The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 23, No. 4, pp.85-91, Aug. 31, 2023. pISSN 2289-0238, eISSN 2289-0246

https://doi.org/10.7236/JIIBC.2023.23.4.85
JIIBC 2023-4-14

클라우드 컴퓨팅 응용 구동을 위한 마이크로서버 성능평가

Performance Evaluation of Microservers to drive for Cloud Computing Applications

오명훈*

Myeong-Hoon Oh*

요 약 국산 마이크로서버인 KOSMOS의 활용을 위해 클라우드 컴퓨팅 분야의 실제 응용 서비스 기반 벤치마크 프로그램인 CloudSuite로 성능 평가 결과를 제시한다. CloudSuite는 오프라인 응용과 온라인 응용의 두 가지 부분에서 클라우드 서비스로 제공되는 몇 가지의 구분되는 응용 프로그램을 컨테이너 기반으로 제공하고 있다. KOSMOS의 유사 스펙의 비교군인 다른 마이크로서버와의 성능 비교에서 전 부분에 걸쳐 KOSMOS가 우수하였으며, 인텔 Xeon CPU 기반 서버와의 비교에서도 일부 오프라인 응용에서는 성능이 더 우수하였다. CloudSuite 오프라인 응용 벤치마크 프로그램인 Graph Analytics 수행 시 KOSMOS의 다수의 노드들을 분산 실행시킨 형상에서 인텔 Xeon CPU 기반 2개의 서버 비교군과 비교하였을 때, 각각 30.3%, 72.3%만큼의 수행시간을 감소시켰다.

Abstract In order to utilize KOSMOS, the performance evaluation results are presented in this paper with CloudSuite, an application service-based benchmark program in the cloud computing area. CloudSuite offers several distinct applications as cloud services in two parts: offline applications and online applications on containers. In comparison with other microservers which have similar hardware specifications of KOSMOS, it was observed that KOSMOS was superior in all CloudSuite benchmark applications. KOSMOS also showed higher performance than Intel Xeon CPU-based servers in an offline application. KOSMOS reduced completion time during executing Graph Analytics by 30.3% and 72.3% compared to two Intel Xeon CPU-based servers in an experimental configuration of multiple nodes in KOSMOS.

Key Words : microserver, cloud computing, CloudSuite

Ⅰ. 서 론

KOSMOS (Korea Supreme Micro Server)는 4개의 CPU를 장착한 컴퓨팅 보드를 1개의 인클로저 내에 16개를 장착시킬 수 있는 국내에서 개발된 마이크로서버이다[1]. x86 CPU 및 arm 계열의 CPU를 장착한 컴퓨팅 보드를 혼합하여 장착할 수 있도록 컴퓨팅 보드의 인터페이스를 단일화하였고, 컴퓨팅 보드 간 혹은 컴퓨팅 보드내의 CPU간 통신을 위해, 이더넷 이외에 별도로 Serial-RapidIO(SRIO) 프로토콜[2] 기반의 I/O 인터페

*정회원, 호남대학교 컴퓨터공학과
접수일자 2023년 7월 2일, 수정완료 2023년 7월 31일
게재확정일자 2023년 8월 4일

Received: 2 July, 2023 / Revised: 31 July, 2023 /
Accepted: 4 August, 2023
*Corresponding Author: mhoh@honam.ac.kr
Dept. of Computer Engineering, Honam University, Korea

이스를 제공한다. 따라서, KOSMOS 한 대의 인클로저 내에 총 64개의 별도의 물리적인 CPU가 존재하고 이들은 고속의 연결망으로 통신이 가능하여 분산 협업 기반의 응용 프로그램 수행에 적합하다. 아울러, 기존 서버에 장착되는 CPU에 비해 저전력, 저발열 특성의 CPU를 채용하여 전력 소비 측면에서도 이득이 존재한다.

이러한 KOSMOS의 전용 연결망 측면에서의 성능 측정한 결과[3]는 이미 제시되어 있으나, 실제 마이크로서버 군들이 타겟으로 하는 응용 애플리케이션 중의 하나인 클라우드 컴퓨팅에 적용하여 성능을 측정해 볼 필요성이 존재한다.

