피드백 제어 이론을 이용한 실시간 웹서버 시스템의 부하 제어 기법

강 봉 직[†]·정 석 용[†]·김 영 일[††]·최 경 희[†††]·정 기 현[††††]·유 해 영[†††††]

요 약

본 논문에서는 실시간 웹 서비스를 제공해야 하는 시스템에서 과도한 웹 요청으로 인해 발행할 수 있는 부하를 조절하는 기법을 제안한다. 많은 기존의 기법은 단순히 고정된 기준에 의해 부하를 조절하기 때문에 시스템의 동적인 상태를 실시간으로 반영하지 않는다. 본 논문에서는 제어 이론을 기초로 시스템의 부하를 모델링하여 동적으로 부하를 반영할 수 있는 제어 방법을 제시한다. 제안 기법에 따라 성능 목표를 만족하는 제어기를 설계하였으며, 설계된 제어기를 적용하여 피드백 제어 시스템을 구성하고 다양한 실험 환경에서 제안된 방법으로 안정된 부하 조절이 가능함을 확인한다.

Load Control Mechanism for Real-Time Web Server Systems Based on Feedback Control Theory

Bong-Jik Kang[†] · Suk-Yong Jung[†] · Young-Il Kim[††]
Kyung-Hee Choi[†††] · Gi-Hyun Jung[††††] · Hae-Young Yoo[†††††]

ABSTRACT

This paper proposes a mechanism for managing overload introduced by excessive web requests in real-time web service system. The many previous mechanisms statically manage the overload and thus the dynamic characteristics are not reflected. This paper proposes a mechanism that models system load based on control theory approach and includes its dynamic characteristics. We design a controller that is able to meet performance requirement. A feedback control system is implemented applying the proposed mechanism and the stable operation of system is verified through various simulation environments.

키워드 : 제어 이론(Control Theory), 실시간 웹 서비스(Real-Time Web Service), 부하 제어(Load Control), 시스템 식별(System Identification), 패킷 접수 비율(Packet Acception Ratio), 피드백 제어 시스템(Feedback Control System)

1. 서 론

인터넷 환경이 급속히 확산됨에 따라 웹 기술의 응용 분야도 넓어져 실시간 내장형 시스템에까지도 적용되고 있다. 실시간 내장형 시스템이 인터넷을 통하여 접근이 가능하려면 반드시 웹서버를 내장하고 있어야 한다. 그런데 이러한 웹서버 시스템들은 인터넷의 개방성으로 인해 웹 요청의 소통량을 예측할 수 없기 때문에 웹서버로의 요청이 폭주하는 경우 커널은 접수되는 패킷을 처리하는데 많은 자원을 소비하게 되고, 웹 요청을 처리하기 위한 작업(이하 웹 태스크)들이 많아지게 되어 시스템은 과부하 상태가 되며, 이는 시스템의 실시간 속성을 만족시키지 못하는 결과를 초래할 수 있다. 그러므로 실시간 웹서버 시스템은 웹 요청이 폭주하는 경우에도 시스템의 과부하를 방지하여 예측 가능한 실시간 서비스를 제공해야 한다.

웹 요청 폭주로 인한 과부하를 방지하는 방법 중에는 웹 요청으로 파생된 웹 태스크의 수를 조절하는 방법 또는 웹 연결의 수를 조절하는 방법이 있다. 웹 태스크의 수를 조절하는 방법은 현재 실행중인 웹 태스크의 수를 감시하면서 새로운 웹 요청이 도착한 경우 헤더를 분석하여, 해당 클라이언트의 서비스 등급에 따라 웹 태스크 실행 여부를 결정하는 방법이다[1, 8, 12]. [1, 8, 12]에서는 클라이언트들 간의 서비스 품질 비율을 미리 정해놓고 웹 요청 분석 결과에 따라 파생되는 태스크 수를 조절하기 때문에 과부하 상태에서도 QoS 비율을 보장할 수는 있으나 과부하 자체를 방지하는 것은 어렵다. 과부하를 방지하여 실시간 서비스를

† 정 회 원 : 동양공업전문대학 전산경영기술공학부 교수
†† 준 회 원 : 아주대학교 컴퓨터공학과
††† 정 회 원 : 아주대학교 정보 및 컴퓨터공학부 교수
†††† 정 회 원 : 아주대학교 전자공학부 교수
††††† 정 회 원 : 단국대학교 정보컴퓨터학부 교수
논문접수 : 2003년 9월 25일, 심사완료 : 2003년 11월 8일

제공하기 위해서는 단순히 태스크 수를 조절하는 것보다 부하를 발생시킬 수 있는 원인을 사전에 제거하는 것이 필요하다. 예를 들면, 과부하가 예상되는 경우에는 연결을 접수한 후에 웹서버가 웹 태스크를 폐기하는 것보다 연결 요청 단계에서 조기에 연결 요청(SYN) 패킷을 폐기하는 것이 좋다. 그 이유는 웹 요청이 폭주하는 경우에는 SYN 패킷을 접수하여 연결을 처리하는 것 자체가 시스템에 큰 부하로 작용하기 때문이다. 하나의 웹 요청을 처리하는 지연(Latency)을 요청 도착부터 연결이 완성되기까지의 연결 지연(Connection Delay)과 연결 후 요청을 분석하고 처리하기까지의 처리 지연(Processing Delay)으로 구분하여 비교할 때, 연결 지연이 처리 지연보다 크다고 알려져 있다[8]. 이것은 SYN 패킷의 도착부터 연결을 완료하기까지의 부하가 웹서버가 분석하고 처리하는 부하보다 상대적으로 크다는 것을 의미한다. 결과적으로 폐기하여야 할 웹 요청이라면 최대한 조기에 폐기하여야 한다는 것이다. 따라서 실시간 서비스를 위해서는 웹 태스크 수를 조절하는 방법보다는 시스템의 부하 정도에 따라 SYN 패킷의 접수 비율을 조절하는 것이 더 적합하다.

웹 연결의 수를 조절하여 과부하를 방지하는 방법은 웹서버가 처리 가능한 연결 수의 임계치를 설정하고 매 SYN 패킷이 도착할 때마다 현재 접수 비율이 임계치 이하인 경우에만 해당 연결을 접수하고, 임계치를 넘는 경우에 SYN 패킷을 조기에 폐기(Drop)하여 웹 연결의 수를 일정 수준 이하로 조절하는 것이다[5, 12, 13]. [5, 12, 13]에서는 단순히 임계치만을 기준으로 하기 때문에 시스템의 동적인 효율 상태를 반영하지 못하여, 시스템의 부하가 낮은 경우에도 접수 비율이 고정되어 있기 때문에 전체적인 시스템 효율이 떨어질 수 있다. 또한 이 방법은 웹 요청의 내용을 파악하기 전에 연결 요청을 접수할지를 결정하기 때문에 중요한 요청이 담긴 패킷이라 할지라도 폐기되는 경우가 발생할 수 있다.

본 논문에서는 과부하를 사전에 방지하기 위하여 SYN 패킷의 접수 비율을 조절하되 단순히 임계치를 가지고 접수 비율을 조절하는 것이 아니라 실시간 시스템의 부하를 측정하여 그에 따라 패킷 접수 비율을 동적으로 조절하는 기법을 제안하고자 한다. 패킷 접수 비율을 동적으로 조절하는 기법은 시스템의 효율을 높임과 동시에 실시간 속성을 만족시켜야 한다. 이러한 알고리즘을 개발하기 위해서는 실시간 웹서버 시스템의 부하를 분석적으로 모델링 하여야 가능하다. 부하를 모델링 하기 위해서는 웹 태스크의 실제 속성들(실행시간 등)을 미리 예측할 수 있어야 한다. 그러나 이것은 웹 요청의 속성에 따라 결정되는 동적인 요소이기 때문에 분석적인 모델링이 쉽지 않다.

따라서 본 논문에서는 동적 시스템 모델링에 적합한 제어 이론을 적용하여 실시간 웹서버 시스템의 부하를 모델링하고, 이를 바탕으로 실시간 속성이 만족되도록 패킷 접수 비율을 동적으로 조절하는 제어 기법을 제안하고자 한다. 동적인 실시간 웹서버 시스템 모델링을 위하여 시스템 식별(System Identification)[2, 7] 기법을 적용하였고 패킷 접수 비율의 조절 문제를 피드백 제어 대상으로 두고 근궤적법(Root-Locus)[10]을 사용하여 피드백 제어기를 설계하였다. 설계된 제어기는 패킷 접수 비율을 시스템의 부하에 따라 동적으로 조절하여 웹서버 시스템의 과부하를 방지하고 실시간 속성이 유지될 수 있도록 한다.

