NUMA affinity 를 고려한 Workload Consolidation 연구

서동유, 김신규, 최찬호, 엄현상, 염헌영
서울대학교 컴퓨터 공학과
e-mail : dyseo@dcslab.snu.ac.kr

A study of workload consolidation considering NUMA affinity

Dongyou Seo, Shin-gye Kim, Chanho Choi, Hyeonsang Eom, Heon Y. Yeom
School of Computer Science & Engineering, Seoul National University

요 약

SMP(Symmetric Multi-Processing)는 Shared memory bus 를 사용함으로써 scalability 가 제한적이었다. 이런 SMP 의 scalability 제한을 극복하기 위해 제안 된 것이 NUMA(Non Uniform Memory Access)이다. NUMA 는 memory bus 를 CPU 별 local 하게 가지고 있어 자신이 가지는 memory 영역에 대해서는 다른 영역을 접근하는 것 보다 더 빠른 latency 를 가지는 구조이다. Local 한 memory 영역의 존재는 scalability 를 높여 주었지만 서버 가상화 환경에서 VM 을 동적으로 scheduling 을 하였을 때 VM 의 page 가 실행되는 core 의 local 한 메모리 영역에 존재하지 않게 되면 remote access 로 인해 local access 보다 성능이 떨어진다. 이 논문에서는 서버 가상화 환경에서 최신 architecture 인 AMD bulldozer 에서 NUMA affinity 가 위반되었을 때 발생하는 성능 저하와 어떤 상황에서 이런 NUMA affinity 가 위반되어도 성능저하가 없는지 연구하였다.

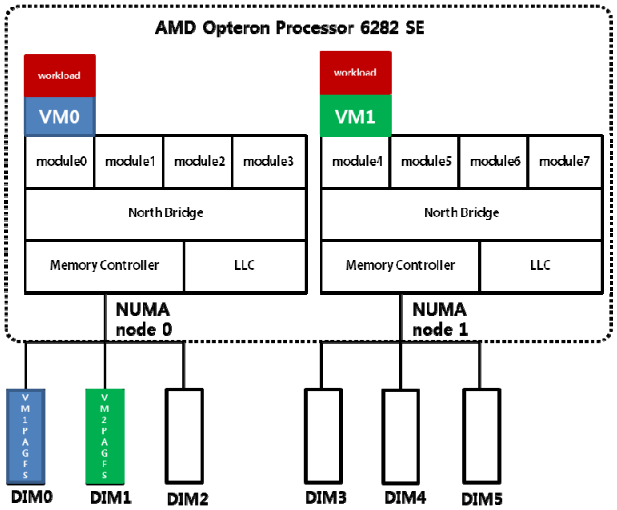
1. 서론

현재 Cloud platform 에서 resource utilization 을 높이기 위해서 서버 가상화 기술이 널리 사용 되고 있다. 하나의 node 에 여러 Virtual Machine 을 사용하여 최대한 idle 한 resource 를 줄여 utilization 을 높이는 기술이다. 서버 가상화에서의 Workload Consolidation 연구가 활발하게 진행 중이며 LLC miss 와 memory bandwidth 와 같은 scheduling metric 들이 제안 되었다. 최신 server 의 CPU 는 Non Uniform Memory Access 구조를 가지고 있어서 이런 scheduling metric 을 이용해 VM 을 schedule 해줄 경우 VM 에 할당된 core 와 VM page 가 존재하는 NUMA node 가 다른 die 에 존재하게 되어 VM 이 자신의 page 에 remote access 를 해야 하는 상황이 발생 한다[1]. 이 논문에서는 VM 이 자신의 page 에 local access 로 접근 할 때와 remote access 로 접근할 때를 비교하여 NUMA affinity 가 위반 된 상황에서 얼마나 성능 저하가 발생하는지 확인 하였다.

2. 실험

2.1 실험방법

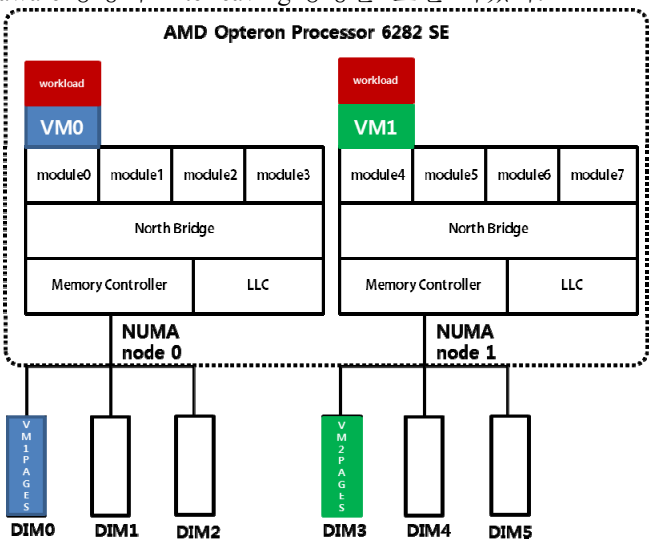
이 실험에서는 3 개의 VM 을 Non NUMA-aware 상황, NUMA-aware 상황 과 Interleaving 3 가지 상황을 만들고 그 상황에서 SPEC CPU 2006 workload[2]들의 수행 시간을 측정 하였다. Non NUMA-aware 상황을 만들기 위해서 VMM(Virtual Machine Monitor)의 VM page 할당 방식을 이용 하였다. VMM 는 VM 의 page 를 VM 이 수행 할 때 필요한 page 를 동적으로 memory 에 할당 한다. 이 특성을 이용하여 NUMA node 0 에 VM0(module 0)과 VM1(module 1)을 pinning 한 후 SPEC CPU 2006 workload 를 한번 실행 시켜 NUMA



