동적 SLA 기반 오토-스케일링 엔진

김기현*, 김호승*, 손무열**, 정창성*
*고려대학교 전기전자전파공학부
**Hanwha Corporation
e-mail:cocoball@korea.ac.kr

Dynamic SLA based Auto-Scaling Engine

Ki-Hyun Kim*, Ho-Seung Kim*, Moo-yeol Son** Chang-Sung Jeong*
*Dept of Electrical Engineering, Korea University
**Commercial explosive business division, Hanwha Corporation

요 약

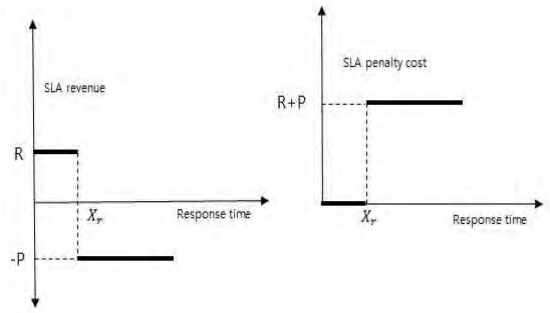
계산 과학 분야에서 자원을 필요한 만큼 빌려 쓸 수 있는 클라우드 기술을 적용하여 과학 클라우드(Science Cloud)를 구축하는 연구가 활발해지고 있다. 특히 의료 분야에서는 3D 볼륨 렌더링과 같은 고성능의 자원을 활용한 대규모 작업 계산 응용이 있다. 이러한 응용의 성공적인 작업 수행과 실시간으로 변화하는 자원 수요에 대처하여 클라우드 자원을 효율적으로 관리하기 위한 오토 스케일링 엔진 개발이 필요하다. 그러나 대부분의 오토 스케일링 엔진은 단순한 하드웨어의 성능을 기반으로 제공되고 있어 클라이언트에 따라 부하를 고려해야한다. 본 논문에는 클라이언트에 따라 가중치를 적용한 동적인 SLA 기반으로 자원 수요를 예측하고 클라우드 자원을 효율적으로 관리하는 오토-스케일링 엔진을 제안한다.

1. 서론

최근 계산 과학 분야에서 자원을 필요한 만큼 빌려 쓸 수 있는 클라우드 기술을 적용하여 과학 클라우드를 구축하는 연구가 세계 여러 곳에서 진행되고 있다. 특히 의료 분야에서는 PACS와 같이 대규모 협업을 필요로 하는 협업 클라우드의 수요도 증가하고 있다. 대규모 작업 계산 응용은 장시간 동안 고성능의 자원을 활용해야 하며 안정적이고 지속적인 자원의 공급이 필수적이다. 이러한 협업 환경에서 성공적인 작업 수행을 위한 실행관리, 자원예측, 자원관리 등을 제공하는 오토-스케일링 엔진 개발이 필요하다. 선행 연구 [1]에서 우리는 사용자가 입력한 SLA 기반으로 동적으로 자원을 관리하는 오토-스케일링 엔진을 제안하였다. 하지만 사용자가 작성한 정적 SLA는 자원 남용이 크고, 자원 수요를 예측하여 미리 자원 생성이 힘들었다. 또한, 클라이언트에 따라 작업 부하 발생 정도가 다르다. 이를 개선하기 위해 클라이언트에 따라 가중치를 적용한 동적인 SLA 기반 오토 스케일링 엔진의 우수성을 보이고자 한다.

2. 동적 SLA를 위한 SLA cost

제안한 동적인 SLA 기반 오토 스케일링 기법은 조건을 위반할 때마다 페널티가 발생한다. 예를 들어 Response time을 기준으로 아래 그림1과 같이 페널티 비용 발생하게 된다. 그림1의 왼쪽 그림에서 요청의 반응시간인 r와 X_r를 비교하여 반응시간이 X_r이하면 SLA 수익 비용 (Revenue cost) 발생하고, 반응시간이 X_r이상으로 넘어가면 벌칙 비용 (Penalty cost) 이 발생하게 된다. 그리고 그림1의 오른쪽과 같이 SLA 수익 비용이 X_r이상이면 벌칙 비용이 발생한다. R+P를 1로 가정하면 다음과 같이 그림1을 수식1로 정의할 수 있다.