본 논문의 구성은 다음과 같다. 2장에서는 일반적인 웹서버가 탑재된 실시간 시스템에 관련된 연구 및 시스템 식별과 근궤적에 관한 연구를 소개하고, 3장에서는 제어 이론을 적용한 웹서버 시스템 모델을 소개한다. 4장에서는 실험을 통하여 성능 평가를 기술한 후, 5장에서 결론 및 향후 연구를 기술한다.

2. 관련 연구

2.1 실시간 웹서버 시스템

실시간 웹 서비스를 제공하는 방법에는 웹서버와 커널의 이원화된 스케줄링을 통합하는 방법이 있다[6]. 이 방법에서는 웹서버의 일부 기능을 커널에 통합하여 스케줄링한다. 웹서버를 탑재한 실시간 내장형 시스템에서는 웹 브라우저가 보내는 요청을 담은 패킷을 운영체제가 분석하지 않는다. 패킷이 웹서버에 전달되어야 비로소 내용을 분석하고 그에 따라 해당 작업(웹 태스크)을 실행하게 된다. 따라서 웹 브라우저에서 실시간성을 가진 서비스가 요청된 경우에도 커널은 패킷에 실시간 서비스를 요청하는 내용이 기록되었는지 모르기 때문에 높은 우선순위로 요청된 실시간 서비스가 웹서버에 의하여 낮은 우선순위의 태스크보다 늦게 실행되는 우선순위 역전 현상이 발생할 수 있다. 이에 따라 장치의 실시간 조작 실패 혹은 실시간 작업의 실행 지연 등으로 인한 문제점이 나타나게 된다. 따라서 실시간 웹서버 시스템에서는 실시간 서비스가 웹 요청에 의해 실행될 수 있기 때문에 패킷을 웹서버의 도움 없이 커널이 직접 분석하고 태스크들을 통합 스케줄링 함으로써 우선순위 역전으로 인한 문제점을 해결하여 실시간 서비스를 제공한다[6].

그런데 패킷분석기는 웹 요청에 대한 연결 과정이 완료한 이후에 비로소 작동하게 된다. 따라서 통합 스케줄링 환경에서도 시스템은 여전히 외부에 노출되어 있기 때문에 웹 요청이 폭주하는 경우에는 연결 과정에 필요한 패킷을 처리하는 부하로 인하여 시스템이 과부하 상태가 되고 그로 인해 실시간 서비스가 실패할 수 있는 문제점을 가지고 있다.

시스템의 과부하를 방지하여 실시간 서비스를 제공하는

방법들도 있다. 웹서버 시스템의 과부하를 방지하는 가장 단순한 방법은 연결 수에 대한 임계치를 사용하는 방법이다[13]. 이것은 상업용 스위치나 라우터에서와 같이 접수되는 요청을 무작위 또는 가장 오래된 순서에 의하여 폐기하는 것이다. 이 방법은 요청 내용의 중요도에 관계없이 연결 요청을 강제로 폐기할 뿐 만 아니라, 시스템의 상태를 전혀 반영하지 못해 부하가 적은 상태에서도 고정된 비율로 연결 요청을 폐기하는 문제를 가지고 있다.

요청의 중요도를 반영하여 커널 수준에서 연결의 수를 조절하거나 연결 이후 실행되는 태스크 수를 조절함으로써 과부하 상태에서도 서비스 품질을 보장하는 방법이 있다[12]. 이 방법은 평상적인 경우에는 태스크 수의 조절만으로도 서비스의 품질 보장이 충분하지만, 과부하 상태에서는 SYN 패킷을 조절하는 메커니즘에 의해서만 목적을 달성할 수 있다. 이것은 과부하가 예상되는 경우에는 조기에 SYN 패킷의 접수를 조절해야 함을 의미한다. 그러나 이 방법에서도 여전히 시스템의 상태를 반영하여 SYN 패킷의 접수 비율을 조절하는 알고리즘은 제공하지 않고 있다.

2.2 시스템 식별(System Identification)

제어기 설계 절차는 시스템의 동작을 분석하여 모델을 만들고, 성능 규격을 만족하는 제어기를 제어이론에 따라 설계하고 성능을 평가하는 절차로 이루어진다[4, 10]. 시스템의 상태에 따라 패킷 접수 비율을 동적으로 조절하는 기법을 개발하기 위해서도 시스템 상태를 분석적으로 모델링하는 과정이 반드시 필요하다.

시스템을 모델링하는 방법은 전체 시스템을 미리 잘 알려진 동작 특성을 가진 서브시스템들로 분할하여 각 서브시스템의 동작을 차분 방정식이나 미분 방정식으로 표현하고, 이들을 수학적으로 결합하여 전체 시스템을 구성하는 방식과 전체 시스템을 블랙박스로 두고 시스템의 입·출력 자료의 분석을 통해 모델을 추론하는 시스템 식별의 방식이 있다[7]. 컴퓨터 시스템을 모델링하기 위하여 기계/전자 시스템과 같이 미분 방정식으로 시스템을 직접 표현하는 것은 시스템의 복잡성으로 인하여 어렵다[11]. 더구나 컴퓨터 시스템은 여러 소프트웨어가 복합적으로 연계되어 동작하고, 시스템을 구성하는 서브시스템의 내용(버전)도 자주 변경되어 모델 변경이 자주 발생한다. 인터넷 접속이 가능한 실시간 웹서버 시스템 또한 여러 가지 복잡한 요소에 의해 운영되는 동적인 시스템이기 때문에 동적인 상태(Dynamics)를 직접 모델링 하기 쉽지 않다. 그런데 시스템 식별 방법은 입·출력 데이터의 관계성을 선형 차분방정식의 형태로 표현하고 있는 모델 구조를 선택하고, 입·출력 데이터만으로 모델 파라미터를 결정하기 때문에 동적인 상태를 미분 방정식으로 직접 표현하지 않고도 동작상태를 모델링을 할 수 있다. 본 논문에서도 시스템 식별 기법을 활용하여 웹서버 시스템 부하를 모델링하고 입·출력 데이터로부터 모델 파라미터를 추정하여 시스템의 효율(Utilization)을 목표 수준으로 제어할 수 있는 제어기를 설계하여 패킷 접수 비율을 동적으로 조절하고자 한다.

2.3 근궤적(Root-Locus)

제어기의 역할은 대상 시스템의 출력이 원하는 특성을 갖도록 조절하는 것이다. 피드백 제어는 출력신호를 입력 단에 피드백해서 제어입력을 결정할 때 사용하는 기법이다. 시스템의 입력신호와 출력신호 사이의 전달특성은 미분방정식으로 표시한다. 이 미분방정식은 라프라스(Laplace)변환이나 z 변환으로 바꾸고 이 변환 함수들 사이의 비를 시스템의 입출력 전달 특성으로 삼는다. 이론적으로 라프라스 변환은 디지털 제어 시스템의 모델링에도 사용할 수 있지만 디지털이나 샘플링된 신호를 포함하는 라프라스 변환식은 전개과정에서 지수항을 포함하여 취급하는 것이 매우 어렵다[4, 10]. 이런 문제점을 해결하기 위하여 디지털 제어 시스템에서는 z 변환을 사용한다. 본 논문에서도 시스템의 모델식을 z 변환으로 시스템의 전달함수를 유도하여 제어기를 설계한다.

제어 시스템 전달함수 값을 무한대가 되게 하는 z의 값을 극점(pole)이라 하며, 함수 값을 0으로 하는 z의 값을 영점(zero)이라고 하는데 극점과 영점은 시스템의 특성에 결정적 영향을 미친다. 제어시스템의 모든 극점이 z 평면에서 단위원($|z|=1$)의 내부에 위치할 경우 제어시스템은 안정하다[4]. 안정된 시스템은 유한한 입력에 대해 유한한 크기의 응답을 보이는 것으로 다양한 부하의 상황에서도 시스템의 목표성능을 달성할 수 있음을 의미한다. 또한 극점이 z 평면의 단위원 내부에서 어느 곳에 위치하는가에 따라서도 시스템의 성능 특성이 달라진다. 따라서 제어기를 통해 시스템 극점의 위치를 변화시킬 경우 원하는 시스템 특성을 얻을 수 있다.

제어기 이득이 바뀜에 따라 변하는 극점의 위치 변화를 z 평면상에 그림으로 나타낸 것을 근궤적(root locus)이라고 한다[10]. 일반적으로 근궤적법을 이용하여 제어기를 설계하기 위해서는 성능규격을 만족하는 적당한 이득을 근궤적에 의해 선정하고, 정해진 이득에 따라 근궤적상에서 극점이 결정되고 그에 따른 성능 특성을 갖는 시스템을 설계할 수 있게 된다. 본 논문에서도 근궤적으로부터 성능목표를 달성하는 극점을 갖는 제어기를 설계한다.

3. 제어 시스템의 설계

제어 시스템을 설계하기 위하여 우선 제어 시스템의 구조(Control System Architecture)를 결정하고, 제어 대상 시스템의 동작을 시스템 식별 기법으로 모델링한 후, 시스템

의 출력이 원하는 특성을 보이도록 적절한 제어기를 설계한다.

