서버 가상화 환경에서의 VM 로드 밸런싱 기법

오원석, 김인혁, 엄영익
성균관대학교 정보통신공학부
e-mail:diki@skku.edu, {kkojiband,yieom}@ece.skku.ac.kr

Dynamic Load Balancing of Virtual Machines in Server Virtualization Environments

Wonsuk Oh, Inhyuk Kim, Young-Ik Eom
Dept of Information and Communication Eng., Sunkyunkwan University

요 약

기업의 데이터 센터의 규모가 커지면서 서버 가상화가 중요한 이슈가 되고 있다. 하지만, 서버의 효율성을 높이기 위해 제안된 가상화 환경의 특징은 가상 머신 내부에서 실제 시스템 사용률을 측정하는데 어려움을 초래했다. 이는 외부에서 가상 머신이 동작중인 물리 머신의 부하를 잘못 인식하게 만들며, 기존 로드 밸런싱 기법 적용의 효율성을 저하시킨다. 이러한 문제를 해결하기 위해 본 논문에서는 가상 머신의 I/O 요청의 총량에 기반을 둔 로드 밸런싱 기법을 제안한다. 제안하는 기법에서 로드 밸런싱 서버는 각각의 물리 서버의 I/O 처리량의 한계값을 알고 있으며, 한계값보다 낮은 I/O 요청이 있는 물리 서버에 속한 가상 머신에게만 작업을 분배한다.

1. 서론

오늘날 기업의 업무에 정보 기술의 적용이 증가됨에 따라 기업의 데이터 센터의 규모 역시 증가하고 있다. 기업용 어플리케이션은 silo-oriented 기법, 즉 각각의 어플리케이션이 서버, 스토리지, 네트워크 등의 인프라스트럭처를 독립적으로 사용하도록 설계됨으로서 관리의 편의성을 증대하며 보안을 높이는 기법으로 설계되는데, 이 기법의 한계로 인하여 기업의 데이터 센터는 높은 유지 및 관리 비용에 비하여 낮은 가동률을 보인다.[3] 이러한 현상의 해결책으로 최근 서버 가상화를 통하여 데이터 센터의 효율성을 높이는 기법의 적용이 증가하고 있다. 그러나 기존의 물리 머신과는 달리, 추상화 레이어의 영향으로 가상 머신은 자신이 동작중인 실제 물리 머신의 사용률을 알 수 없으며, 이러한 차이점 때문에, 기존의 로드 밸런싱 기법으로는 가상 머신을 대상으로 적절한 부하의 분배가 이루어지지 않는 문제가 있다. 이에 본 논문에서는 기존의 로드 밸런싱 기법을 가상화에 적합하게 변형한 기법을 제안한다.

본 논문의 구성은 다음과 같다. 2장에서는 가상화와 로드 밸런싱 기법에 대해 설명하고 3장에서는 기존의 물리 머신 환경과의 차이로 발생하는 문제점과 이를 해결하기 위한 가상화 환경에서의 동적 로드 밸런싱 기법을 제안한다. 마지막 4장에서는 결론 및 향후 연구계획을 설명한다.

2. 관련 연구

본 절에서는 가상화의 개념, 대표적인 가상화 솔루션 중 하나인 Xen, 로드 밸런싱 기법에 대해 설명한다.

2. 1. 서버 가상화

서버 가상화는 물리적인 컴퓨터 자원의 추상화를 지칭하는 넓은 의미의 용어이다.[2] 최근의 가상화 플랫폼은 하이퍼바이저(hypervisor)라 불리는 가상 머신 모니터(Virtual Machine Monitor, VMM)을 통하여 물리 자원을 추상화하며, 추상화 된 자원은 가상 머신 운영체제 상에서 하나의 독립적인 자원으로서 사용된다. 최근의 가상화 솔루션은 전가상화(total virtualization) 혹은 반가상화(para-virtualization) 기법을 기반으로 한다.

2. 1. 1. 전가상화

전가상화 기법은 ring compression, binary patch와 같은 기술을 기반으로 구현된 하이퍼바이저를 사용하여 가상 머신을 동작시키는 방식이다.[2] 이 기법은 가상 머신 운영체제의 수정 없이 바로 사용이 가능하다는 장점이 존재하지만, 물리 머신에 비하여 부하의 폭이 크다는 단점을 있다. 전가상화 기법을 사용하는 대표적인 솔루션에는 VMware가 있다.

