붙임1

연구논문/작품 제안서

2018 년도 제 1학기

논문/작품	○논문(√) ○작품( ) ※ 해당란에 체크
제목	메모리 공유율 기반 가상머신 이주 기법
GitHub URL	https://github.com/persona0220/graduation-thesis
팀원명단	박 수 진 (박수진 (학번: 2014314920)

2018 년 3월 21일

지도교수 : 엄 영 익 서명

1. 과제의 필요성

Abstract

최근 모바일 단말 보급의 증가로 IoT, 모바일 엣지 컴퓨팅(MEC, Mobile Edge Computing)와 같은 기술의 개발과 함께 모바일 단말에서 가상화 기술을 적용시키기 위한 연구가 진행되고 있다. 가상화 환경에서 여러 대의 가상 머신들은 같은 자원을 공유하며 동작하기 때문에 시스템의 한정된 물리적 자원을 효율적으로 공유하여 사용할 수 있다. 그러나 다수의 가상 머신들이 동시에 구동되기 위해서는 각 가상 머신에서 사용하는 페이지들이 모두 호스트의 물리 메모리를 공유하기 때문에, 호스트의 메모리는 가상머신의 수와 성능을 결정하는 중요한 요인이 된다. 만약 가상머신간의 메모리 경쟁이 심해질 경우, 호스트의 메모리에서 지속적으로 스왑핑(swapping)이 발생하고, 이는 가상머신에 심각한 성능 저하를 초래한다. 따라서 호스트는 메모리를 확보하기 위해 특정 가상머신을 선택해 종료하거나 다른 물리 호스트로 이주시키는 방법을 취할 수 있다. 호스트에서 필요한 만큼의 메모리를 확보하기 위해서는 이주시킬 대상이 되는 가상머신을 신중하게 선정하는 기법이 필요하다. 따라서 본 연구에서는 각 가상머신의 메모리 공유율을 실시간으로 측정하는 방법을 고안하고, 이 정보를 바탕으로 각 가상머신을 종료했을 때 호스트에서 확보할 수 있는 메모리를 예측한다. 또한, 예측한 메모리를 통해 이주 대상이 될 가상머신을 선정하는 기법을 연구한다.

서론

최근 다양한 분야에서 가상화 기술이 사용됨에 따라, 가상화 환경에서의 오버헤드를 줄이고, 성능을 향상시키기 위한 연구가 진행되고 있다. 가상화는 컴퓨터 하드웨어 플랫폼이나, 스토리지 장치, 네트워크와 같은 자원을 가상으로 구현하는 기술로, 하나의 물리적 환경 위에서 다수의 컴퓨터 시스템을 사용할 수 있도록 해준다. 가상화 기술을 통해 하나의 컴퓨터에서 동시에 여러 개의 운영체제를 가동시킬 수 있으며, 서버 통합, 가상 실험 환경 구축, 데이터 센터 등과 같이 수많은 곳에 활용된다. 최근 많은 서비스들은 데이터 센터에 의해 제공되고 있으며, 데이터 센터는 수천 대의 컴퓨터에서 여러 개의 응용 프로그램들을 동시에 구동한다. 하나의 서버에서 다수의 서비스를 제공하기 위해 데이터 센터에서도 가상화 기술이 사용되고 있다.

가상화 환경에서 여러 대의 가상 머신들은 같은 자원을 공유하며 동작하기 때문에 시스템의 한정된 물리적 자원을 효율적으로 공유하여 사용할 수 있다. 하지만, 한정된 자원을 공유해야 하기 때문에 가상화 기술에서 메모리 효율성이 매우 중요하게 작용하며, 메모리는 하나의 물리적 서버에서 제공할 수 있는 가상머신의 수를 결정하는 가장 중요한 제한요인이 된다. 따라서 가상머신의 수가 증가하면, 자원을 분배하기 위한 오버헤드가 증가하고 가상화 시스템의 전체 성능이 저하될 수

있다. 특히 모바일 단말은 서버나 데스크탑에 비해 상대적으로 한정된 자원을 가지므로 가상화 기술을 적용하는데 한계점이 존재한다.

