다. PM과 VM의 자원은 CPU와 메모리로 두 종류이다. 그리고 VM의 자원 요구량을 다양하게 하기 위해 가상 CPU 코어 개수는 1~4개 범위로 랜덤변수에 의해 결정되였으며 메모리 역시 512~4096MBytes 범위에 512MBytes의 단위로 랜덤변수에 의해 결정되었다. 랜덤 변수는 균등분포를 따른다. 그리고 VM 수가 이주열의 수행시간과 VM 재배치시간에 미치는 영향을 살펴보기 위하여 VM 수를 1000개, 2000개로 하여 각각 시뮬레이션을 진행하였다.

표 1. 시뮬레이션 환경변수
Table 1. Simulation variables

변수	값
클러스터의 network bandwidth	1Gbits
클러스터의 PM수	2000개
PM의 CPU 코어 개수	8개
PM의 메모리 량	8Gbytes
클러스터에서 실행되는 VM수	1000개, 2000개
VM의 가상 CPU 코어 개수	1~4개
VM의 메모리	512~4096MBytes
VM migration 방식	Pre-copy live migration

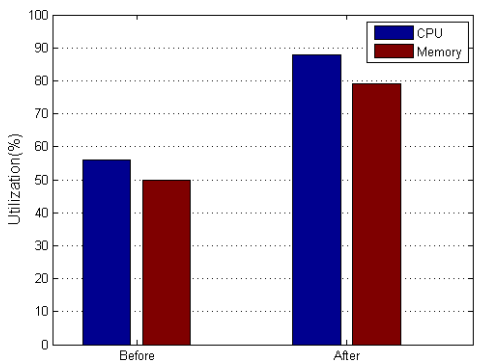

그림 3. PM들의 평균 자원 활용률
Fig. 3. Average utilization of PMs

그림 3은 FFD를 통해 VM 맵을 구하고 재배치를 했을 때 PM들의 평균 자원 활용률의 변화를 보여준다. PM의 자원 활용률은 VM들에 의해 점유되는 자원 량과 총 자원량의 비율을 말한다. 그림 3에서 볼 수 있듯이 클러스터에 실행되는 VM개수가 1000개일 때와 2000개 일 때 FFD를 통해 생성된 두 VM맵은 PM들의 CPU와 메모리 활용률을 약 56, 50%에서 88, 79%로 향상시킨다.

제안하는 기법은 VM 맵에서 VM들의 목적지를 교환하여 VM들을 재매핑하기 때문에 일부 PM들의 자원 활용률을 변화시킬 수 있으나 PM들의 전체 자원 활용률에는 영향을 주지 않는다.

그림 4는 클러스터에서 실행되는 VM개수에 따라 이주열의 최장 수행시간을 보여준다. FFD의 경우 이주열의 최장 수행시간이 VM이 1000개 때 777.9초, VM이 2000개일 때 893.3초 이다. 그에 반해 제안하는 기법의 경우 이주열의 최장 수행시간이 VM이 1000개일 때 359.3초, VM이 2000개일 때 390.9초이다.

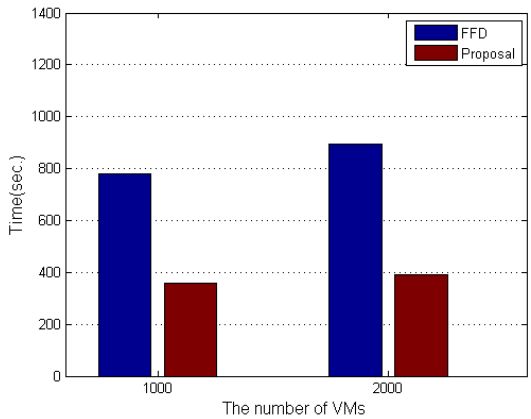

그림 4.이주열의 최장 수행시간
Fig. 4. The longest execution time of migration queue

그림 4에서 VM 개수는 두 배로 증가했음에도 불구하고 이주열의 최장 수행시간 증가폭은 작을 것을 알 수 있다. 이는 VM 개수가 증가하면 이주열의 개수는 늘어나지만 한 이주열에 속하는 평균 VM 개수의 증가폭은 크지 않기 때문이다. 그러나 비록 소폭일지라도 이주열의 수행시간이 증가하면 제안한 기법의 이주열 수행시간의 단축률 역시 증가한다. 이주열의 수행시간이 증가하면 VM들의 목적지를 교환하여 수행시간을 단축시킬 수 있는 기회 역시 많아지기 때문이다. 이러한 이유로 제안하는 기법은 VM이 1000개일 때보다 2000개일 때 이주열의 최장 수행시간을 더 많이 단축한다. 제안하는 기법은 FFD와 비교하여 이주열의 최장 수행시간을 VM이 1000개, 2000일 때 각각 53.9, 56.2% 단축하였다.