2. 1. 1. 반가상화

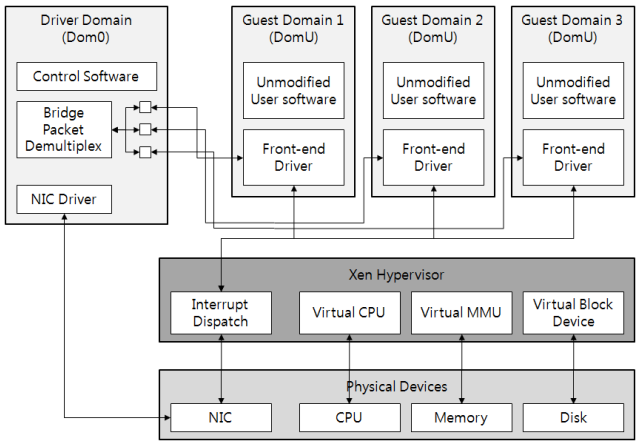
반가상화 기법은 물리 머신에서 사용하는 특권명령 대신에 하이퍼바이저에서 제공하는 하이퍼콜(hypercall)을 사용하여 특권 명령을 호출하도록 수정된 가상 머신 운영체제를 동작시키는 방식이다. 이 기법은 물리 머신에 근접한 성능을 얻을 수 있다는 장점이 존재하지만, 가상 머신 운영체제의 수정이 필요하다는 단점이 있다. 반가상화 기

법을 사용하는 대표적인 솔루션에는 Xen[1]이 있다.

2. 2. Xen

Xen은 하나의 물리 머신 위에서 여러 개의 가상 머신 운영체제를 동시에 실행할 수 있는 반가상화 기법이 적용된 x86 가상 머신 모니터이다.[4]

Xen 상에서 동작하는 가상 머신은 크게 두 부분으로 나눌 수 있는데, 한 부분은 Xen의 전반적인 관리를 담당하는 Domain0(Dom0)이며, 다른 부분은 Xen 상에서 서비스를 제공하는 가상 머신인 DomU이다. 각각의 물리 머신에 단 하나만 존재하는 Dom0는 DomU에 서비스를 제공하는 역할을 한다. Dom0는 물리 머신의 부팅 시간에 생성이 되고, 변경되지 않은 장치 드라이버를 관리한다.[8] 반면 DomU는 Dom0의 지원을 통하여 사용자에게 서비스를 제공하는 역할을 담당한다.



(그림 1) Xen I/O [2]

Xen 상에서의 네트워크는 그림 1과 같이 각각의 DomU에 front-ends(FE)라 불리는 가상 네트워크 인터페이스를 통하여 구현된다.[2] 또한 DomU의 FE에 상응하여 Dom0에 back-ends(BE)가 생성되며, FE의 관문 역할을 하게 된다. 각각의 FE와 BE는 가상 I/O 채널 상에서 연결되어있으며, 가상 I/O 채널은 page-flip 기술을 통하여 동작한다.[2]

2. 3. 로드 밸런싱 기법

로드 밸런싱 기법은 다수의 머신에 작업을 분배하는 기법으로서, 로드 밸런싱 기법의 적용을 통하여 프로세서가 가동되지 않는 시간을 줄이고, 높은 확장성을 꾀하며, 빠른 속도로 연산을 수행할 수 있다. 로드 밸런싱 기법은 동일기종 간의 로드 밸런싱 기법과 이기종 간의 로드 밸런싱 기법이 있는데, 동일기종 간의 로드 밸런싱 기법은 고정된 인자 값을 기반으로 하는 반면, 이기종 간의 로드 밸런싱 기법은 각각의 시스템에 배정하는 고정되지 않은 인자 값을 기반으로 하는 것이 특징이다.[9]

3. 제안 기법

본 절에서는 가상화 환경의 특징으로 발생하는 문제점을 살펴보고 본 논문에서 제시하는 가상 머신을 위한 로드 밸런싱 기법에 대해 설명한다.

