제18회 한국정보처리학회 추계학술발표대회 논문집 제9권 제2호 (2002.11)

대기이론기법 기반 웹서버의 이산사건 모델링

민병석*, 남의석*, 김학배*
*연세대학교 전기전자공학과
e-mail: hbkim@yonsei.ac.kr

A Discrete-Event Modeling of a Webserver Using Queueing Schemes

Byungseok Min*, Euiseok Nahm*, Hagbae Kim*
*Dept of Electrical & Electronic Engineering, Yonsei University

요 약

원활한 웹서버의 성능유지를 위하여, HTTP 트래픽의 특성에 대한 분석과 웹서버의 적절한 튜닝이 요구되고 있지만 이에 대한 연구는 아직 미진한 상태이다. 특히, 현재 대부분의 어플리케이션이 HTTP 1.1에 기반하여 구현되고 있음에도 불구하고, 대부분의 연구들이 HTTP 1.0에 기반하여 성능 분석이 이루어진 반면 HTTP 1.1에 대한 모델링과 성능분석은 거의 이루어지지 못하였다. 따라서, 본 논문에서는 지속연결을 지원하는 HTTP 1.1 프로토콜을 기반으로 하여 서버내의 세부 하드웨어 특성 등을 고려하여, 웹서버가 사용자의 요청을 받아들이면서부터 서비스를 마칠 때까지의 과정을 Tandem 네트워크 큐잉 모델을 사용하여 해석적인 웹서버 모델을 제안한다.

1. 서론

현재의 웹서버는 TCP/IP 에서 브라우저를 통해 구현되는 HTTP 프로토콜을 기반으로 서비스를 하고 있으며, 패킷 네트워크 대부분의 트래픽을 차지하고 있다. 이와 연관하여, 접속 연결 과정 및 접속 연결 유지 시간의 변화에 따른 웹 요청 처리 성능에 대해 많은 연구[1, 2, 3]가 이루어져 왔다. 그러나 대부분의 연구가 웹서버 내의 메모리 관리 방식이나 서버 하드웨어 플랫폼 및 네트워크 대역폭, I/O 버퍼 관리, 파일 사이즈, 캐쉬 메모리 관리 같은 HTTP와 TCP/IP 하위레벨의 세세한 과정 및 요소들을 간과한 채 이루어져, 웹 서비스 요청율과 전체 서비스 처리시간, 프로세서 개수 같은 단지 몇 개의 변수만을 가지고 웹서버의 성능을 평가하기에는 큰 무리가 있다. 따라서, 본 논문에서는 Queueing 웹서버 모델로서 Persistent Connection 을 지원하고, 쓰레드 개수나 TCP 요청 대기큐 개수, 메모리 크기나 I/O 버퍼 및 네트워크 대역폭 같은 하드웨어 및 소프트웨어적인 세부요소를 고려하여, 웹 요청이 들어오면서부터 TCP 서브시스템과 HTTP 서브시스템 그리고 Network 서브시스템의 3개 서브시스템을 거치면서 웹 요청을 처리해 나가는 Tandem Queueing 모델을 제안한다.

2. HTTP 트래픽 모델링

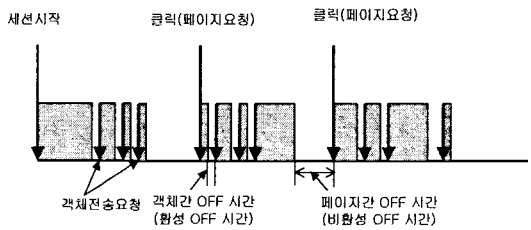

그림 1. 사용자 세션 정의

HTTP 1.1에서 사용자 세션은 그림 1에서 볼 수 있듯이 웹 서버 내의 어떤 페이지에 대한 요청을 함으로써 시작하고, 연결 지속시간이 만료됨으로써 종료한다. 그리고 요청된 페이지에 내재된 객체들에 대한 연속적인 전송 요청 및 그 파일에 대한 전송으로 이루

