KIPS Tr. Comp. and Comm. Sys.
Vol.4, No.11 pp.369~382 pISSN: 2287-5891

http://dx.doi.org/10.3745/KTCCS.2015.4.11.369

Performance Improvement of an Energy Efficient Cluster Management Based on Autonomous Learning

Sungchul Cho† · Kyusik Chung††

ABSTRACT

Energy aware server clusters aim to reduce power consumption at maximum while keeping QoS(quality of service) compared to energy non-aware server clusters. They adjust the power mode of each server in a fixed or variable time interval to activate only the minimum number of servers needed to handle current user requests. Previous studies on energy aware server cluster put efforts to reduce power consumption or heat dissipation, but they do not consider energy efficiency well. In this paper, we propose an energy efficient cluster management method to improve not only performance per watt but also QoS of the existing server power mode control method based on autonomous learning. Our proposed method is to adjust server power mode based on a hybrid approach of autonomous learning method with multi level thresholds and power consumption prediction method. Autonomous learning method with multi level thresholds is applied under normal load situation whereas power consumption prediction method is applied under abnormal load situation. The decision on whether current load is normal or abnormal depends on the ratio of the number of current user requests over the average number of user requests during recent past few minutes. Also, a dynamic shutdown method is additionally applied to shorten the time delay to make servers off. We performed experiments with a cluster of 16 servers using three different kinds of load patterns. The multi-threshold based learning method with prediction and dynamic shutdown shows the best result in terms of normalized QoS and performance per watt (valid responses). For banking load pattern, real load pattern, and virtual load pattern, the numbers of good response per watt in the proposed method increase by 1.66%, 2.9% and 3.84%, respectively, whereas QoS in the proposed method increase by 0.45%, 1.33% and 8.82%, respectively, compared to those in the existing autonomous learning method with single level threshold.

Keywords : Power Mode Control, QoS, Power Consumption, Autonomous Learning, Prediction Algorithm, Hybrid

자율학습기반의 에너지 효율적인 클러스터 관리에서의 성능 개선

조 성 철† · 정 규 식††

요 약

에너지 절감형 서버 클러스터에서는 에너지 절감을 고려하지 않는 기존 서버 클러스터에 비해 서비스 품질을 보장하면서 전력소비를 절감하는 것을 목표로 하며, 현재의 부하를 처리하는 데 필요한 최소수의 서버들만 ON 하도록 고정 주기 또는 가변 주기로 서버들의 전원 모드를 조정한다. 이에 대한 기존 연구들은 전력을 절감하거나 열을 낮추는데 노력해왔지만 에너지 효율성을 잘 고려하지 못했다. 본 논문에서는 기존 자율학습기반의 서버 전원 모드 제어 방법의 단위전력당 성능과 QoS를 높이기 위한 에너지 효율적인 클러스터 관리기법을 제안한다. 제안 방법은 다중임계기반의 자율학습 방법과 전력소모 예측 방법을 결합한 서버 전원 모드 제어이다. 일반적인 부하 상황에서는 다중임계 학습기반의 서버 전원 모드 제어를 적용하고, 급변하는 부하 상황에서는 예측기반의 서버 전원 모드 제어가 적용된다. 일반적 상황과 급변하는 상황의 구별은 현재의 사용자 요청과 관찰된 과거 몇 분의 사용자 요청의 비율에 따라 이루어진다. 또한, 동적종료 기법을 추가로 적용해 서버가 OFF 하는 데 소요되는 시간을 단축한다. 제안 방법은 16대 서버로 구성된 클러스터 환경에서 3가지 부하 패턴을 이용하여 실험을 수행한다. 다중임계 학습, 예측, 동적종료를 함께 이용한 실험에서 단위전력당 성능(유효응답 수)과 표준화된 QoS 측면에서 가장 우수한 결과를 보여준다. 제안하는 방법과 파라미터 로드된 단일임계 학습을 비교할 때 뱅킹 부하패턴, 실제 부하패턴, 가상 부하패턴에서 단위전력당 유효응답 수가 각각 1.66%, 2.9%, 3.84% 향상되고, QoS 관점에서는 각각 0.45%, 1.33%, 8.82% 향상되었다.

키워드 : 전원 모드 제어, QoS, 소비전력, 자율학습, 예측 알고리즘, 결합

※ 이 논문은 2012년도 정부(교육과학기술부)의 재원으로 한국연구재단의 기초연구사업 지원을 받아 수행된 것임(NRF-2012R1A1A2006602).
† 정 회 원 : 숭실대학교 전자공학과 박사
†† 정 회 원 : 숭실대학교 스마트시스템소프트웨어학과 교수
Manuscript Received : July 6, 2015
Accepted : August 27, 2015
* Corresponding Author : Kyusik Chung(kchung@q.ssu.ac.kr)

1. 서 론

데이터 센터에서 소비되는 에너지는 IT 장비에서 52%, 쿨링 장비에서 40%를 소비하고, 나머지 부분을 기타 장비에서 소비한다. 이를 간략히 표현하면 $P_{DC} \fallingdotseq P_{cool} + P_{IT}$ 로 나타낼 수 있다. 즉, 전체 소비전력은 IT 장비가 소비하는 전력과 장비를 냉각하는 데 필요한 전력의 합으로 표현하고 이 부분에서 대부분을 소비하고 있다[1]. 본 논문은 IT 장비에서 소비되는 에너지 효율을 개선하는 부분을 다룬다.

에너지 소비 문제를 해결하기 위한 에너지 절감형 서버 클러스터에서는 에너지 절감을 고려하지 않는 기존 서버 클러스터에 비해 서비스 품질을 보장하면서 전력소비를 절감하는 것을 목표로 한다.

데이터 센터에서 에너지 소모 비중이 가장 높은 서버들에서 에너지 소모를 줄이는 방법들로는 1) 에너지 절감형 서버 하드웨어를 사용하는 방법[2], 2) 멀티코어/멀티프로세서를 사용하는 서버를 사용하는 경우 한 서버 내 CPU 전력 소모를 최소화하도록 서버 운영체제에서 에너지 절감형 스케줄링을 사용하는 방법[3], 3) 서버 클러스터에서 부하 상황에 따라 필요한 수만큼의 서버들만 ON 하고 나머지는 OFF 하는 에너지 절감형 서버클러스터 관리기법[4], 4) 이종 혼합 서버를 구축하여 에너지 절감형 서버클러스터를 구축하는 방법[5] 등이 있다.

Abdul Hameed 연구팀[6]은 에너지 자원 할당기술 측면에서 에너지 효율적 데이터 센터 구축에 대한 분류를 제시하고 있다. 예측(predictive)에 의한 방법, 자원의 상호 반응(reactive)에 따라 결정하는 방법, 예측과 상호 반응에 따른 결정 방법을 결합(hybrid)해 사용하는 방법으로 구분한다. 그리고 할당기술에 따라 전력 절감(power-aware)과 열 절감(thermal-aware)으로 분류하고, 분배동작에 따라 서비스 마이그레이션과 서비스 종료로 분류한다. 또한 상호운영 방식에 따라 동종운용과 이종운용으로 분류한다.

본 논문에서는 서버 클러스터환경에서 에너지 효율적인 서버 클러스터 관리기법을 연구한다. 조건부 선택은 자율학습기반의 전원 모드 제어 기법과 예측기반의 전원 모드 제어 기법을 선택하는 방법이다. 결합된 알고리즘은 일반적인 트래픽 변화에 다중임계 학습을 선택하여 단위전력당 유효응답 수를 높이는 데 이용하고, spike성 워크로드에서는 예측을 선택하여 더 많은 사용자 요청을 처리하는 데 이용한다. 또한, 조건부 선택기법에 동적종료를 함께 이용하여 서버를 빠르게 OFF 하고, 소비전력을 낮추는 역할을 한다. 위 방법을 이용한 결합은 QoS는 유지하면서 단위전력당 유효응답 수를 높이도록 하는 방법이다.

본 논문의 구성은 다음과 같다. 2절에서는 에너지 절약을 위한 서버 전원 모드 제어에서 기존 방법에 대한 연구를 소개한다. 3장에서는 제안하는 방법인 자율학습과 예측의 조건부 선택을 통한 서버 전원 모드 제어 방법을 소개한다. 4절에서는 실험 및 토론을, 마지막으로 5절에서는 결론 및 향후 연구 방향을 제시한다.

2. 연구 배경

2.1 서버 클러스터링 시스템

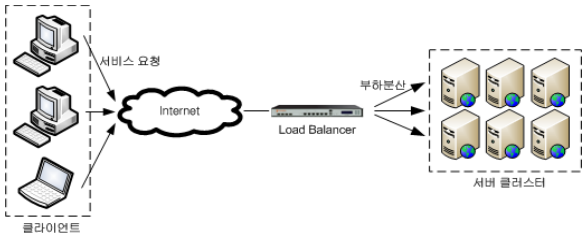


Fig. 1. Server Cluster Environment

Fig. 1은 클러스터링 환경을 나타내는 그림이다. 외부의 클라이언트가 부하분산기를 거쳐 서버 클러스터에 서비스를 요청되게 되는데 얼마나 많은 서비스 요청 트래픽이 요구될지 알 수 없으므로 미리 용량계획이 이루어져야 한다. 부하분산기의 용량계획은 동적으로 클러스터의 서버 수를 조절해야 하므로 제안하는 시스템에서는 이 부분을 동적 서버 프로비저너라 부른다. 여기의 용량계획기반의 동적 서버 프로비지닝은 요청 트래픽 양에 대해 서버당 처리능력으로 나누어 필요한 서버 대수를 결정한 뒤 해당 서버는 ON 하고 나머지 서버들은 OFF 한다.

일반적인 부하분산기는 가용성을 높이고, 클러스터 확장과 처리용량을 높이며, 서비스 요청에 대한 빠른 응답을 제공하고, 추가 기능으로 SSL offloading, DoS 방어 등을 제공하고 있다.

2.2 기존 에너지 절감형 연구

에너지 절감형 알고리즘의 핵심은 필요한 서버 대수를 어떻게 결정하느냐에 달려있다. 서버 대수 결정 알고리즘의 유형으로는 일정 주기마다 서버모드를 정적으로 제어하는 방법, 사용자요청 트래픽 변화를 예측하는 방법, 학습결과를 통해 제어하는 방법 등이 있다. 학습은 과거의 정보를 기반으로 운영에 필요한 최적값을 얻는 방식으로, 고정된 임계치를 사용하는 단일임계치 방법과 변경 가능한 다중임계치로 구분할 수 있다. 필요한 서버 대수가 결정되면 나머지 서버들을 OFF 하게 되는데 OFF 하는 방식으로 서버와 클라이언트 간 연결 수가 0이 될 때까지 수동적으로 기다리는 방법과 능동적으로 서버 OFF 시키는 방법으로 구분할 수 있다. 이에 대한 연구사례는 아래와 같다.

