쿠버네티스 환경에서 파드 사이의 CPU 자원 경쟁에 따른 응답시간 분석

한승주 [1], 김동균 [1], 유헌창 [1]
[1] 고려대학교 정보대학 컴퓨터학과

{andy030407, kdonggyun97, yuhc}@korea.ac.kr

An Analysis of Response Time Depending on CPU Resource Contention between Pods in Kubernetes Environment

Seungjoo Han[1], Donggyun Kim[1], Heonchang Yu[1]
[1]Dept. of Computer Science and Engineering, Korea University

요 약

클라우드 서비스는 자원 효율성을 위해 서로 다른 애플리케이션들이 같은 물리적 자원을 공유하며 제공되고 이로 인해 자원 경쟁이 발생할 수 있다. CPU 자원 경쟁이 발생하는 경우 병목 현상과 캐시 상태 변화로 사용자는 느린 응답속도를 겪게 된다. 본 논문에서는 쿠버네티스 환경에서 CPU 자원 경쟁에 따른 타깃 애플리케이션의 성능을 측정하여 그 영향을 분석한다. 실험 결과 애플리케이션의 응답시간은 노드의 CPU 사용량이 늘어날수록 평균 361%, 최대 855% 증가하고, 작업 수가 많아질수록 최대 19.4% 증가한다. 이 결과를 바탕으로 자원 효율성과 파드 성능을 고려한 가이드라인을 제공하는 것이 목적이다.

1. 서론

클라우드 서비스의 성장과 대중화로 클라우드 공급자는 다양한 서비스를 제공하게 되었다. 쿠버네티스는 클라우드 서비스를 위한 컨테이너 오케스트레이션 플랫폼으로, 복잡한 클라우드 환경에서 컨테이너 애플리케이션을 효율적으로 배포하고 관리하는 데 핵심적인 도구로 사용된다.

클라우드 환경에서 자원의 활용률을 높이는 것은 매우 중요한 문제다. 그러나, 서비스를 제공하는 구글의 production cluster 와 Amazon AWS EC2 의 평균 CPU 활용률은 각각 20%와 7%~17% 수준으로 낮게 나타났다[1-2]. 자원 효율성 문제를 해결하기 위해 업계와 학계에서는 여러 애플리케이션이 동일한 물리적 자원을 공유하며 실행하는 연구가 진행됐다.

자원 공유 환경에서 일부 인스턴스는 실제로 요청한 것보다 더 많은 자원을 사용하는 경우가 발생한다. 이로 인하여 자원 경쟁이 발생하고, 개별 서비스의 성능이 저하되는 문제가 나타난다[3].

본 논문에서는 쿠버네티스 환경에서 파드 간 CPU 자원 경쟁에 따른 타깃 애플리케이션의 성능을 분석한다.

2. 쿠버네티스 파드 간 자원 경쟁

쿠버네티스는 클러스터를 관리하는 마스터 노드와 애플리케이션을 실행하는 하나 이상의 워커 노드로 구성된다. 애플리케이션은 쿠버네티스의 가장 작은 논리적 단위인 파드로 배포되어 스케줄러에 의해 자원이 충분한 노드에 배치된다. 그러나 파드가 사용하는 노드의 자원은 실시간으로 변화하며, 파드에 많은 요청이 발생하면 그에 따른 많은 자원을 점유하게 된다. 하나의 노드에 여러 개의 파드가 존재할 경우, 파드 간에는 한정된 자원을 두고 경쟁이 발생할 수 있다. 만약 특정 파드가 노드의 자원 대부분을 점유하고 있다면, 남은 가용 자원이 제한됨에 따라 다른 파드에 영향을 주어 작업 시간이 증가한다. 따라서 노드의 CPU 사용 현황에 따른 타깃 애플리케이션의 성능이 어떻게 변화하는지 비교 분석한다.

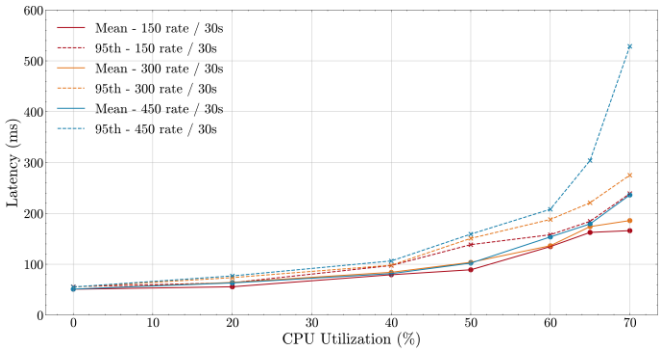
3. CPU 자원 경쟁에 따른 응답시간 분석

3.1 실험 환경

본 실험에서는 Intel® Core™ I5-12600K CPU 와 DDR5 32GiB 메모리로 구성된 시스템 환경을 사용했다. 통제된 환경을 위해 가상머신을 구성하여 쿠버네티스 클러스터 환경을 구축했다. 클러스터는 마스터 노드 1 개와 워커 노드 3 개로 구성하였으며, 각각 2 개의 vCPU 와 4GiB 의 RAM 을 할당하였다.

노드의 고정된 자원 점유 및 부하 환경을 구성하기 위해 progrium/stress 이미지를 사용하여 노드의 CPU 점유율을 설정했다. 타깃 애플리케이션은 동일 노드에 삼각함수 계산을 수행하는 php-apache 를 사용하고, 애플리케이션의 서비스 응답시간 측정을 위해 클러스터 외부에서 HTTP 요청을 보냈다.