3.1 피드백 제어 시스템 구조

과도한 웹 연결 요청으로 인한 부하를 조절하기 위해서 피드백 제어 시스템 구조에서는 (그림 1)과 같이 SYN 패킷 접수 비율($v(t)$)에 따라 선별적으로 연결을 허용하는 SYN 패킷 접수기(SYN Adaptor)가 필요하다. 네트워크 인터페이스 카드(NIC : Network Interface Card)에 수신된 SYN 패킷을 통한 신규 TCP 연결 요청은 하드웨어 인터럽트를 발생시켜 네트워크 서브시스템의 SYN 패킷 접수기를 구동시키고, SYN 패킷 접수기는 부하 조절을 위해 제어기에 의해 조절되는 SYN 패킷 접수 비율에 따라 일부 SYN 패킷의 경우 즉시 폐기하고, 나머지 패킷에 대하여는 새로운 소켓을 만들어 귀기울임 큐(Listen Queue)에 부착시킨 후 TCP 3-단계 핸드쉐이크(3-way handshake)를 성공적으로 완료하여 웹서버로 하여금 서비스를 제공할 수 있도록 한다. 이때 폐기된 SYN 패킷의 경우는 SYN+ACK 패킷을 기다리는 사용자의 타임아웃을 유발하게 되어 웹 서비스 요청을 지연시킴으로써 CPU 효율이 조절되는 효과가 있다. SYN 패킷의 접수 비율을 결정하는 제어기는 동적으로 변하는 시스템 상태를 정확히 반영하기 위하여 주기별로 CPU 효율($u(t)$)을 관측하여 초기 설정된 CPU 효율 목표값(U)에 도달할 때까지 제어 법칙(Control Law)에 따라 접수 비율을 결정한다.

3.2 시스템 모델링

제어 대상 시스템의 동작을 모델링하기 위해서 자동회귀 잉여입력(ARX : Autoregressive, eXtra input)[7] 모델로부터 전달 함수(Transfer Function)를 도출(Formulation)하고, 최소제곱 추정법(Least Square Estimation)[2, 7]에 의해 파라미터를 추정한다.

실시간 웹서버 시스템의 현재의 출력(CPU 효율)이 과거의 입력(패킷 접수 비율)과 출력에 의존한다고 가정할 때, 입력 $x(t)$와 출력 $y(t)$의 관계는 미지의 파라미터들을 갖는 차분 방정식(Difference Equation)으로 표현한다. 차분 방정식은 식 (1)과 같이 표현된다.

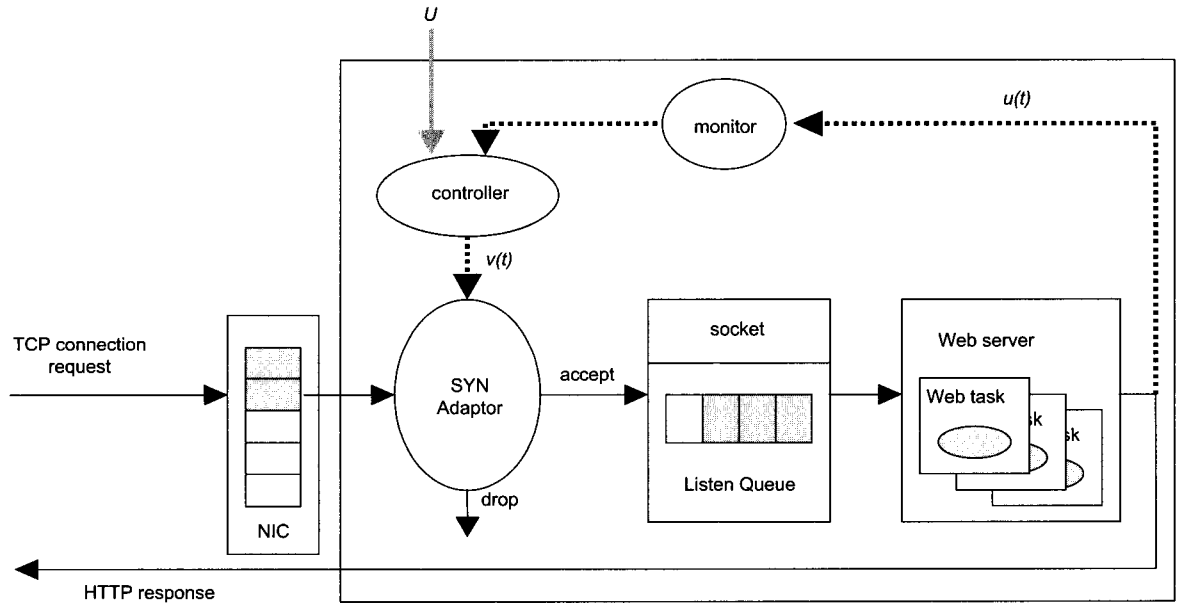
$$y(t) = \sum_{i=1}^{n} a_i y(t-i) + \sum_{j=0}^{m} b_j x(t-j) \qquad (1)$$

식 (1)의 파라미터 a_i, b_j들이 결정되면 식 (1)은 선형 시스템의 전달 함수가 된다. 용이한 분석 작업을 위하여 식 (1)을 시공간에서 주파수 공간으로 변환하는 z 변환 ($Y(z) \sum_{t=0}^{\infty} y(t) z^{-t}$)을 적용한다. 식 (1)을 z 변환하면 식 (2)와 같다.

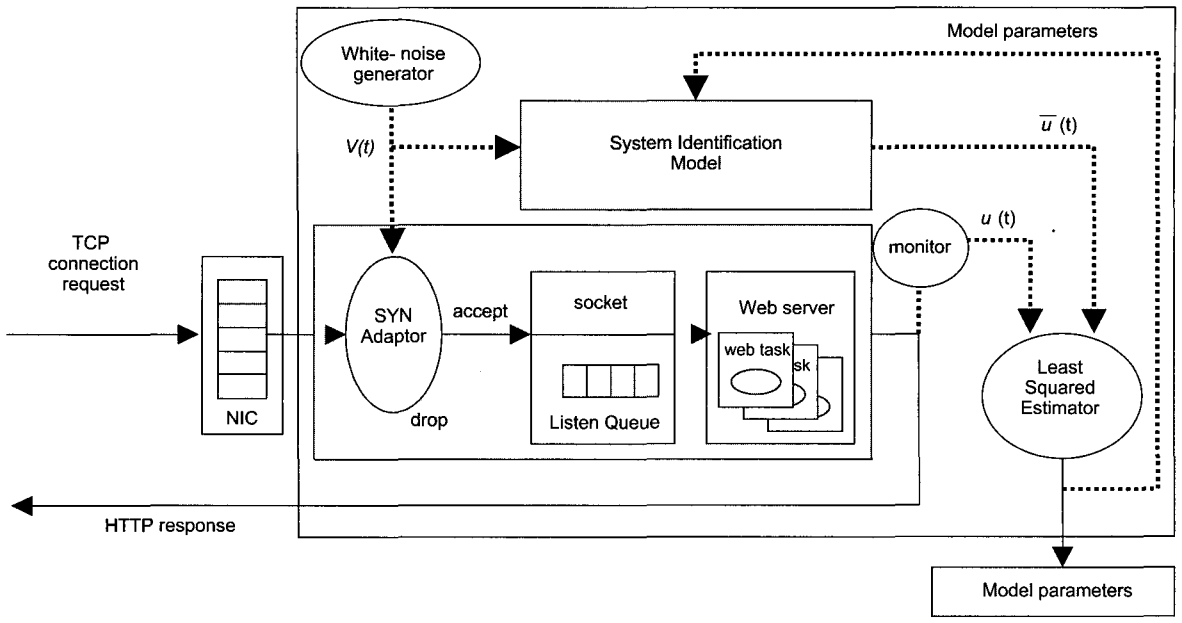
$$H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{j=0}^{m} b_j x^{m-j}}{z^n - \left(\sum_{i=1}^{n} a_i z^{n-i} \right)} \qquad (2)$$

식 (2)로부터 식 (3)의 시스템의 z 평면에서의 전달 함수 $G(z)$를 구한다. 시각 t에서 CPU 효율을 $u(t)$라하고, 패킷 접수 비율을 $v(t)$라고 할 때, $n=2$, $m=1$인 경우 시스템의 전달 함수 $G(z)$는 식 (3)과 같다.

$$G(z) = \frac{U(z)}{V(z)} = \frac{b_0 z + b_1}{z^2 - a_1 z - a_2} \qquad (3)$$



(그림 1) 피드백 제어 시스템 구조



(그림 2) 모델 파라미터 추정 구조

전달 함수 $G(z)$는 시스템의 내부 과정은 사용하지 않고 블랙박스와 같은 하나의 컴포넌트로 간주하여 모델링한 것이다.