1) 서버 클러스터 환경에서 자율학습기반의 에너지 효율적인 클러스터 관리 기법[8]

본 연구팀이 수행한 방법으로 기존 용량계획기반의 동적 서버 프로비저닝 대신에 학습기반으로 최적의 서버 대수를 결정한다. 기존 용량계획기반에서는 클러스터에 입력되는 워크로드(작업부하) 패턴에 무관하게 동작하는 반면에, 이 방법에서는 입력되는 워크로드 패턴을 분류하고 그에 따라

최적의 서버 대수를 결정할 수 있게 하는 학습 방법을 적용하며, 고정된 단일임계치을 사용한다. 다중임계치가 아닌 단일임계치를 이용한 이유는 최소 단위의 고정 제어 주기로 서버모드 제어를 실험한 경우와 성능 비교를 하기 위해서이다.

워크로드 패턴은 각 서버의 네트워크 트래픽 및 cpu 부하값들에 의해 결정되는데 이 연구에서는 워크로드 패턴을 트래픽 및 cpu 부하값들의 표준편차들에 의해 분류한다. 여기에서 표준편차를 사용하는 의미는 클러스터 환경에서 각 서버들이 처리하고 있는 워크로드 패턴이 평균에서 얼마나 떨어져 있는지의 정도를 가지고 워크로드 패턴을 분류한다는 뜻이다. 각 서버의 현재 트래픽값과 cpu 부하값들이 주어지면 트래픽값 표준편차와 cpu 부하값 표준편차를 구한 뒤, 이들 정규화된 값들(각각을 con, cpu 특성값이라고 부름)로 서버 클러스터의 현재 워크로드 패턴을 규정한다.

실시간으로 측정되는 cpu 사용률 및 con 사용률을 가지고, cpu 사용률에 대해 가중치(이를 상태 가중치(ratio)로 부름)를 곱하고 con 사용률에 (1 - 상태 가중치)를 곱하여 얻는, 각 사용률에 대한 가중치를 결합한 값으로 총 이용률(utilization)을 얻는다.

$$total\,utilization = \sum_{i=0}^{n-1} (1-ratio) \times con[i] + ratio \times cpu[i],$$
$$i = 0, \cdots, n-1$$
$$server\,count = \frac{total\,utilization}{threshold} \quad (1)$$

전체 클러스터를 구성하는 각 서버의 총 이용률을 구하고, 단일임계치로 나누어 필요한 서버 대수를 결정한다. 이 논문에서는 현재 주어진 워크로드 패턴에 대해 서버 클러스터 전체 소비전력당 서버 클러스터의 성능을 최대로 할 수 있는 최적 서버 대수를 결정하기 위해 학습 방법을 적용한다. 런타임에 트래픽 및 cpu 부하에 대한 상태가중치를 얻어 최적 서버 대수를 찾는다.

일단 학습이 한번 수행되면 그 학습 결과(트래픽 및 cpu 부하에 대한 상태가중치 및 서버의 워크로드 임계값)를 테이블에 기록하여 그다음부터는 동일한 워크로드 패턴이 나타나면 기록된 학습 결과를 이용하여 서버 프로비저닝이 동작한다. 자세한 내용은 논문[8]을 참고하기 바란다.

2) 에너지 절감형 서버 클러스터에서 급변하는 부하처리를 위한 다중임계치기반의 서버 전원 모드 제어[9]

에너지 절감형 서버 클러스터에 관한 기존 연구에서는 현재의 사용자 요청을 처리하는 데 필요한 최소의 서버 대수를 계산하여 해당 서버만을 활성화하도록 서버 전원 모드를 일정주기마다 제어한다. 부하가 급격하게 변하는 상황에서는 서버 수를 빨리 증가하지 못하기 때문에 기존 연구에서는 QoS가 떨어진다. 이 문제를 해결하기 위해, 본 연구팀이 수행한 이 논문에서는 부하추세를 급증, 증가, 완만, 감소, 급감하는 5가지 상황으로 분류하여 필요한 서버 대수를 계산할 때 각 상황에 맞는 다중임계치를 적용한다. 또한 부하추세를 5등급으로 나누는 기준을 서버가 부하를 추가로 감당할 수 있는 잔여용량에 따라 유연하게 조정하는 방법을 추가로 사용한다. 이 논문의 제안 방식은 부하변화량뿐만 아니라 부하를 추가로 감당할 수 있는 서버의 잔여용량을 같이 고려하여 부하추세를 판단한다. 고정 주기로 서버 전원 모드를 제어하면서 부하가 급증 또는 급감하는 경우 바로 서버 전원 모드를 추가적으로 제어하는 방법을 사용한다. 이 방법들은 부하가 변하는 상황을 서버의 잔여용량을 감안하여 여러 가지로 상정하여 각 상황에 따라 필요 서버 수를 달리 계산하여 보다 빠르고 세밀하게 서버 클러스터를 제어한다.

이 논문에서는 SPECweb이라는 전문 벤치마킹 툴을 이용하여 부하가 급격하게 변화하는 패턴들을 생성하여 클러스터 환경에서 실험하였다. 실험 결과는 QoS 측면에서 에너지 절감을 고려하지 않는 기존의 클러스터링 방식 수준으로 향상되었으며, 소비전력은 부하 패턴에 따라 최대 약 50% 절감되었음을 보여준다. 실험을 통해, 부하가 급격하게 변하는 상황에서 이 논문의 제안 방법이 Single Threshold 알고리즘과 Dual Threshold 알고리즘을 적용한 경우보다 QoS를 더 높일 수 있음을 확인하였다. 반면에 기존 대비 전력 소모는 늘어남을 확인하였는데 전력 추가 소모에 대한 타당성을 분석하였다. 자세한 내용은 논문[9]를 참고하기 바란다.

3) 서버 클러스터 환경에서 에너지 절약을 위한 서버 전원 모드 제어에서의 동적 종료[10]

특정서버를 OFF 하기로 결정한 상황에서 해당 서버가 처리 중인 서비스가 종료된 다음 서버를 OFF 해야 한다. 본 연구팀이 수행한 이 논문은 서버의 상태를 계속 감시하다가 서비스가 종료되었음을 확인하면서 서버 OFF 하는데 서버 상태 감시방법에 관한 연구에 속한다.

각 서버의 네트워크 상태 정보를 부하분산기가 알아야 종료될 서버의 동적인 종료를 할 수 있는데, “netstat”이라는 명령어를 통해 서버의 실제 연결 개수인 ESTABLISHED 정보를 알아낼 수 있다. 즉, ESTABLISHED 정보가 없다면 모든 부하 처리를 마친 상태이기 때문에 종료를 해도 QoS를 보장할 수 있다. 서버의 네트워크 상태 정보를 이용하여 종료 시점을 결정하는 동적종료 방법은 다음과 같다.

a) 각 서버의 상태 정보 중 ESTABLISHED 정보를 LVS[11]로 매초 전송한다. 이 정보는 서버가 클라이언트의 요청에 대한 부하 처리의 개수, 즉 실제로 연결된 개수를 뜻하며, 연결 중인 서버에서는 다수의 연결이 존재한다.

b) 부하분산기(LVS)는 종료해야 할 서버를 선택하고, 그 서버의 연결 정보를 계속 체크하면서, 종료 시점을 기다린다.

c) 선택된 서버의 연결 정보, 즉 연결된 개수가 0이 되는 시점에 서버를 종료하게 된다. 이때, 시간이 얼마나 걸리든지 QoS는 만족하게 된다.

본 연구팀이 수행한 동적종료 연구[10]는 클라이언트의 요청을 서버가 모두 처리하였는가에 근거하여 동적으로 최적 종료 시간을 결정하는 방식으로 상황에 따라 부하 처리

필요 시간을 동적으로 운영함으로써 소비전력과 QoS를 모두 만족시키고 있다.

4) 에너지 절감형 서버 클러스터 환경에서 QoS 향상을 위한 소비전력 예측[12]

부하가 급증 또는 급감하는 비정상적인 상황에서 QoS를 보장할 수 없는 이유는 서버가 OFF에서 ON으로 바뀌는 데 필요한 지연시간 때문에 ON 서버 대수를 당장 증가시킬 수 없기 때문이다. 이러한 문제를 해결하기 위해, 본 연구팀이 이 논문에서 수행한 방법은 짧은 시간에 급격한 트래픽 변화가 발생하는 상황을 다루기 위해 패킷 유입량을 기준으로 시계열 모델을 사용한 예측 알고리즘을 이용해서 서비스 요청 증가, 감소를 예측하여 서버의 ON/OFF 상태를 제어하는 알고리즘이다. 이 논문에서는 정상적인 상황뿐만 아니라 비정상적인 상황에서도 QoS를 향상시키는 새로운 소비전력 예측 알고리즘을 제안하며, 이 알고리즘은 기존 시계열 분석에 기반한 예측과 추세를 반영한 예측 조정의 두 부분으로 구성된다.

MMA(Modified Moving Average), MWMA(Modified Weighted Moving Average), MES(Modified Exponential Smoothing), MESTA(Modified Exponential Smoothing with Trend Adjusted) 방식을 구현하고 비교실험을 수행하였다. 추세조정 부분을 적용한 트래픽 예측을 통해 5~20초 가량 서버를 빠르게 ON 하여 서비스 요청 처리를 준비함으로써 QoS의 향상 효과를 가져왔다. 자세한 내용은 논문[12]를 참고하기 바란다.

2.3 접근 방식

기존 자율학습기반의 서버 전원 모드 제어 방법[8]의 에너지 절감 효과 및 성능을 개선하기 위해 본 논문에서는 에너지 절감 효과 및 여러 가지 성능개선 기법을 결합하여 사용한다. 첫째, 기존 방법에서는 필요한 서버 대수를 결정할 때 단일임계치를 사용하였는데 본 논문에서는 다중임계치를 사용한다. 둘째, 기존 논문에서는 학습 방법만을 적용하여 필요한 서버 대수를 정하는 반면에 본 논문에서는 spike성 트래픽이 들어올 경우 트래픽 변화량을 미리 예측하여 필요한 서버 대수를 조정하는 방법을 조건부로 추가로 사용한다. 셋째, 기존 방법에서는 전원을 OFF 하기한 서버를 일정시간을 기다렸다가 서버를 OFF 하는 수동적인 방법을 사용하였는데 본 논문에서는 더 일찍 서버를 OFF 하기 위해 서버 상태를 감시하는 능동적 방법을 사용한다.