3. 1. 기존 기법의 문제점

3. 1. 1. 이기종 간의 동적 로드 밸런싱 알고리즘의 문제점

기존에 상용되는 이기종 간의 동적 로드 밸런싱 알고리즘은 단일한 운영체제가 동작하는 물리 머신을 대상으로 한다. 동일기종 간의 로드 밸런싱 기법과는 달리, 이기종 간의 동적 로드 밸런싱 알고리즘에는 이기종 간의 성능 차이를 비교하는 부하 색인이 존재한다. 또한 부하 색인의 생성을 위하여 기존의 동적 로드 밸런싱 알고리즘은 작은 규모의 성능 테스트를 수행하여 이를 반영한다.

그러나 이러한 방법은 실제 서비스의 요청과는 상관없는 성능 테스트 프로그램을 수행해야 할 뿐 아니라, 성능 테스트 프로그램의 수행으로 발생하는 부하가 단일한 운영체제가 동작하는 물리 머신에 비하여 가상 머신의 개수만큼 발생한다는 문제점이 존재한다.

또한 시스템 사용률에 기반을 둔 일부 로드 밸런싱 기법은, 가상 머신에서 확인 가능한 시스템 사용율과 가상머신이 동작하는 실제 물리 머신의 시스템 사용률이 서로 다르기 때문에, 잘못된 부하 분배가 발생한다.

3. 1. 2. 가상 머신의 I/O 상의 문제점

Xen 상에서의 스케줄링의 정확성은 사용 가능한 네트워크의 대역폭과 밀접한 연관성이 존재한다.[7] 적절하지 못한 스케줄링은 네트워크 I/O 처리의 지체로 인한 대기로 대역폭이 낮아지게 된다. 또한 주어진 시간 내에 처리되지 못한 네트워크 I/O로 인하여 Dom0 혹은 DomU의 NIC의 queue 오버플로가 발생하여 TCP 재전송이 발생하여 사용가능한 네트워크의 대역폭이 더욱 낮아진다.[2]

그러므로 다량의 I/O가 요청될수록, 혹은 다수의 가상화 머신이 활성화 될수록 Dom0는 I/O 처리를 위한 충분한 시간을 배정받지 못할 수 있으므로 전체적인 네트워크 대역폭의 하락이 발생한다.

또한 가상 머신의 I/O를 구현하기 위해서는 물리 머신의 I/O를 동작시키는 것에 비하여 최소 3배 이상의 CPU 자원이 필요하다. 따라서 I/O 요청이 증가할수록 물리 머신에 비하여 머신의 성능에 더욱 악영향을 끼치게 된다.

3. 2. 가상 머신을 위한 서버 로드 밸런싱 기법

본 논문에서 제안하는 가상화 환경에서의 동적 로드 밸런싱 기법은 가상 머신의 I/O 요청의 총량을 기반으로 한다.

3. 2. 1. 시스템 모델

제안하는 기법은 다음과 같은 상황을 가정한다.

① N개의 물리 서버 $P_1, P_2, \cdots, P_N$으로 구성되어 있으며, 각각의 물리 서버는 M개의 가상 서버 $V_1, V_2, \cdots, V_M$가 있다.

② 각각의 물리 서버는 물리 서버에 속해있는 가상 서버를 관리하는 역할을 수행하며, 가상 서버는 사용자에 의해 요청된 작업을 처리하는 역할을 수행한다. 가상 서버는 단일한 종류의 작업만을 처리한다.

③ 각각의 물리 서버는 미션 크리티컬한 성능 보장을 위한 I/O 요청량의 한계가 존재한다. 이는 각각의 서버가 성능을 유지하며 작업을 처리하기 위하여 존재한다.

④ 요청되는 작업의 양은 충분히 많으며, 이를 처리하기 위하여 다수의 물리 서버에 위치한 다수의 가상 서버를 활용한다. 작업은 로드 밸런싱 알고리즘에 의해 분배된다.

⑤ 로드 밸런싱 처리를 위한 별도의 서버가 존재하며, 요청된 작업의 분배는 이 서버를 통하여 이루어진다.