모델 파라미터의 추정 과정은 시스템 식별 기법의 핵심이다. 식 (1)의 파라미터 a_i, b_j들은 최소제곱 추정법을 사용하여 추정할 수 있다. 최소제곱 추정법을 사용하여 파라미터를 추정하기 위한 구조는 (그림 2)와 같다.

(그림 2)에서 시스템은 주기마다 소음 생성기(White-noise generator)에 의해 임의로 변경되는 패킷 접수 비율 $v(t)$에 따라 요청을 접수한다. 접수된 요청에 한하여 웹서버가 응답을 한다. 감시기(Monitor)는 주기마다 출력값 $u(t)$를 측정한다. 최소제곱 추정기(Least Square Estimator)는 재귀 최소제곱 추정법(Recursive Least Square Estimation)에 따라 매 주기에서 반복적으로 현재 파라미터 추정량을 식 (1)에 대입하여 추정된 출력값 $\overline{u}(t)$를 계산한다. 추정 오차는 실제 출력값과의 차이 $(u(t)-\overline{u}(t))$가 된다. 최소제곱 추정기는 오차의 제곱이 최소화되도록 각 주기에서 파라미터 값 a_i, b_j를 반복적으로 계산하게 된다[2, 7]. 4.2절의 모델 파라미터 추정 실험 결과 파라미터 (a_1, a_2, b_0, b_1)의 값은 (0.454743, 0.519224, 0.15674, -0.14613)로 추정되었다.

3.3 제어기 설계

시스템 모델링 다음 과정은 제어 대상 시스템이 성능 목표를 충족하면서 동작할 수 있도록 제어기를 설계하고 평가하는 것이다. 실시간 시스템의 성능을 나타내기 위한 여러 가지 지표들을 사용하여 대상 시스템이 목표로 하는 성능을 나타낼 수 있다. 정상 상태와 더불어 과도 상태의 성능을 동적으로 나타내기 위하여 안정도, 과도 상태 응답, 정상 상태 오차 등으로 실시간 시스템의 동적인 성능 규격을 나타낸다[9]. 본 논문에서도 [9]에서 제시한 지표에 따라 시스템의 성능 목표를 정한다.

시스템은 안전성(Stability)을 보장하여야 한다. 안정된 시스템이란 범위 내 입력에 대하여 범위 내의 출력을 갖는 시스템은 말한다. 따라서 운영 중에 시스템의 CPU 효율이 특정 범위 내에서 제한되어야 함을 의미한다. 과도 상태 응답은 정상 상태 이전의 응답 특성을 말한다. 이것은 퍼센트 오버슈트(%OS : Overshoot)와 정착시간(Settling Time)으로 표현할 수 있다. 오버슈트는 최악의 과도 성능으로, 특히 실시간 시스템에서의 높은 오버슈트는 실시간 시스템의 실패를 유발할 수도 있는 중요한 지표이다[3]. 따라서 시스템의 실시간성 보장을 위하여 오버슈트를 최소화하여야 한다. 정착 시간은 안정 상태에 도달할 때까지의 시간으로 시스템이 얼마나 빨리 목표로 하는 CPU 효율에 도달하는지를 의미한다. 효율이 좋은 시스템이 되기 위해서는 짧은 정착 시간을 가져야 한다. 정상 상태 오차(Steady State Error)는 정상 상태에서 목표 값과 평균 CPU 효율의 차이이다. 따라서 제어 시스템의 정확성을 높이기 위해서는 정상 상태 오차가 0에 가까워야 한다.

(그림 1)과 같은 피드백 제어 시스템에서 제어기를 설계하기 위하여 제어 파라미터들이 필요하다. 제어 변수(Controlled Variable)는 샘플링 구간의 CPU 효율, $u(k)$이다. 조작 변수(Manipulated Variable)는 SYN 패킷 접수 비율 $v(k)$이다. 참조 변수(Performance Reference)는 희망하는 CPU 효율, U가 된다. 제어 입력(Control Input)은 $d(k)$, $v(k+1) = v(k) + d(k)$ 이다. 제어기는 희망하는 CPU 효율 U를 달성하기 위하여 오차 $e(k) = U - u(k)$를 0에 가깝게 줄여야 한다.

직관적으로 오차 $e(k)$가 음수인 경우 CPU 효율을 감소시키기 위해 $u(k)$를 줄여야 한다. 본 논문에서는 제어기 설계를 위하여 비례 적분(PI) 제어법[4, 10]을 사용하였다. z 평면에서 비례 적분 제어기는 아래와 같은 방정식으로 표현된다.

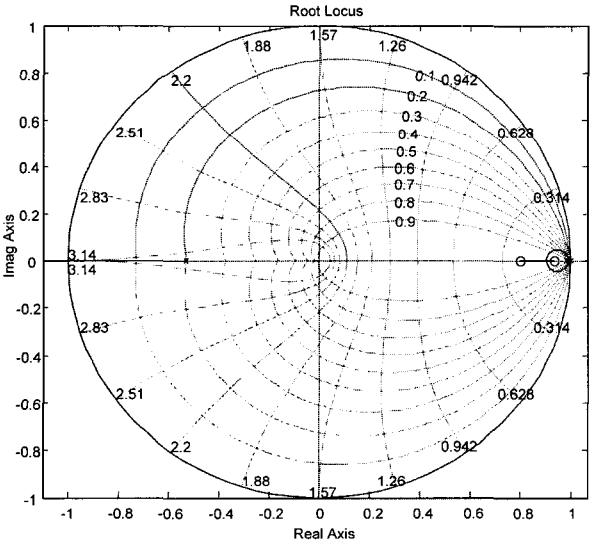
$$D(z) = K_P + \frac{K_I T_z}{z-1} = \frac{(K_P + K_I T)z - K_P}{z-1} = \frac{g(z-r)}{z-1} \quad (4)$$

$$where \quad g = K_P + K_I T, \; r \frac{K_P}{K_P + K_I T}$$

위 식에서 K_P는 비례 이득(Proportional Gain)이고 K_I은 적분 이득(Integral Gain)이다. 따라서 (그림 1)에 표현된 전체 폐회로 시스템의 전달 함수 $G_c(z)$는 다음과 같다.

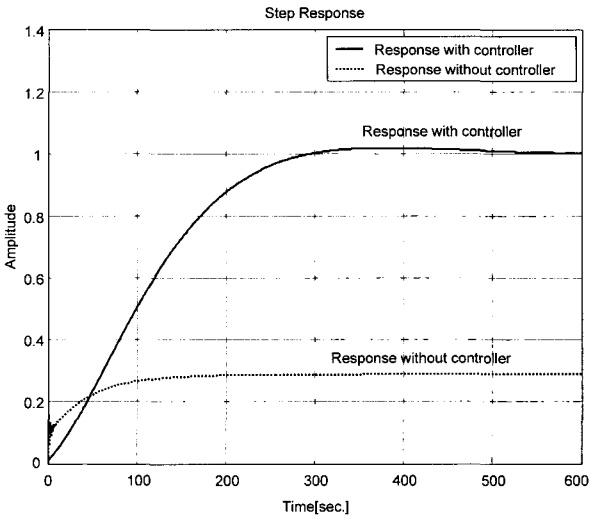
$$G_c(z) = \frac{D(z)G(z)}{1 + D(z)G(z)} \quad (5)$$

비례 적분 제어기의 이득을 결정하기 위해서 g와 r을 구해야 한다. 본 논문에서는 근궤적법을 이용하여 g와 r을 결정한다. 근궤적법은 제어 파라미터의 변화에 따른 폐루프 시스템의 극점의 위치 변화를 z 평면상에 그림으로 표현하는 기법이다. 결정된 모델 파라미터를 바탕으로 폐루프 시스템의 극점을 MATLAB의 근궤적 도구를 사용하여 그린 결과는 (그림 3)과 같다.



(그림 3) 근궤적도

(그림 3)의 근궤적도에서 점선으로 그려진 감쇠비(Damping Ratio) 선도과 근궤적이 교차하는 지점을 선택하여 제어기 파라미터 g와 r이 각각 0.1164, 0.8로 결정되었고 극점은 $p_0 = -0.5436$, $p_{1,2} = 0.9900 \pm 0.0078i$ 에 배치되었다. (그림 4)는 단위 계단 입력에 대하여 제어기가 없는 시스템과 결정된 제어기 파라미터를 갖는 피드백 제어 시스템의 응답 특성을 보여준 것이다. (그림 4)의 하단 그래프와 같이 제어기가 없는 경우에는 정상 상태 오차가 약 0.7 정도로 크다는 것을 알 수 있다. 반면 피드백 제어 시스템은 상단 그래프와 같이 목표 값 1이 달성되고 있음을 알 수 있다. 이것은 목표 성능을 달성하기 위해서는 제어기 설계가 반드시 필요하다는 것을 의미한다.