제안하는 방법의 알고리즘 중에서 다중임계 학습 방법은 단위전력당 유효응답 수를 향상하는 부분을, 동적종료는 빠른 OFF 시간을 통해 전력 소비를 줄여주는 부분을, 예측 방법은 사용자 요청이 급격히 변화가 발생 했을 때의 QoS 를 보장하는 부분 담당한다. 자율학습기반의 전원 모드 제어에서 급격히 사용자 요청이 변화하는 경우에 대응하기 위해 예측 기법을 선택하여 트래픽을 처리하도록 구성하고 있다.

3. 제안된 시스템

Fig. 2의 설계된 부하분산기의 내부 구조를 크게 두 분류로 나는 것은 기존의 부하분산 시스템에 학습기반의 동적 서버 프로비저너를 쉽게 추가하기 위함이다. 부하분산기 상반부의 동적 서버 프로비저너는 다중학습 모듈, spike 예측 모듈, 서버 전원 모드 결정모듈, 서버 전원 모드 제어모듈로 구성되고, 부하분산기 하반부의 부하분산기는 부하 모니터, 부하분산 모듈로 구성된다.

Fig. 2의 동작과정은 다음과 같다. 클러스터를 구성하는 서버 부하 정보가 부하분산기의 하반부 부하 모니터(Traffic

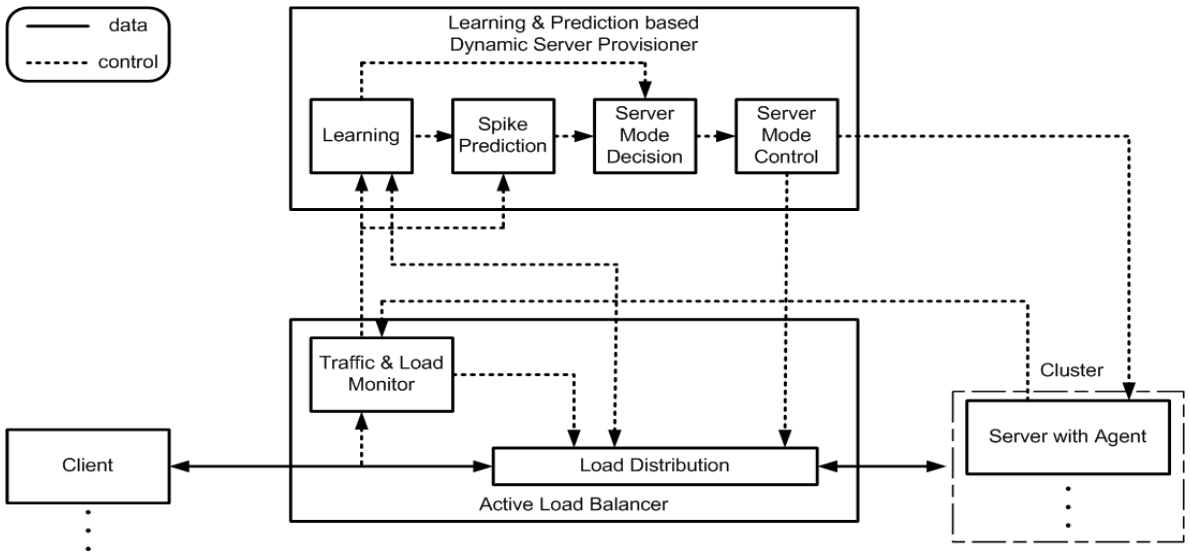


Fig. 2. Internal Structure of Load Balancer

& Load Monitor)에서 수집된다. 이 부하 정보는 spike 판단 로직에서 일반적 워크로드에 해당하는 사용자 요청에는 학습 모듈에 전달하고, 급격한 워크로드에 해당하는 사용자 요청에는 예측기반 모듈에 전달한다. 학습 모듈 혹은 예측 모듈에서 찾아진 서버 수에 따라 ON/OFF 할 서버를 결정하고 서버모드 제어 모듈은 서버 상태를 제어하며 상반부에서 결정한 서버의 부하분산 목록에 해당하는 서버들에게 하반부의 부하분산기는 부하분산을 하도록 한다.

3.1 자율학습 방법에서 다중임계치 적용

기존 용량계획기반 정적 서버모드 제어[13]에서는 예상 소비전력을 모니터링하여 단일 값의 소비전력값(임계치)으로 나누어 서버 대수를 결정하는데, 본 논문은 다중임계치 방식의 학습을 이용하여 단위전력당 성능을 높이고 있다. 기존의 다중임계치기반의 정적 서버 전원 모드 제어[9]에서는 5단계 임계치를 사용하지만, 학습기반의 방식에서는 다중 임계치 방식에서는 3단계 임계치를 사용한다. 임계치 개수가 줄어든 이유는 학습의 오버헤드에 해당하는 임계치와 상태가중치에 대한 모든 조합을 찾는 시간을 단축하여 학습 런타임 실행속도를 줄이기 위함이다.

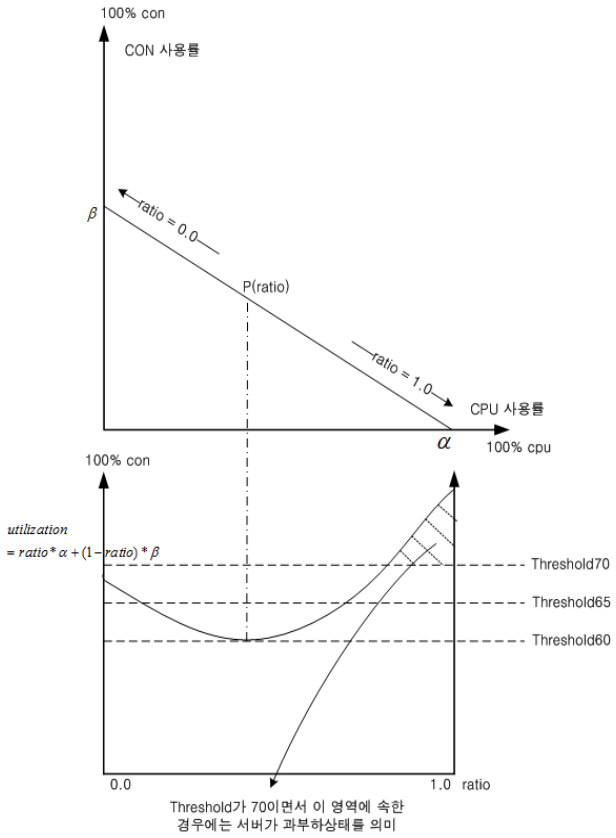


Fig. 3. Utilization Change in Ratio and Multi-Level Threshold

Fig. 3은 Equation (1)을 설명한다. Fig. 3의 상단 그림은 워크로드 패턴 변화를 보여주는데 x축은 cpu 사용률, y축은 connection 사용률을 나타낸다. 어느 한 순간의 워크로드는 α에서 β로 이어지는 직선상에서 상태가중치(ratio)값에 따라 결정되는 하나의 지점으로 표시된다. α에 가까울수록 상태가중치는 1에 근접하며 β에 가까울수록 상태가중치는 0에 근접한다. Fig. 3의 하단 그림은 서버의 utilization을 보여준다. x축은 ratio값, y축은 서버 utilization값을 나타낸다. 이 그림에서 최소값을 가지는 서버 utilization 지점(그림상에서 P)이 최소의 서버 대수로 현재 트래픽을 처리하는 최적의 지점에 해당한다.

Fig. 3의 하단 그림에서 수평방향의 3개의 점선은 3종류의 임계치를 각각 나타낸다. 여기서 임계치는 서버 1대당 적정 처리용량을 의미한다. 만일 임계치를 70을 사용할 경우 Fig. 3 하단 그림에서 오른쪽에 사선 친 영역이 있다. 이는 상태가중치값이 그 영역으로 들어오면 서버가 과부하상태로 서버 추가 배치가 필요한 상황을 나타낸다. 만일 임계치를 60을 사용할 경우 P 지점을 제외한 모든 구간에서 서버가 과부하상태로 서버 추가 배치가 필요한 상황을 나타낸다. 자율학습 방법은 주어진 워크로드에 대하여 상태가중치값을 계속 바꾸어 가면서 최적의 지점을 찾아가는 과정이다.

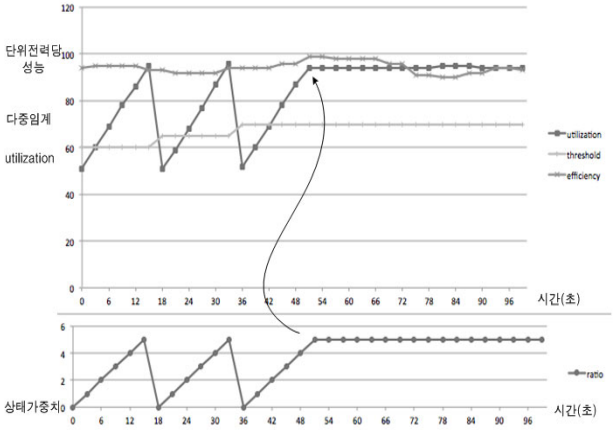


Fig. 4. Process of Finding the Most Efficient Operating Parameters Via Autonomous Learning with Multi-Level Threshold

Fig. 4는 3단계 임계치를 계속 바꾸어 적용하면서 최적의 상태가중치를 찾아가는 학습 과정을 보여준다. Fig. 4의 하단 그림에서 x축은 시간, y축은 상태가중치의 단계를 나타낸다. x축에서 시간 0~15 구간은 임계치 60을 적용한 구간이고, 시간 18~33은 임계치 65를 적용한 구간이며, 시간 36~51 구간은 임계치 70을 적용한 구간이다. 이는 Fig. 4 상단 그림의 threshold 그래프가 보여준다. y축 상태가중치가 원래 0과 1사이의 값을 가질 수 있는데 본 논문에서는 그 구간을 0.2 간격으로 나누어 총 6단계(0.0, 0.2, 0.4, 0.6, 0.8, 1.0)의 값만 고려한다. Fig. 4의 하단 그림에서 y축은 단계 0~5를 나타낸다. 상태가중치값을 6단계만 고려한 것은 경우의 수를 줄여서 학습시간을 단축하기 위함이다. Fig. 4의 상단 그림에서 x축은 시간, y축은 서버 utilization, 임계값, 단위전력당 성능을 각각 나타낸다. 특정 시간의 utilization은 하단 그림에서 주어진 해당 시점에서의 상태가중치를 이용

하여 계산한 값이고 이 지점에서의 단위전력당 성능은 상단 그림 맨 위에 있는 efficiency 그래프의 한 점에 해당한다.