3.2 CPU 사용률에 따른 응답시간 분석



[그림 1] CPU 사용률에 따른 평균 및 95 번째 응답시간

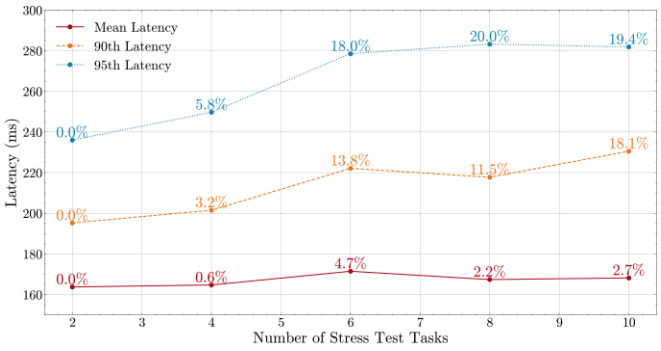
실험은 노드에 타깃 애플리케이션만 존재할 때와 stress 애플리케이션이 CPU 를 각각 20%, 40%, 50%, 60%, 65%, 70%를 점유하고 2 개의 작업을 수행할 때, 부하 생성기를 통해 30 초 동안 150, 300, 450 번의 요청을 타깃 애플리케이션으로 보내 응답시간을 측정하였다. [그림 1]과 [표 1]은 측정 결괏값의 평균과 95 번째 값을 나타낸다. 노드에 CPU 가용 자원이 줄어들수록 응답시간이 전체적으로 증가한다. CPU 활용률이 60% 이상인 경우 요청량이 증가할수록 응답시간이 길어지는 경향을 보인다. Stress 애플리케이션으로 인해 타깃 애플리케이션이 사용 가능한 CPU 자원이 점점 줄어들고, 요청량이 증가함에 따라 작업 요청이 레디 큐에 병목되어 처리 시간이 길어지기 때문이다.

<표 1> CPU 사용률에 따른 평균 응답시간 (ms)

CPU / Rate	0%	20%	40%	50%	60%	65%	70%
150	50 (0%)	55 (9%)	79 (55%)	89 (74%)	134 (164%)	162 (219%)	166 (226%)
300	51 (0%)	64 (25%)	84 (64%)	103 (103%)	136 (166%)	174 (240%)	185 (263%)
450	51 (0%)	63 (23%)	81 (59%)	102 (99%)	153 (200%)	179 (250%)	236 (361%)

3.3 CPU 작업 수에 따른 응답시간 분석

다음으로, stress 애플리케이션이 65%로 동일한 비율의 CPU 를 점유하도록 하고 CPU 부하 작업이 2, 4, 6, 8, 10 개일 때 30 초간 400 개의 요청을 타깃 애플리케이션으로 보낼 때의 서비스 응답시간을 비교하였다. [그림 2]는 평균값과 90, 95 번째 응답시간을 나타낸다. CPU 작업 수가 증가함에 따라 평균 응답시간은 변화가 크지 않지만, 90, 95 번째 응답시간은 상승하는 모습을 보인다. 사용 가능한 CPU 자원은 동일함에도, 프로세스가 많아지면 이들을 전환하는 과정에서 캐시와 TLB 의 상태가 변화하여 context switch overhead 가 증가한다[4]. 이로 인해 일부 요청들이 레디 큐에서 CPU 에 할당되기까지 대기시간이 발생하여 처리 시간의 증가로 응답시간에 영향을 미친 것으로 추정된다.



[그림 2] CPU 작업 개수에 따른 응답시간

4. 결론

본 논문에서는 노드의 CPU 자원 사용량과 작업 개수에 따른 타깃 애플리케이션의 성능에 미치는 영향을 분석하였다. 다른 파드의 CPU 자원 점유율 증가 및 노드의 작업 개수가 증가할수록 응답시간도 함께 증가하였다. 이러한 결과는 각각 가용 가능한 자원이 줄어들어 병목현상이 발생하고, 캐시와 TLB 상태 변화로 인해 응답시간이 늘어난 것으로 보인다. 해당 실험으로 노드의 CPU 활용 정도에 따라 파드의 성능에 영향을 미치는 것을 확인하였다. 이를 바탕으로 노드의 자원 효율성과 파드의 성능을 모두 고려한 자원 배치 방안을 찾을 수 있을 것으로 기대한다.

Acknowledgement

본 연구는 2024 년 과학기술정보통신부 및 정보통신기획평가원의 SW 중심대학사업 지원을 받아 수행되었음(2023-0-00044)

참고문헌

[1] REISS, Charles, et al. Heterogeneity and dynamicity of clouds at scale: Google trace analysis. In: *Proceedings of the third ACM symposium on cloud computing*. 2012. p. 1-13.

[2] LIU, Huan. A measurement study of server utilization in public clouds. In: *2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing*. IEEE, 2011. p. 435-442.

[3] GUO, Jing, et al. Who limits the resource efficiency of my datacenter: An analysis of alibaba datacenter traces. In: *Proceedings of the international symposium on quality of service*. 2019. p. 1-10.

[4] TSAFRIR, Dan. The context-switch overhead inflicted by hardware interrupts (and the enigma of do-nothing loops). In: Proceedings of the 2007 workshop on Experimental computer science. 2007. p. 4-es.