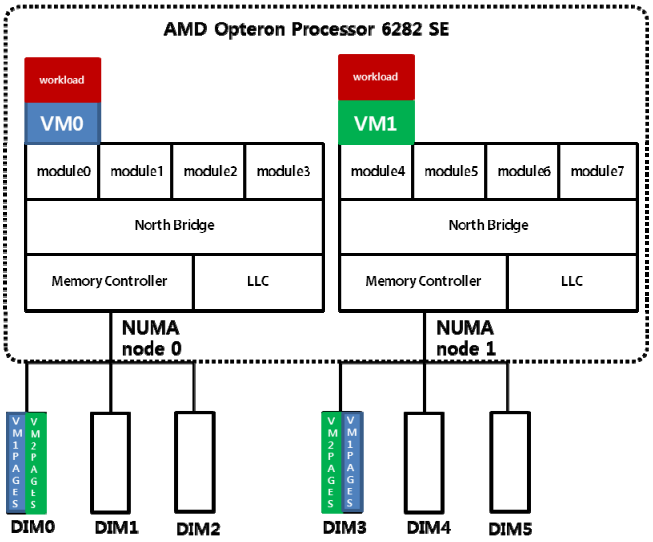
(그림 1) Non NUMA-aware 상황

node 0 에 두 VM 의 page 들을 몰아 넣은 후 VM1 을 다른 NUMA node(module 4)에 pinning 을 시키고 이전에 실행 했던 똑 같은 SPEC CPU 2006 workload 를 수행 시킨다. 이때 VM1 은 똑 같은 page 를 사용 하기 때문에 NUMA node 1 에 속해 했지만 NUMA node 0 에서 자신의 page 를 remote access 한다. 그림 1 은 Non NUMA aware 를 표현 한 것이다. NUMA-aware 상황은 VM 1(module 4)의 page 가 NUMA node 0 에 있는 것이 아니라 처음부터 VM 1 을 module 4 에 pinning 을 시켜 VM1 의 page 가 NUMA node 1 에 할당 시켜놓고

workload 를 실행 시킨 상황이다. Interleaving 은 hardware 적으로 모든 page 를 골고루 node 별로 분산 시켜 주는 기능이다. 이 기능을 하용 하면 VM 0 과 VM 1 의 page 가 NUMA node 0 과 1 에 골고루 분산된 상황이 만들어 진다. 그림 2 와 3 은 각각 NUMA-aware 상황과 Interleaving 상황을 표현 하였다.



(그림 2) NUMA-aware 상황



(그림 3) Interleaving 상황

2.2 실험환경

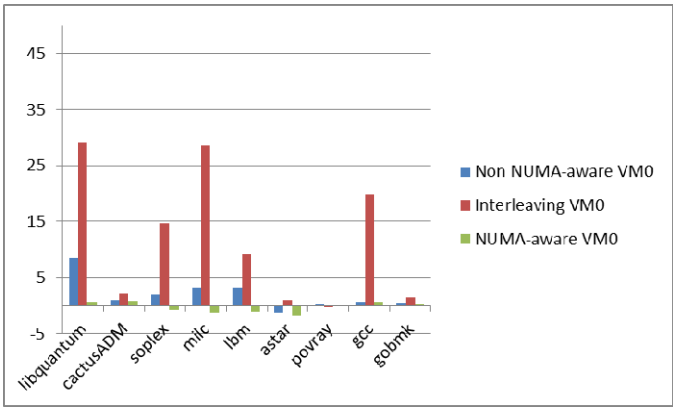
이 실험에서는 AMD Opteron Processor 6282 SE 를 사용하였으며 이 CPU 의 architecture 는 최신 architecture 인 bulldozer 이다. 128GB memory 환경에서 실험을 하였고 서버가상화를 구성하는 Hypervisor 로 KVM 을 사용하였다. 실험에 사용한 workload 는 SPEC CPU 2006 benchmark 이며 그 중에서 Memory Intensive Workload 5 개(libquantum, soplex, milc, lbm, gcc)와 CPU Intensive Workload 4 개(cactusADM, astar, povray, gobmk)를 사용하였다. 표 1 은 NUMA-aware 한 상황에서 workload 혼자 돌았을 때 실행 시간이다. 표 1 은 이 실험에서 baseline 으로 사용되었다.