다수의 가상머신을 하나의 물리적 시스템 위에서 동시에 실행시켰을 때, 가상머신들은 디스크, 네트워크, 메모리 등과 같은 호스트의 물리 자원들을 함께 사용한다. 이 때 각 가상머신들이 가지고 있는 가상메모리는 호스트의 물리메모리에 함께 올라가기 때문에 호스트의 메모리는 동시에 구동할 수 있는 가상머신의 개수를 결정하는 중요한 요소로 작용한다. 따라서 호스트는 다양한 상황에서 필요에 따라 선택적으로 가상머신을 중지하거나 이주시킴으로서 필요한 메모리를 확보한다. 특정 가상머신에서 동작하는 응용프로그램이 과도한 메모리를 사용할 경우 호스트의 한정된 메모리를 경쟁적으로 사용하여 전체 시스템의 성능을 악화시킬 수 있다. 그 외에도 전력을 아끼기 위해 자원을 확보하거나, 높은 성능을 내야 하는 특정 응용 프로그램을 위해 자원을 확보해야 할 때 선택적으로 가상머신을 종료하거나 상대적으로 메모리가 충분한 다른 호스트로 이주시켜 문제를 해결할 수 있다.

가상머신 실시간 이주(live VM migration)란, 실행중인 가상머신을 멈추지 않고 물리적으로 다른 곳에 위치한 호스트로 가상머신을 이주시키는 기술이다. 가상머신 이주 기술은 특정 호스트에 가해지는 부하를 분산하고, 열 분산으로 인한 전력을 감소시킬 수 있으며, 서버의 유지보수를 용이하게 할 수 있다. 가상머신을 멈추지 않고 이주시키기 위해서는 가상머신의 CPU상태, 메모리, I/O 상태와 같은 가상머신의 모든 정보를 소스 호스트(source host)에서 타겟 호스트(target host)로 전송하는 것이 필요하다. 하지만 가상머신이 실시간으로 실행중이기 때문에 메모리를 전송하는 동안 메모리의 내용이 변경될 수 있으며, 변경된 메모리 역시 전송해야 한다. 또한, 소스 호스트에서 실행되던 가상머신이 타겟 호스트에서 재시작 하는데 가동 중단시간(downtime)이 발생한다.

가상머신을 이주시키기 위해서는 이주 대상이 되는 가상머신을 선정해야 한다. 호스트의 메모리 확보를 최우선으로 고려하였을 때, 어떤 가상머신을 내쫓는 것이 가장 효율적으로 메모리를 확보할 수 있는지 실시간으로 결정할 수 있어야 한다. 호스트에서 구동되는 다수의 가상머신은 각각 가상의 메모리를 가지고 구동되며, 가상머신을 완전히 이주시켰을 때 호스트는 가상머신이 사용하던 메모리를 확보한다.

가상화 환경에서 하나의 호스트에서 다수의 가상머신을 구동했을 때, 가상 머신간에 중복된 페이지를 사용하게 될 경우 호스트의 메모리에는 중복된 페이지가 존재하게 된다. 이러한 중복 페이지들은 가상 머신들이 같은 응용프로그램을 실행하거나, 같은 데이터를 다룰 때 발생한다. 호스트의 메모리는 가상화 환경의 성능을 결정하는 중요한 요인이므로, 메모리를 효율적으로 관리하기 위한 다양한 연구가 진행되어 왔다. 그 중 KSM(Kernel Samepage Merging) 모듈은 리눅스 커널 2.6.32 버전부터 지원되는 메모리 중복제거 모듈로, 주기적으로 시스템에 할당된 페이지를 스캔하여 페이지의 내용을 기반으로 중복 페이지를 찾아내어(contents-based page

sharing) 하나의 KSM 페이지로 병합한다. KSM은 서로 다른 프로세스의 익명 페이지를 대상으로 하며, 페이지의 내용을 인덱스로 사용하는 두 개의 레드블랙트리를 이용하여 공유페이지와 단일페이지를 관리한다. KSM을 사용할 경우 동일한 중복 페이지가 해제되고 하나의 페이지로 병합되기 때문에 메모리를 확보하고, 더 많은 가상머신이나 응용 프로그램을 실행할 수 있게 된다.