(그림 4) 단위 계단 응답

제어 이론에 의하면 피드백 시스템의 극점들이 모두 z 평면상의 단위 원 내부에 존재할 때 시스템의 안정성은 보장된다고 한다[2, 4, 10]. 따라서 본 논문에서 설계한 피드백 시스템의 극점 p_0, p_1, p_2들도 모두 z 평면상의 단위 원 내부에 존재하기 때문에 안정도가 보장됨을 의미한다. 그리고 과도 응답의 경우, 피드백 시스템의 오버슈트는 1.87로 최악의 과도 응답에도 목표 지표의 1.87% 이내의 응답 특성을 보여준다. 정착 시간은 제어 이론에 의하면 극점과 원점 사이의 거리가 가까울수록 짧아진다. 위와 같은 g, r에 대하여 정착 시간은 379초이고 정상 상태 오차는 0이다. 이것은 설계된 제어기를 갖는 피드백 제어 시스템이 목표로 하는 CPU 효율을 달성할 수 있음을 의미한다.

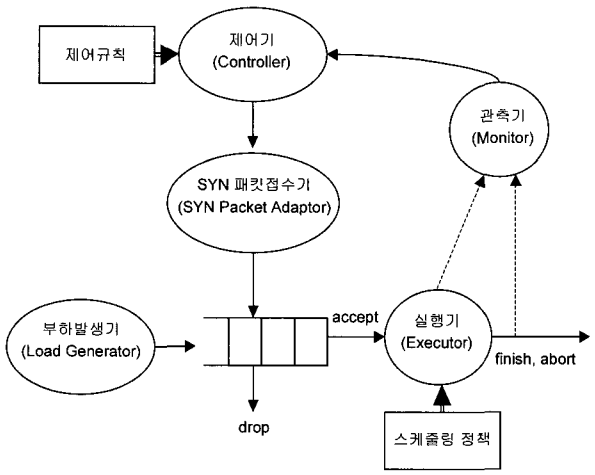
결과적으로 피드백 제어 이론의 시스템 모델링 기법과 근궤적법을 이용하여 시스템의 CPU 효율을 제어하는 실시간 시스템의 제어기를 설계할 수 있다. 이것은 제어 이론 기반의 기법이 동적인 컴퓨터 시스템에의 적용 가능성이 높다는 것을 의미한다.

4. 성능 평가

4.1 실험 환경 및 방법

본 절에서는 모의실험을 통하여 피드백 제어 시스템의 성능을 평가한다. 다양한 부하 상황에서도 목표 효율 U를 달성할 수 있도록 설계된 제어기를 웹서버 시스템에 적용하여 피드백 제어 성능을 실험한다. 모의실험을 위하여 (그

림 5)의 구조를 갖는 실시간 웹서버 시스템 모의 실험기를 제작하였다. 모의실험기는 부하발생기(Load Generator), 실행기(Executor), 관측기(Monitor), 제어기(Controller), SYN 패킷접수기(SYN Adaptor)의 5개 컴포넌트로 구성하였다.



(그림 5) 모의실험기 구조

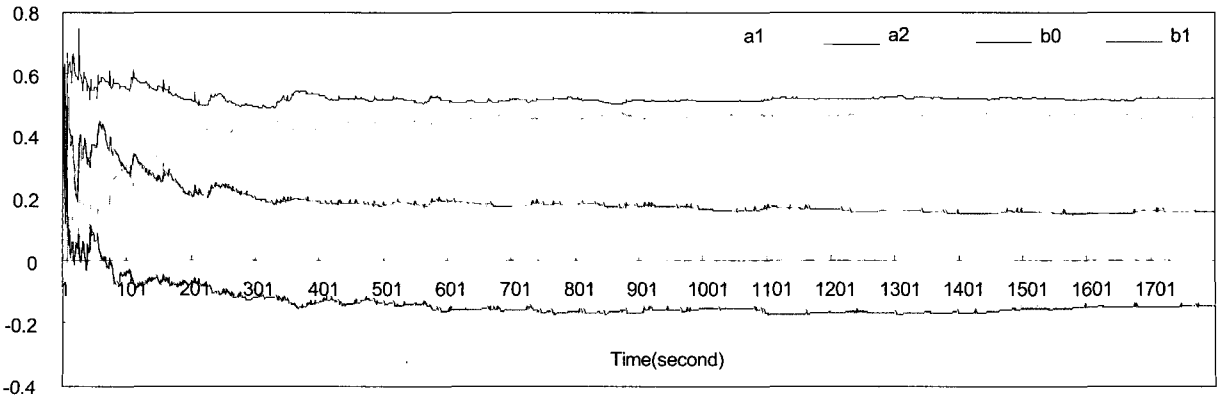
부하발생기는 성능 실험에 사용할 태스크를 발생시킨다. 모의실험에 사용될 웹 태스크들은 도착율이 평균 λ를 갖는 포아송 분포를 따른다고 가정하였다. 실행기는 태스크의 실행과 스케줄링을 수행한다. 시스템 내 상존하는 평균 태스크 수가 k인 경우에 시스템의 평균 효율이 1(100%)이 되도록 태스크의 실행 시간을 $1/k$로 정하였다. 따라서 시스템의 평균 부하는 $\rho = \lambda / k$이다. 태스크의 마감시간은 1초로 설정하였다. 관측기는 1초를 샘플링 주기로 하여 제어변수인 시스템 효율 U를 측정한다. 시스템 효율은 해당 주기에 대한 CPU 점유시간의 백분율로 구한다. 제어기는 관측기를 통해 주기적으로 측정한 시스템 효율과 목표 효율의 차이에 따라 제어 입력인 SYN 패킷 접수 비율을 결정한다. 제 3장의 제어기 설계를 통해 도출한 PI 제어를 적용하여 제어기를 구현하였다. PI 제어기 전달함수의 파라미터 g, r은 제 3장에서 근궤적법으로 성능 목표를 달성하도록 결정된 0.1164, 0.8을 사용하였다. 제어기는 폐루프 시스템이나 개루프 시스템에 대하여 각각 실험할 수 있도록 하기 위하여 제어기의 동작을 켜거나 끌 수 있도록 구성하였다. SYN 패킷접수기는 시스템이 목표 효율을 유지할 수 있도록 SYN 패킷 접수 비율에 따라 선별적으로 태스크를 접수하도록 하였다.

과부하 상황에서 피드백 제어 시스템의 성능을 평가하기 위하여 다음과 같은 순서로 실험을 수행하였다. ① 시스템 모델을 결정하기 위하여 모델 파라미터를 추정한다. ② 과부하에 따른 마감 시간 초과 비율과 태스크 처리량의 변화를 실험하여 과부하로 인한 실시간 시스템의 특성과 성능을 알아보고 과부하 제어의 당위성을 보인다. ③ 다양한 과부하 상황에서 시스템의 목표 효율이 피드백 부하 제어에 따라 효과적으로 유지되는지의 여부를 실험하여 제어기 성능을 확인한다. ④ 웹으로부터의 부하가 동적으로 변하는 경우와 목표 효율을 운용 중에 변경시키는 경우에도 제어기가 설정된 시스템의 목표 효율을 달성하는지의 여부를 실험하여 제어기 성능을 확인한다.

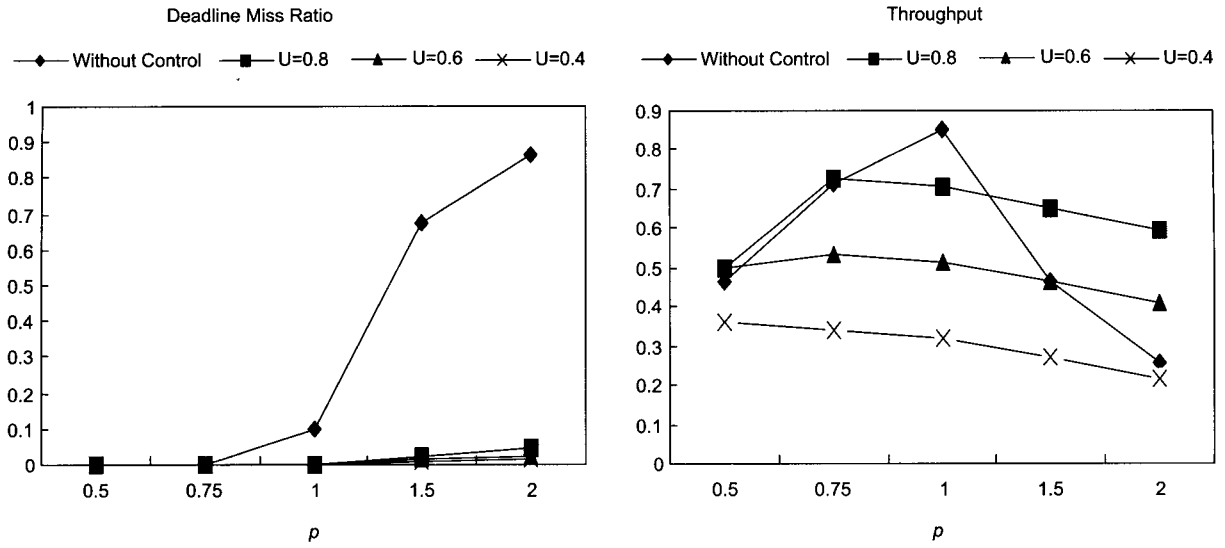
모의실험에서는 서로 다른 U, ρ 변수 값을 갖는 태스크 집합에 대하여 실험하였으며, 각 태스크 집합은 30분 동안 도착한 태스크로 구성하였다.