어진다. 전송사이간 OFF 시간은 사용자가 브라우저를 보면서 생각하는 시간에 해당하는 비활성 OFF 시간과, 단일 웹 객체를 구성하는 파일들의 전송간에 브라우저가 웹 파일들을 파싱(parsing)하고 새로운 TCP 연결을 시작하기 위해 준비하는데 소요하는 처리 시간에 해당하는 활성 OFF 시간으로 이루어진다. HTTP 1.1은 비활성 OFF 시간이 정해진 타임아웃 시간보다 작을 때까지는 연결을 지속하게 하고, 요청한 파일에 대한 전송이 이루어진 후 ACK를 기다리지 않고 연이어서 다음 파일의 전송을 시작하는 파이프라인을 지원한다. 이때, 한 객체를 진송 받고 나서 다음 객체에 대해 전송을 시작할 때까지의 OFF 시간(활성 OFF 시간)과 한 페이지를 모두 전송 받고 나서 클릭한 후 다음 페이지에 대한 요청이 있기까지의 OFF 시간(비활성 OFF 시간), 웹서버가 제공하는 파일 크기, 페이지의 이동을 나타내는 클릭 횟수, 파일 전송 시간등은 모두 랜덤변수로 정의되며 웹서버에서 서비스하는 파일에 대한 요청빈도와 시간적인 지역성과 함께 시스템의 특성을 결정한다.

3. 웹서버의 큐잉모델 분석

3.1 Tandem Queueing 모델

본 논문에서 제시하는 웹 서버 모델은 [3]에서 제시하는 큐잉모델을 기반으로 웹 서비스를 3개의 서브 시스템을 거치는 Tandem Queueing 모델로 나타내고, 각 서브시스템에서 서비스가 이루어질 수 없을 때 재시도하는 상황과, HTTP 1.1에서 제공하는 연결지속 시간을 고려한 세션종료 확률에 따른 시스템 변화를 고려한다. 전체적인 시스템 모델은 아래 그림처럼, TCP 서브시스템과 HTTP 서브시스템, 그리고 Network 서브시스템의 3개의 서브 시스템이 Tandem으로 연결된 잭슨 네트워크(Jackson Network)로 해석될 수 있다.

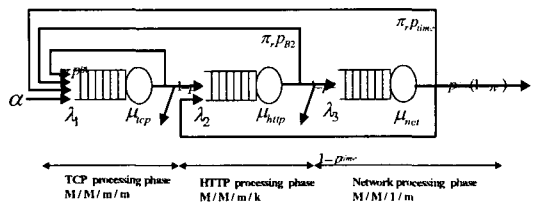

그림 2. 웹서버의 텐덤큐잉 모델

잭슨의 이론에 의하면, 임의의 시간 t에 각 큐들에 있는 트랜잭션 요청의 개수는 독립적인 랜덤 변수로 해석될 수 있고, 정상상태에서 세개의 큐에 각각 N_1, N_2, N_3개의 트랜잭션 요청이 있을 확률은 TCP 큐에 N_1개, HTTP 큐에 N_2개, 네트워크 큐에 N_3개 있을 확률을 곱한 것과 같다.

3.2 TCP 서브시스템

TCP 서브 시스템은 서버 데몬이 서비스하는 TCP 요청 대기큐로 구성된다. TCP 연결 설정 단계에서 서버는 요청 대기큐에 있는 슬롯의 개수 만큼만 들어오는 요청을 서비스 할 수 있고, 슬롯이 꽉 찬 뒤의 요청은 기다리는 공간이 없이 그냥 버려지기 때문에, 서버의 개수가 슬롯 개수 m_{tcp}이고, 서비스 시간은 서버가 SYN-ACK를 보낸 후부터 클라이언트로부터 ACK를 받을 때까지의 시간이므로 결국 RTT(Round Trip Time)값을 가지게 된다. 따라서 RTT 값이 평균이 μ_1인 지수분포를 이루고, 큐의 사이즈가 m_{tcp}이면 이 서브 시스템은 $M/M/m_{tcp}/m_{tcp}$ 큐잉 모델로 해석될 수 있다. 따라서 이 서브 시스템으로 들어오는 서비스 요청율 λ_1 과 서비스 처리율 μ_1, TCP 요청 대기큐가 꽉 차서 추가로 들어오는 요청이 블라킹 될 확률 P_{B1}, 그리고 TCP 서브 시스템에 있는 평균 트랜잭션 요청의 수 $E[N_{tcp}]$는 다음과 같이 구할 수 있다.