메모리를 확보하기 위해 특정 가상머신을 호스트에서 쫓아내기 위해서는 각 가상머신을 종료하였을 때 확보할 수 있는 메모리의 크기를 알아야 한다. 이 때, 가상머신의 특성에 따라 다른 가상머신과 공유하고 있는 페이지의 수가 현저히 달라진다. 만약 페이지 공유율이 높은 가상머신을 해제할 경우, 해당 가상머신이 사용하고 있던 페이지 중 대다수는 다른 가상머신에서도 사용되고 있기 때문에 호스트 메모리에서 해제될 수 없다. 이는 결과적으로 예상한 것보다 더 적은 페이지가 호스트 메모리에서 해제되어 가상머신을 쫓아내었을 때의 결과를 확실할 수 없게 된다.

따라서, 본 연구에서는 가상화 환경에서 각 가상머신의 메모리 공유율을 측정하고, 이에 기반하여 호스트의 메모리를 확보하기 위해 이주 대상이 될 가상머신을 선정하는 기법을 연구한다.

2. 선행연구 및 기술현황

2.1 KSM (Kernel Samepage Merging) 모듈

KSM 모듈은 리눅스 커널 2.6.32 버전부터 지원되는 메모리 중복제거 모듈로, 주기적으로 시스템에 할당된 페이지를 스캔하여 중복 페이지를 찾아내고 하나의 페이지로 병합한다. KSM은 각 페이지의 내용을 기준으로(contents-based page sharing) 중복 페이지를 탐색하며, 이 과정을 통해 중복된 페이지들이 메모리에서 해제되어 메모리를 절약할 수 있다. KSM은 각 프로세스에서 병합대상으로 지정된 영역에 대해 익명페이지(anonymous page)를 대상으로 페이지를 관리하며, KVM (Kernel-based Virtual Machine)은 호스트의 관점에서 일반 프로세스와 동일하게 여겨진다. 이 때 각 프로세스에서 병합대상으로 지정한 페이지들은 순차적으로 탐색되고, 호스트의 지정된 전체 메모리 페이지는 두 개의 레드블랙트리를 이용하여 공유 페이지는 안정 트리(stable tree)로, 단일 페이지는 불안정 트리(unstable tree)로 나누어 관리된다. 트리의 각 노드는 페이지의 내용을 기반으로 구성되며 이는 레드 블랙 트리에서 키의 역할을 한다. 불안정 트리의 페이지는 write-protect 페이지가 아니기 때문에 실시간으로 내용이 변경될 수 있기 때문에 매 스캔 라운드가 지날 때마다 불안정트리는 재구성이 필요하다.

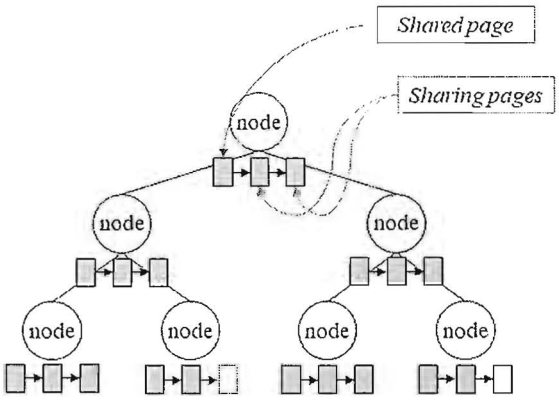

그림 1. KSM 안정트리(stable tree) 의 구조

페이지가 스캔될 때, KSM은 해당 페이지가 안정 트리 내에 있는지 가장 먼저 확인한다. 만약 동일한 내용의 페이지가 안정 트리에서 발견되었다면, 해당 노드와 병합하여 연결 리스트 구조로 해당 페이지를 연결한다. 만약 발견되지 못할 경우, 불안정 트리에서 페이지를 다시 검색하고, 불안정 트리에서 동일한 내용의 페이지가 발견될 경우 두 페이지는 병합되어 안정 트리로 옮겨오게 된다. 마지막으로, 불안정 트리에서도 동일한 페이지가 발견되지 못할 경우 해당 페이지는 불안정 트리에 새로운 노드가 되어 삽입된다. 위와 같은 방법은 레드 블랙 트리의 특성을 활용하여 노드의 인덱스를 확인하지 않고도 트리의 균형을 유지할 수 있으며, 최악의 경우 O(logN)의 시간을 보장한다.