Fig. 4를 종합적으로 설명하면, 다중임계치 영역 중 가장 낮은 임계치(60)에서 상태가중치를 모두 찾고, 그 다음 임계치(65)에서 상태가중치를 바꾸면서 최고임계치(70)까지 모든 조합에 대한 검사하는 과정을 거친다. 모든 조합의 검사 과정마다 단위전력당 성능(유효응답 수)을 기록한다. 검사를 마치면 모든 기록을 가진 인덱스 테이블에서 가장 높은 단위전력당 성능의 운영 변수 조합(임계값, 상태가중치)을 선택한다. Fig. 4의 예는 70, 상태가중치 5일 때 최대 효율을 가진다. 이러한 학습 과정은 기존의 워크로드 패턴 유형에 대한 학습결과가 없을 때마다 이루어져야 한다. Fig. 3과 Fig. 4는 정해지지 않은 다중임계치(Multi threshold) 중에서 단위전력당 성능이 가장 높은 임계치와 상태가중치를 찾아가는 학습과정을 나타낸 그림이다.

학습과정에서 다중임계치를 사용한 이유는 소비전력당 서버 클러스터의 성능을 최대로 할 수 있는 최적 서버 대수를 결정하는 임계치가 워크로드의 형태에 따라 높거나 낮음에 따라 달라질 수 있기 때문이다. 학습이 수행되고 나면 그 학습 결과(트래픽와 cpu 부하에 대한 상태가중치 및 서버의 워크로드 임계치)를 패턴 매트릭스에 기록하여 그다음부터는 동일한 워크로드 패턴이 나타나면 기록된 학습 결과를 이용하여 동적 서버 프로비저닝이 동작한다. 해당 워크로드 패턴에서 찾은 상태가중치와 임계치를 이용하여 클러스터를 구성하는 전체 서버에 대한 총 이용률을 구하고 해당 임계치로 나누어 필요한 서버 대수를 결정한다.

3.2 자율학습 방법과 예측 방법의 결합

본 논문에서는 자율학습 방법[8]과 예측 방법[12]을 결합하여 사용한다. 사용자 요청 트래픽이 일반적인 상황에서는 학습 방법을 적용하고 사용자 요청 트래픽이 급변하는 상황에서는 예측 방법을 적용한다. Fig. 5에서 일반적 사용자 요청에 해당하는 워크로드에서는 학습부분(우측 점선 부분)으로 제어흐름을 선택하고, 급격한 사용자 요청 변화가 있는 워크로드에서는 예측부분(좌측 점선 부분)으로 제어흐름을 선택하도록 구성했다. load info가 주어지면 우선 spike성 워크로드인지를 구별한다. 만일 spike성 워크로드라면 예측부분을 수행하고, 아니라면 예전에 학습이 수행되었는지를 확인한다(checking parameter). 만일 예전 결과가 있으면 그것을 이용하고 그렇지 않다면 학습부분을 수행한다. 여기서의 학습은 Fig. 2의 상단부 Learning에 해당하고 여기서의 예측은 Fig. 2의 상단부 Spike Prediction에 해당한다.

Fig. 5에서 우측의 점선부분으로 표시한 학습동작 흐름을 자세히 설명하면 다음과 같다.

1) 자율학습 부분[8]

• 단계 1. CPU-Connection 표준편차 확인

cpu와 트래픽 사용률에 대한 표준편차를 구한다. 해당 값은 정규화된 값으로 변환된다.

• 단계 2. 학습 패턴 테이블 확인

패턴 매트릭스에 정규화된 표준편차에 대한 해당 엔트리(cpu, con)가 존재하는지 확인한다. 패턴 매트릭스에서 엔트리가 채워져 있으면 학습된 상태, 없으면 학습 이전 상태가 된다. 엔트리가 빈 상태에서 학습을 거치며 엔트리를 채우게 된다.

• 단계 3. 상태가중치-임계치 선택

학습이 존재한다면 정규화된 표준편차에 대한 엔트리(cpu, con)에서 상태가중치, 임계치를 선택한다. 패턴 매트릭스에

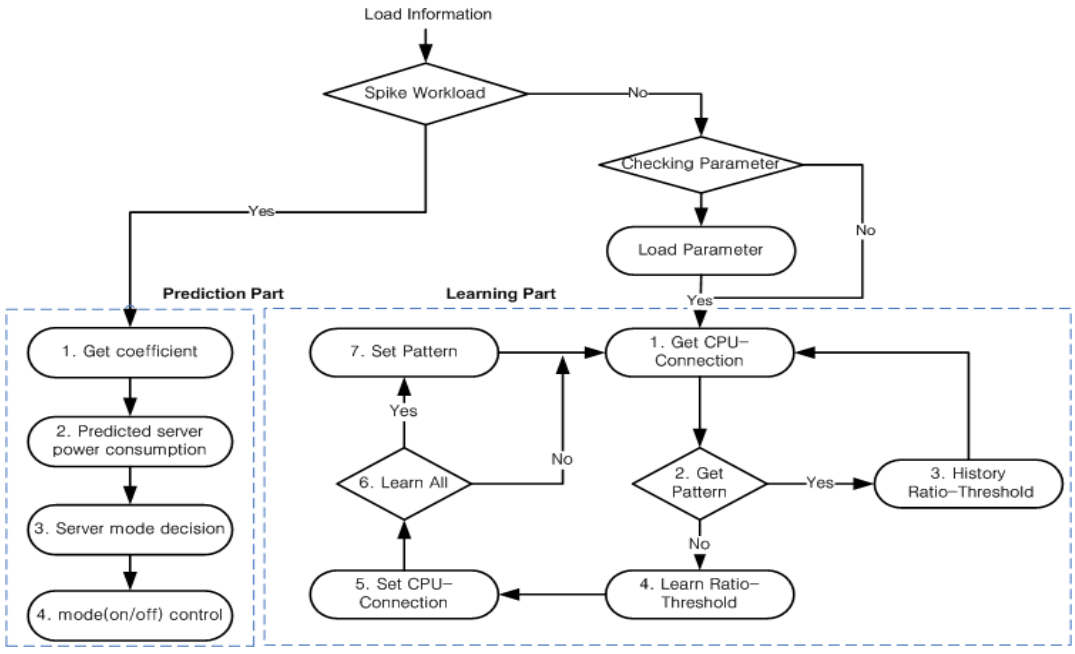


Fig. 5. Conditional selection of autonomous learning and prediction

이미 학습된 엔트리의 값을 선택하게 된다.

• 단계 4. 상태가중치-임계치에 대한 학습

이전의 2단계에서 기존 학습패턴이 없다면, 새로운 패턴이므로 학습을 수행하도록 한다. 해당 엔트리(cpu, con)에 대한 상태가중치, 임계치를 바꾸어가며 최적의 조합을 선택한다.

• 단계 5. 상태가중치, 임계치 기록

해당 상태가중치, 임계치로 부하분산을 수행하고, 인덱스 테이블에 임계치와 상태가중치를 기록한다.

• 단계 6. 모든 조합에 대한 학습 여부 확인

모든 조합에 대해 학습이 이루어졌는지 확인한다. 전체 인덱스 테이블을 검색하고 정규화된 표준편차에 대한 엔트리(cpu, con)에서 최적의 단위전력당 성능을 보인 (상태가중치, 임계치)값을 선정한다.

• 단계 7. 패턴 매트릭스 테이블에 기록

선정된 엔트리(cpu, con)에서 (상태가중치, 임계치)값을 패턴 매트릭스 테이블에 기록한다.

자율학습 부분의 자세한 동작과정은 자율학습기반의 에너지 효율적인 클러스터 관리 기법 논문[8]을 참고하기 바란다.

2) 예측 부분[12]

소비전력 예측에 대한 기존 연구[12]에서는 MESTA(Modified Exponential Smoothing with Trend Adjusted)가 가장 우수한 성능을 보였으나, 본 논문에서는 MES(Modified Exponential Smoothing)를 선택하여 사용하였다. 그 이유는 전체 처리 부하가 늘어난 새로 구성된 실험환경에서 두 알고리즘 중 MES가 좀 더 우수하게 나타났기 때문이다. 알고리즘의 구체적인 동작 방법은 아래와 같다.

Fig. 5에서 좌측의 점선부분으로 표시한 예측 동작 흐름을 자세히 설명하면 다음과 같다.

• 단계 1. 계수값 측정

계수값 측정부분은 Input bytes량 대비 서버 소비전력 량을 일정 시간동안 수집하여 평균을 얻는다. 이 값을 계수값으로 저장한다.

$$\sum \frac{(\text{총소비전력})}{input\,Byte} / (\text{측정시간}) = Coefficient \qquad (2)$$

• 단계 2. 서버 소비전력 예측

서버의 소비전력 예측부분은 부하분산기에서 측정된 트래픽 유입량을 10초 이동평균을 하여 계수값과 곱하여 예측된 서버 소비전력을 계산한다.

$$(\text{예측된 } input\,Byte \text{ 량}) \times Coefficient = (\text{예측된 서버 소비 전력}) \qquad (3)$$

• 단계 3. 필요한 서버 대수가 결정되어 어느 서버들을 ON할지를 결정(Fig. 6의 server mode decision 부분에 해당)

• 단계 4. 해당 서버들을 ON 또는 OFF(Fig. 6의 server mode control에 해당)

예측부분의 자세한 동적과정은 에너지 절감형 서버 클러스터 환경에서 QoS 향상을 위한 소비전력 예측 논문[12]을 참고하기 바란다.

3) 학습 방법 또는 예측 방법의 선택 기준

Table 1에 보이는 바와 같이 부하 추세에 따라 자율학습 방법 또는 예측 방법을 선택한다. 부하가 일정한 상태에서 과부하 상태로 급변하는 경우를 찾기 위해서 부하분산기의 입출력정보($\frac{\text{현재 입출력}}{\text{관찰된입출력}(1\text{분이동평균}, 3\text{분이동평균})}$)를 보고 급격한 부하의 상승과 급격한 부하의 하강을 검출한다.