$$\lambda_1 = \frac{\alpha}{(1-P_{B1})(1-\pi_r)}, \quad \mu_1 = \frac{1}{RTT} \tag{1}$$

$$P_{B1} = \frac{a^{m_{tcp}}}{m_{tcp}! \sum_{k=0}^{m_{tcp}} \frac{a^k}{k!}}, \quad a = \frac{\lambda_1}{\mu_1} \tag{2}$$

$$E[N_{tcp}] = \frac{\lambda_1}{\mu_1} = \frac{\alpha}{\mu_1(1-\pi_r)(1-P_{B1})} \tag{3}$$

3.3 HTTP 서브시스템

HTTP 서브 시스템은 크게 HTTP 요청 대기큐와, 여러 개의 HTTP 쓰레드를 가진 HTTP 데몬으로 구성된다.

HTTP 서브시스템에서 쓰레드의 개수를 m_{http}라 하고, HTTP 요청 대기큐의 사이즈를 Q 라 하면 이 서브 시스템은 $M/M/m_{http}/(m_{http}+Q)$ 시스템으로 해석

할 수 있다. 그리고 이 서브 시스템에서의 서비스 시간은 요청한 파일을 패치하는 시간과 메모리 버퍼에 복사하는 시간, 또는 만약 메모리 버퍼가 모두 사용중이라면 사용 가능한 버퍼가 생길 때까지 기다리는 시간으로 이루어진다. 따라서 HTTP 서브 시스템에서의 서비스 요청율 λ_2와 서비스 처리율 μ_2, HTTP 요청 대기큐가 꽉 차서 요청이 거부될 확률 P_{B2}는 다음과 같이 구할 수 있다.

$$\lambda_2 = \frac{\alpha}{(1-\pi_r)\{1-(1-P_{time})(1-P_{B2})\}} \quad (4)$$

$$\mu_2 = \frac{1}{T_{fetch} + T_{wait\ I/O} + T_{write\ I/O}} \quad (5)$$

$$P_{B2} = \frac{a^{m_{http}}}{m_{http}!}\left(\sum_{k=0}^{m_{http}-1}\frac{a^k}{k!} + \frac{a^{m_{http}}(1-\rho^{Q+1})}{m_{http}!(1-\rho)}\right)^{-1} \quad (6)$$

여기에서 $a = \frac{\lambda_2}{\mu_2}$, $\rho = \frac{\lambda_2}{m_{http}\mu_2}$

3.4 네트워크 서브 시스템

네트워크 서브 시스템에서, 메모리 버퍼에 있는 파일들은 네트워크를 통해 전송할 수 있는 크기인 1개의 MSS 블록으로 나누어진다. 이 과정은 서버가 메모리 버퍼에 접근하여, 버퍼에 저장되어 있는 파일을 가져와서 MSS 블록사이즈로 나눈 뒤, 해당 블록을 출력버퍼에 놓고 네트워크를 통해 그 블록을 보내면, 클라이언트가 네트워크 카드를 통해 해당 블록을 읽고 ACK를 보내주는 과정으로 구성된다. 따라서, 네트워크 서브시스템에서의 서비스 처리시간은 메모리에 올라와 있는 파일들을 MSS 블록으로 나누는 시간 $T_{partition}$ 과 각각의 블록들을 출력버퍼에 놓는 시간 T_{buf}, 네트워크를 통해 서버 시스템에서 클라이언트 시스템으로 전송하는 시간 $T_{transfer}$, 그리고 클라이언트가 네트워크 카드를 통해 해당 블록을 읽는 시간 T_{client} , 클라이언트로부터 ACK가 오는 시간 T_{ACK} 값을 더한 값이 된다. 따라서 네트워크 서브시스템에서의 서비스 요청율 λ_3와 서비스 처리율 μ_3는 다음과 같이 나타낼 수 있다.

