논문 13-38C-06-03

한국통신학회논문지 '13-06 Vol.38C No.06
http://dx.doi.org/10.7840/kics.2013.38C.6.512

가상화 클라우드 데이터센터에서 가상 머신 간의 균등한 성능 보장을 위한 제어 알고리즘

김 환 태*, 김 황 남°

Control Algorithm for Virtual Machine-Level Fairness in Virtualized Cloud Data center

Hwantae Kim*, Hwangnam Kim°

요 약

본 논문은 가상 머신 기반의 클라우드 데이터센터에서 가상 머신의 CPU 스케줄링으로 인해 발생할 수 있는 네트워크 불평등 현상을 해결하는 가상머신 수준의 제어 알고리즘을 제안한다. 이를 위해 이기종 호스트들로 구성된 클라우드 데이터 센터 테스트베드를 구축하고, 가상 머신간의 네트워크 불평등 현상이 발생함을 실험적으로 보인다. 그리고 이를 해결할 수 있는 PID 제어 기법 기반의 가상 머신 네트워크 성능 보장 제어 알고리즘을 설계하고, 이를 실제 시스템에 구현하기 위한 방안을 설명한다. 실제 테스트베드에 제안하는 알고리즘을 구현하여 알고리즘 동작 결과를 분석한다.

Key Words : cloud system, datacenter, virtual machine, network fainrness, control algorithm

ABSTRACT

In this paper, the control algorithm which can resolve the unfairness in network performance of the virtual machine arised from the CPU scheduling in cloud datacenter has been proposed. We first describe the evaluation and analysis results of the network unfairness phenomenon of virtual machine through the heterogeneous cloud datacenter testbed and we propose the control algorithm which can guarantee the fairness of the network performance based on the PID control scheme. Through the implementation and evaluation results, we verify the performance of the proposed algorithm.

Ⅰ. 서 론

최근 클라우드 컴퓨팅이 인터넷 서비스의 새로운 패러다임을 형성하고 있다. 클라우드 컴퓨팅은 데이터센터 네트워크 위에 구축되어 사용자가 언제 어디서든 접속 가능한 서비스를 제공하고 있다. 클라우드 컴퓨팅의 기반 기술 중 가장 중요한 것이 가상화 기술이다. 하드웨어 가상화, 네트워크 가상화 등의 기술을 통해 보다 적은 자원과 비용을 사용하여 효율적인 서비스가 가능하며, 다양한 사용자의 요구를 만족시킬 수 있는 클라우드 컴퓨팅 환경을 구축할 수 있다[1,2].

기존의 서버 기반 서비스 패러다임과 클라우드 기반 서비스 패러다임의 가장 큰 차이점은 저사양

※ 본 연구는 미래부가 지원한 2013년 정보통신·방송(ICT) 연구개발사업의 연구결과로 수행되었습니다.
※ 본 연구는 2013년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업 입니다.
(No. NRF-2013R1A1A2010388)
◆ 주저자 : 고려대학교 전기전자전파공학부 무선네트워크 연구실, kht2k@korea.ac.kr, 학생회원
° 교신저자 : 고려대학교 전기전자전파공학부 부교수, hnkim@korea.ac.kr, 종신회원
논문번호 : KICS2013-03-147, 접수일자 : 2013년 3월 29일, 최종논문접수일자 : 2013년 5월 24일

의 장비를 많이 사용함으로써 인프라 구축을 위해 사용되는 비용을 줄이면서 기존의 인프라 용량을 유지하는 것이다[3-4]. 사용자에게 더 나은 서비스를 제공하기 위해 저장 공간, 네트워크 장비, 그리고 CPU 등의 장비를 업그레이드 하는 것이 아니라 저사양의 더 많은 장비를 사용하여 저비용으로 필요로 하는 서비스 용량을 구축한다. 이러한 패러다임을 'Scale Out'이라 한다. 'Scale Up' 대신에 'Scale Out'을 통해 클라우드 시스템을 지속적으로 확장하게 되면, 결국 전체 클라우드 인프라가 이기종 장비들로 구성되게 된다[5]. 그림 1은 "Scale Out" 패러다임의 개념도이다.

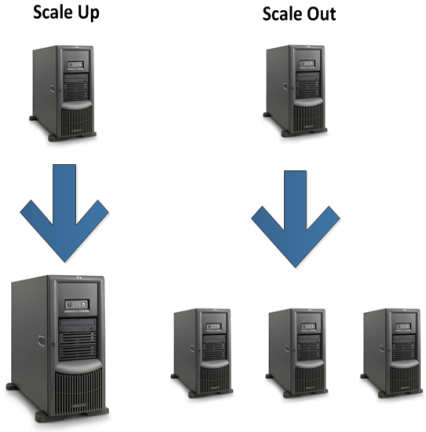


그림 1. Scale-Out 방식의 개념도
Fig. 1. The conceptual representation of the "Scale Out"

"Scale Out" 방식의 클라우드 시스템 구축을 통해 클라우드 인프라의 최초 구성 비용 및 서비스 운영 비용을 크게 줄일 수 있다. 하지만 'Scale Out'으로 인해 발생하는 클라우드 구성요소의 이기종성으로 인한 다양한 문제점이 발생하게 된다. 특히 가상 머신 기반의 클라우드 시스템에서 가상 머신의 이기종성에 따른 CPU 사용률의 차이에 따라 네트워크 성능에 심각한 불균형 현상이 발생한다. 상대적으로 성능이 더 좋은 서버에 존재하는 가상 머신이 다른 가상 머신보다 네트워크 자원을 더 많이 차지하게 되고 이는 네트워크 성능에 있어 심각한 불균형을 초래하게 된다[6,7].