[3]에서는 KSM을 사용해 호스트에서 중복 페이지를 삭제하고 메모리를 좀 더 효율적으로 사용하는 것을 여러 환경에서의 실험을 통해 보여준다. 두 개의 가상머신에서 동일한 OS와 동일한 응용 프로그램을 실행했을 때, 많은 양의 중복페이지가 발생하는 것을 볼 수 있다. 위 연구에서는, 4GB 메모리의 호스트에서 유사한 페이지를 많이 발생시키는 동일한 2GB의 프로세스를 구동하여 KSM 모듈의 메모리 공유 효율성을 측정하였다. 이 때 KSM을 사용하지 않았을 때에는 250MB의 메모리만이 공유되었지만, KSM을 실행한 후에는 750MB의 페이지가 공유되어 기존 2개의 프로세스만을 구동할 수 있던 환경에서, 3개의 프로세스까지 동시에 구동할 수 있는 것을 보여주었다.

따라서, KSM은 동일한 페이지를 많이 발생시키는 다수의 가상머신이 구동될 때 상당한 양의 메모리를 공유할 수 있으며, 따라서 이주 대상이 되는 가상머신을 선정할 때 가상머신 간의 메모리 공유율이 고려되어야 한다.

2.2 가상머신 내부와 가상머신 간의 메모리 공유

KVM(Kernel-based Virtual Machine)은 내부적으로 KSM API를 사용하여, KSM 모듈을 호출하여 메모리를 관리한다. 메모리의 각 페이지들은 그 내용을 기반으로 동일한 페이지인지의 여부를 검사하여, 중복된 페이지가 발견되었을 경우 새로운

페이지를 생성해 중복된 페이지들을 모두 맵핑하고, 기존의 중복된 페이지들은 모두 해제함으로서 전체 메모리 사용량을 줄일 수 있다.

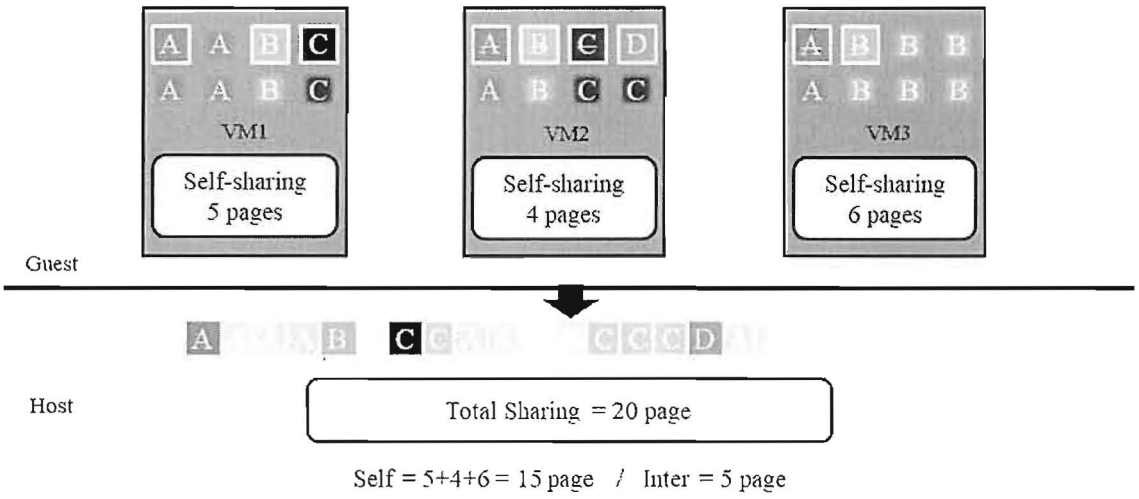

그림 2. 3개의 가상머신을 동시에 구동할 때 가상머신 내부 공유율과 가상 머신 간 공유율

위 [그림2]는 3개의 가상머신을 한 개의 호스트에서 동시에 구동하였을 때 어떻게 각 페이지들이 공유되는지 간단히 보여주고 있다. 먼저, 각 가상머신의 페이지들은 madvise 함수를 통해 모두 공유 가능한 영역으로 지정되고, 첫 번째 순차 탐색에서 내용이 동일한 페이지들이 병합된다. 이 때, 결과적으로 총 24개의 페이지 중에서 중복된 페이지 20개가 해제되어, 결과적으로 4개의 페이지만이 호스트의 물리 메모리에 남게 된다.