클라우드 컴퓨팅은 사용자 요청에 따라 확장 가능한 클라우드 서비스를 온라인으로 제공하고 있다. 실제 클라우드 컴퓨팅 플랫폼에서 널리 운영되고 있는 클라우드 서비스, 예를 들어, 웹 검색, 소셜 네트워킹 및 비즈니스 분석 등의 특징은 비교적 대량의 작업 세트, 높은 수준의 병렬 처리 및 실시간 처리 등이다.

이러한 특성 때문에 클라우드 서비스 성능 측정을 위한 벤치마크 프로그램 측면에서, 데스크톱 PC용 벤치마크 프로그램[4], 병렬 처리 전용 벤치마크 프로그램[5] 및 전통적인 상용 서버 응용 벤치마크 프로그램[6]과는 차별성이 존재한다. 따라서, 클라우드 컴퓨팅 분야에 대한 성능 비교를 위해 실제 클라우드 온라인 서비스를 기반으로하는 벤치마크 프로그램이 요구된다.

CloudSuite는 클라우드 서비스를 위한 벤치마크 프로그램으로 스위스 로잔 연방 공과대학에서 관리 유지되고 있다[7]. 세 번째 배포 버전은 데이터 센터에서 널리 사용되는 8 개의 응용 프로그램으로 구성되며, 벤치마크 프로그램은 실제 소프트웨어 스택을 기반으로 각각의 응용 프로그램 설정을 반영한다. 현재 네 번째 배포 버전은 세 번째 배포 버전에 2가지의 응용 프로그램이 더 추가되었다.

본 논문에서는 KOSMOS의 주요 목적 애플리케이션 분야인 클라우드 서비스의 실제에 가까운 성능 측정을 위해 CloudSuite 세 번째 배포 버전을 사용한다. 아울러, KOSMOS와 같은 형태의 마이크로서버 군의 사용 분야확장을 위해, 다양한 마이크로서버 군의 실제 제품과 고사양의 서버의 성능도 측정하여 비교 분석한다.

II. CloudSuite 벤치마크 프로그램

CloudSuite는 오늘날의 데이터 센터에서 수행되는 광범위한 클라우드 기반 응용 프로그램 범주를 포함한다. 첫 번째 배포 버전에는 데이터 분석, 데이터 제공, 미디어 스트리밍, 대규모 계산 집약적 작업, 웹 검색 및 웹 서비스가 포함되었고, 두 번째 배포 버전에서는 그래프 분석 및 데이터 캐싱 기능을 확장하였다.

세 번째 배포 버전인 CloudSuite 3.0은 벤치마크 및 인프라 측면에서 이전 배포 버전에 비해 크게 향상된 기능을 포함한다. 즉, 널리 사용되는 비디오 공유 웹 사이트 설정에 따라, 새로운 실시간 비디오 스트리밍 벤치마크인 Apache Spark[8]를 사용하는 메모리 내 데이터 분석과 웹 서버 소프트웨어 스택을 미러링하는 웹 서빙과 같은 엄격한 대기 시간 제약 조건으로 대규모 데이터 조작을 위한 벤치마크 프로그램이 포함된다.

CloudSuite 3.0은 크게 offline과 online 벤치마크 프로그램으로 나뉜다. Offline 벤치마크 프로그램은 대규모 데이터셋으로 기계학습 알고리즘을 수행한다. 실시간 제약사항이 없고 주어진 입력 데이터에 따라, 수행결과를 출력하는 데까지의 걸리는 수행 시간을 측정한다. 멀티 노드 환경에서는 마스터 노드에서 수행 전에 다수의 슬레이브 노드를 구성하여 수행할 작업을 분산시켜 실행 후 결과값을 취합하는 형태로 동작한다.

Online 벤치마크 프로그램은 대규모 데이터셋으로 클라이언트에서 서버쪽으로 서비스를 요청하여 특정 서비스가 수행될 때의 단위 시간당 실시간 처리량을 측정한다. 아울러, 서비스 품질을 위해 백분위 수 지연 시간(N-th percentile latency)도 측정한다. 멀티 노드 환경에서는 서버 쪽에 다수의 멀티 노드를 할당하여 서버의 부하를 분산시킬 수 있으며, 특정 벤치마크 프로그램에서는 클라이언트에 멀티 노드를 할당하여 서버쪽에 부하를 부가할 수도 있다.

