2001년 한국정보처리학회 춘계 학술발표논문집 제8권 제1호

QoS 제공을 위한 Diff-HTTP

현은실, 이윤정, 김태윤
고려대학교 컴퓨터학과
{eunsil, genuine, tykim}@netlab.korea.ac.kr

Diff-HTTP for QoS

Eun-sil Hyun, Yoon Jung Rhee, Tai-Yun Kim
Dept. of Computer Science & Engineering, Korea University

요 약

HTTP/1.0 은 동일한 서버로부터 각각의 개체에 대하여 개별적인 TCP 연결을 생성하기 때문에 다중의 요구를 비효율적으로 처리한다. 이러한 문제를 해결하기 위한 방안으로 제안된 HTTP/1.1 은 TCP 연결을 지속적인 연결(Persistent connection)이라는 개념을 도입하여 하나의 TCP 연결 상에서 다중의 요구(Request)를 처리하도록 하고 있다[9]. 네트워크가 발전됨에 따라 사용자가 늘어나고 다양해지면서 서비스의 차별화 문제가 중요한 문제로 대두되었다[3,5]. 본 논문에서 제시하는 Diff(Differentiated)-HTTP 은 웹 서버에 서비스를 요청한 클라이언트들에게 차별화 된 서비스를 제공하기 위해서 사용자를 두 등급, 기본 등급과 우선 순위를 고려한 상위 등급으로 구분한다. 각 등급은 제한 시간(Holding Time)으로 차별화 되고 상위 등급에 속한 클라이언트에게 제한된 시간을 증가시켜 지연을 최소로 함으로써 고품질의 서비스를 제공하는 방안을 제안한다.

1. 서론

네트워크가 발전됨에 따라 사용자가 늘어나고 다양해지면서 차별화 된 서비스의 제공이 중요한 문제로 대두되었다. 현재의 WWW 서비스는 인터넷의 단일 서비스에 의해 동일한 서비스의 품질을 제공한다. 모든 사용자들에게 동등한 서비스 제공하는 것도 중요하지만 서비스의 품질을 보장하는 방법으로 고급 사용자들에게 고품질의 서비스 제공이 이슈로 대두되고 있다. 이러한 문제점을 고려할 때 Web 서버에서의 차별화 된 서비스 제공 메커니즘이 요구된다[3,5].

HTTP/1.0 에서는 각각의 문서마다 별도의 연결을 만들어야 하는 추가적인 부담이 발생한다. 따라서 많은 문서를 갖고 오려고 할 때 HTTP 프로토콜은 성능상의 저하를 발생시키게 된다. 최근에 구현된 HTTP 서버의 연결 해제 정책은 제한 시간(Holding Time 혹은 Time Out)방식으로 일정한 시간을 주어 그 시간 안에 다른 요청이 없을 시 연결을 해제 시키고 시간이 지나기 전에 또 다른 요청이 있을 경우 그 시점에서 새로운 시간을 할당해주는 방식이다[1].

본 논문에서는 Web 서버의 고품질의 차별화 된 서비스 제공을 위하여 효율적인 연결 관리 기법을 제시한다. 제안된 메커니즘은 제한 시간 정책을 이용하며 서버가 각기 다른 사용자에게 차별화 된 서비스를 제공하기 위해 사용자를 두 등급 즉, 우선 순위를 고려한 상위 등급과 기본 등급으로 나눈 후 제한 시간을 차별화 하여 상위 등급에 포함된 사용자에게 제한 시간을 증가 시켜 지연을 최소로 함으로써 고품질의 서비스를 제공한다.

2. HTTP 관련연구

본 장에서는 현재까지 진행되어 온 HTTP 프로토콜의 문제점을 분석하고 관련 연구를 살펴본다.

2.1 HTTP/1.0 연결관리

HTTP/1.0 상에서 클라이언트는 같은 서버에게 연속적인 문서 요청을 함에도 불구하고 서버는 클라이언트가 요청하는 문서 파일을 전송하고 나면 자동적으로 연결을 해제하는 구조로 되어있다. 각 요청마다 새로운 TCP 연결이 설정/해제되기 때문에 다음과 같은 많은 문제점이 야기된다[9].