$$\lambda_3 = \frac{(1-P_{B2})\alpha}{(1-\pi_r)(1-(1-P_{time})(1-P_{B2}))} \quad (7)$$

$$\mu_3 = \frac{1}{T_{partition} + k(T_{buf} + T_{transfer} + T_{client} + T_{Ack})} \quad (8)$$

네트워크 컨트롤러는 메모리 버퍼에 있는 정보를 출력 버퍼로 보낼 때, 사용 가능한 출력버퍼가 없을 때 해당하는 요청을 출력 버퍼가 생길 때까지 무한대로 기다리게 할 수 있으며, 네트워크 컨트롤러에 의해 어느 한 순간에 출력 버퍼 m_{net}개 중 단 한 개만을 네트워크에 내보낼 수 있고, 출력 버퍼의 개수만큼 유지하고 있을 수 있으므로, 결국 1개의 서버를 가지고 m_{net}의 버퍼를 가진 시스템으로 해석이 가능하다. 따라서 네트워크 서브 시스템은 $M/M/1/m_{net}$ 으로 해석할 수 있다. 따라서 정상 상태에서 네트워크 서브 시스템에 있는 트랜잭션의 수가 k개일 확률은 식(9)와 같이 나타낼 수 있다.

$$P[N=k] = \frac{(1-\rho)\rho^k}{1-\rho^{m_{net}+1}} \quad (9)$$

4. 검증 및 시뮬레이션

본문에서 제시한 웹서버 모델을 검증하기 위하여, 이산 사건 시뮬레이션을 통한 성능과, 실험실 내의 LAN 환경에서 벤치마킹을 통해 웹서버의 성능을 측정한 값을 비교하였다. 먼저, 실험실 내의 서버 구성은 그림 3과 같이, 부하를 발생시키고 각 클라이언트 컴퓨터들의 요청들에 대한 웹서버의 상태를 기록하는 1대의 컨트롤러 컴퓨터와 1대의 웹서버, 그리고 8대의 컴퓨터를 고속 이더넷(100Mbps)을 통해 연결하여, 한 대의 컨트롤러 컴퓨터가 여러 대의 클라이언트 컴퓨터들이 웹서버에 서비스 요청을 하게 함으로써 그에 따른 서버의 대역폭을 측정하는 WEB BENCH 4.1[4]을 통하여 수행하였다.

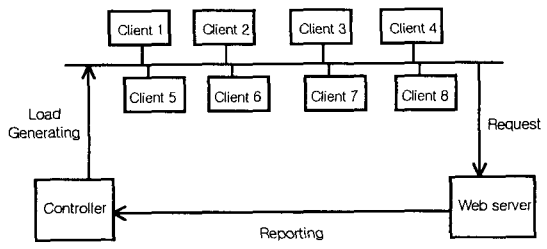

그림 3. 테스트 환경

컨트롤러 컴퓨터는 클라이언트 서버로 하여금 웹서버에 저장되어 있는 파일들을 요청하게 함으로써 부하를 발생시키고, 그 요청하는 클라이언트 컴퓨터의 수를 조정함으로써 요청수를 변화시킬 때마다 웹서버에서 내보내는 대역폭을 측정한다. 그리고 웹서버는 펜티엄-II 300MHz CPU 와 128M RAM, 10Mb/s 랜카드로 구성되고, OS는 Linux 커널 2.4.2이며 아파치 버전 1.3.19 웹서버 프로그램을 구동하였다. 웹 벤치 4.1을 통해 얻은 웹서버의 성능과 시뮬레이션에 의한 웹서버의 성능 비교는 아래 그림과 같다.