CloudSuite 3.0의 Offline 벤치마크 프로그램으로는 Data Analytics, In-Memory Analytics, Graph Analytics가 존재한다. Data Analytics는 기계학습 타스크 분석에 필요한 맵리듀스 프레임워크를 가정하고, 위키피디아의 정보와 기계학습 라이브러리인 Mahout[9]으로 맵리듀스를 수행하여 특정 분야의 단어 수를 추출한다.

In-memory Analytics는 영화 추천 응용 프로그램의 일종으로 movielens[10]의 9066개의 영화와 671명의 사용자로부터의 십만 여 개의 평가 정보를 사용한다. Spark의 기계학습 라이브러리인 Spark MLlib[11]에서 제공하는 ALS(alternating least squares) 알고리즘[12]을 수행하여 추천 시스템을 동작시킨다.

표 1. CloudSuite 3.0 벤치마크 프로그램 구성 요약
Table 1. Summary of configurations of benchmark programs in CloudSuite 3.0

종류	서버	클라이언트	사용 데이터	응용 분야
Data Analytics	Apache Hadoop	Apache Mahout	Wikipedia	기계학습 기반 맵리듀스
Graph Analytics	Apache Spark	Apache GraphX	twitter	페이지 랭크
In-Memory Analytics	Apache Spark	Apache MLlib	movielens	영화추천 시스템
Data Caching	Memcached	Twitter user emulator	twitter	Memcached server warming up -> Twitter 사용자 요청
Data Serving	Apache Cassandra	Emulator	YCSB	NoSQL DB 서버 생성 -> YCSB stress
Media Streaming	NGINX	Httperf traffic generator	Video data	NGINX 웹 서버 -> httperf traffic 발생
Web Serving	NGINX Memcached MySQL	Faban traffic generator		Elgg SNS에 접속
Web Search	Apache Solr	Faban traffic generator		인터넷 검색 엔진

Graph Analytics는 하이퍼 링크 구조를 가지는 문서에서 상대적 중요도에 따라 가중치를 부여하는 페이지랭킹 수행 시 걸리는 시간을 측정한다. 트위터 데이터 세트를 이용하여 Spark의 GraphX[13]로 시스템을 수행한다.

Offline 벤치마크 프로그램으로는 Data Caching, Data Serving, Media Streaming, Web Serving, Web Search의 5개의 프로그램이 존재한다. Data Caching은 기본적으로 memcached[14] 서버의 성능 측정을 위한 것으로, 트위터 사용자를 가정하고, 미리 관련 데이터를 메모리에 옮겨 놓은 캐싱된 데이터를 클라이언트 단에서 사용요청 후 응답 시 단위 시간당 처리율을 측정한다.

Data Serving은 NoSQL 분산 데이터베이스인 Cassandra[15]에 읽기, 쓰기, 삭제 명령 수행 시의 지연시간과 단위 시간당 처리율을 측정한다. 분산 데이터베이스 생성 시에는 YCSB[16] 프레임워크의 데이터를 사용한다.

Media Streaming은 비디오 스트리밍 전송에 대한 성능을 측정한다. 클라이언트에서는 서로 다른 비디오 길이를 갖는 요청 트래픽을 발생시키고, 이를 처리할 수 있는 웹서버는 이벤트 구동 기반 방식의 NGINX[17]를 사용한다.

Web Serving은 실제 SNS 서비스를 구현하고 성능을 측정한다. 서버에서 구동되는 SNS 서비스는 elgg[18]를 기반으로 구현되며, 클라이언트에서는 Faban[19]의 트래픽 발생기로 SNS 서비스에 로그인을 요청한다. 수행 시에는 SNS 서비스의 내부 각 단계의 지연시간을 측정한다.

마지막으로 Web Search는 검색엔진을 구현하기 위해, 서버에서는 아파치 Solr 검색 플랫폼[20]을 사용하고, 클라이언트에서는 Faban[19] 트래픽 발생기로 검색 요청을 발생시킨다. 이후 서버내의 ISN(Index Serving Node)에서 검색 수행한 후 검색 시간을 측정한다.