본 논문에서는 이기종 호스트로 구성된 클라우드 네트워크에서 가상 머신간의 성능 차이로 인해 발생하는 네트워크 불평등 현상을 실험적으로 보이고, 이것이 CPU 스케줄링에 따른 결과임을 실험적으로 분석한다. 그리고 가상 머신 플랫폼 수준에서 네트워크를 제어함으로써 네트워크 성능 불평등 현상을 해결할 수 있는 기법을 제안한다.

본 논문은 2장에서 Xen 기반의 가상머신 기술에 관해 간략히 소개하고, 가상 머신의 스케줄링으로 인한 네트워크 불평등 현상을 실험적으로 보인다. 3장에서는 이를 해결하기 위한 PID 제어 이론 기반의 가상 머신 네트워크 성능보장 알고리즘의 제안 동기 및 동작 방식을 설명한다. 4장에서는 본 논문에서 제안하는 네트워크 제어 알고리즘이 효율적으로 동작함을 테스트베드를 통한 실험으로 검증하고 5장에서 본 논문을 결론 맺는다.

Ⅱ. 가상 머신 네트워크 불균형 현상

본 장에서는 클라우드 시스템 구축을 위해 널리 사용되는 Xen 가상화 기법에 대해 간략히 설명하고, 가상 머신의 CPU 스케줄링에 따른 네트워크의 불균형 현상을 실험을 통해 확인한다.

2.1. Xen 가상 머신

Xen 클라우드 구축을 위해 가장 널리 사용되는 오픈소스 가상 머신 플랫폼으로 x86 기반의 아키텍처에서 동작하는 반가상화 가상 머신 플랫폼이다. Xen은 크게 dom0 와 domU로 구성된다. dom0는 부팅시에 생성되어 다른 도메인을 생성, 종료시키고, CPU 스케줄링 및 다른 컴퓨팅 자원의 할당을 제어하는 도메인이다. 또한 dom0 에 각종 드라이버가 포함되어 반가상화 기반의 각 가상 머신들이 물리적인 장비에 직접적으로 접근하기 위한 인터페이스의 기능을 수행한다. domU는 Xen에서 동작하는 각각의 게스트 가상 머신을 의미한다[8].

Xen 가상 머신은 Credit 스케줄러를 사용한다[9,10]. 각각의 가상 머신들이 가중치와 최대치로 정의되는 파라미터 조건 하에서 균등하게 CPU를 차지하여 사용하게 된다. Non -preemptive 스케줄링 알고리즘으로 작업 보존성의 특징을 갖는 스케줄러이다. 로드 밸런싱 기법이 적용되어 특정 가상 머신이 많은 CPU를 필요로 하는 경우 CPU 자원을 효율적으로 분배한다.

이러한 가상머신의 CPU 스케줄링이 네트워크 성능의 심각한 불균형을 초래할 수 있다. 가상 머신으로 패킷이 전달되어 처리되어야 하는 시점에 CPU를 할당 받지 못해 패킷을 처리하지 못하면, TCP 알고리즘에 의해 혼잡 윈도우가 느리게 증가하거나 네트워크 혼잡이 발생하지 않았음에도 타임아웃 등이 발생하여 혼잡 윈도우가 급격히 감소하여 네트워크 성능이 저하될 수 있다. 또한 작업 보존성의

특징 때문에 특정 가상 머신으로부터의 패킷을 지속적으로 처리하는 과정에서 해당 가상 머신의 혼잡 윈도우는 급격히 증가하지만, 다른 가상머신의 혼잡 윈도우는 느리게 증가하게 되어 네트워크 불균형 현상을 야기할 수 있다. 본 논문에서는 CPU 스케줄링으로 인한 네트워크 성능 저하 현상을 실험적으로 분석한다.

2.2. Explicit Congestion Notification

본 논문에서 각 가상 머신간의 균등한 성능 보장을 위해 Explicit Congestion Notification (ECN)을 사용한다. ECN은 IP와 TCP를 기반으로 네트워크 혼잡 발생 시 송신측의 혼잡 윈도우를 감소시키도록 하는 혼잡 제어 기법이다. 기존의 TCP 혼잡 제어 기법은 전송 되는 패킷의 손실 혹은 중복되는 ACK 패킷의 수신을 통해 송신측이 인지하고 혼잡 윈도우를 줄이는 방식을 사용한다. ECN은 중복 ACK의 수신이나 패킷 손실 없이 라우터에서 혼잡 상황을 인지하면 이를 송신측에 전달해 주어 불필요한 패킷 손실 없이 송신측이 네트워크 혼잡 상태를 인지하고 혼잡 윈도우를 조절할 수 있도록 하는 기법이다. ECN은 다음과 같은 방식으로 동작한다.