4.2 모델 파라미터 추정 실험 결과

우선 시스템 모델을 결정하기 위하여 3.2절에 기술한 방법에 따라 태스크 집합에 대해 재귀 최소제곱 추정기법을 적용하여 식 (1)로 표현된 시스템 모델의 파라미터 a_i, b_j의 값을 추정하였다. (그림 6)은 최초 (a_1, a_2, b_0, b_1)의 값을 (0.5, 0.5, 0.5, 0.5)로부터 파라미터 추정을 시작하여 재귀 최소제곱 추정 과정에 따라 파라미터 값들이 시간에 따라 수렴하고 있음을 알 수 있다. 본 논문에서는 차수가 2인 경우 가장 적은 오차가 보여주었기 때문에 식 (1)의 차수를 $n = 2$, $m = 1$로 고정하였다. 이 경우 파라미터 (a_1, a_2, b_0, b_1)의 값은 (0.454743, 0.519224, 0.15674, -0.14613)로 추정되었다.



(그림 6) 시스템 모델 파라미터 추정 과정



(그림 7) 마감 시간 초과 비율과 처리량

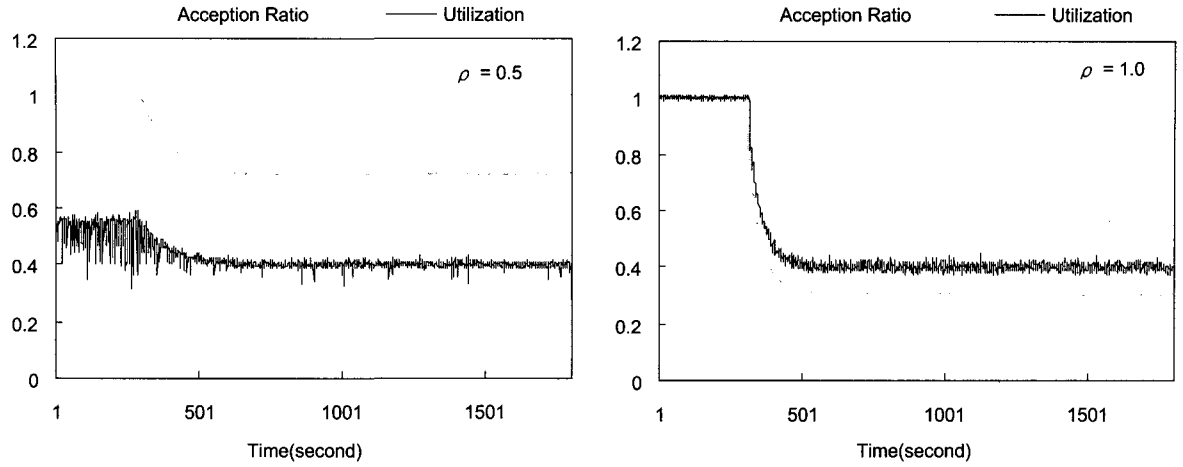
4.3 과부하에 의한 실시간 시스템의 특성 및 성능 실험 결과

과부하로 인하여 야기되는 실시간 시스템의 특성과 성능을 실험하였다. (그림 7)은 여러 가지의 서로 다른 부하 ρ (0.5, 0.75, 1.0, 1.5, 2.0) 환경에서 다양한 목표 효율 U(0.8, 0.6, 0.4)에 대하여 제어를 하는 경우와 제어를 하지 않는 경우 마감 시간 초과 비율(Deadline Miss Ratio)과 처리량(Throughput)을 측정한 결과이다. 그림에서 제어를 하지 않는 경우에는 부하가 증가함에 따라 마감 시간 초과 비율이 비례하여 증가함을 알 수 있다. 그러나 제어기가 동작하여 목표 효율(Utilization)을 유지하는 경우에는 마감 시간 초과 현상이 거의 발생하지 않음을 알 수 있다. 이것은 과부하 상황에서 연결 요청 수를 적절히 조절함으로써 실시간 속성이 만족될 수 있다는 것을 의미한다. 한편 처리량(실행 완료 태스크 비율) 관점에서 성능 측정 결과를 살펴보면, 제어를 하지 않는 경우에는 웹 요청의 증가에 따라 처리량도 증가하지만 과부하($\rho>1$) 상황에서는 처리량이 급격히 감소하여 성능이 떨어짐을 알 수 있다. 물론 과부하 상태에서 CPU 효율 값은 1이다. 이는 과부하 상황에서 CPU 효율 값은 지표상으로는 높더라도 처리 성능은 떨어진다는 것을 의미한다. 반면 제어를 하는 경우에는 과부하 상황에서도 처리량의 급격한 감소 현상이 발생하지 않고 유지됨을 알 수 있다. 따라서 과부하 상황에서도 CPU 효율을 적절한 값으로 제어할 수 있다면 시스템의 실시간 속성을 만족하면서 좋은 성능을 얻을 수 있다는 것을 의미한다. 물론 실시간 속성을 만족하면서 최고의 성능을 얻기 위해서는 최적의 CPU 효율 값 U를 제어 목표로 정해야 한다. 그러나 이러한 최적의 CPU 효율 값 U를 결정하는 문제는 환경에 따라 달라지고, 본 논문에서 관심을 가지고 있는 제어 기법과는 다른 문제이므로 더 이상 다루지 않는다.

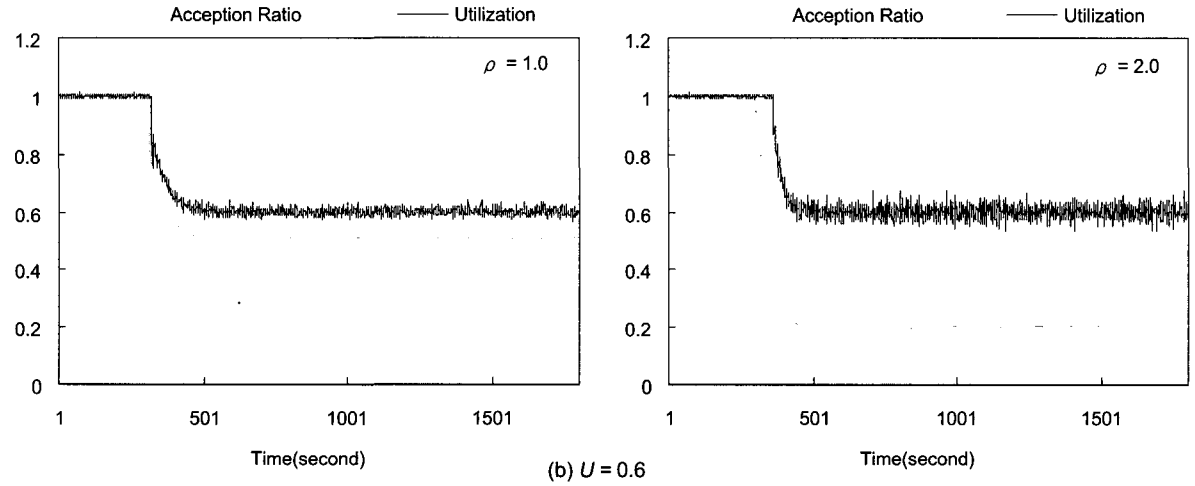
4.4 제어 성능 실험 결과

피드백 제어 시스템의 제어 성능을 평가하기 위하여 식 (4)의 PI 제어기 파라미터 g, r이 각각 0.1164, 0.8인 제어기를 적용하여 다양한 부하 ρ와 목표값 U에 대해 실험하였다. 그러나 $\rho \leq U$인 경우는 제어가 필요하지 않기 때문에 $\rho>U$인 경우에 한해 실험하였다. 이 실험에서 초기 접수 비율(Acception Ratio)은 모든 패킷을 접수하도록 초기화하였으며, 제어기가 없는 경우와의 비교를 위하여 제어기는 300초부터 작동하도록 하였다. (그림 8)은 목표 효율을 고정하고 세 개의 서로 다른 부하 ρ(0.5, 1.0, 2.0) 값에 따른 제어 결과를 보인 것이다. 제어기가 동작하기 이전에는 과부하 상태이면서 트래픽 제어를 하지 않고 모든 패킷을 받아들이기 때문에 CPU 효율이 목표 효율 U보다 높은 상태이다. 그러나 제어기가 작동하면서 세 개의 서로 다른 부하 ρ(0.5, 1.0, 2.0)에 대해 패킷 접수 비율을 조절하여 목표 효율이 달성되고 있음을 알 수 있다. 반면 그림에서 부하가 $\rho=2.0$인 경우에는 제어기가 작동하여 접수 비율이 감소한 이후에도 효율이 즉시 제어되지 않고 있음을 보여준다. 이는 제어기가 접수 비율을 점차적으로 감소하여 SYN 패킷을 폐기하여도 과부하 상태이기 때문에 일정 시간 이후에야 제어 효과가 나타나 효율값이 감소한다는 것을 의미한다.