표 1은 CloudSuite 3.0의 각 벤치마크 프로그램의 서버, 클라이언트 측에 사용된 주요 엔진 및 데이터 소스를 요약하고 있고, 관련 애플리케이션을 표시한다.

Ⅲ. 비교 대상 서버 및 실험 방법

KOSMOS에서 수행하는 클라우드 응용 프로그램의 수행 성능을 비교하기 위해 유사 제품군인 마이크로서버인 HP사의 Moonshot[21]과 Supermicro 사의 MicorBlade[22]를 비교 대상으로 실험한다. 아울러, 기존 인텔 x86기반 고성능 서버로 각각 E5-2440과 E5-2630 Xeon CPU[23]를 탑재한 서버를 비교 대상으로 실험한다. 표 2는 이들 서버에 대한 기본 스펙을 요약한다.

표 2. 비교 대상 서버의 규격 요약
Table 2. Summary of specifications in servers for comparisons

모델	코어 수	CPU 모델	캐시 (MB)	메모리 (GB)	내부망
MoonShot	64	Intel Atom S1260 @2.00GHz	8	128	ETH (1G)
MicroBlade	128	ntel Atom C2750 @2.40GHz	16	128	ETH (2.5G)
KOSMOS	128	Intel Atom C2758 @2.40GHz	16	128	ETH (2.5G) SRIO
E5-2440	10	Intel Xeon E5-2440 @2.0 GHz	15	64	
E5-2630	10	Intel Xeon E5-2630 @2.2 GHz	25	64	

마이크로서버 형태 군은 모두 내부에 16개의 내부 컴퓨팅 모듈을 장착 가능하고, 컴퓨팅 모듈끼리는 서로 내부연결망으로 연결된다. 따라서, 테스트에 사용된 모든 마이크로서버 형태의 실험군의 코어 수, 캐시, 메모리 크기는 각 컴퓨팅 모듈의 것에 16을 곱한 수치이다. KOSMOS의 경우는 내부 연결망으로 이더넷 이외에 SRIO 기반 연결망[24]을 별도로 가진다.

마이크로서버군과 고성능 서버군 사이에서는 전체 코어 수, 캐시, 총 메모리 용량에서 마이크로서버군이 더 높은 수치로 유리하게 보이지만, 마이크로서버군은 코어나 메모리 자원을 분산 처리[25]하기 위해서는 내부 연결망을 통해야 하는 오버헤드가 존재한다. 고성능 서버군에서는 CPU 칩 내 혹은 메인보드 내부에서 모든 자원을 접근 가능하다. 다시 말해서, 저성능의 여러 CPU를 별도의 연결망으로 사용하는 것과 고성능 CPU를 단일하게 사용하는 비교점이 존재할 수 있다.

마이크로서버는 다수의 컴퓨팅 자원(컴퓨팅 노드)을 지원하며, 다수의 컴퓨팅 노드들을 수행 응용에 따라 동적으로 구성하여 수행한다. 따라서, 본 실험에서도 실제 사용환경을 반영한 보다 의미있는 데이터 추출을 위해, CloudSuite 3.0의 다양한 응용에서 컴퓨팅 노드의 숫자를 변경하여 실험한다. Offline쪽 벤치마크 프로그램에서는 단일 마스터 노드에 다수의 슬레이브 노드로 설정하고 실행하며, Online쪽 벤치마크 프로그램에서는 클라이언트 혹은 서버 단을 다수의 노드로 할당하여 수행한다. 이에, 비교 대상 및 노드 수에 따른 구성에 따라 성능 비교가 가능하다. 실제 실험에서는 주요 비교 대상인 MicroBlade가 활용 노드 수가 최대 8개로 제한적이어서, 모든 대조군의 최대 멀티 노드의 수를 6개로 한정하고 마스터, 혹은 슬레이브의 수에 따라, 1-to-1, 1-to-2, 1-to-4, 1-to-6의 환경을 구성하였다.