3. 2. 2. 제안하는 로드 밸런싱 기법

로드 밸런싱 설정 단계에서, 각각의 물리 서버는 자신이 수용 가능한 I/O 한계치를 조사하여 로드 밸런싱 서버에 통보하며, 각각의 가상 머신은 자신이 수행하는 작업의 작업 별 I/O 요청 횟수를 로드 밸런싱 서버에 통보한다. 로드 밸런싱 과정에선, 가상 서버는 주기적으로 자신의 부하를 로드 밸런싱 서버에 통보하며, 로드 밸런싱 서버는 가상 서버가 실제 동작중인 물리 서버의 I/O 상황에 기반을 두어 작업을 분배한다.

가상 머신을 위한 로드 밸런싱 기법의 동작은 다음 네 단계로 구성된다.

① 물리 서버의 I/O 요청 처리 한계값 측정
② 작업 별 I/O 요청량 측정
③ 가상 머신의 부하 측정
④ 작업 분배

3. 2. 3. 물리 서버의 I/O 요청 처리 한계값 조사

각각의 물리 서버는 초기 설정 단계에서 I/O 요청 처리 한계값을 조사한다. 이를 위하여, 각각의 가상 머신의 I/O 요청 횟수의 총합을 측정하며, 가상 머신의 I/O 요청 횟수는 물리 머신과 가상 머신 간의 메모리 page-flip 횟수를 기반으로 한다. 물리 서버의 I/O 요청 처리 한계값 측정은 별도의 수행이 없는 한, 로드 밸런싱 기법의 적용 초기 1회만 발생한다.

<표 1> 물리 서버의 I/O 요청 처리 한계값 측정 알고리즘

```
// num_of_pm: number of physical machines
// num_of_vm: number of virtual machines
// mp_exchange: memory page exchanges
// limit: limit about acceptable I/O request

for (pm = 0; pm < num_of_pm; pm++) {
  for (vm = 0; vm < num_of_vm[pm]; vm++) {
    limit[pm]+= mp_exchange[vm]
  }
}
```

3. 2. 4. 작업 별 I/O 요청량 측정

로드 밸런싱 서버는 초기 설정 단계에서 작업 별 I/O 요청 횟수를 측정한다. 물리 서버의 I/O 요청 처리 한계값 조사와 마찬가지로 가상 머신의 I/O 요청 횟수는 물리 머신과 가상 머신 간의 메모리 page-flip 횟수를 기반으로 하며, 로드 밸런싱 기법의 적용 초기 1회만 발생한다.

<표 2> 작업 별 I/O 요청량 측정 알고리즘

```
// num_of_vm: number of virtual machines
// mp_exchange: memory page exchanges
// io_per_jobs: IO request per each jobs.

for (vm = 0; vm < num_of_vm; vm++) {
  io_per_jobs[vm] = mp_exchange[vm] / jobs[vm]
}
```

3. 2. 5. 가상 머신의 부하 측정

가상 머신이 부하 측정을 위하여, 단위 시간당 처리되는 요청된 작업의 개수를 측정한다. 이 과정을 주기적으로 수행하여 로드 밸런싱 서버에 통보한다.

$$load_{vm} = jobs_{vm} / t_{vm} \quad (1)$$

$jobs_{vm}$	가상 머신에 요청된 작업의 개수
t_{vm}	요청된 작업의 처리에 소요되는 시간
$load_{vm}$	단위 시간동안 처리되는 작업의 개수

3. 2. 6. 작업 분배

로드 밸런싱 서버는 작업 별 I/O 요청 횟수와 가상 머신에 요청된 작업 처리량을 바탕으로 물리 서버 전체에 요청된 I/O 총량을 측정한다. 로드 밸런싱 서버는 물리 서버에 요청된 I/O 총량을 물리 서버의 I/O 요청 처리 한계값과 비교하여, 한계값보다 작은 경우에만 작업을 분배한다.