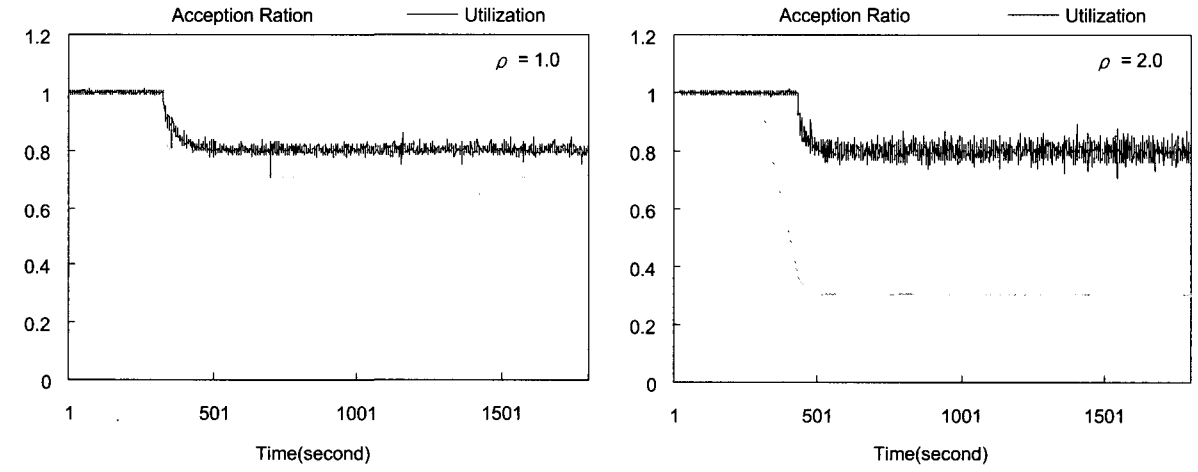
(그림 9)는 고정된 부하 환경에서 세 개의 서로 다른 목표 효율 U(0.4, 0.6, 0.8)에 따른 제어 결과를 보인 것이다. 이 경우에도 제어기가 동작한 이후에는 목표 효율이 달성되고 있음을 알 수 있다. 반면 그림에서 부하가 큰 경우가 작은 경우보다 오차가 큰 것을 알 수 있다. 이것은 안정된 상태에서 부하가 큰 환경이 제어기의 접수 비율 변화에 더욱 민감하다는 것을 의미한다.



(a) $U = 0.4$



(b) $U = 0.6$

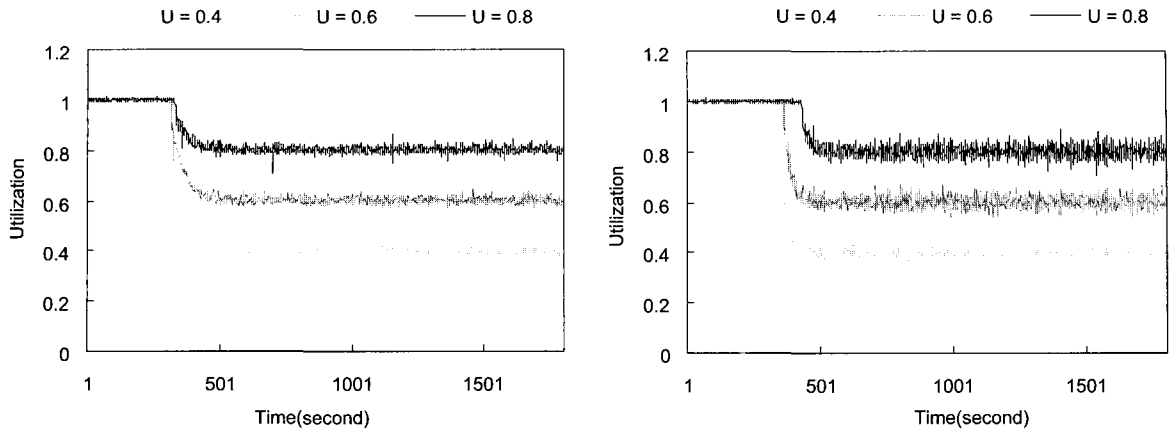


(c) $U = 0.8$

(그림 8) 제어 성능

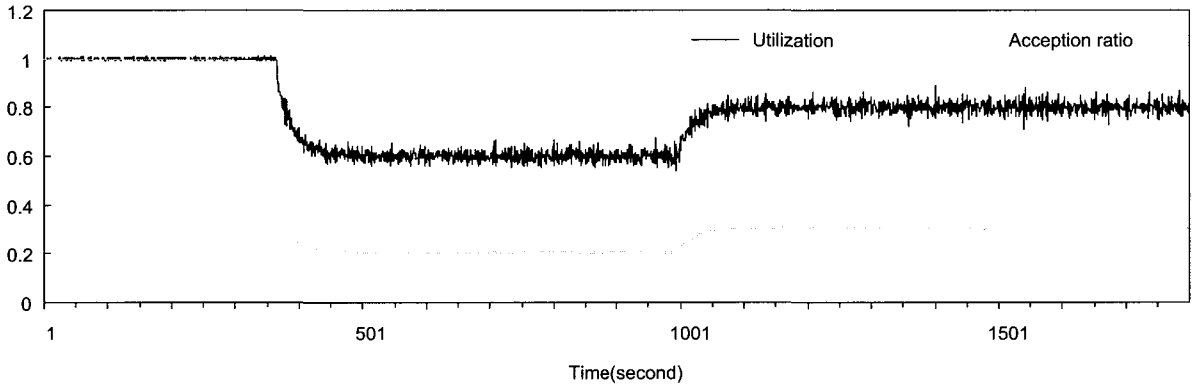
4.5 부하 및 목표효율 변동에 따른 제어성능 실험 결과

시스템 운영 중에 트래픽이 변동하는 경우에도 제어기가 목표 효율을 달성할 수 있는지 확인하기 위한 모의실험을 하였다. (그림 10)은 목표 효율을 고정하고 부하 ρ의 값을 시작부터 1000초까지는 0.8로 유지하고 그 이후에는 1.2로 부하를 증가시킨 경우 제어 결과를 보인 것이다. 이 경우에



(a) ρ = 1.0 (b) ρ = 2.0

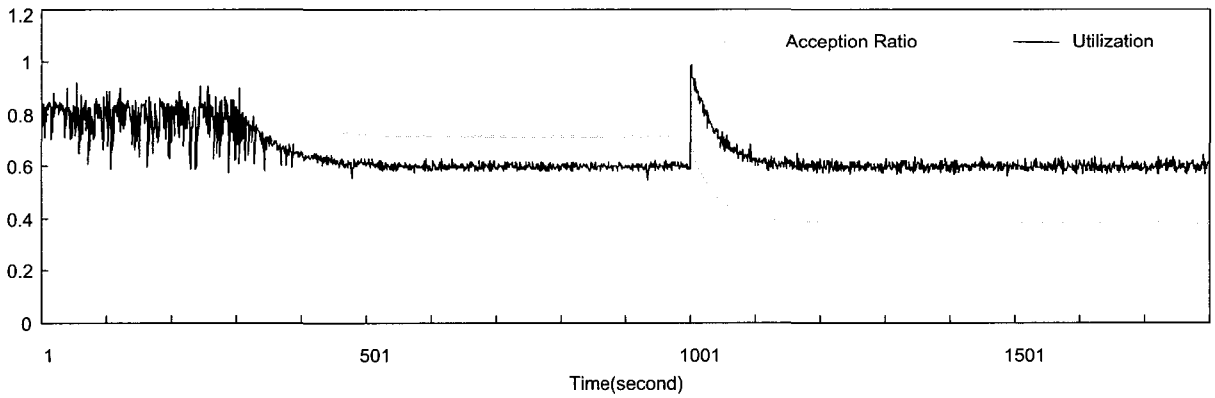
(그림 9) 목표 효율 U에 따른 제어 성능



(그림 10) 트래픽 변경시 제어 성능 (U = 0.6)

도 제어가 동작한 이후에는 패킷 접수 비율이 조절되어 목표 효율이 유지되고 있으며 부하가 증가한 1000초부터는 일시적으로 CPU 효율이 증가되었으나 제어기의 접수 비율 제어에 따라 곧 목표치에 도달하고 있음을 알 수 있다. 이는 부하가 변동하는 경우에도 시스템의 목표 성능을 달성할 수 있음을 의미한다.