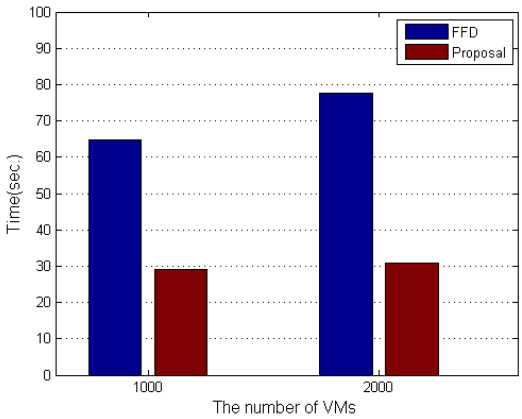

그림 5. 이주열의 수행시간 표준편차
Fig. 5. The standard deviation of migration queue execution time

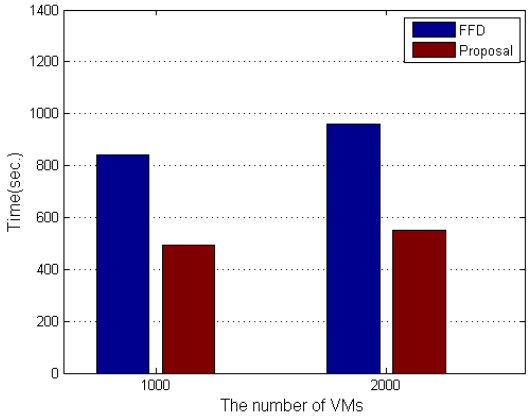

그림 6. VM 재배치 시간
Fig. VM relocation time

그림 5는 VM개수에 따른 이주열의 수행시간 표준편차를 보여준다. FFD는 VM 개수가 1000개, 2000개일 때 각각 64.67, 77.56초의 수행시간 표준편차를 보여준다. 반면에 제안하는 기법은 VM 개수가 1000개, 2000개일 때 각각 29.1 30.8초의 수행시간 표준편차를 보여준다. FFD의 경우 VM 개수가 많아짐에 따라 표준편차가 증가한다. 그러나 제안한 기법은 VM들 간의 목적지를 교환하여 이주열의 수행시간을 평준화하기 때문에 VM 개수가 많아져도 대체적으로 일정한 표준편차를 보여준다.

그림 6은 VM 개수에 따른 VM 재배치시간을 보여준다. 우리는 3장에서 VM 배재치 시간은 이주열의 최장 수행시간에 의해 결정된다고 언급하였다. 그러나 그림 4와 6을 비교해 보면 VM 재배치시간이 이주열의 최장 수행시간보다 큰 것을 알 수 있다. 이는 네트워트 대역폭의 공유와 집중적인 I/O로 인한 오버헤드 때문이다. VM 재배치 동안 많은 수의 VM들이 네트워크 대역폭을 공유하여 일제히 이주하며 이때 PM들에서 집중적인 I/O로 인한 오버헤드가 발생한다. 따라서 VM들의 이주시간은 지연된다. 이러한 이유로 비록 이주열의 최장 수행시간과 VM 재배치 시간 간에 차이는 있지만 최장 수행시간이 단축됨에 따라 VM 재배치 시간이 단축되는 것을 확인할 수 있다. 제안한 기법은 VM개수가 1000개 일 때 와 2000개 일 때 각각 VM 재배치 시간이 493.3, 549.5초로 FFD의 839.7, 959.1초에 비해 각각 41.3, 42.7% 단축시켰음을 보여준다.