Ⅳ. 실험 및 결과

본 장에서는 CloudSuite 3.0의 여러 벤치마크 프로그램을 3장에서 제시한 실험 군에 적용하여 측정한 결과를 제시하고 비교 분석한다. 아울러, 마이크로서버 군에서는 실제 컴퓨팅 노드 수를 변경한 다양한 구성에 따른 결과를 측정하였으며, 도커 swarm 모드[26]를 사용하였다. 비교 분석 과정에서 마이크로서버 군의 Offline 계열의 Data Analytics, Graph Analytics, In-Memory Analytics 벤치마크 프로그램 결과는 기 출판된 결과[3]의 데이터를 사용한다. online 벤치마크 프로그램으로는 본 논문에서 마이크로 서버군의 인프라 환경에 따라 제약이 있었던 3개의 프로그램을 제외한 실행이 가능했던 Data Serving과 Web Serving을 분석한다.

1. Data Analytics

기계 학습을 위해 위키피디아 데이터로부터 모델을 생성한다. 마스터에서는 분류를 위해 위키피디아 문서를 슬레이브로 보내고, 슬레이브에서는 생성된 모델을 사용하여 위키피디아 문서를 분류한 후 결과를 마스터로 전송한다. 슬레이브의 수를 늘려가면서 맵리듀스 실행을 분산시켜 보았으며, 측정 기준은 수행 시간이다.

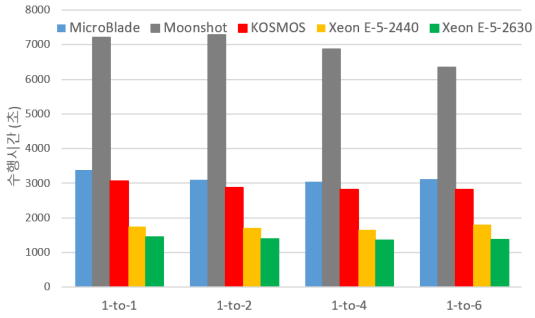

그림 1. Data Analytics 수행시간 실험 결과
Fig. 1. Experiment results of completion time in data Analytics

그림 1은 Data Analytics의 수행시간 결과를 나타낸다. 전반적으로 슬레이브 수가 증가함에 따라 큰 성능의 변화가 없음이 관찰되었고, 이는 슬레이브로 분산되어 수행되는 Mahout의 워크로드가 그다지 크지 않음을 알 수 있다. 아울러, 분산 워크로드가 애초에 작기 때문에, 별도의 연결망을 통해 워크로드를 분산시키고 결과 값을 모으는 마이크로서버 형태보다 단일 프로세서에서 다중 프로세스 형태로 노드를 증가시키는 Xeon 서버의 수행시간이 더 적은 것으로 평가되었다. Moonshot의 경우 슬레이브의 수를 늘릴수록 수행 시간이 감소되는데, 이는 분산된 워크로드가 큰 경우는 슬레이브를 늘려 분산시키는 것이 이득이 될 수 있음을 알려준다. 그럼에도 불구하고 KOSMOS의 성능은 MicroBlade 보다 최대 9.5%, Moonshot 보다 최대 60.5 % 수행시간을 감소시켰다.

2. Graph Analytics

페이지랭킹 애플리케이션을 수행한다. 트위터 데이터 세트로부터의 사용자 그래프 정보를 통해, 각 슬레이브에 그래프 분석을 위한 GraphX의 분산 처리 과정을 수행시킨다. 측정 기준은 역시 수행시간이다.

Graph Analytics의 수행 시간을 나타내는 표 3과 같이, 전반적으로 슬레이브의 숫자가 증가함에 따라서 수행시간은 감소되는 추세이며, 마이크로서버 계열을 제외한 Xeon 계열에서는 슬레이브 4개, 6개에서는 오히려 시간이 더 증가함을 할 수 있다. 이는 Graph Analytics의 슬레이브에서 수행되는 GraphX의 워크로드가 앞의 Data Analytics 보다는 더 큼을 알 수 있으며, 각 물리 컴퓨팅 노드에 분산시켜 실행하는 마이크로서버 군보다 한 프로세서에서 다수의 컨네이너를 생성시키는 Xeon에서 슬레이브의 숫자가 일정 이상보다 늘어나게 되면 전반적으로 성능이 감소됨을 추론할 수 있다.