(그림 1) SLA revenue (왼쪽), SLA penalty cost (오른쪽)

$$P(q)=\begin{cases}0 & \text{if } r_{time} \le X_r \\ 1 & otherwise\end{cases} \quad (1)$$

또한 전체 요청 개수 L에 따른 평균 SLA 벌칙 비용 AC는 다음과 같은 수식2로 정의한다.

$$AC=\frac{1}{L}\sum_{r}P(r) \quad (2)$$

예를 들어 클라이언트가 10개의 request를 요청해서 1

개의 응답을 놓쳤을 때, SLA 벌칙 비용은 0.1이 된다. 가중치 벌칙 비용 함수는 다음과 같이 수식3으로 정의한다.

$$P_w(r,i) = P(r,i) \times w(i) \qquad (3)$$

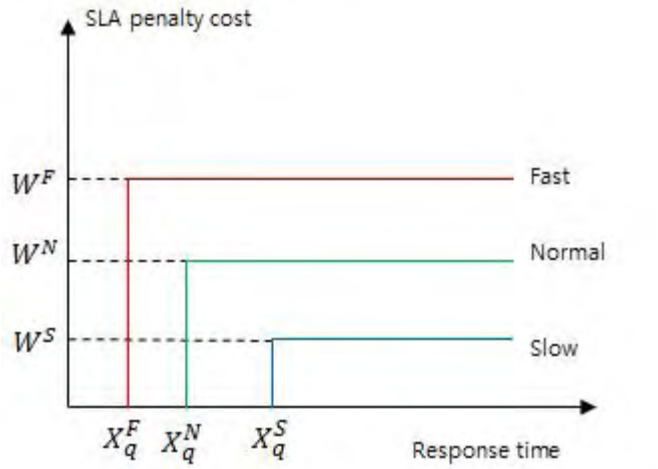
t개의 간격이내에서 평균 SLA 벌칙 비용은 수식4와 같이 정의한다. 여기서 $L(i,t)$는 전체 SLA 벌칙 비용으로 정의한다.

$$AC(i,t) = \frac{1}{L(i,t)} \sum_{T\times(t-1) \le r_s \le T\times t} P(r,i) \qquad (4)$$

t개의 간격에서 SLA 벌칙 비용은 수식5로 정의한다.

$$SLA(t) = \sum_{i=1}^{N} \sum_{T\times(t-1) \le r_s \le T\times t} P_w(r,i)$$
$$= \sum_{i=1}^{N} AC(i,t) \times L(i,t) \times w(i) \qquad (5)$$

그림2와 같이 전체 클라이언트를 반응시간에 따라 Fast, Normal, Slow로 나누고 반응시간에 따라 가중치를 주어 오토-스케일링의 SLA에 반영한다. 시간 간격에 따라 동적 SLA는 최적의 자원 수요 예측 값을 계산한다.



(그림 2) 가중치 SLA 페널티 비용

이전 시간 간격이 끝나고 그림 3과 같이 동적 SLA는 다음 최적 방향을 결정하게 된다. 그림과 같이 최적 방향을 결정하기 위해 3-D 배열인 $d_i = [d_{i1}, d_{i2}, d_{i3}]$을 사용한다. t번째 간격에서 우리는 CPU, memory, workload를 각각 수식8로 정의한다. 표1를 참조해서 보면 다음 방향을 N, 간격을 t로 정의하고 새로 추가되는 자원은 $V(t)$라 정의한다. 수식6은 다음 방향 N을 정의한다.

$$N = \arg_D \min SLA(t) \qquad (6)$$

수식6에서 최적의 다음 방향 값으로 자원 수요를 예측하기 위해 수식7에서 SLA(t) 값을 정의한다. 이 값을 이용하여 다음 자원 수요를 예측하고 오토 스케일링을 한다. 그리고 간격 t동안 i번째 클라이언트의 평균 SLA 비용은 수식8을 사용하여 계산한다.