- 송신측, 수신측에서 ECN의 사용이 활성화 되어 있는 경우 IP 헤더의 DiffServ 필드를 '10(ECT(0))' 혹은 '01(ECT(1))' 으로 인코딩 한다.
- ECN이 활성화 되어 있는 라우터에서 네트워크 혼잡 상황을 인지하면 라우팅 하고자 하는 패킷에 대해 IP 헤더의 DiffServ 필드의 값을 '11(CE)' Set 한 뒤 해당 패킷을 드롭 하지 않고 수신측으로 전송한다.
- 수신측에서 IP 헤더의 DiffServ 필드가 CE로 설정되어 있는 패킷에 대해 TCP 헤더의 ECE, CWR 비트를 사용하여 송신측에게 혼잡 발생 여부를 알려준다.
- ECE, CWR 비트가 Set 되어 있는 ACK를 수신한 송신측은 혼잡 윈도우의 크기를 줄임으로써 네트워크 혼잡 상황에 대처한다.

2.3. 가상 머신 간 네트워크 불균형 현상

2.3.1. 클라우드 데이터 센터 테스트베드

본 논문에서는 이기종 장비들이 혼재되어 있는 클라우드 데이터 센터를 단순화 하여 서버 PC 1대, 노트북 2대, 그리고 1Gbps 라우터로 구성된 테스트베드를 구축하였다. 각각의 서버 장비에 두개의Xen 가상 머신을 생성하였다. 그림 2는 테스트베드의 구성도를 나타내고, 표 1은 테스트 베드 구성 장비들의 사양을 나타낸다. 이기종 장비들로 구성된 시스템을 테스트하기 위해 각각 다른 사양의 서버 장비들을 사용하여 테스트베드를 구성하였다.

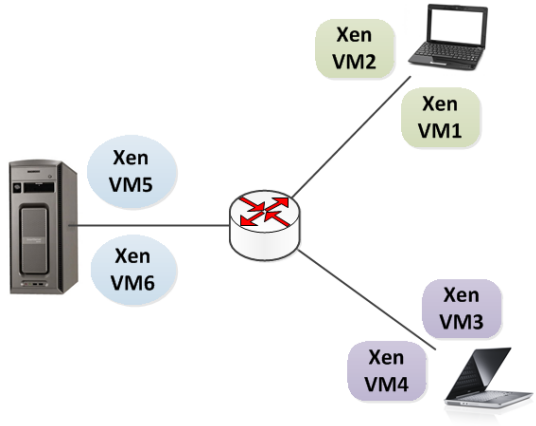


그림 2. 클라우드 데이터센터 테스트 베드 구성도
Fig. 2. The cloud datacenter testbed

표 1. 테스트 베드 구성 장비 사양
Table 1. The specification of the testbed components

	Model	Specification
Server 1	PC	2GHz dual core 1GB RAM 1Gbps Ethernet
Server 2	Laptop (Fujitsu Series A6010)	1.83GHz dual core 512MB RAM 1Gbps Ethernet
Server 3	Laptop (Lenovo 7668-A19)	1.66GHz dual core 1GB RAM 1Gbps Ethernet
Router	Buffalo (wzr-hp-g450h)	5-Port 1Gbps Ethernet

2.3.2. 가상 머신 네트워크 성능 감소 현상 분석

그림 3은 VM1과 VM5, VM3과 VM6 사이에 단독으로 트래픽을 발생시켰을 경우의 수율을 나타낸다. VM1은 871Mbps, VM3는 911Mbps의 평균 수율을 보였다. 그림 3의 결과를 통해 일반적인 상황에서 VM1이 VM3 보다 더 낮은 네트워크 성능을 가짐을 확인할 수 있고, 테스트베드가 이기종성을 잘 반영하고 있음을 확인할 수 있다.

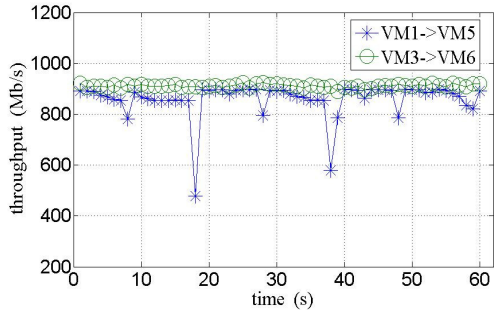


그림 3. VM1, VM3의 업로드 단독 트래픽 수율
Fig. 3. Throughputs of individual upload traffic of VM1 and VM3

그림 4는 VM1, VM3에서 VM5, VM6으로 동시에 업로드 트래픽을 발생시킨 경우의 수율을 나타낸다. 그림 5는 VM1은 VM5로 업로드, VM3은 VM6으로 다운로드 하는 상황에서의 수율을 나타낸다. VM1, VM3에서 업로드 트래픽을 발생시킨 경우 각각 206Mbps, 717Mbps의 수율을 보였고, VM1에서 업로드, VM3에서 다운로드 트래픽을 발생시킨 경우 249Mbps, 662Mbps의 수율을 보였다. 이를 통해 일반 적인 상황에서 낮은 네트워크 성능을 보이는 VM1이 VM3와 동시에 네트워크를 사용할 때 심각한 성능 저하를 보임을 확인할 수 있다.