Table 1. Selection Algorithm According to the Load Information of the Input Data

부하의 변경정도	부하 추세	알고리즘 선택
load average(α, β) >> 1	급증	예측
load average(α, β) > 1	상승	학습
1	상태 유지	학습
load average(Θ, γ) < 1	하강	학습
load average(Θ, γ) << 1	급감	예측

Table 1의 α, Θ는 1분간 관찰된 입출력 이동평균 대비 현재의 입출력 비율이고, β, γ는 3분간 관찰된 입출력 이동평균 대비 현재의 입출력 비율로 상승이면 1보다 크게 되고, 감소이면 1보다 작은 수가 된다. α, Θ는 좀 더 최근의 트래픽 추세를 알려주고, β, γ는 이보다 긴 트래픽 추세를 알려주게 된다.

3.3 서버의 동적종료 방법 적용

기존 동적종료[10]에서는 클라이언트와 서버의 연결 수가 0이 될 때까지 기다려 모든 사용자 요청을 처리하고 있다. 반면에 제안 방식에서는 사용자 요청이 일부 남아있더라도 서버의 부하가 낮게 유지되고 서버의 연결 수가 적으면 timeout 시간에 의해 서버를 OFF 하도록 하고 있다. 이러한 서버의 빠른 OFF는 소비전력을 줄여 단위전력당 성능(유효응답 수)을 높이는 데 기여한다.

기존의 동적종료 제어는 부하분산 목록에서 제외된 서버의 연결 수(ESTABLISHED)가 0이 되어 해당서버가 sleep 되는 방식이라면, 제안 방법은 keepalive가 길게 설정된 경우에도 서버 OFF가 처리 가능하도록 한다. 서버의 부하가 충분히 낮고 네트워크 연결이 적게 유지되고 있으면, 부하분산 목록에서 제외하고 일정시간이 지난 후에 서버를 sleep 상태가 되도록 한다.

Fig. 6의 동적종료의 처리는 학습기반의 프로비저닝이나 예측기반의 프로비저닝에 의해 서버 대수가 결정된 후 서버 모드 제어에서 동적종료를 처리하도록 구성한다. Fig. 6의 동적종료를 포함한 서버모드 제어 흐름은 다음과 같다. 자율학습에서는 다중임계치와 상태가중치를 이용해 총 이용률을 구하고 임계치로 나누어 필요한 서버 수를 서버모드 결

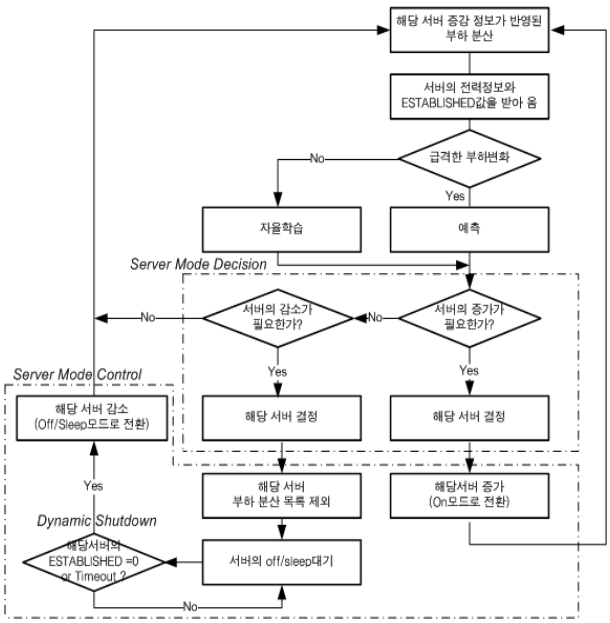


Fig. 6. Control Flow of Dynamic Shutdown

정 모듈에 전달해주고, 예측에서는 소비전력 예측을 통해 필요한 서버 수를 서버모드 결정 모듈에 전달해준다. 서버 모드가 결정되면 서버 ON/OFF를 제어하게 되는데 여기에 동적종료 판단부가 OFF를 결정하는 데 관여한다. 판단 조건은 일정 시간 안에 서버의 연결 수가 0이면 OFF 하도록 하고, 일정시간이 지나 부하가 낮은 상태로 감소 중이고 네트워크 연결수가 적으면 OFF가 되도록 결정한다.

4. 실 험

4.1 실험 환경

Fig. 7은 실험 전체 구성을 나타내며 부하 테스트환경인 SPECweb[14]의 구성은 모두 상용 x86프로세서를 이용하며 리얼 서버 16대, prime client 1대, client 6대로 구성된다. prime client는 client가 여러 대일 때 동기화 실행을 관리하고, Besim(Backend Simulator)는 client의 동작(로그인, 계좌이체 등)을 제어한다. 또한, 부하분산기 1대와 기가비트 스위

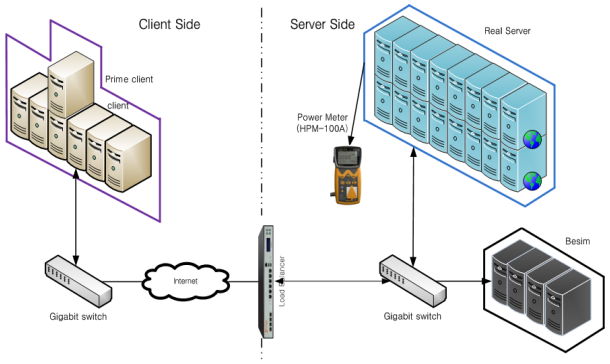


Fig. 7. The Overall Configuration of Experiments

치 2대를 이용한다. 각 서버에 설치된 아파치[15]의 경우 서버의 웹 처리용량을 늘리기 위해 mpm-event 모드를 이용하고, 클러스터를 구성하는 서버는 Fedora를 설치하고 클라이언트는 Ubuntu를 설치하여 실험하였다.

전력의 측정은 SPECweb의 warmup 구간이 끝나는 시점부터 뱅킹 부하패턴과 실제 부하패턴은 1시간, 가상 부하패턴 30분간의 클러스터의 전력소비를 측정하였다.

Table 2. Hardware and Software Specification for Experiment

항목	대수	하드웨어	소프트웨어
server	16	i5-3550, 8G, 400G HDD	Fedora 18, PHP 5.5.3, python 3.3.2, Apache 2.4.6
client	6	i5-3550, 8G, 400G HDD	Ubuntu 12.10
Prime Client	1	i5-3550, 8G, 400G HDD	Ubuntu 12.10
LVS	1	i5-3550, 8G, 400G HDD	Fedora 13, ipvsadm
Besim	4	i5-3550, 8G, 400G HDD	Fedora 13, PHP 5.3.6
SPECweb	1	장비에 분산 설치됨	Besim, Client, PrimeClient에 설치
PowerMeter	1	HPM-100A	serial com
network switch	2	SG92-24 Gigabit switch	

Table 2는 클러스터의 구성 하드웨어 요소들과 설치되어지는 소프트웨어에 대한 설명이다. 여기에서 SPECweb은 별도의 하드웨어에서 동작하는게 아니라 client, primeclient, besim에 나누어 분산 설치 되어진다.

4.2 실험 방법

참고논헌 [8]에는 일반적인 트래픽 상황인 SPECweb 뱅킹 부하패턴, 실제 부하패턴, 트래픽 변화가 심한 비일반적 상황인 가상 부하패턴으로 총 3가지 부하 패턴에 대하여 always-on (모든 서버가 항상 ON), 정적제어, 예측 방법, 단일임계치를 사용하는 학습 방법 각각의 실험 결과가 나와있다. 본 논문에서는 참고논헌 [8]과 마찬가지로 총 3가지 부하 패턴에 대하여 새로운 실험들을 각각 추가하였다. 우선은 본 논문에서 설명하는 다중임계치를 이용하는 학습 방법을 먼저 수행하였다. 기존의 단일임계치를 사용하는 학습 방법에서 얻은 학습 결과를 갖고 하는 실험(이를 파라미터 로드된 단일임계치 학습 실험이라 부름), 다중임계치를 사용하는 학습 방법에서 얻은 학습 결과를 갖고 하는 실험(이를 파라미터 로드된 다중임계치 학습 실험이라 부름)을 수행하였다. 이 두 가지 유형의 파라미터 로드된 방법에 본 논문에서 설명하는 방법들 중에서 1) 예측 방법만을 추가 적용, 2) 동적종료만을 추가 적용 3) 예측 방법 및 동적종료를 추가하는 방법을 적용, 각 유형별 4가지씩 총 8종류의 실험을 수행하였다. 파라미터 로드된 단일임계치 학습 방법(실험1), 파라미터 로드된 단일임계치 학습 방법+예측 방법(실험2), 파라미터 로드

된 단일임계치 학습 방법+동적종료(실험3), 파라미터 로드된 단일임계치 학습 방법+예측 방법+동적종료(실험4), 파라미터 로드된 다중임계치 학습 방법(실험5), 파라미터 로드된 다중임계치 학습 방법+예측 방법(실험6), 파라미터 로드된 다중임계치 학습 방법+동적종료(실험7), 파라미터 로드된 다중임계치 학습 방법+예측 방법+동적종료(실험8)이다.

첫 번째 실험은 Fig. 8에서 보이는 SPECweb banking으로 온라인 은행 업무에 근거하고, 이 작업 부하에 있는 모든 요청패턴을 만들어 생성하고 패턴에 대한 실험을 하였다.

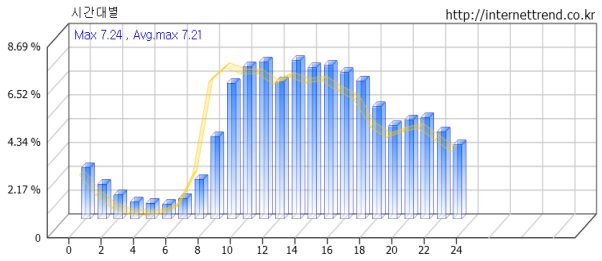


Fig. 8. InternetTrendTM - Banking Load Pattern

Session: 2069 1242 730 491 402 446 774 2128 6118 6400 6050 6050 5708 5909 5656 5722 5269 4964 3870 3513 3750 3878 3453 2970
실험시간: 60분(그래프 하단의 24시간은 하루를 나타내며, 전체 실험 소요시간을 60분으로 전환하여 실험), 최대 session: 6400

InternetTrendTM[16]에는 일반사이트 및 모바일WEB/APP의 로그분석 솔루션 및 ASP 서비스 LoggerTM에 의해 선별된 실측 로그분석에 의한 데이터 가운데 샘플링 데이터를 통계 처리하여 생성된 웹 분석 데이터의 평균값 데이터들이 카테고리/기간별로 DB화 되어 일반에게 공개되어있다. 따라서 원하는 카테고리와 시간대를 입력하면 해당 트래픽 추세를 확인할 수 있다. 두 번째 실험으로 Fig. 9에 보이는 실제 워크로드 패턴은 InternetTrendTM[16]의 Website Performance 카테고리의 주문집중 시간대 자료 중 2012/9/1~2012/9/7까지의 금융/부동산/경제 파트의 시간대별 집중도를 따라서 생성하였다.