$$\text{SLA(t)} = \sum_{i=1}^{N} AC(i,t) \times L(i,t) \times w(i) \qquad (7)$$

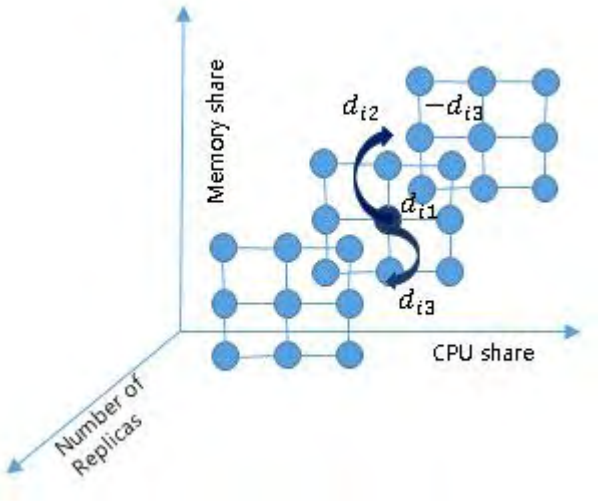
$$AC(i,t) = f_i(cpu(i,t-1) + d_{i1} mem(i,t-1) + d_{i2,} V(t-1), L(i,t)) \qquad (8)$$

$$\sum_{i=1}^{N} d_{i1} = 0 \quad \sum_{i=1}^{N} d_{i2} = 0 \qquad (9)$$

$$D_3 = d_{13} = d_{23} = \ldots = d_{N3} = 0 \qquad (10)$$

<표 1> 표기 및 정의

표기	정의
t	Interval
T	The length of the interval
N	Type of clients
$V(t)$	The number of VM
r	Request
r_s	The start time of the request
$P(r)$	SLA penalty cost function for request r
$w(i)$	Weight for the i-th class of client
$P_w(r,i)$	Weight SLA penalty cost for the i-th class of clients
$L(i,t)$	The total number of requests for the i-th class of clients during the k-th interval
$CPU(i,t)$	The CPU shares for the i-th class of clients during the k-th interval
$mem(i,t)$	The memory shares for the i-th class of clients during the t-th interval
$AC(i,t)$	The average SLA cost for the i-th class of clients during the t-th interval

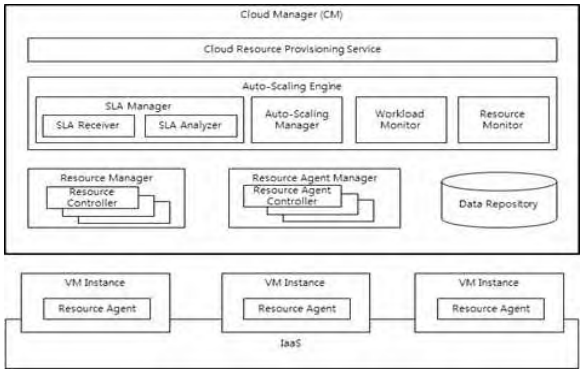


(그림 3) 탐색 방향

i번째 클라이언트의 CPU와 memory 활용률을 $cpu(i,t-1), mem(i,t-1)$라 정의한다. 예를 들어 공유하는 CPU 활용률이 20이고, 1024MB의 memory를 사용하고, 방향이 $d_{i1}=10, d_{i2}=-100$ 이라면 평균 SLA 비용 측면에서 CPU 활용률 30과 메모리 사용률 924MB가 된다.