시스템 운영 중에 목표 효율을 변경하는 경우에도 제어가 가능한지 확인하기 위해 모의실험을 하였다. (그림 11)은 목표 효율을 변경한 경우 제어 결과를 보인 것이다. 그림에서 1000초까지는 효율이 목표값 0.6으로 제어되고 1000초부터 효율이 변경된 목표값 0.8을 유지하도록 제어되고 있음을 알 수 있다. 이는 시스템의 목표 효율을 동적으로



(그림 11) 목표 효율 변경시 제어 성능 (ρ = 2.0)

변경하는 경우에도 제어가 가능함을 의미한다.

이상의 모의실험을 통해, 피드백 제어 시스템은 안정적인 성능을 보여주었으며 여러 가지 과부하 환경에서 목표로 하는 효율을 달성할 수 있도록 항상 제어 가능하다는 사실을 확인할 수 있었다.

5. 결 론

과도한 웹 요청으로 발생하는 부하를 조절하기 위한 기존의 방법은 실제 시스템의 동적인 효율 상태 변화나 외부의 부하 변동 상황에서 효과적으로 부하를 조절하기 어렵다. 이는 웹 연결 수나 웹 태스크 발생 수를 시스템의 동적인 효율 상태와 연동하여 조정하지 않고 단순히 임계치만을 가지고 조절하기 때문이다. 본 논문에서는 효과적인 부하 조절을 통해 과부하를 사전에 방지하기 위하여 실시간 시스템의 부하를 측정하고 그에 따라 패킷 접수 비율을 동적으로 조절하는 기법을 제안하였다. 제안된 기법은 실시간 웹서버 시스템을 시스템 식별 기법을 통해 모델링하고 근궤적법에 따라 성능 규격을 만족하는 피드백 제어기를 설계함으로써 시스템 부하에 따라 패킷 접수 비율을 동적으로 조절할 수 있도록 하였다. 다양한 실험 결과 제안된 기법이 여러 가지 부하 상황에서도 안정된 제어 성능을 보임을 확인하였다.

설계된 피드백 제어기는 제어기 파라미터의 선택에 따라 그 성능이 달라지며, 이는 제어 대상 시스템의 성능 규격을 고려해 결정해야 한다. 적절한 성능 규격의 결정은 응용 분야에 의존적이지만, 제어 이론을 적용한 기법이 효과적인 제어기를 설계할 수 있는 체계적인 방법론임을 알 수 있었다.

본 연구팀은 향후 시스템 효율뿐만 아니라 다른 실시간 시스템 속성을 조절하는 기법을 제어 이론을 바탕으로 연구하고, 이를 통해 컴퓨터 시스템에 대한 제어 이론의 유용성이 확대될 것으로 기대한다.

참 고 문 헌

[1] J. Almeida, M. Dabu, A. Manikutty and P. Cao, "Providing Differentiated Levels of Service in Web Content Hosting," *Proc. Workshop on Internet Server Performance*, pp. 91-102, March, 1999.

[2] K. J. Astrom and B. Wittenmark, *Adaptive Control*(2nd Ed.), Addison-Wesley, 1995.

[3] S. Brandt, G. Nutt, T. Berk and J. Mankovich, "A Dynamic Quality of Service Middleware Agent for Mediating Application Resource Usage," *Proc. IEEE Real-Time Systems Symposium*, Madrid, Spain, pp.307-317, December, 1998.

[4] G. F. Franklin, *Digital Control of Dynamic Systems*, 1998.

[5] R. Iyer, V. Tewari, and K. Kant, "Overload Control Mechanisms for Web Servers," *Proc. Workshop on Performance and QoS of Next Generation Networks*, Nagoya, Japan, pp. 225-244, November, 2000.

[6] S. Jung, B. Kang, K. Choi and K. Chung, "Integrated Scheduling for Reducing the Delays by Priority Inversion in Real-Time Web Service," *IEICE transactions on communications*, Vol.E86-B, No.7, pp.2143-2153, July, 2003.

[7] L. Ljung, *System Identification Theory for the User*, Prentice Hall, 1999.

[8] C. Lu, T. Abdelzaher, J. Stankovic and S. H. Son, "A Feedback Control Approach for Guaranteeing Relative Delays in Web Server," *Proc. IEEE Real-Time Technology and Applications Symposium*, pp.51-62, June, 2001.

[9] C. Lu, J. A. Stankovic, T. F. Abdelzaher, G. Tao, S. H. Son, and M. Marley, "Performance Specifications and Metrics for Adaptive Real-Time Systems," *Proc. IEEE Real-Time Systems Symposium*, Orlando, FL, pp.13-23, December, 2000.

[10] N. S. Nise, *Control System Engineering*, John Wiley & Song, Inc, 2000.

[11] S. Parekh, N. Gandhi, J. Hellerstein, D. Tilbury, T. Jayram, and J. Bigus, "Using control theory to achieve service level objectives in performance management," *Proc. IFIP/IEEE International Symposium on Integrated Network Management*, Seattle, WA, pp.841-854, May, 2001.

[12] T. Voigt and P. Gunningberg, "Adaptive resource-based web server admission control," *Proc. Symposium on Computers and Communication*, Taormina/Giardini Naxos, Italy, pp.219-224, July, 2002.

[13] "Cisco TCP intercept," http://www.cisco.com

강 봉 직

e-mail : bjkang@dongyang.ac.kr

1989년 서울대학교 계산통계학과 학사

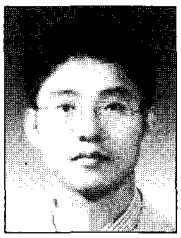
1991년 한국과학기술원 전산학과 석사

1991년~1994년 포스데이타 근무

2000년 아주대학교 컴퓨터공학과 박사 과정 수료

1995년~현재 동양공업전문대학 전산경영기술공학부 재직

관심분야 : 운영체제, 실시간 시스템, 내장형 시스템



정 석 용

e-mail : syjung@dongyang.ac.kr

1987년 서울대학교 계산통계학과 학사

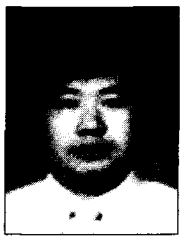
1993년 한국과학기술원 전산학과 석사

1987년~1995년 LG정보통신 근무

2000년 아주대학교 컴퓨터공학과 박사 과정 수료

1996년~현재 동양공업전문대학 전산경영기술공학부 재직

관심분야 : 운영체제, 실시간 시스템, 내장형 시스템



김 영 일

e-mail : yikim@kerinet.re.kr

1981년 충남대학교 전자교육공학과(학사)

1992년 아주대학교 전자계산학과(석사)

1989년~1998년 경기도 과학교육원, 경기도 교육정보연구원

1999년~2002년 경기도교육청 장학사

2002년~현재 고색중학교 교감

1995년~현재 아주대학교 컴퓨터공학과 박사과정(수료)

관심분야 : 운영 체제, 분산시스템, 실시간 및 멀티미디어시스템

최 경 희

e-mail : khchoi@madang.ajou.ac.kr

1976년 서울대학교 사범대학 수학교육과 학사

1979년 프랑스 그랑데꼴 ENSEEIHT 정보공학과 석사

1982년 프랑스 Paul Sabatier 정보공학과 박사

현재 아주대학교 정보 및 컴퓨터공학부 재직

관심분야 : 운영체제, 분산 시스템, 실시간 및 멀티미디어시스템 등

정 기 현

e-mail : khchung@madang.ajou.ac.kr

1984년 서강대학교 공과대학 전자공학과 학사

1988년 University of Illinois, EECS 석사

1990년 Purdue University 박사

1984년~1986년 FACOM Korea 근무

1991년~1992년 현대전자 반도체 연구소 근무

현재 아주대학교 전자공학부 재직

관심분야 : 컴퓨터구조, VLSI 설계, 멀티미디어 및 실시간 시스템 등

유 해 영

e-mail : yoohy@dankook.ac.kr

1979년 단국대학교 수학과(이학사)

1982년 단국대학교 대학원 수학과 수료(이학석사)

1994년 아주대학교 대학원 컴퓨터공학과 수료(공학박사)

1983년~현재 : 단국대학교 정보컴퓨터학부 재직

관심분야 : 소프트웨어공학, 정보화 전략계획수립 및 컨설팅, 웹 트래픽 튜닝 등