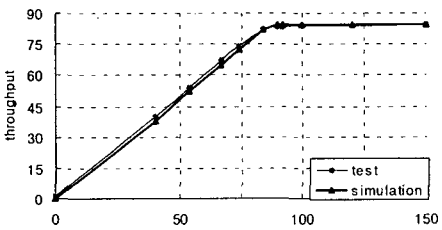

그림 4. 실험값과 시뮬레이션 결과의 비교

시스템에서의 여러 변수들 중에서도 HTTP 1.1은 세션이 종료되기 전에 비활성 OFF 시간이 이미 설정된 타임 아웃시간을 초과하여 세션이 종료되는 확률과, 세션이 종료되었을 때 그 페이지를 다시 서비스 받기 위해 재연결을 시도하는 확률에 강력하게 의존하고 있기 때문에 앞에서 제시한 웹서버 모델이 검증되기 위해서는 시스템 처리 상황에 따라 다양한 재시도 확률 π_r 값과 세션이 타임 아웃되는 확률 $P_{timeout}$ 값이 적용되어야만 했고 그 값은 웹 서버가 어떤 종류의 서비스를 하는가와 그에 따른 웹서버의 Persistent Connection 지속시간에 강력히 의존한다. 테스트 결과 우리는 $\pi_r = 0.15$과 $P_{timeout} = 0.3$ 인 값에서 WEB BENCH의 결과 값과 유사한 시뮬레이션 결과를 얻을 수 있었다. 앞에서 제시하고 있는 웹서버 모델은 아래 그림에서 볼 수 있는 것처럼 실험실에서 웹벤치를 통해 얻은 결과와 거의 유사함을 볼 수 있다. 지속연결 종료확률은 타임 아웃값을 어떻게 설정하느냐에 의존하고, 연결이 종료되었을 때 재 시도할 확률은 서버가 어떤 종류의 서비스를 운영하고 있는 지에 의존하므로, 이 두 값을 찾은 뒤, 시뮬레이션의 결과를 이용하면 더 효율적인 웹서버의 운영이 가능하다.

5. 결 론

본 논문에서는 웹서비스를 세개의 텐템 서브시스템으로 연결된 큐잉 모델을 제시하고, 각각의 서브 시스템 내에서 일어나는 과정을 해석적으로 나타내었다. 제시된 모델은 실험실 랜 환경에서 웹벤치 4.1을 통한 벤치마킹과 실험실 환경과 유사한 값에 기반하여 해석적으로 이산사건 시뮬레이션을 통하여 검증하였다. 이 모델이 HTTP 1.1을 기반으로 모델링 되었기 때문에, 세션종료 전에 타임아웃 되어 연결이 종료되는 확률과, 타임아웃 되었을 때 재시도하는 확률은 아파치와 같은 웹서버 프로그램내의 HTTP 1.1 Persistent Connection 타임아웃 값 설정에 따라 큰 영향을 받기 때문에, 시스템 성능 평가에 앞서 그 웹서버의 설정에 따른 성능 분석이 요구된다.

일반적인 웹서버에 대해 제안한 웹서버의 모델을 보완하고 일반화하기 위하여 고려되어야 할 사항은 다음과 같다. 먼저 웹 트래픽이 Poisson 분포를 따르지 않으며, 웹서버에 요청하는 파일의 크기가 지수 분포가 아닌 Heavy-tailed 분포를 따른다는 점, 또한 각 서브시스템 내에서의 서비스 시간이 지수분포가 아닌 일반분포를 가진다는 점과 웹서버 내의 캐시에 대한 영향등이 더 고려되어야 할 것이다.

참고문헌

[1] Zhen Liu, N. Niclause, C. Jalpa-Villaneva, "Traffic model and Performance Evaluation of web servers," Performance Evaluation 46, pp 77-100, 2000

[2] P. Barford, M.E. Crovella, "Generating representative web workloads for network and server performance evaluation," Proceedings of the ACM Sigmetrics '98, 1998.

[3] R.D. Van der Mei, R. Hariharan, P.K. Reeser, "Web Server Performance Modeling," Proceedings of 4th Informs Telecom Conference, Special Issue of Telecommunication.

[4] http://www.etestinglabs.com.