3. 오토 스케일링 엔진 구조

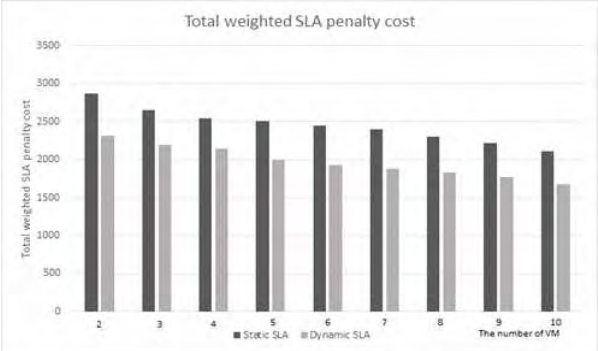
2장에서 제안한 오토 스케일링을 구동시키기 위한 오토 스케일링 구조는 그림4와 같다. 본 논문에서 제안한 오토 스케일링 엔진은 SLA Manager, Auto-Scaling Manager, Workload Monitor, Resource Monitor로 구성된다. SLA Manager는 SLA를 관리하는 역할을 한다. SLA는 크게 사용자가 직접 입력한 정적 SLA와, 2장에서 제안한 동적 SLA가 있다. Workload Monitor는 반응 시간 정보를 모니터링 한다. Resource Monitor는 CPU, Memory share를 모니터링 한다. 이와 같이 모니터링으로 수집된 정보와, SLA 조건을 비교하여 최종적으로 오토 스케일링을 판단하는 모듈이 Auto-Scaling Manager이다. 가상머신 상에는 Resource Agent가 구동되어 모니터링 정보와, 프로세스 정보를 수집하고 전달하는 역할을 한다. 이를 통해 클라우드 자원 관리의 효율을 증진시키고, 비용을 절감할 수 있다.



(그림 4) 오토 스케일링 엔진 구조

4. 성능평가

2장에서 제안한 동적 SLA의 성능을 평가한다. 그림 5는 사용자가 입력한 정적 SLA와, 2장에서 제안한 모델링으로 구동되는 동적 SLA의 전체 가중치 SLA 페널티 비용을 비교한 결과이다. SLA 페널티 비용이 높을수록 클라우드 자원 남용이 심하다. x축이 가상머신 개수인데 개수를 증가시킬수록 페널티 비용이 감소하고 전반적으로 동적 SLA가 성능이 뛰어난 것을 보여준다. 이를 통해 우리가 제안한 동적 SLA기반 오토 스케일링의 우수성을 검증한다.



(그림 5) SLA 페널티 비용 비교

5. 결론

본 논문에서는 클라이언트 별로 가중치를 적용한 동적 SLA기반 오토 스케일링 엔진을 제안하였다. 성능 평가를 통해 사용자가 직접 입력하는 정적 SLA와 2장에서 제안한 모델링을 적용한 동적 SLA의 성능을 비교하여 동적 SLA가 클라우드 자원 사용 비용을 크게 절감시켜주는 것을 증명하였다. 제안한 오토 스케일링 엔진을 통해 과학 클라우드나 의료 협업 클라우드에 클라우드 자원 사용을 증진시키고 비용을 감소 시킬 수 있음을 확인하였다.

향후에는 클라이언트 특성 뿐 만 아니라, 응용의 특성을 고려한 의미론적인 스케일링과 저전력 컴퓨팅 기술을 활용한 에코 스케일링 기법을 추가할 예정이다.

Acknowledgement

본 연구는 미래창조과학부 및 정보통신기술진흥센터의 대학ICT연구센터육성 지원사업 (IITP-2015-H8501-15-1004)과, 2015년도 정부 (미래창조과학부)의 재원으로 한국연구재단-차세대정보컴퓨팅기술개발사업 (NRF-2014039205)과, 문화체육관광부 및 한국콘텐츠진흥원의 2012년도 문화콘텐츠산업기술지원사업의 연구결과로 수행되었음 (R2012030096)

참고문헌

[1] 김기현, 정창성, “클라우드 기반 협업 플랫폼을 위한 SLA기반 오토-스케일링 엔진”, 대한전자공학회 하계학술대회 제37권 1호, 2014.

[2] Ki-Hyun Kim and Chang-sung Jeong, “Adaptive Resource Management Platform”, Jokull journal, Vol 64, No 11, Nov 2014

[3] Ki-Hyun Kim and Chang-sung Jeong, “Cloud based Medical Collaboration Platform”, Jokull journal, Vol 64, No 11, Nov 2014

[4] M. Mao and M. Humphrey, “Auto-scaling to minimize cost and meet application deadlines in cloud workflows”, High Performance Computing, Networking, Storage and Analysis (SC), 2011 International Conference for, Nov, 12-18, 2011.

[5] Rodrigo N., Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, “Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms”, Software: Practice and Experience, 41-1, pp. 23-50, 2011.