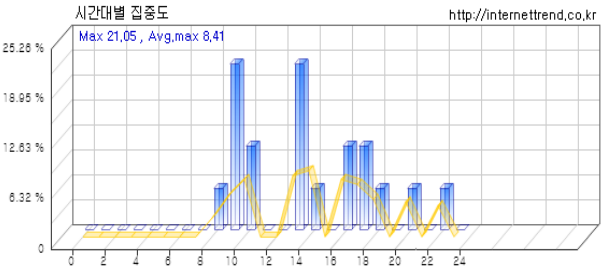


Fig. 9. InternetTrendTM - Website Performance

Session: 0 0 0 0 0 0 0 0 1599 6400 3202 0 0 6400 1599 0 3202 3202 1599 0 1599 0 1599 0
실험시간: 60분(그래프 하단의 24시간은 하루를 나타내며, 전체 실험 소요시간을 60분으로 전환하여 실험), 최대 session: 6400

세 번째 실험인 Fig. 10에 보이는 가상 부하 패턴은 급격한 부하변화에 대한 성능을 확인하고자 생성하였다.

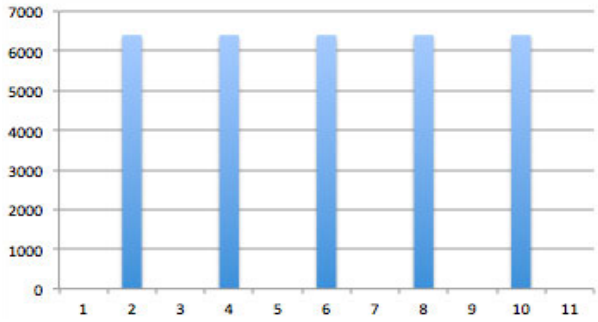


Fig. 10. Virtual Load Pattern

Session: 0 6400 0 6400 0 6400 0 6400 0 6400 0
실험시간: 30분(그래프 하단의 24시간은 하루를 나타내며, 전체 실험 소요시간을 30분으로 전환하여 실험), 최대 session: 6400

단일임계 자율학습과 다중임계 자율학습 방법의 서버 전원 모드 제어는 정적 서버 전원 모드 제어와 동일하고 필요한 서버 수를 산정하는 부분에 자율학습을 적용한 방식에 해당한다. 이 경우 역시 1초 단위의 서버 전원 모드 제어를 하고, 3초 단위로 자율학습과 네트워크 연결 가중치를 변경하는 방식이다. 소비전력 예측기법의 경우 부하분산기로 유입되는 네트워크의 부하에 대해 추세 조정된 지수평활방법(modified exponential smoothing)을 동일하게 1초 단위로 서버 전원 모드를 제어하는 방식이다. 기존에 실험된 결과와 새로 추가된 8종류의 실험방법을 설명하면 다음과 같다.

파라미터 로드된 단일임계 학습과 예측의 조건부 선택은 하나의 임계치를 가지고 네트워크 트래픽과 cpu의 상태가중치를 변경하여 학습결과 파라미터를 저장하고 재활용하는 방법과 소비전력 예측기법의 선택을 통해 급격한 트래픽 변화에 대한 처리를 위한 개선방법이다. 파라미터 로드된 단일임계 학습에 동적종료 적용은 단위전력당 유효응답 수를 개선하기 위한 실험이다. 파라미터 로드된 단일임계 학습과 소비전력 예측의 조건부 선택에 동적종료는 향상된 QoS와 단위전력당 유효응답 수를 개선하기 위한 실험이다.

단일임계 자율학습이 정적 서버모드 제어와 동일하게 하나의 임계치를 가지도록 하여 성능비교를 했다면, 다중임계 자율학습은 단위전력당 유효응답 수와 QoS 개선을 위해 여러 개의 임계치를 가지도록 구성한 경우의 실험에 해당한다. 여기에 학습으로 인한 오버헤드를 줄이기 위해 파라미터 로드된 다중임계 학습을 실험하였다. 또한, 다중임계 자율학습과 소비전력 예측의 조건부 선택을 적용하여 급격히 트래픽이 증가하는 경우에 대응하도록 실험하였다. 파라미터 로드된 다중임계 자율학습의 동적종료 실험은 QoS 개선과 함께 단위전력당 유효응답 수를 개선하기 위한 실험에 해당한다. 파라미터 로드된 다중임계 학습과 소비전력 예측기법의 조건부 선택에 대해서도 단위전력당 유효응답 수를 더 개선하기 위해 동적종료 방법을 추가하여 실험하였다.

4.3 실험 결과

SPECweb workbench 그래프에서 time good은 클라이언트의 요청에 2초 이내 응답 받은 수를, time tolerable은 4초 이내 응답 수, time fail은 응답하는 데 4초 이상 걸린 수를 의미한다. time tolerable과 time fail의 합이 클라이언트 요청에 따른 서버의 유효응답 총수에 해당한다. 실험 결과 중 Table 3의 각 요소를 비율(%)로 나타내었는데 이는 유효응답 총수에 대한 각 요소의 비율을 나타낸 것이다. 실험 결과표의 Power 항목은 해당 실험에서 소모된 서버들의 전력량을 의미한다. 실험 결과표의 처리율(%)은 서버 전원 모드 제어 방법마다 유효응답 총수가 다른데 항상 ON 방법 대비 해당 방법의 유효응답 총수 처리 비율을 의미한다.

특정 방법에서의 time good 비율은 그 해당 방법의 유효응답 총수 대비 2초 이내 응답한 수의 비율을 나타내는데 이는 해당 방법에서의 서비스품질에 해당한다. 각 방법에서의 유효응답 총수가 각기 다르므로 항상 ON 방법 대비 해당 방법의 서비스품질을 정의하여 비교하려고 한다. 이를 실험 결과표에서 표준화된 서비스품질(normalized QoS)이라고 부른다. 이 항목은 해당 방법의 서비스품질에 해당방법의 처리율을 곱함으로써 계산된다.

또한, 실험 결과표에서 단위전력당 good(유효) 응답 수 항목을 추가로 사용한다. 이는 해당 방법에서 서비스 품질에 해당하는 time good에 해당되는 응답 수를 그것을 처리하기 위해 소모한 전력량으로 나눈 값이다. 이는 전력 소모 측면에서 얼마나 양질의 good 응답인지 구분하기 위해 도입한다. 아래의 실험 결과표에서 표준화된 서비스품질과 단위전력당 good 응답 수를 서버 전원 모드 제어 방법과 비교할 때 중요한 의미를 갖는다[9].

1) 뱅킹 부하패턴

Table 3. Results of Experiment(banking load pattern)

서버 전원 모드 제어 방법	Power (W)	유효 응답 총수	Time (%)			처리율 (%)	표준화된 QoS (%)	단위 전력당 good 응답 수
			Good	Tolerable	Fail			
always-on	645.8	1902342	99.8	100.0	0.0	100.0	99.8	2939.82
정적	406.1	1678156	99.3	99.5	0.5	88.22	87.6	4103.44
예측	430.3	1736408	99.5	99.7	0.3	91.28	90.82	4015.17
단일 임계	415.5	1712947	99.5	99.6	0.4	90.04	89.59	4102
파라미터단일 임계 (실험1)	401.4	1715508	99.5	99.6	0.4	90.18	89.73	4252.44
파라미터 단일 임계+ 예측 (실험2)	400.8	1722114	99.5	99.6	0.4	90.53	90.08	4275.21
파라미터단일 임계+ 동적 종료 (실험3)	368.8	1777669	99.4	99.6	0.4	88.08	87.55	4306.01
파라미터단일 임계+ 예측+ 동적 종료 (실험4)	394.2	1803847	99.5	99.6	0.4	89.8	89.35	4311.89
다중임계	415.6	1721447	99.5	99.7	0.3	90.49	90.04	4121.37
파리미터다중 임계 (실험5)	403.8	1724851	99.6	99.7	0.3	90.67	90.31	4254.46
파라미터다중 임계+ 예측 (실험6)	404.5	1739259	99.5	99.7	0.3	91.43	90.97	4278.28
파라미터다중 임계+ 동적 종료 (실험7)	394.6	1802867	99.5	99.7	0.3	89.93	89.48	4313.75
파라미터다중 임계+ 예측+ 동적 종료 (실험8)	396.6	1813136	99.6	99.7	0.3	90.49	90.13	4323.01

Table 3은 뱅킹 부하패턴에 대한 실험 결과 비교표이다. 파라미터 로드된 다중임계 학습과 예측 결합에 대한 동적종료와 정적 전원 모드 제어 방법에 비해 단위전력당 성능 면에서 5.35% 높고, QoS 측면에서는 2.89% 높다.

파라미터 로드된 다중임계 학습과 예측의 조건부 선택에 대한 동적종료는 파라미터 로드된 단일임계 학습에 비해 단위전력당 good(유효) 응답 수 면에서 1.66% 높다. QoS 측면에서는 0.45% 높다. 실험 결과에 대한 자세한 분석은 4.4절에 나와있다.

2) 실제 부하패턴

Table 4. Results of Experiment(real load pattern)

서버 전원 모드 제어 방법	Power (W)	유효 응답 총수	Time (%)			처리율 (%)	표준화된 QoS (%)	단위 전력당 good 응답 수
			Good	Tolerable	Fail			
always-on	617.3	667531	99.9	100	0	100	99.85	1080.29
정적	214.8	459575	98.1	98.3	1.7	68.85	67.54	2098.9
예측	219.1	471585	98	98.2	1.8	70.65	69.24	2109.33

서버 전원 모드 제어 방법	Power (W)	유효 응답 총수	Time (%)			처리율 (%)	표준화된 QoS (%)	단위 전력당 good 응답 수
			Good	Tolerable	Fail			
단일 임계	203.8	469491	98	98.2	1.8	70.22	68.92	2257.61
파라미터단일 임계 (실험1)	208.7	485422	97.9	98.2	1.8	72.72	71.19	2277.09
파라미터단일 임계+ 예측 (실험2)	225.6	501803	98	98.3	1.7	75.17	73.67	2179.82
파라미터단일 임계+ 동적 종료 (실험3)	201.5	505648	98.1	98.2	1.8	70.35	69.01	2286.25
파라미터단일 임계+ 예측+ 동적 종료 (실험4)	209.5	527930	97.8	98.1	1.9	73.44	71.82	2288.64
다중 임계	213.4	472628	97.9	98.2	1.8	70.8	69.31	2168.24
파라미터다중 임계 (실험5)	207.7	495118	98	98.2	1.8	74.17	72.69	2336.14
파라미터다중 임계+ 예측 (실험6)	221.8	504304	98.1	98.2	1.8	75.55	74.11	2230.49
파라미터다중 임계+ 동적 종료 (실험7)	199.5	513926	98	98.2	1.8	71.31	69.88	2338.36
파라미터다중 임계+ 예측+ 동적 종료 (실험8)	205.5	529168	98	98.2	1.8	73.61	72.14	2343.16

Table 4는 실제 부하패턴에 대한 실험 결과 비교표이다. 파라미터 로드된 다중임계 학습과 예측의 조건부 선택에 대한 동적종료와 정적 전원 모드 제어 방법에 비해 단위전력당 유효응답 수 면에서 11.64% 높고, QoS 측면에서는 6.81% 높다. 파라미터 로드된 다중임계 학습과 예측의 조건부 선택에 대한 동적종료는 파라미터 로드된 단일임계 학습에 비해 단위전력당 유효응답 수 면에서 2.9% 높다. QoS 측면에서는 1.33% 높다. 실험 결과에 대한 자세한 분석은 4.4절에 나와 있다.