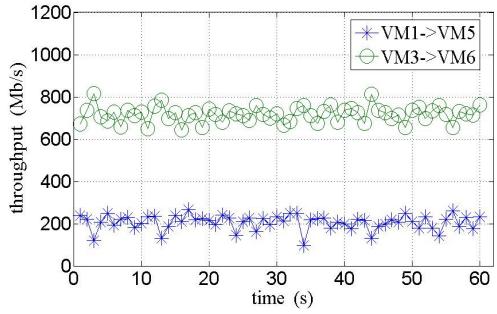


그림 4. VM1, VM3의 동시 업로드 수율
Fig. 4. Throughputs of simultaneous upload traffic of VM1 and VM3

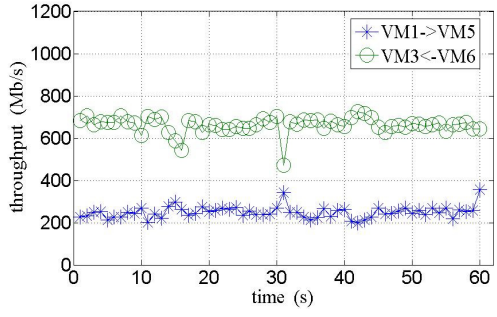


그림 5. VM1 업로드, VM3 다운로드 동시 트래픽 수율
Fig. 5. Throughputs of simultaneous upload traffic of VM1 and download traffic of VM3

그림 6은 VM1과 VM3에서 가상 머신이 아닌 서버 1로의 업로드 트래픽을 발생시킨 경우이다. 이 실험에서는 VM1과 VM3가 거의 비슷한 수율을 보였다. 이를 통해 그림 4,5의 네트워크 성능 불평등 현상이 가상 머신간의 서버 1상에서 동작하는 가상 머신 VM5, VM6 간의 스케줄링으로 인해 발생하는 것임을 확인할 수 있다.

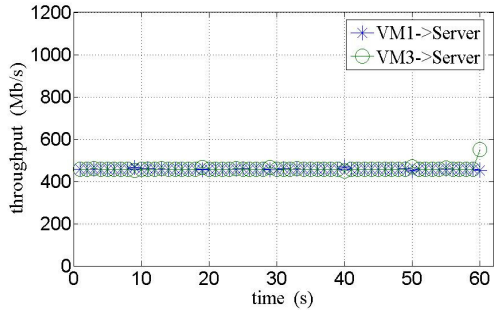


그림 6. VM1, VM3의 서버 1로의 업로드 단독 트래픽 수율
Fig. 6. Throughputs of individual upload traffic of VM1 and VM3 to Server 1

그림 7은 VM1과 VM3가 동시에 업로드를 행하는 상황에서 VM5, VM6의 CPU 사용률을 나타낸다. 네트워크 성능이 뛰어난 VM3과 통신하는 VM6가 VM5보다 더 많은 CPU 사용률을 보임을 확인할 수 있다. 따라서 이기종 장비로 구성된 클라우드 네트워크에서 네트워크 성능이 상대적으로 좋지 않은 가상 머신에 대해 CPU가 적게 할당되고 네트워크 성능에 심각한 불균형이 초래한다는 것을 실험적으로 확인할 수 있다.

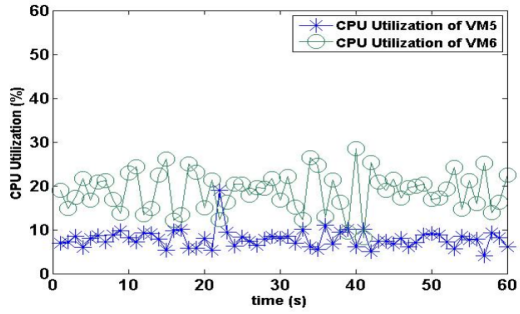


그림 7. VM5와 VM6의 CPU 사용률
Fig. 7. CPU utilization of VM5 and VM6

Ⅲ. 가상 머신 네트워크 성능 보장 제어 알고리즘

본 장에서는 2장에서 분석한 이기종 호스트로 구성된 클라우드 네트워크의 불균형 현상을 해결할 수 있는 제어 알고리즘을 설명하고, 실제 시스템에 구현하기 위한 방안을 설명한다.

3.1. 가상 머신 네트워크 성능보장 제어 알고리즘

이기종 호스트상에서 동작하는 가상 머신간에 균등한 네트워크 성능 보장을 위해 각 가상 머신에 동일한 네트워크 사용 시간을 할당하고 사용 시간을 초과하여 사용한 가상 머신의 혼잡 윈도우를 감소시켜 네트워크 균형을 유지하도록 하는 알고리즘을 제안한다, 본 알고리즘에서는 PID 제어 기법을 적용한 Decision Probability를 사용한다. Decision Probability에 따라 Sender의 혼잡 윈도우를 감소시켜 다른 Sender의 네트워크 성능을 보장할 수 있도록 알고리즘을 구성한다. Decision Probability는 다음 식 (1),(2)를 통해 결정된다.

$$e_i[k] = \frac{U_i[k] - C_i[k]}{C_i[k]} \tag{1}$$

$$P_i[k] = K_p e_i[k-1] + K_i \sum_{t=1}^{k} e_i[t-1] + K_d \frac{e_i[k-1] - e_i[k-2]}{T} \tag{2}$$

Decision Probability P는 매 Time Slot마다 결정되며, T는 Time Slot 간의 시간 간격을 의미한다. U_i 는 가상 머신 i가 각 Time Slot에 네트워크를 사용한 시간이고, C_i는 가상 머신 i에 할당된 네트워크 사용시간이다. 식 (2)를 통해 계산된 Decision Probability를 바탕으로 네트워크 자원을 많이 사용하는 가상 머신의 혼잡 윈도우를 줄이도록 유도하여 모든 가상 머신 간의 균등한 성능을 보장한다.