KOSMOS는 MicroBlade, Moonshot과 비교하였을 때, 전 영역에서 우수하며, 각각 최대 30.3%, 72.3%만큼을 수행시간에서 감소시켰다. 아울러, KOSMOS는 E5-2440, E5-2630보다 각각 1-to-6, 1-to-4 이상에서 오히려 성능이 더 좋음을 알 수 있었고, 1-to-6 기준으로 각각 64.7%, 67.6% 만큼의 수행시간을 감소시켰다. 제약사항으로 인해 6개의 슬레이브까지만 측정하였지만, 6개보다 더 많은 슬레이브를 추가하였을 때는 Xeon 서버에 비해 성능 차이는 더 벌어질 것으로 예상된다.

표 3. Graph Analytics 수행시간 실험결과
Table 3. Experiment results of completion time in Graph Analytics

단위(초)	1-to-1	1-to-2	1-to-4	1-to-6
MicroBlade	1,556,688	730,679	125,583	81,291
Moonshot	4,672,461	2,099,243	286,133	194,483
KOSMOS	1,311,229	581,054	103,872	56,631
E5-2440	715,887	435,385	96,644	160,508
E5-2630	702,140	177,745	126,003	174,636

3. In-Memory Analytics

애플리케이션은 영화 추천 시스템이다. 마스터에서는 rating 행렬, 사용자 및 아이템 벡터를 분리하여, 슬레이브 노드에 분산 전송한다. 슬레이브에서는 지역 행렬 인수 분해(local matrix factorization)를 사용하여 결과를 마스터에 전송한다.

표 4는 In-Memory Analytics의 수행시간 측정 결과이다. 전반적으로 슬레이브 노드의 숫자가 증가할수록 실행시간은 오히려 더 길어진다. 이는 마스터에서 슬레이브 노드에 처리 데이터를 분산 전송할 때, 분산 노드의 숫자가 많아지면 마스터에서의 처리량이 더 가중됨을 알 수 있다. 또한, 슬레이브 노드에 분산되어 실행되는 지역 행렬 인수 분해 연산의 크기는 분산시켜도 크게 성능에는 영향을 미치지 않을 정도로 적은 것으로 판단된다. 표 3의 실험 결과처럼 MicroBlade나 KOSMOS에 비해 Xeon의 실행 시간 증가폭이 더 큼을 알 수 있으며, Xeon E5-2630 대비 KOSMOS는 1-to-4, 1-to-6에서 더 짧은 실행 시간을 나타낸다.

표 4. In-Memory Analytics 수행시간 실험결과
Table 4. Experiment results of completion time in In-Memory Analytics

단위 (밀리초)	1-to-1	1-to-2	1-to-4	1-to-6
MicroBlade	64,870	72,890	90,023	113,068
Moonshot	182,679	197,888	293,033	360,528
KOSMOS	69,000	86,388	121,465	180,388
E5-2440	31,891	48,405	102,237	174,898
E5-2630	38,521	52,344	121,624	216,461

4. Data Serving

NoSQL 분산 데이터베이스에 읽기, 쓰기, 삭제 수행 시의 지연시간과 성능(초당 처리수)을 측정한다. 멀티 노드 환경에서 실행 시에는 단일 클라이언트에서 다수 개의 노드로 이루어진 데이터베이스 서버에 접근한다.

표 5는 Data Serving 실행 후 성능을 측정한 결과를 나타낸다. 전반적으로 데이터베이스 서버 쪽의 노드의 개수, 즉, 데이터베이스 서버를 여러 개 구동시켜도 각 비교군 별로 성능에서는 크게 차이를 보이지 않고, 오히려, Xeon의 경우는 2개 내지 4개의 데이터베이스 서버 노드 개수 이후에는 오히려 성능이 감소됨을 알 수 있다. 이는 클라이언트의 요청 빈도가 일정한 경우, 데이터베이스 서버를 여러 개의 노드로 분리했을 때, 오히려 분리한 노드 간 데이터 전송 등의 이유로 성능이 감소함을 알 수 있다.