- TCP 연결 설정 시간 지연
- 슬로우 스타트(Slow Start)로 인한 지연
- TIME_WAIT 상태 문제

2.2 HTTP/1.1 지속적인 연결

HTTP/1.1 은 클라이언트의 요청에 대한 응답을 보낸 후 서버는 즉시 연결을 해제하지 않고 TCP 연결을 계속 유지한다. 동일한 클라이언트와 서버 사이에서 연속적인 요청/응답을 단일 연결을 통해서 보낼 수 있다[7,9].

2.2.1 지속적인 연결에서의 연구와 문제점

HTTP/1.1 은 TCP 연결을 명시적인 'CLOSE' 연산을 실행하기 전까지 계속 연결 상태를 유지해야 하기 때문에 휴지 상태의 연결에 대한 해제 시점을 정하는 원칙이 매우 중요하다. 그러나 무조건 연결을 계속 유지시키는 것은 서버 자원의 낭비뿐만 아니라 서버의 오버헤드와 자원 부족으로 인하여 새로운 클라이언트와의 연결을 지연시키는 결과를 야기한다. 결국 휴지상태의 연결이 많을 시에는 서버의 처리율을 저하시키는 요인으로 작용한다[2].

2.2.2 차별적인 서비스 제공을 위한 연구

최근의 연구 동향에서 보여지는 것은 클라이언트가 요청한 서비스에 대하여 서버가 클라이언트를 차별화 하여 서비스를 제공하기 위한 여러 방안이다.
동시에 많은 클라이언트가 서버에 작업 요청을 했을 경우 우선 순위 기반의 스케줄링 기법을 제공한다. 서버의 운영체제에서 FCFS 알고리즘이 아닌 우선 순위가 높은 요청들을 낮은 요청들에 비해 먼저 처리해 주는 알고리즘이다[3].

각 서비스를 상위 그룹(foreground)와 하위 그룹(background)으로 나누어 두 등급의 서비스를 실시한다. 상위 그룹 요청에 대한 응답을 위해서는 하위 그룹 요청에 비해 서버의 자원을 더 할당한다. 우선 서버는 상위 그룹과 하위 그룹의 두 개의 큐를 가지고 하위 그룹의 큐가 사용하는 서버의 자원을 제한한다. 따라서 이것은 상위 그룹의 요청은 하위 그룹의 요청들의 로드가 증가해도 크게 영향을 받지 않는다[5].

3. 차별화된 서비스 제공을 위한 Diff-HTTP

3.1 Diff-HTTP 모델

본 논문에서 제안한 방법은 이전의 HTTP/1.1 의 제한 시간 방식을 기본으로 하되, 우선 순위를 부여한 상위 등급을 만든 후 제한 시간의 조절을 통하여 상위 등급 사용자가 서비스를 받을 경우 연결의 설정/해제 과정의 비율을 상대적으로 낮추어 상위 등급 사용자로 하여금 지연을 덜 느끼도록 하는 것이다.

작동원리는 다음과 같다.

(1) 클라이언트는 TCP 연결을 설정한다.

(2) 클라이언트는 HTTP 요청을 서버에게 보낸다. 서버는 먼저 이 사용자가 등급 중 어느 등급에 속하는지 분석 한 후 그 등급에 부합된 제한 시간을 부여한다. 디스크로부터 클라이언트가 요구한 것을 조회해서 서비스를 완수한다. 그리고 클라이언트에게 응답을 보낸다.

(3) 서버는 클라이언트가 연결을 해제하지 않고 다른 요청이 들어올 때까지 TCP 연결을 유지한다. 각 등급에 따라 클라이언트에 대한 제한 시간이 지나기 전에 클라이언트의 요청이 있다면 다시 해당 클라이언트에 부합된 제한 시간이 설정되고 해당 클라이언트의 요청이 더 이상 없을 경우 연결을 해제한다.

상위 등급과 기본 등급에 따른 차별화 된 서비스는 제한 시간을 등급에 따라 부여함으로써 정의 할 수 있다. 그림 5 에서 볼 수 있듯이 우선 순위를 부여 받은 상위 등급의 클라이언트는 기본 등급의 클라이언트에 비해 제한 시간에 추가 시간 *'a'* 가 부여된다.

3.2 Diff-HTTP 시나리오

3.2.1 Diff-HTTP 환경 설계

서버와 클라이언트의 연결이 설정되는 과정에서 클라이언트들의 등급 결정 과정을 담당하는 것을 Broker 라 정의한다. Broker 를 통해 서비스를 요청한 클라이언트가 어느 등급에 포함되는 지의 여부를 결정한 후 서버는 해당 클라이언트의 등급에 합당한 제한 시간을 부여하여 클라이언트의 요청을 처리한다.