그림 8은 본 논문에서 제안하는 알고리즘의 동작 흐름을 나타낸다.

- 각각의 가상 머신에 대해 네트워크 사용시간을 균일하게 할당한다.
- 일정 시간 주기로 각각의 가상 머신 네트워크 사용 시간을 계산하고, 식 (2)를 통해 Decision Probability를 계산한다.
- 매 패킷마다 Decision Probability를 통해 혼잡 윈도우를 제어하기 위한 의사결정을 수행한다.
- TCP와 IP의 ECN을 사용하여 네트워크를 많이 사용하는 송신측의 혼잡 윈도우를 감소시켜 다른 가상 머신의 네트워크 성능을 향상시킨다.

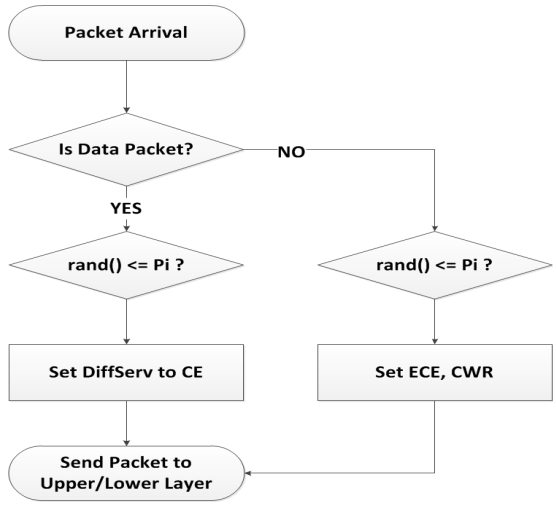


그림 8. 알고리즘 동작 순서도
Fig. 8. The flow chart of the proposed algorithm

3.2. 성능 보장 알고리즘 구현

Xen은 dom0의 Xen Network Backend 모듈을 통해 domU로의 네트워크 인터페이스를 제공한다. 따라서 Xen Network Backend 모듈에서 사용되는 정보들을 바탕으로 알고리즘을 실제 시스템에 구현한다. 그림 9는 Xen의 네트워크 구성 방식을 나타낸다. Xen은 domU의 네트워크 장비(eth0)를 dom0의 가상 네트워크 인터페이스(vif)와 연결하고, 가상 네트워크 인터페이스와 실제 장비의 네트워크 디바이스(peth0)와 브리지를 통해 연결하는 방식으로 게스트 운영체제의 네트워크 연결을 구성한다. Xen Network Backend를 실제 호스트의 물리적 네트워크 장치 드라이버를 추상화 하여 게스트 운영체제의 네트워크 드라이버에 인터페이스를 제공한다.

그림 10은 Xen Network Backend에서 성능 보장 제어 알고리즘의 구현방안을 나타낸다. 게스트 운영체제에서 패킷 전송이 발생하면(hard_start_xmit) Network Backend의 xen_netbk_tx_submit() 함수가 호출되고, xenvif_receive_skb() 함수를 통해 브리지를 거쳐 물리적 장치로 패킷을 전달한다. 이때 성능 보장 모듈로 패킷을 전달하고, 성능 보장 모듈에서 해당 패킷의 크기를 지속적으로 기록한다. 패킷을 수신하게 되면 실제 장치 드라이버로부터 xen_netbk_rx_action() 함수가 호출되고, 그 후 패킷을 성능 보장 모듈로 전달하여 패킷의 크기를 측정하고, Decision Probability에 따라 ECN 사용 여부를 결정한다. 그리고 해당 패킷이 Xen Network Backend의 수신 처리 함수(xen_netbk_rx_action) 다시 전달된 뒤 게스트 운영체제 드라이버의

netif_rx() 함수로 전달된다.

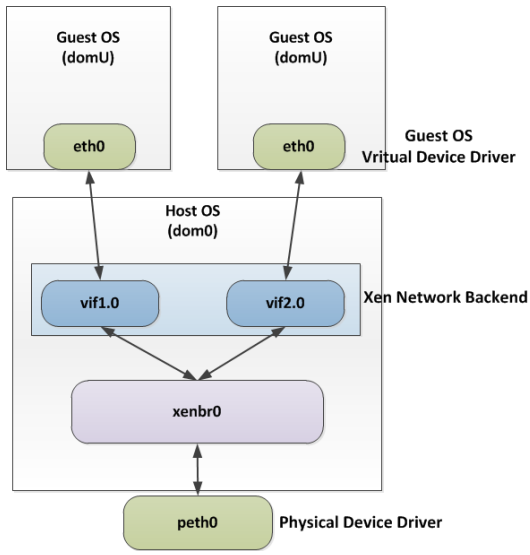


그림 9. Xen 가상 머신의 네트워크 구조
Fig. 9. The virtual architecture of the Xen virtual machine