KOSMOS는 다른 마이크로서버인 MicroBlade, Moonshot에 비해 각각 최대 36.7%, 370 % 이상의 성능증가가 측정되였으나, Xeon에 비해서는 성능이 떨어진다. 이는 Data Serving 벤치마크 프로그램은 분산처리 개념보다는 데이터베이스 접근 시 개별 쿼리 처리를

측정한다는 측면에서, 개별 노드의 성능이 떨어지는 KOMOS에서 큰 성능을 기대할 수 없기 때문이다.

표 5. Data Serving 성능 측정 실험결과
Table 5. Experiment results of throughput in Data Serving

단위(ops/s)	1-to-1	1-to-2	1-to-4	1-to-6
MicroBlade	198.3	210.4	186.2	207.8
Moonshot	95.4	72.6	64.5	60.4
KOSMOS	344.6	311.3	258.1	284
E5-2440	700.3	839.6	709.2	590.3
E5-2630	584.45	663.13	709.2	639.39

5. Web Serving

Web Serving 벤치마크 프로그램은 클라이언트에서 웹 서버(NginX), 캐시 서버(Memcached), 응용 서버(ELGG), 데이터베이스 서버(MySQL)로 이루어진 서버 단에 서비스 접근 요구를 생성하고 이에 대한 응답 시간을 측정한다. 구체적으로, 응용 서버의 ELGG를 실행 시 SNS 각 내부 단계(BrowsetoElgg, DoLogin, PostSelfWall, SendChatMessage, AddFriend, Register, Logout, UpdateActivity, ReceiveChatMessage)의 지연시간을 측정하여 제시한다. 실험 시에는 클라언트의 숫자를 증가시켰다. 표 6은 클라언트 노드 수에 따른 각 실험 군의 PostSelfWall 단계 평균 응답시간 측정 결과를 나타낸다.

표 6. Web Serving 응답시간 측정 실험결과
Table 6. Experiment results of response time in Web Serving

단위(초)	1-to-1	1-to-2	1-to-4	1-to-6
MicroBlade	0.301	0.316	0.34	0.404
Moonshot	0.663	0.688	0.739	1.174
KOSMOS	0.177	0.178	0.181	0.179
E5-2440	0.162	0.153	0.15	0.141
E5-2630	0.113	0.103	0.104	0.105

표 6에서와 같이 KOSMOS의 경우, 두 마이크로 서버 군인 MicroBlade와 Moonshot 보다 각각 최대 55.7%, 84.8% 더 낮은 지연시간을 나타냈다. 그러나 더 중요한 관찰은 노드의 수가 증가, 즉, 클라이언트의 접속 요구 빈도가 늘어남에 따라, Moonshot과 MicroBloade는 응답시간이 증가하지만, KOSMOS의 경우는 다른 Xeon 서버와 마찬가지로 응답시간의 변화가 거의 없다는 점이다. 이는 실험의 클라이언트 노드 6개보다 더 증가하면 두 마이크로 서버 보다 성능 격차가 더 심해질 수 있음을 예상할 수 있다. 아울러, 서버 단의 각 서버 구성 시 단일 CPU 내에서의 여러 논리 프로세서를 사용하는 것보다 물리적인 다수의 CPU를 연결망으로 사용하는 KOSMOS가 Xeon 서버보다 성능 측면에서 불리한 것으로 분석된다.

V. 결 론

클라우드 서비스 관점에서 마이크로서버로 분류되는 KOSMOS의 클라우드 컴퓨팅 인프라로서의 효용성을 살펴보기 위해 클라우드 서비스로 실제 사용되고 있는 다양한 카테코리의 응용을 내포한 CloudSuite를 벤치마크 프로그램으로 사용하였다.

KOSMOS는 전반적으로 같은 마이크로서버 군인 Moonshot과 MicroBlade 보다 전 영역에서 우수한 성능 나타내었다. 인텔 Xeon CPU 기반 서버와의 비교에서는 KOSMOS가 단일 CPU의 성능이 낮음에도 불구하고, 분산 처리 후 분석을 위한 일부 offline 계열의 응용에서 높은 성능을 나타내었다. offline 벤치마크프로그램인 Graph Analytics에서 KOSMOS는 인텔 E5-2440, E5-2630 Xeon CPU 기반 서버보다 각각 64.7%, 67.6% 만큼의 수행시간을 감소시켰다.