2.3 실험결과

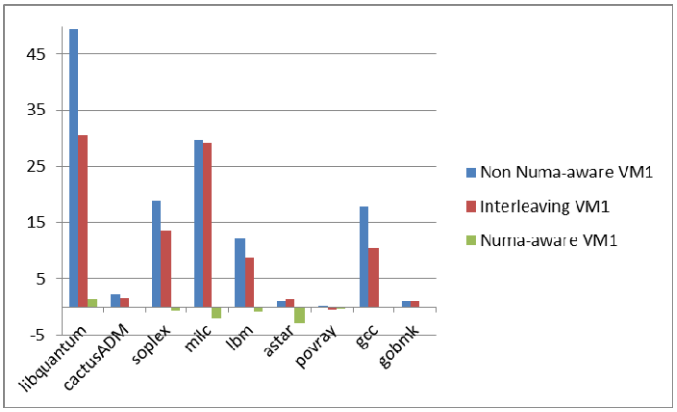
그림 1 은 앞서 설명 한 3 가지 상황에서 VM 0 의 workload 수행 시간을 나타낸 것이다.

<표 1> SPEC CPU 2006 workload baseline

Workload	runtime(sec)
libquantum	918
cactusADM	2048
soplex	506
milc	650
lbm	556
astar	803
povray	411
gcc	675
gobmk	788



(그림 4) VM 0 에서 workload 수행 시간



(그림 5) VM 1 에서 workload 수행 시간

VM 0 의 page 는 Non NUMA-aware 와 NUMA-aware 상황에서 모두 NUMA node 0 에 모든 page 가 있다. 하지만 Non NUMA-aware 상황에서는 VM 0 과 VM 1 이 같은 NUMA node 를 공유 하기 때문에 contention 이 발생하여 VM 0 의 성능이 저하 된다. 하지만 CPU Intensive Workload (cactusADM, astar, povray, gobmk)는 memory access 가 적기 때문에 Non NUMA-aware 상황에서도 성능 저하가 거의 없었다. Interleaving 상황에

서는 VM 의 page 들을 node 별로 나누기 때문에 VM 0 은 page 의 절반을 remote access 하기 때문에 성능 저하가 Memory Intensive Workload (libquantum, soplex, milc, lbm, gcc)에서 심하게 발생한다. 그림 5 는 VM 1 의 workload 성능을 나타낸 것이며 실제 Non NUMA-aware 상황에서 VM 1 은 모든 page 를 NUMA node 0 로 remote access 하는 상황이다. Non NUMA-aware 상황에서 VM 0 에서 libquantum 의 성능 저하는 8.3%였던 반면 VM 1 에서는 50%로 remote access 로 인한 성능 저하가 큰 것으로 나타났다. 전체적으로 Interleaving 보다 Non NUMA-aware 상황에서 더 큰 성능 저하가 발생 하였다. 반면에 NUMA-aware 상황에서는 VM 0,1 의 모든 workload 가 거의 baseline 의 성능을 보였다.

3. 결론

서버 가상화를 사용하는 Cloud platform 에서 NUMA affinity 를 가진 CPU 를 사용 할 경우 VM scheduling 을 할 때 NUMA affinity 가 위반되어 성능 저하가 발생 하는 경우가 있다. 이 실험에서 3 가지의 상황 Non NUMA-aware, Interleaving, NUMA-aware 상황을 재현하고 Non NUMA-aware 상황이 얼마나 성능이 저하가 되는지 보여주었다. 또한 Memory Intensive Workload 에서는 성능 저하가 심했지만 CPU Intensive Workload 는 모든 상황에서 거의 baseline 과 비슷한 성능을 보였다. 결론적으로 NUMA affinity 가 위배되면 성능저하가 발생하지만 CPU Intensive Workload 의 경우에는 그렇지 않은 것으로 확인 되었다. VM scheduling 을 구성할 때 NUMA affinity 가 위배된 VM 들의 page 를 local 한 NUMA node 로 migrate 하는 과정이 필요하지만 CPU Intensive Workload 를 수행하는 VM 의 page 는 migrate page 과정을 생략해도 상관 없다는 결론을 낼 수 있다.

4. 감사의 글

이 논문은 2011 년도 정부(교육과학기술부)의 재원으로 한국연구재단-차세대정보컴퓨팅기술개발사업의 지원을 받아 수행된 연구임. (No. 2011-0020521) 이 연구를 위해 연구장비를 지원하고 공간을 제공한 서울대학교 컴퓨터연구소에 감사 드립니다.

This research was supported by Next-Generation Information Computing Development Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education, Science and Technology. (No. 2012-0006424) The ICT at Seoul National University provided research facilities for this study.

참고문헌

[1] Blagodurov,S., Zhuravlev,s., Mohammad, D., and Fedoravo, A., “A case for numa-aware contention management on multicore processors”

[2] SPEC CPU 2006, http://www.spec.org/benchmarks.html