(1) Broker

Broker 의 역할은 크게 두 가지로 구분된다.

1) 등급테이블의 관리

서버에 접근하는 클라이언트의 접속 정보 히스토리를 기반으로 접속 횟수와 시간 정보에 따라 클라이언트들을 상위 등급과 기본 등급으로 구분한다. 전체의 클라이언트 중에 일정한 비율만큼을 상위 등급으로 규정한다. 최신의 접속 정보의 히스토리를 바탕으로 상위 등급을 관리한다.

따라서 본 논문에서는 세 가지 알고리즘을 제안하고 이중에서 LRFR 방식으로 설계한다.

① LRR(Latest Recently Request) 알고리즘

가장 최근에 서비스를 요청한 사용자를 상위 등급 안에 포함시키는 방안으로 최적 알고리즘의 근사치이다. 즉 오랜 기간 동안 해당 서버에게 서비스를 요청하지 않은 사용자를 상위 등급에서 제외시키기 위한 것으로 상위 등급 안에 포함된 사용자는 가장 최근에 서버에게 서비스를 받은 사용자들이 될 것이다. 그러나 최근에 접속한 사용자들이 곧 앞으로도 많은 서비스를 해당 서버에게 요청한다는 보장이 없기 때문에 이 알고리즘에 의해 만들어진 상위 등급은 서버에 간헐적으로 접근하는 사용자를 포함한다.

② MFR(Most Frequently Request)알고리즘

해당 서버에게 서비스를 많이 요청한 사용자들을 요청 횟수 순으로 상위 등급 안에 포함시키는 방안이다. 이것은 활발하게 서버에게 서비스를 요청하는 사

용자는 앞으로도 많은 서비스를 요청한다는 가정 하에서 만들어진 것이다. 그러나 이 알고리즘은 어떤 사용자가 전 단계에서 해당 서버에게 많은 서비스를 요청했지만 그 후로 다시 서비스를 요청하지 않을 경우에 문제점을 야기한다. 접속 횟수로는 많은 횟수가 되어 상위 등급 안에 포함되지만 더 이상의 서비스를 요청하지 않는다면 상위 등급의 한정 수를 하나 차지하기 때문에 다른 사용자 즉 최근에 많은 서비스를 요청하는 사용자가 상위 등급 안에 포함되지 않을 수 있기 때문이다. 따라서 이 알고리즘에서 가장 중요한 것은 어떤 일정한 시간 간격으로 정보를 수정해야 한다는 것이다.

③ LRFR(Latest Recently/Frequently Request)알고리즘

앞에서 정의한 두 알고리즘의 문제점을 해결하기 위한 방안으로 만들어진 것이다. 이 알고리즘은 일정한 시간 간격 내에 가장 많은 서비스를 요청했던 사용자들을 상위 등급 안에 포함시키는 방안이다. 즉 사용자 접속 횟수와 최후 접속 시간 정보를 포함하고 있어 오랜 시간 동안 서버에게 요청을 하지 않는 사용자를 상위 등급 안에서 제외시키는 것이다. 따라서 시간 정보를 저장하고 있기 때문에 오랜 시간 동안 서버에게 요청을 하지 않는 간헐적인 사용자를 상위 등급에서 제외시킬 수 있다.

2) 클라이언트의 제한 시간 결정

클라이언트가 서버에게 HTML 문서를 받기 위한 연결을 요청하는 경우 그 요청을 처리 하기 앞서 해당 클라이언트가 어느 등급에 해당하는 지를 검사하여 서버로 하여금 그 클라이언트에게 해당하는 제한 시간을 부여하도록 한다.

해당 클라이언트의 히스토리를 바탕으로 접근한 사용자가 일정한 상위 그룹 내에 포함되어 있다면 기본 제한 시간 + 'α ' 를 부여하고, 그렇지 않은 경우는 일반 제한 시간을 부여하여 제한 시간을 할당한다.