[5]에서는 KSM을 통해 메모리 공유가 일어날 때, 페이지 공유의 특성에 따라 가상 머신 내부 공유(self-sharing)와 가상 머신 간 공유(inter-sharing)로 메모리 공유를 구별하였다. 이러한 분류는 특정한 가상머신을 축출했을 때 얻을 수 있는 페이지를 계산하기 위해 중요한 정보가 된다. 만일 위 [그림2]에서 첫 번째 가상머신(VM1)을 이주시켰다고 해도, 결과적으로 다른 가상머신들에서 A, B, C 세 개의 페이지들이 모두 사용 중이기 때문에 호스트의 물리 메모리는 전혀 확보되지 못하기 때문이다.

2.3 가상머신 실시간 이주(Live VM Migration)

가상머신 실시간 이주는 실행중인 가상머신을 멈추지 않고 물리적으로 다른 곳에 위치한 호스트로 가상머신을 이주시키는 기술이다. 이 기술은 데이터 센터와 같은 가상화 환경에서 빠르게 hot-spots를 제거하거나, 전력을 아끼기 위해 자원을 확보해야 할 때, 데이터 센터의 단편화(fragmentation)를 방지하기 위한 경우 등 다양한 목적으로 사용된다. 가상머신을 종료하지 않고 다른 호스트로 이동시키기 위해서는 가상머신의 상태를 모두 전송해야 하며, 모든 메모리 상태와 CPU, I/O 상태 등

을 포함한 데이터의 양은 수 GB에 달하여 시스템 성능에 오버헤드가 발생된다.

더욱이, 가상머신을 호스트 간에 전송하는 동안 상당한 네트워크 트레픽이 발생하며, 이는 곧 가상머신 내에서 실행되고 있는 네트워크 집중적인 응용 프로그램의 성능에까지 영향을 미친다. 만약 해당 가상머신이 큰 그룹 내에서 다른 가상머신들과 협력하여 작업하는 상황이라면, 이러한 하나의 가상머신의 성능 저하는 전체 그룹의 성능까지 악화시킬 수 있다.

3. 작품/논문 전체 진행계획 및 구성

1년간의 연구논문 일정과 연구의 계획 및 일정을 아래 [표 1] 에 표시해 두었다. 먼저, 리눅스 커널 기반 메모리 공유 모듈 KSM을 분석하고, 이를 바탕으로 가상머신의 내부 메모리 공유율과 가상머신 간 메모리 공유율을 측정 및 분석한다. 그 후, 다수의 가상머신이 동시에 구동되는 상황에서 각 가상머신이 종료되었을 때 확보될 수 있는 메모리의 크기(페이지의 수)를 실시간으로 측정한 정보를 바탕으로 예측하고, 이를 통해 이주 대상이 될 가상 머신을 선택하는 기법을 제시한다.

또한, 연간 연구논문 일정에 맞추어 3월부터 11월까지 Github 에 연구 내용을 상세히 기록하고, 3월에는 제안서와 서약서, 9월에 중간보고서, 11월에 최종보고서를 제출하고 논문 발표회에 참석한다.