3) 가상 부하패턴

Table 5. Results of Experiment(virtual load pattern)

서버 전원 모드 제어 방법	Power (W)	유효 응답 총수	Time (%)			처리율 (%)	표준화된 QoS (%)	단위 전력당 good 응답 수
			Good	Tolerable	Fail			
always-on	325.6	764140	99.6	100	0	100	99.63	2338.48
정적	149.6	308126	90	91	9	40.32	36.29	1853.7
예측	171.6	454491	92.9	93.5	6.5	59.48	55.26	2460.5
단일 임계	150.2	415167	95.1	95.6	4.4	54.33	51.67	2628.65
파라미터단일 임계 (실험1)	143.7	403973	95.2	95.6	4.4	52.87	50.33	2676.29
파라미터단일 임계+ 예측 (실험2)	157.3	438960	95.2	95.6	4.4	57.44	54.68	2656.64
파라미터단일 임계+ 동적 종료 (실험3)	135.8	425182	94.8	95.3	4.7	50.33	47.71	2684.74
파라미터단일 임계+ 예측+ 동적 종료 (실험4)	149.9	47123	95.2	95.5	4.5	57.06	54.32	2769.18
다중 임계	157.4	438764	95.3	95.7	4.3	57.42	54.72	2690.75
파라미터다중 임계 (실험5)	152.5	431667	95.1	95.7	4.3	56.49	53.72	2691.9
파라미터다중 임계+ 예측 (실험6)	154.6	444884	95.3	95.6	4.4	58.22	55.48	2742.4
파라미터다중 임계+ 동적 종료 (실험7)	142.2	457539	95.1	95.5	4.5	54.32	51.66	2776.17
파라미터다중 임계+ 예측+ 동적 종료 (실험8)	150.6	477707	95.2	95.6	4.4	57.53	54.77	2779.13

Table 5는 가상 부하패턴에 대한 실험 결과 비교표이다. 파라미터 로드된 다중임계 학습과 예측의 조건부 선택에 대

한 동적종료와 정적 전원 모드 제어 방법에 비해 단위전력당 유효응답 수 면에서 49.92% 높고, QoS 측면에서는 50.92% 높다. 파라미터 로드된 다중임계 학습과 예측의 조건부 선택에 대한 동적종료는 파라미터 로드된 단일임계 학습에 비해 단위전력당 유효응답 수 면에서 3.84% 높다. QoS 측면에서는 8.82% 높다. 실험 결과에 대한 자세한 분석은 4.4절에 나와있다.

4.4 분석 및 토론

1) QoS 측면에서의 결과 분석

Table 6. Comparison of QoS for Each Pattern

알고리즘 유형	가상 부하 패턴	실제 부하 패턴	뱅킹 부하 패턴
정적 서버 전원 모드 제어(실험0)	36.29	67.54	87.6
소비전력 예측	55.26	69.24	90.82
단일임계 학습	51.67	68.92	89.59
파라미터 로드된 단일임계 학습 (실험1)	50.33	71.19	89.73
파라미터 로드된 단일임계 학습+ 예측(실험2)	54.68	73.67	90.08
파라미터 로드된 단일임계 학습+ 동적종료(실험3)	47.71	69.01	87.55
파라미터 로드된 단일임계 학습+ 예측+동적종료(실험4)	54.32	71.82	89.35
다중임계 학습	54.72	69.31	90.04
파라미터 로드된 다중임계 학습 (실험5)	53.72	72.69	90.31
파라미터 로드된 다중임계 학습+ 예측(실험6)	55.48	74.11	90.97
파라미터 로드된 다중임계 학습+ 동적종료(실험7)	51.66	69.88	89.48
파라미터 로드된 다중임계 학습+ 예측+동적종료(실험8)	54.77	72.14	90.13

Table 6에서 정적 서버모드 제어(실험0)를 파라미터 로드된 다중임계 학습과 예측의 조건부 선택에 대한 동적종료(실험8)를 비교할 때 뱅킹 부하패턴, 실제 부하패턴, 가상 부하패턴에서 QoS가 각각 2.89%, 6.81%, 50.92% 향상되었다. 또한, 파라미터 로드된 다중임계 학습과 예측의 조건부 선택에 대한 동적종료(실험8)와 파라미터 로드된 단일임계 학습(실험1)를 비교할 때 뱅킹 부하패턴, 실제 부하패턴, 가상 부하패턴에서 QoS가 각각 0.45%, 1.33%, 8.82% 향상되었다. 계산과정은 Equation (4)와 같다.

$$\text{상승률} = \frac{\text{실험8} - \text{기존방법}}{\text{기존방법}} \quad (4)$$

본 논문에서는 다중임계 학습 방법, 예측 방법, 동적종료 방법을 사용하고 있는데 각 방법의 QoS 향상에 미치는 영향을 분석하기 위해 각 방법의 기여도를 계산하였다.

Table 7. Contribution Analysis in QoS

실험방법	다중임계 학습	동적종료	예측
뱅킹 부하패턴	37.1%	-16%	78.8%
실제 부하패턴	48.2%	-42.1%	93.8%
가상 부하패턴	34.3%	13.5%	52.2%

아래 2) 단위전력당 응답 수 분석에서도 동일하게 적용된다.

실험8의 다중임계 학습과 예측의 선택률을 보면 뱅킹 부하패턴에서 다중임계 학습은 95.26%이고, 예측은 4.74%이고, 실제 부하패턴에서 다중임계 학습은 85.53%이고, 예측은 14.47%이고, 가상 부하패턴에서 다중임계 학습은 70.09%이고, 예측은 29.91%이다. 조건부 선택률은 기여도 분석에 참고하기 위해 기술하였다.

Table 7의 기여도 분석은 다중임계 상승률(실험5), 예측 상승률(실험6), 동적종료 상승률(실험7)의 총합을 100%로 하고, 각 적용기법에 의한 상승률을 부분 기여도로 하여 Equation (5)와 같이 계산하였다.

$$\text{부분기여도} = \frac{\text{적용기법 상승률}}{\text{다중임계상승률} + \text{예측상승률} + \text{동적종료상승률}} \quad (5)$$

가상 부하패턴, 실제 부하패턴, 뱅킹 부하패턴의 순서로 예측 선택률이 높게 측정된 결과와 마찬가지로 급증과 급감이 많은 가상 부하패턴, 실제 부하패턴에서 예측과 다중임계 학습에 의한 QoS 상승률이 높았다.

실제 부하패턴에서 동적종료가 음의 기여도를 갖는 이유는 트래픽 감소구간(11시, 15시, 19시 지점)에서 낮은 부하를 유지하므로 동적종료에 의한 서버의 빠른 OFF를 위해 사용자 요청 손실이 있음을 감수했기 때문이다. 또한, 뱅킹 부하패턴에서도 트래픽 감소구간에서 낮은 부하가 유지되고 있어 동적종료에 의한 OFF를 하였기 때문이다. 반면에 뱅킹 부하패턴 중 은행 오전 개장 시간(9시) 이후에 급격한 트래픽 증가구간이 있어 예측의 조건부 선택률은 작지만 예측 효과가 크게 나타났다. 가상 부하패턴의 경우 트래픽 급증구간은 예측의 조건부 선택률이 높아 예측 효과가 크고, 트래픽 감소구간은 부하감소 시간이 짧아 동적종료에 의한 손실로 잡히지 않았다.

결론적으로 QoS 측면에서 예측은 모두 양의 기여도를 갖고 가장 높게 나왔으며, 다중임계치 학습 역시 모두 양의 기여도를 가져 두 알고리즘은 QoS 측면에서 유효한 알고리즘을 보여준다. 동적종료는 가상 부하패턴에서 그 효과가 크다.

2) 단위전력당 유효응답 수 측면에서의 결과 분석

Table 8에서 정적 서버모드 제어(실험0)를 파라미터 로드된 다중임계 학습과 예측의 조건부 선택에 대한 동적종료(실험8)를 비교할 때 뱅킹 부하패턴, 실제 부하패턴, 가상 부하패턴에서 단위전력당 유효응답 수가 각각 5.35%, 11.64%, 49.92% 향상되었다. 또한, 파라미터 로드된 단일임계 학습(실

Table 8. Comparison of the Numbers of Good Response Per Watt for Each Pattern

알고리즘 유형	가상 부하 패턴	실제 부하 패턴	뱅킹 부하 패턴
정적 서버 전원 모드 제어(실험0)	1853.7	2098.9	4103.44
소비전력 예측	2460.5	2109.33	4015.17
단일임계 학습	2628.65	2251.61	4102
파라미터 로드된 단일임계 학습(실험1)	2676.29	2277.09	4252.44
파라미터 로드된 단일임계 학습+예측(실험2)	2656.64	2179.82	4275.21
파라미터 로드된 단일임계 학습+동적종료(실험3)	2684.74	2286.25	4306.1
파라미터 로드된 단일임계 학습+예측+동적종료(실험4)	2768.18	2288.64	4311.89
다중임계 학습	2690.75	2168.24	4121.37
파라미터 로드된 다중임계 학습(실험5)	2691.9	2336.14	4254.46
파라미터 로드된 다중임계 학습+예측(실험6)	2742.4	2230.49	4278.28
파라미터 로드된 다중임계 학습+동적종료(실험7)	2776.17	2338.36	4313.75
파라미터 로드된 다중임계 학습+예측+동적종료(실험8)	2779.13	2343.16	4323.01

험1), 파라미터 로드된 다중임계 학습과 예측의 조건부 선택에 대한 동적종료(실험8)를 비교할 때 뱅킹 부하패턴, 실제 부하패턴, 가상 부하패턴에서 단위전력당 유효응답 수가 각각 1.66%, 2.9%, 3.84% 향상되었다. 계산과정은 Equation (4)와 같다.