<표 3> 작업 분배 알고리즘

```
// num_of_pm: number of physical machines
// num_of_vm: number of virtual machines
// mp_exchange: memory page exchanges
// limit: limit about acceptable I/O request

for (vm = 0; vm < num_of_vm; vm++) {
  total_io_request[pm] += load[vm] * io_per_jobs[vm]
}

for (pm = 0; pm < num_of_pm; pm++) {
  if (total_io_request[pm] < limit[pm]) {
    send_job (); // send job request
  }
}
```

3. 3. 기존 기법과 제안 기법과의 비교 · 분석

단일한 운영체제가 동작하는 물리 머신은 머신의 성능이 부하의 처리 속도에 직접적인 영향을 미친다. 그러나 가상화 환경에서는 각각의 가상 머신의 성능은 유동적이므로, 물리 머신의 성능과 물리 머신에 속한 각각의 I/O의 부하에 따라 부하의 처리 속도가 달라진다. 표 2는 기존의 환경과는 변화된 가상화 환경에서 요구되는 로드 밸런싱 기법을 4가지 선정 한 뒤, 이를 기반으로 기존 동적 로드 밸런싱 기법과 제안 기법을 비교한 결과이다.

<표 4> 기존 기법과 제안 기법 비교

기준 요소	기존 기법	제안 기법
작업 응답 시간	○	○
물리 머신 성능 반영	×	○
가상 머신 I/O 측정	×	○
성능 테스트 프로그램 사용 안함	×	○

표의 결과에 따르면 기존 기법과 제안 기법은 부하 측정을 위하여 작업 응답 시간을 기준으로 하는 것을 확인할 수 있다. 그러나 기존 기법은 한 가지 요소만을 만족시키는 반면, 제안 기법은 4가지 요소를 모두 만족시키는 것을 확인할 수 있다.

4. 결론 및 향후계획

본 논문에서는 가상화 환경에서 문제가 되는 로드 밸런싱의 오차 현상을 해결하기 위한 기법으로 가상 머신의 I/O 프로세싱에 필요한 CPU 비용 측정을 이용한 가상 머신을 위한 로드 밸런싱 기법을 제안했다. 가상 머신을 위한 로드 밸런싱 기법의 장점으로는 기존의 기법의 문제점이던 성능 테스트 프로그램의 중복 실행 및 I/O의 대역폭 급감을 감지 못하던 문제의 해결 가능, 적은 부하로 인한 실행 시 성능상의 효율성 등이 있다. 향후 연구 계획으로는 제안 기법에 대한 실제 구현 및 테스트를 통해 실질적인 성능평가를 실시하고자 한다.

참고문헌

[1] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, A. Warfield, “Xen and the art of virtualization”, Proceedings of the nineteenth ACM symposium on Operating systems principles, October 19-22, 2003

[2] G. Liao, D. Guo, L. Bhuyan, S. R. King, "Software techniques to improve virtualized I/O performance on multi-core systems", Proceedings of the 4th ACM/IEEE Symposium on Architectures for Networking and Communications Systems, November 06-07, 2008

[3] P. Padala, K. G. Shin, X. Zhu, M. Uysal, Z. Wang, S. Singhal, A. Merchant, K. Salem, "Adaptive control of virtualized resources in utility computing environments", Proceedings of the 2nd ACM SIGOPS/EuroSys European Conference on Computer Systems 2007, March 21-23, 2007

[4] A. Menon, J. R. Santos, Y. Turner, G. Janakiraman, W. Zwaenepoel, "Diagnosing performance overheads in the xen virtual machine environment", Proceedings of the 1st ACM/USENIX international conference on Virtual execution environments, June 11-12, 2005

[5] L. Cherkasova, R. Gardner, "Measuring CPU overhead for I/O processing in the Xen virtual machine monitor", Proceedings of the annual conference on USENIX Annual Technical Conference, p.24-24, April 10-15, 2005

[6] Ross, J.W. "Creating a strategic IT architecture competency: Learning in stages". MIS Quarterly Executive 2, 1 2003

[7] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, A. Warfield, "Live migration of virtual machines", Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation, p.273-286, May 02-04, 2005

[8] K. Fraser, S. Hand, R. Neugebauer, I. Pratt, A. Warfield, M. Williamson. "Reconstructing I/O". Technical Report UCAM-CL-TR-596, University of Cambridge, August 2004.

[9] C. A. Bohn, G. B. Lamont. "Load balancing for heterogeneous clusters of PCs". Future Generation Computer Systems, 18:389 - 400, 2002