표 1. 연간 연구논문 일정 및 연구 계획

내용 \ 월	3	4	5	6	7	8	9	10	11
연구노트 Github 작성	→	→	→	→	→	→	→	→	→
리눅스 커널 기반 메모리 공유 모듈 KSM(Kernel Same-page Merging) 분석	→	→							
가상머신 내부 메모리 공유율과 가상머신 간 공유율 측정 및 분석			→	→					
가상머신을 종료했을 때 호스트에서 확보할 수 있는 페이지 수 실시간 측정					→	→			
이주 대상이 되는 Victim-VM 선택 기법 제시							→	→	
연구논문 일정	제안서 서약서						중간보고서		최종보고서 & 발표회

4. 기대효과 및 개선방향

하나의 호스트에서 구동되는 여러 대의 가상머신에 대해 실시간으로 메모리 사용량과 공유율을 측정하고, 이 정보를 기반으로 각 가상머신을 종료했을 때 확보할 수 있을 것으로 예측되는 메모리의 크기, 혹은 페이지의 수를 실시간으로 출력한다. 가상머신 이주(live VM migration)는 서비스 다운타임이나, 네트워크 경쟁으로 인한 가상머신 내의 응용프로그램의 성능 감소 등 상당한 비용을 필요로 하기 때

문에 꼭 필요한 경우 신중하게 수행해야 한다. 따라서 본 연구를 통해 호스트의 메모리를 확보하기 위해 어떤 가상머신을 이주시켜야 하는지 미리 예측하여 최소한의 비용으로 필요한 메모리를 확보할 수 있을 것으로 기대한다.

5. 참고문헌

[1] KVM (Kernel-based Virtual Machine) [online]. Available: http://www.linux-kvm.org/page/Main_Page.

[2] F. Bellard, "Qemu, A Fast and Portable Dynamic Translator," *Proc. of the USENIX conference on Annual Technical Conference,* pp. 46, 2005.

[3] A. Arcangeli, I. Eidus, and C. Wright, "Increasing Memory Density by Using KSM," *Proc. of the Linux Symposium,* pp. 19-28, 2009.

[4] A. Kivity, Y. Kamay, D. Laor, U. Lublin, and A. Liguori, "KVM: The Linux Virtual Machine Monitor," *Proc. of the Linux Symposium,* pp. 225-230, 2007.

[5] S. Barker, T. Wood, P. Shenoy, and R. Sitaraman, "An Empirical Study of Memory Sharing in Virtual Machines," *Proc. of the USENIX conference on Annual Technical Conference,* pp.273-284, 2012.

[6] K. Miller, F. Franz, M. Rittinghaus, and M. Hillenbrand, and F. Bellosa, "XLH: More Effective Memory Deduplication Scanners Through Cross-layer Hints," *Proc. of the USENIX conference on Annual Technical Conference,* pp. 279-290, 2013.

[7] G. Milos, D.G. Murray, S. Hand, and M. A. Fetterman, "Satori: Enlightened Page Sharing," *Proc. of the conference on USENIX Annual Technical Conference,* 2009.

[8] D. Gupta, S. Lee, M. Vrable, S. Savage, A. C. Snoeren, G. Varghese, G. M. Voelker, and A. Vahdat, "Difference Engine: Harnessing Memory Redundancy in Virtual Machines," *Proc. of the USENIX Syumposium on Operating Systems Design and Implementation,* pp.309-322, 2008.

[9] K. Miller, F. Franz, T. Groeninger, M. Rittinghaus, M. Hillenbrand, and F. Bellosa, "KSM++: Using I/O-based Hints to Make Memory-Deduplication Scanners More Efficient," *Proc. of the ASPLOS Workshop on Runtime Environments, Systems, Layering and Virtualized Environments,* 2012.

[10] P. SHarma and P. Kulkarni. "Singleton: System-Wide Page Deduplication in Virtual Environments," *Proc. of the 21st international symposium on High-Performance Parallel and Distributed Computing,* pp.15-26, 2012.

[11] L. Xia and P. A. Dinda, "A Case for Tracking and Exploiting Inter-node and

Intra-node Memory Content Sharing in Virtualized Large-scale Parallel Systems," *Proc. of the international workshop on Virtualization Technologies in Distributed Computing Date,* pp.11-18, 2012.

[12] T. Wood, G. Tarasuk-Levin, P. J. Shenoy, P. Desnoyers, E.Cecchet, and M. D. Corner, "Memory Buddies: Exploiting Page Sharing for Smart Colocation in Virtualized Data Centers," *Proc. of the international conference on Virtual Execution Environments,* pp.31-40, 2009.