KOSMOS와 같은 마이크로서버 형태의 서버는 클라우드 서비스 실행 시 워크로드가 큰 데이터 분석을 다수의 논드에 분산시켜 실행하는 형태의 응용에서 성능에서 이득이 있을 것으로 예상된다.

References

[1] ETRI develops microserver with 10-fold greater density and extremely low power consumption, https://www.mk.co.kr/news/english/view/2017/08/551867/

[2] Zhang Jingchao, Qiao Liyan and Chen Liqun, “Development of serial RapidIO high-speed data transmission based on VPX bus,” 2019 14th IEEE International Conference on Electronic Measurement & Instruments (ICEMI)
DOI: 10.1109/ICEMI46757.2019.9101789

[3] Myeong-Hoon Oh, “Performance Evaluation of Interconnection Network in Microservers," Journal of the Institute of Internet, Broadcasting and Communication (IIBC), Vol. 21, No. 6, pp. 91-97, Dec. 31, 2021.
DOI: https://doi.org/10.7236/JIIBC.2021.21.6.91

[4] Standard Performance Evaluation Corporation, https://www.spec.org/benchmarks.html

[5] Princeton Application Repository for Shared-Memory Computers (PARSEC), https://parsec.cs.princeton.edu/

[6] TPC Benchmarks Overview, https://www.tpc.org/information/benchmarks5.asp

[7] A Benchmark Suite for Cloud Services, http://cloudsuite.ch/

[8] Unified engine for large-scale data analytic, https://spark.apache.org/

[9] For Creating Scalable Performant Machine Learning Applications, https://mahout.apache.org/

[10] MovieLens, https://grouplens.org/datasets/movielens/

[11] MLlib is Apache Spark's scalable machine learning library, https://spark.apache.org/mllib/

[12] Arthur Szlam, Andrew Tulloch, and Mark Tygert, "Accurate Low-Rank Approximations Via a Few Iterations of Alternating Least Squares," SIAM Journal on Matrix Analysis and Applications, vol. 38, issue 2, pp. 425, 433, Jan. 2017. DOI: https://doi.org/10.1137/16m1064556

[13] GraphX Programming Guide, https://spark.apache.org/docs/latest/graphx-programming-guide.html

[14] What is Memcached?, https://memcached.org/

[15] Open Source NoSQL Database, https://cassandra.apache.org/_/index.html

[16] Yahoo! Cloud Serving Benchmark https://github.com/brianfrankcooper/YCSB

[17] NGINX: Advanced Load Balancer, Web Server, & Reverse Proxy, https://www.nginx.com/

[18] Introducing a powerful open source social networking engine, https://elgg.org/

[19] Helping measure performance, http://faban.org/

[20] Welcome to Apache Solr - Apache Solr https://solr.apache.org/

[21] HP Mootshot, https://www.hpe.com/us/en/servers/moonshot.html

[22] Supermicro MicroBlade, https://www.supermicro.com/en/products/microblade/

[23] Intel® Xeon® Processor E5 v3 Family, https://ark.intel.com/content/www/us/en/ark/products/series/78583/intel-xeon-processor-e5-v3-family.html

[24] RapidIO Specifications, https://rapidio.org/rapidio-specifications/

[25] Hyokyung Bahn, "Real-time Task Aware Memory Allocation Techniques for Heterogeneous Mobile Multitasking Environments," Journal of the Institute of Internet, Broadcasting and Communication (IIBC), Vol. 22, No. 3, pp. 43-48, June 30, 2022. DOI: https://doi.org/10.7236/JIIBC.2022.22.3.43

[26] Swarm mode overview, https://docs.docker.com/engine/swarm/

저 자 소 개

오 명 훈(정회원)

- 2005년 2월 : 광주과학기술원 정보통신공학과 공학박사
- 2005년 4월 ~ 2021년 2월 : 한국전자통신연구원 책임연구원
- 2021년 2월 ~ 현재 : 호남대학교 컴퓨터공학과 조교수
- 주관심분야 : 임베디드시스템, 고성능 컴퓨터구조, 클라우드컴퓨팅

※ 본 논문은 2022년 호남대학교 연구과제 지원으로 연구되었음