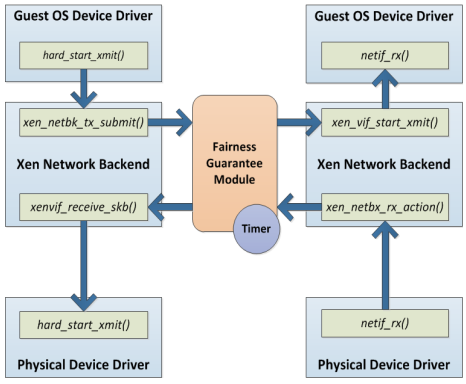


그림 10. 성능 보장 제어 알고리즘 구현 방안
Fig. 10. Implementation of the performance gurantee control algorithm

이와 같은 성능 보장 알고리즘은 서로 다른 호스트에 존재하는 가상머신과 통신을 하는 호스트의 dom0 상에 구현된다. 본 논문에서는 테스트 베드의 서버 1 호스트에 알고리즘을 구현한다.

각 가상 머신의 네트워크 사용 시간은 특정 시간 간격 동안 전송된 패킷의 양을 통해 추정한다. 서버 1 호스트에서 송/수신 되는 모든 패킷의 양을 각각의 가상 머신 별로 기록하고, 특정 시간 주기 마다 네트워크 사용 시간을 식 (3)과 같이 계산한다.

$$U_i[k] = \frac{B_i[k]}{NC} \quad (3)$$

이때, $B_i[k]$는 가상머신 i가 Time Slot 동안 송수신한 바이트의 총량이고, NC는 네트워크 대역폭이다. 본 논문의 테스트 베드에서 NC는 1Gbps 이다. 이를 통해 네트워크 사용시간을 추론하여 성능 보장 알고리즘에 반영한다.

Ⅳ. 실험 결과

본 장에서는 3장에서 설명한 네트워크 성능 보장 알고리즘을 실제 테스트 베드에 구현하여 동작하는 실험 결과를 보인다. 본 장의 실험 결과는 5장에서 소개하는 관련 연구들과 네트워크 성능의 공평성 유지라는 측면에서 같은 결과를 보이지만, 기존 시스템의 스케줄러나 이벤트 처리 방식의 변경 없이 모듈 단위에서 네트워크 트래픽의 양을 제어함으로써 균등한 네트워크 성능을 보장한다는 점에서 차이를 보인다.

그림 11은 VM1과 VM3에서 VM5와 VM6로의 트래픽을 발생시킨 그림 4의 실험을 네트워크 성능 보장 알고리즘 동작 환경 하에서 수행한 결과이다. VM1의 업로드 평균 수율은 414Mbps, VM2의 업로드 평균 수율은 500Mbps로 측정되었다. 약간의 불평등이 발생하는 이유는 일정 시간 간격을 주기로 Decision Probability를 계산하게 되는데 CE비트를 처리한 후 다음 Decision Probability를 계산하기까지의 시간 동안 VM3의 혼잡 윈도우가 다시 급격히 증가하기 때문이다. 실험 결과를 통해 VM1, VM3의 업로드 상황에서 완전히 균등한 네트워크 수율을 보이지는 않지만 알고리즘이 동작하지 전보다 VM1의 수율이 약 100% 정도 증가했음을 확인할 수 있다.

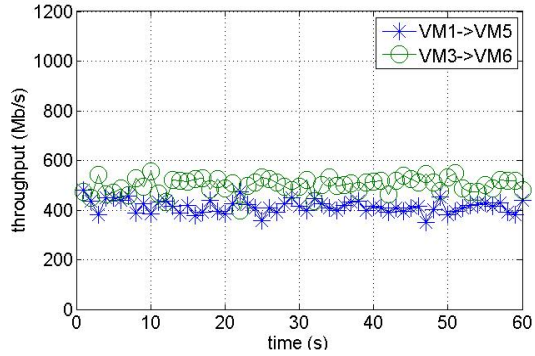


그림 11. VM1, VM3의 동시 업로드 수율
Fig. 11. Throughputs of simultaneous upload traffic of VM1 and VM3

그림 12는 그림 5에서와 같이 VM1은 업로드 트래픽을 발생시키고, VM3에서 다운로드 트래픽을 발생시킨 경우의 실험 결과이다. 이 경우에는 VM1, VM3간의 네트워크 성능이 거의 균일하게 보장되었음을 확인할 수 있다.