IV. 결 론

클라우드 클러스터에 VM들을 재배치할 때는 PM의 한정된 자원량 때문에 VM들이 순차적으로 이주해야하는 상황이 발생하며 이는 VM 재배치 시간을 증가시킨다. 본 논문에서는 VM 맵에서 VM들의 목적지를 교환함으로써 VM 재배치 시간을 단축하는 VM 재매핑 기법을 제안한다. 제안하는 기법은 입력으로 주어진 VM 맵으로부터 순차적으로 이주해야하는 VM들을 찾아 다음 두 가지 조건을 만족하는 VM들의 목적지를 교환한다. 첫째. VM들의 목적지 PM들을 교환했을 때 그 PM들이 VM들의 자원 요구량을 수용할 수 있어야 한다. 둘째, 목적지 교환을 통해 VM 재배치시간을 단축할 수 있어야 한다. 목적지 교환은 VM 재배치 시간이 더 이상 단축될 수 없을 때까지 반복된다.

시뮬레이션에서는 실행되는 VM개수가 각 1000개, 2000개인 두 클러스터에서 생성된 각 VM 맵을 기반으로 하여 제안한 기법의 성능을 평가하였다. 시뮬레이션 결과는 VM 개수가 각 1000개, 2000개인 클러스터에서 제안한 기법으로 VM 맵을 재매핑 했을 때 재매핑하지 않은 경우에 비해 VM 재배치시간을 각각 41.3, 42.7% 단축시켰음을 보여준다.

참고문헌

[1] Kim, Changhyeon, Wonjoo Lee, and Changho Jeon. "A Resource Reduction Scheme with Low Migration Frequency for Virtual Machines on a Cloud Cluster," KSII Transactions on Internet

and Information Systems (TIIS), vol. 7, no. 6, 1398-1417, 2013

[2] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: State-of-the-art and research challenges", Internet Service and Applications, vol. 1, no. 1, pp 7-18, 2010

[3] M. Wang, X. Meng, and L. Zhang, "Consolidating Virtual Machines with Dynamic Bandwidth Demand in Data Centers," in proc. of IEEE INFOCOM, pp 71-75, 2011.

[4] We. Song, Z. Xiao, Q. Chen and H. Luo, "Adaptive Resource Provisioning for the Cloud Using Online Bin Packing," IEEE Transactions on Computers, vol. 99. 2013.

[5] D. Wilcox, A. McNabb, and K. Seppi, "Solving virtual machine packing with a reordering grouping genetic algorithm," in proc. of IEEE Congress on Evolutionary Computation, pp. 362-369, 2011.

[6] T. Ferreto, C. A. F. De Rose, and H. Heiss, "Maximum migration time guarantees in dynamic server consolidation for virtualized data centers," in proc. of the 17th International Conference on Parallel processing - Volume Part I, pp. 443-454, 2011

[7] Y. Ho, P. Liu and J. Wu, "Server consolidation algorithms with bounded migration cost and performance guarantees in cloud computing," in proc. of 4th IEEE International Conference on Utility and Cloud Computing, pp. 154-161, Dec. 2011.

[8] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," Future Generation Computer Systems, vol. 28, no. 5, pp. 755-768, 2012.

[9] SimCloudIS(Simulator for Cloud computing Infrastructure and service) project, https://code.google.com/p/simcloudis/

저 자 소 개

김 창 현

2008: 경일대학교
컴퓨터공학과 공학사
2010: 한양대학교
컴퓨터공학과 공학석사
현 재: 한양대학교
컴퓨터공학과 박사과정
관심분야: 센서 네트워크, 성능 분석,
그리드 컴퓨팅,
클라우드 컴퓨팅
Email : ctcquatre@hanyang.ac.kr

김 준 상

2003: 한양대학교
전자컴퓨터공학부 공학사
2005: 한양대학교
컴퓨터공학과 공학석사
2008-2012: 해군사관학교
컴퓨터학과 전임강사
현 재: 한양대학교
컴퓨터공학과 박사과정
관심분야: 성능 분석, 그리드 컴퓨팅,
클라우드 컴퓨팅
Email : kjspbe@hanyang.ac.kr

전 창 호

1977: 한양대학교 전자공학과 학사
1982: Cornell University
컴퓨터공학과 석사
1986: Cornell University
컴퓨터공학과 박사
현 재: 한양대학교 컴퓨터공학과 교수
관심분야: 병렬처리시스템, 성능 분석,
그리드 컴퓨팅,
클라우드 컴퓨팅
Email : chj5193@hanyang.ac.kr