Table 9. Contribution Analysis in the Numbers of Good Response Per Watt

실험방법	다중임계 학습	동적종료	예측
뱅킹 부하패턴	2.4%	68.5%	29%
실제 부하패턴	80.1%	83.2%	-63.4%
가상 부하패턴	8.5%	55%	36.4%

Table 9의 단위전력당 유효응답 수에 대한 기여도 분석은 다중임계 학습, 동적종료, 예측 방법의 결합 대비 적용기법의 상승률로 Equation (5)와 같이 계산하였다.

뱅킹 부하패턴에서는 급증하거나 급감하는 패턴변화가 상대적으로 적고 오후 2시 이후부터 계속 트래픽이 감소하여 동적종료의 효과가 크게 나타났다. 실제 부하패턴은 부하가 계단식으로 감소하는 (11시, 15시, 19시 지점) 구간이 상대적으로 많이 분포하고 있기 때문에 동적종료의 효과가 크게 나타났으며, 가상 부하패턴의 경우 트래픽 급증과 급감이 반복되며 부하의 감소속도가 빨라 동적종료 효과가 가장 크게 나타났다. 실제 부하패턴에서 예측이 음의 기여도를 갖는 이유는 예측으로 인해 켜진 서버들이 운영되는 시간이 짧아 낮은 기여도를 나타내기 때문이다.

실험 결과를 종합적으로 보면, QoS 측면은 실험8이 QoS가 가장 높게 측정된 실험6과 큰 차이가 없으며, 나머지 다른 실험보다 QoS가 우수하였다. QoS 기여도는 예측, 다중임계치 학습, 동적종료의 순으로 나타났다. 또한, 단위전력당 유효응답 수 측면은 실험8이 가장 우수하게 측정되었다. 단위전력당 유효응답 수 기여도는 동적종료, 다중임계치 학습, 예측순으로 나타났다.

이에 따라 제안된 방법, 즉 다중임계치 학습과 예측의 조건부 선택에 대한 동적종료 방법(실험8)이 기존 정적 서버 모드 제어 방식(실험0)과 기존 단일임계치 학습(실험1)보다 단위전력당 성능, QoS 측면에서 개선이 이루어지고 있음을 보여준다. 실험을 통해 단일임계 학습보다 다중임계 학습에서 단위전력당 유효응답 수와 QoS 측면에서 우수함을 확인할 수 있고, 동적종료는 사용자 요청에 약간의 손실이 있지만 빠른 OFF로 소비전력을 줄였다. 또한, 소비전력 예측 기법은 spike성 부하에서 QoS 향상에 우수함을 확인하였다.

5. 결론 및 향후 연구 방향

본 논문에서는 일반적 사용자 워크로드에서 자율학습기반의 동적 프로비지너닝을 선택하고 사용자 워크로드가 급변하는 상황에서는 예측기반의 동적 프로비저닝을 선택하는 방법을 제안하였다.

워크로드의 형태를 뱅킹 부하패턴, 실제 부하패턴, 사용자 요청이 급변하는 가상적인 경우로 나누어 실험을 진행하였고, 실험 결과는 다중임계 학습의 동적종료에서 spike성 부하에 대한 예측 추가를 통해 단위전력당 성능(유효응답 수)과 QoS 측면에서 가장 우수한 결과를 보여주었다.

정적 서버모드 제어, 파라미터 로드된 다중임계 학습과 예측의 조건부 선택에 대한 동적종료를 비교할 때 뱅킹 부하패턴, 실제 부하패턴, 가상 부하패턴에서 단위전력당 유효응답 수가 각각 5.35%, 11.64%, 49.92% 향상되고, QoS 관점에서는 각각 2.89%, 6.81%, 50.92% 향상되었다. 또한, 파라미터 로드된 단일임계 학습, 파라미터 로드된 다중임계 학습과 예측의 조건부 선택에 대한 동적종료와 비교할 때 뱅킹 부하패턴, 실제 부하패턴, 가상 부하패턴에서 단위전력당 유효응답 수가 각각 1.66%, 2.9%, 3.84% 향상되고, QoS 관점에서는 각각 0.45%, 1.33%, 8.82% 향상되었다.

현재의 실험은 리눅스 서버클러스터를 구성하여 부하의 유형에 따라 동적 재구성하도록 구성하였으며, 상용서버의 cpu의 고성능화, 메모리의 고용량화, 디스크의 단위용량당 비용이 저렴해져가고, KVM, XEN, VMware 같은 가상화 툴의 성능 향상 및 발전으로 인해 가상화 환경에서 서버 클러스터를 지원하도록 할 필요가 증가하고 있다. 또한, OpenStack과 같은 가상화 플랫폼을 적용한 클러스터를 구성하고 서버의 연결 종료시간을 단축하기 위해 동적 마이그레이션을 적용하는 실험을 통해 동적 프로비저닝 및 전력을 절감하는 추가적인 연구가 필요할 것으로 본다.

References

[1] Fanxin Kong and Xue Liu, "A Survey on Green-Energy-Aware Power Management for Datacenters," in *ACM Computing Surveys(CSUR)*, 2014.

[2] Chenguang Liu, Jianzhong Huang, Qiang Cao, Shenggang Wan, and Changsheng Xie, "Evaluating Energy and Performance for Server-Class Hardware Configurations," *6th IEEE International Conference on Networking*, Architecture and Storage, 2011.

[3] J. Mair, K. Leung, Z. Huang, "Metrics and task scheduling policies for energy saving in multicore computers," *11th IEEE/ACM International Conference on Grid Computing (GRID)*, 2010.

[4] G. Chen et. al., "Energy-Aware Server Provisioning and Load Dispatching for Connection-Intensive Internet Services," *NSDI'08 Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, 2008.

[5] A Krioukov, et al., "NapSAC: design and implementation of a power-proportional web cluster," *ACM SIGCOMM computer communication overview*, 2011.

[6] Abdul Hameed, Alireza Khoshkbarforoushha, Rajiv Ranjan, Prem Prakash Jayaraman, Joanna Kolodziej, Pavan Balaji, Sherali Zeadally, Qutaibah Marwan Malluhi, Nikos Tziritas, Abhinav Vishnu, Samee U. Khan, and Albert Zomaya, "A survey and taxonomy on energy efficient resource allocation techniques for cloud computing systems," *Springer Computing*, Jun., 2014.

[7] Anton Beloglazov, Rajkumar Buyya, Young Choon Lee, and Albert Zomaya, "A Taxonomy and Survey of Energy-Efficient Data Centers and Cloud Computing Systems," The University of Melbourne, Australia, The University of Sydney, Australia, 2010.

[8] Sungchul Cho, Hukeun Kwak, and Kyusik Chung, "An Energy Efficient Cluster Management Method based on Autonomous Learning in a Server Cluster Environment," *KIPS Transactions on Computer and Communication Systems*, Vol.4, No.6, pp.185-196, 2015.

[9] Taejune Ahn, Sungchul Cho, Seokkoo Kim, Kyongho Chun, and Kyusik Chung, "A Flexible Multi-Threshold Based Control of Server Power Mode for Handling Rapidly Changing Loads in an Energy Aware Server Cluster," *KIPS Transactions on Computer and Communication Systems*, Vol.3, No.9, pp.279-292, 2014.

[10] Hoyeon Kim, Chihwan Ham, Hukeun Kwak, and Kyusik Chung, "Dynamic Shutdown of Server Power Mode Control for Saving Energy in a Server Cluster Environment," *KIPS Transactions on Computer and Communication Systems*, 2013.

[11] LVS(Linux Virtual Server) [Internet], http://www. linuxvirtualserver.org.

[12] Sungchul Cho, Sanha Kang, Heungsik Moon, Hukeun Kwak, and Kyusik Chung, "Prediction of Power Consumption for Improving QoS in an Energy Saving Server Cluster Environment," *KIPS Transactions on Computer and Communication Systems*, Vol.4, No.2, pp.47-56, 2015.

[13] Hoyeon Kim, Chihwan Ham, Hukeun Kwak, Hulung Kwon, Youngjoung Kim, Kyusik Chung, "A Dynamic Server Power Mode Control for Saving Energy in a Server Cluster Environment," *The KIPS Transactions: PartC*, Vol.19, No.3, pp.135-144, 2012.

[14] SPECweb [Internet], http://www.spec.org/benchmarks.html/.

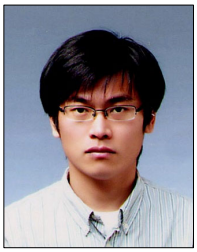
[15] Apache [Internet], http://www.apache.org/.

[16] InternetTrend [Internet], http://www.internettrend.co.kr.

[17] Direct Routing [Internet], http://www.linuxvirtualserver.org/VS-DRouting.html.

[18] H. Kwak, A. Sohn and K. Chung, "Autonomous Learning of Load and Traffic Patterns to Improve Cluster Utilization," *Cluster Computing*, Vol.14, Issue 4, Dec., 2011.

[19] Hukeun Kwak, Kyusik Chung, Hyung Won Choi, and Andrew Sohn "Enabling Scalabe Cloud Infrastructure using Autonomous VM Migration," *2012 IEEE 14th International Conference on High Performance Computing and Communications, 2012.*



조 성 철

e-mail : sccho@q.ssu.ac.kr
1998년 호서대학교 전자공학과(학사)
2000년 숭실대학교 전자공학과(석사)
2002년~2004년 한단정보통신
2007년~2008년 아시아나IDT
2009년~2011년 스트라텍
2015년 숭실대학교 전자공학과 박사
관심분야 : 임베디드 시스템 및 네트워크 컴퓨팅

정 규 식

e-mail : kchung@q.ssu.ac.kr
1979년 서울대학교 전자공학과(공학사)
1981년 한국과학기술원 전산학과 (이학석사)
1986년 미국 University of Southern California(컴퓨터공학석사)
1990년 미국 University of Southern California(컴퓨터공학박사)
1998년~1999년 미국 IBM Almaden 연구소 방문연구원
1990년~현 재 숭실대학교 스마트시스템소프트웨어학과 교수
관심분야 : 임베디드 및 네트워크 컴퓨팅