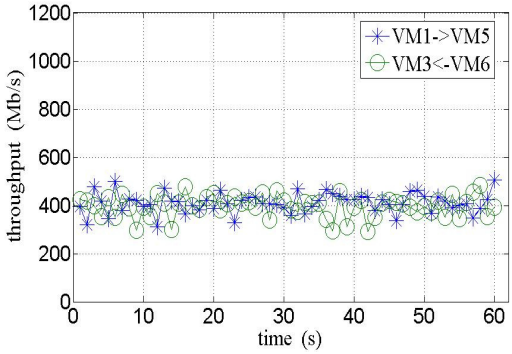


그림 12. VM1 업로드, VM3 다운로드 수율
Fig. 12. Throughputs of simultaneous upload traffic of VM1 and download traffic of VM3

그림 13, 14는 일반적인 가상 머신의 운영 시나리오에서 제안하는 알고리즘의 동작 성능을 나타낸다. 그림 13은 VM1이 VM5로 업로드를 수행 하는 도중에 VM3가 VM6로의 전송을 시작하는 시나리오에서의 네트워크 수율을 나타낸다. 60초까지는 VM1 만 단독으로 네트워크 자원을 사용하다 60초부터 120초까지는 VM1과 VM3가 네트워크 자원을 함께 사용하게 되고 이후 VM1 이 다시 네트워크 자원을 모두 사용하게 된다. 실험 결과를 통해 제안하는 알고리즘이 동적인 네트워크 환경 변화에 잘 반응함을 확인할 수 있다. 각 VM별 네트워크 사용 시간을 추론할 때 Time Slot을 사용하는데 Time Slot의 길이가 짧기 때문에(200ms) 급격한 변화에도 잘 반응할 수 있다.

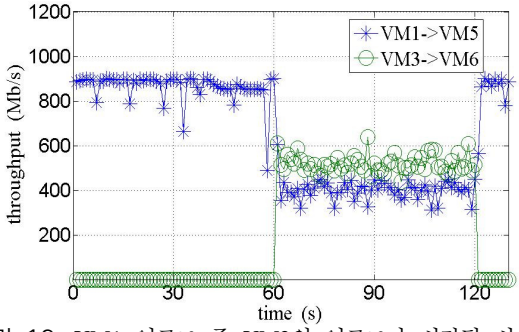


그림 13. VM1 업로드 중 VM3의 업로드가 시작된 시나리오에서의 수율
Fig. 13. Throughputs of the scenario where VM3 starts upload in the middle of the transmission of VM1

그림 14는 VM1에서 VM5로의 업로드 트래픽이 존재하는 환경에서 VM3의 다운로드 전송이 시작되었을 경우의 수율을 나타낸다. 그림 14의 결과를 통해 그림 13의 결과와 마찬가지로 제안하는 알고리즘이 동적인 네트워크 상황 변화에도 잘 반응하여 동작함을 확인할 수 있다.

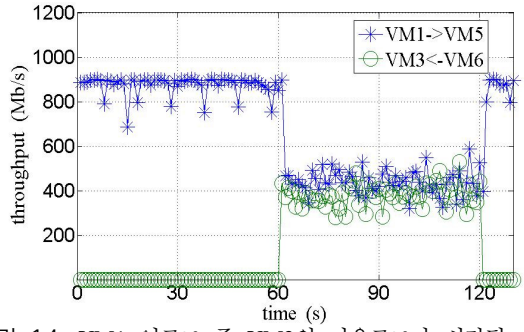


그림 14. VM1 업로드 중 VM3의 다운로드가 시작된 시나리오에서의 수율
Fig. 14. Throughputs of the scenario where VM3 starts download in the middle of the transmission of VM1

그림 11, 12, 13, 14의 실험 결과를 통해 본 논문에서 제안하는 네트워크 성능 보장 알고리즘이 이기종 호스트로 구성된 클라우드 시스템 환경에서 효율적으로 각 가상 머신의 네트워크 성능을 보장할 수 있음을 확인할 수 있다.

V. 관련 연구

가상 머신의 I/O 스케줄링에 관한 다양한 연구들이 진행 되어 왔다. 특히, 참고문헌 [7], [8], [11]은 각 가상 머신에서 네트워크 인터페이스를 포함한 I/O를 공평하게 사용가능 하게 하는 가상 머신 스케줄링 방식을 제안한다. 참고문헌 [7]에서는 VMM (Virtual Machine Monitor)에서 발생하는 Semantic Gap으로 인한 I/O 불평등 현상을 해결하기 위한 Task-Aware 스케줄링 기법을 제안한다. Semantic Gap이란 VMM에서 각각의 게스트 도메인의 Taks를 알지 못해 I/O 이벤트 처리가 지연되는 현상을 의미한다. 이를 해결하기 위해 I/O 중심의 Task의 일반적인 특징과 혼재된 워크로드 간의 Correlation을 통한 I/O Task 추론 기법, 그리고 Task 레벨에서 CPU Preemption을 수행하는 스케줄링 기법을 제안한다. 참고문헌 [11]에서는 Xen의 Credit 스케줄러가 가지는 I/O 불평등의 원인을 고찰하고, 공평한 I/O 성능을 보장하는 SEDF 스케줄러를 제안한다. Credit 스케줄러에서 Boost 모드로 인한 게스트 도메인의 CPU Preemption을 최소화하고, 각 Task의 Deadline을 기반으로 스케줄러의 Queue를 재배치함으로써 공평한 I/O 사용이 가능하

도록 한다. 참고문헌 [8]은 각 가상 머신들간에 QoS에 따른 서비스 차별화를 위한 스케쥴링 기법을 제안한다. 네트워크 자원 사용에 대한 Credit, 그리고 Leaky-bucket 제어기를 통한 스케쥴링을 통해 가상 머신레벨의 차별화된 네트워크 성능 제공을 가능하도록 한다. 이와 같은 연구들 모두 가상 머신 간의 네트워크 불평등 현상을 해결할 수 있는 방안을 제시한다. 하지만 기존의 스케쥴러를 새로운 스케쥴러로 대체하여 균일한 네트워크 성능을 제공한다는 점에서 본 연구와 차이점을 보인다. 기존의 스케쥴러가 CPU분배의 측면에서는 효율성을 보이는데, I/O 성능의 균형을 위해 새로운 스케쥴러를 도입하는 것은 복잡도가 매우 커지고, CPU 분배를 성공적으로 수행하지 못할 수 있다. 이와 다르게 본 연구에서 제안하는 알고리즘은 가상 머신의 스케쥴러는 기존과 동일하게 유지하면서 네트워크 트래픽의 양을 제어함으로써 네트워크 성능의 균형을 유지하기 때문에 구현상의 복잡도를 낮추면서도 효율적으로 동작할 수 있다.

Ⅵ. 결 론

본 논문에서는 이기종 호스트들로 구성된 가상 머신 기반의 클라우드 데이터 센터에서 CPU 스케쥴링으로 인한 네트워크 불평등 현상을 실험적으로 분석하였다. 그리고 이를 해결하기 위한 PID 제어 이론 기반의 가상 머신 네트워크 성능 보장 알고리즘을 제안하였다. 이를 실제 시스템에 구현하기 위한 방법을 논의하고 실제 구현된 결과를 바탕으로 성능 평가를 수행하였다. 성능 평가 실험 결과를 통해 본 논문에서 제안하는 알고리즘이 이기종 호스트로 구성된 가상 머신 기반 클라우드에서 효율적으로 동작하는 것을 확인할 수 있다.

References

[1] I. Y. Jung, I. Jo, and Y. Yu, “Trust assurance of data in cloud computing environment,” *The Journal of the Korea Information and Commuication Society,* vol. 36, no. 9, pp. 1066-1072, Sep. 2011.

[2] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, “A view of cloud computing,” *Communnications of the ACM*, vol. 53, no. 16, pp. 50-58, Apr. 2010.

[3] M, Peter, and T. Grance. ″The NIST definition of cloud computing,″ *NIST special publication,* SP 800-145, Sep. 2011, from http://csrc.nist.gov/publications/PubsSPs.html#

[4] A. Greenberg, P. Lahiri, D. A. Maltz, P. Patel, and S. Sengupta, “Towards a next generation data center architecture: scalability and commoditization,” in *Proc. ACM Workshop on Programmable Routers for Extensible Services of Tomorrow*, pp. 57-62, Seattle, U.S.A, Aug. 2008.

[5] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel, “The cost of a cloud: research problems in data center networks,” *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 1, pp. 68-73, Jan. 2009.

[6] A. Madhavapeddy, R. Mortier, J. Crowcroft, and S. Hand, “Multiscale not multicore: efficient heterogeneous cloud computing,” in *Proc. 2010 ACM-BCS Visions of Computer Science Conerence,* Article no. 6, Edinburgh, U.K, Apr. 2010.

[7] H. Kim, H. Lim, J. Jeong, H. Jo, and J. Lee, “Task-aware virtual machine scheduling for I/O performance,” in *Proc. 2009 ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments*, pp. 101-110, Houston, U.S.A., Mar. 2009.

[8] S. K. Lee, H. Kim, J.-G. Ahn, K. J. Sung, and J. Park, “Provisioning service differentiation for virtualized network devices,” in *Proc. The International Conference on Networking and Services (ICNS 2011)*, pp. 152-156, Venice, Italy, May 2011.

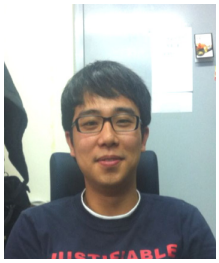
[9] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, “Xen and the art of virtualization,” *ACM SIGOPS Operating System Review,* vol. 37, no. 5, pp. 164-177, Dec. 2003.

[10] L. Cherkasova, D. Gupta, and A. Vahdat, “Comparison of the three CPU schedulers in Xen,” *ACM SIGMETRICS Performance*

Evaluation Review, vol. 35, no. 2, pp. 42-51, Sep. 2007.

[11] D. Ongaro, A. L. Cox, and S. Rixner, "Scheduling I/O in virtual machine monitors." in *Proc. 4th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments,* pp. 1-10, Seattle, U.S.A., Mar. 2008.

김 환 태 (Hwantae Kim)



2012년 2월 고려대학교 전기전자전파공학부 졸업

2012년 3월~현재 고려대학교 전기전자전파공학부 석사과정

<관심분야> 통신공학, 네트워크공학, 융합IT

김 황 남 (Hwangnam Kim)

1992년 2월 부산대학교 컴퓨터 공학과 졸업

1994년 2월 서울대학교 컴퓨터 공학과 석사

2004년 2월 미국 어바인-샴페인 소재 Illinois 주립대학 컴퓨터과학과 박사

1994년~1999년 LG 전자 주임연구원

2000년~2001년 Bytemobile 소프트웨어 엔지니어

2004년~2005년 미국 Illinois 주립 대학 Post Doctorate Fellow

2005년~2006년 삼성전자 책임연구원

2006년~현재 고려대학교 전기전자전파공학부 교수

<관심분야> 통신공학, 네트워크공학, 융합IT, CPS