(2) 요청/응답 프로세싱

서버는 Broker 에 의해 주어진 제한 시간을 기반으로 하여 해당 클라이언트와의 연결을 유지한다. 클라이언트가 요청한 서비스를 제공해 주고 만약 휴지 상태일 경우 해당 클라이언트의 제한 시간이 0 으로 만기 될 때까지 더 이상의 요청이 없을 경우 연결을 해제한다. 제한 시간이 만기 되기 전에 해당 클라이언트가 다른 서비스를 다시 요청한다면 그 요청을 처리해 주고 새로이 클라이언트에 해당하는 제한 시간을 적용한다.

3.2.2 서버와 클라이언트 알고리즘

차별화 된 서비스를 제공하기 위해서 서버 관점에서 알고리즘을 설명하면 다음과 같다.

(1) 서버

서버는 새로운 클라이언트의 요청이 들어오면 broker()함수를 호출한다. broker()를 사용하여 서버는 해당 클라이언트의 히스토리를 분석한 후 그 히스토리에 따라 등급을 결정한다. broker 에 의해 결정되어진 등급에 따라 해당 클라이언트에게 제한 시간을 부여한다. 만약 클라이언트가 상위 등급에 속한 경우 그 클라이언트에게 기본적인 제한 시간보다 더 많은 제한 시간을 부여한다. 그렇지 않은 경우는 기본적인 제한 시간을 제공한다. 클라이언트가 휴지 상태에 있을 때 제한 시간이 감소하기 시작하여 시간이 만기 되면 자동적으로 해제된다. 또는 클라이언트로부터 'CLOSE' 메시지를 받을 경우에도 연결을 해제할 수 있다.

(2) 클라이언트

클라이언트가 더 이상 서비스 요청이 없을 경우 서버에게 'CLOSE' 메시지를 보낼 수 있다. 또는 휴지 상태로 있다면 서버의 broker 에 의해 설정된 제한 시간이 만기 되어 서버에 의해 연결이 자동적으로 해제된다.

4. 결론 및 향후 연구 방향

본 논문에서는 HTTP/1.1 에서 하나의 TCP 연결 위에 다수의 요청들을 실행하기 위한 지속적인 연결에서의 적절한 TCP 연결 해제 시점을 서버 측에서 차별화 된 정책을 사용하여 지원할 수 있는 알고리즘을 제안하였다. 사용자가 다양해지고 서비스를 요구하는 클라이언트가 증가함에 따라 서비스를 차별화 하여 제공하는 것이 중요한 문제로 대두되었다. 본 논문에서는 서버에게 서비스를 요청하는 클라이언트를 상위 등급과 기본 등급의 두 개의 등급으로 나누고 우선순위를 부여하여 상위 등급의 클라이언트에게 서비스 제한 시간의 가중치를 주어 서비스를 제공하는 Diff-HTTP 를 제안하였다. 먼저 클라이언트를 차별화 하여 다양한 서비스를 제공하였다. 차별적인 서비스를 제공하는 데 있어 지연을 최대한 줄여 높은 서비스의 품질을 제공하였다. 이는 사용자에게 가중치의 제한 시간을 부여하여 상위 등급의 사용자는 기본 등급의 사용자에 비해 차별성과 신뢰성을 제공하기 위해서다. 향후 연구 과제는 이것을 구현하고, 효율적인 차별화 서비스를 위한 Broker 의 효율적인 알고리즘의 연구가 더욱더 필요하다. 또한 프로토콜 차원에서의 서비스의 차별화로부터 더 나아가 OS 차원에서 사용자의 요청에 따른 서비스를 등급 별로 나누어 서버의 리소스를 차별화 하여 제공하는 방법을 연구하고자 한다.

참고 문헌

[1] Edith Cohen, Haim Kaplan and Jeffrey Oldham, "*Managing TCP connections under persistent HTTP*," Elsevier Science B.V, 1999.

[2] M. Elaoud, C. J Screenan, P. Ramanathan and P. Agrawal, "Use of server load to dynamically select connection closing time for HTTP/1.1 servers," http://www.cae.wisc.edu/~elaoud/research.html

[3] Jussara Almeida, Mihaela Dabu, Anand Manikutty and Pei Cao, "Providing Differentiated Levels of Service in Web Content Hosting, In Proceedings of the 1998 SIGMETRICS Worshop on Internet Server Performance, Madison, WI, USA, PP. 91-102, June 1998.

[4] H.Saran and S.Keshav, "An Empirical Evaluation of Virtual Circuit Holding Times in IP-over-ATM Networks," Journal on Selected Areas Communication 13(1995).

[5] Lars Eggert and John Heidemann, "Application-Level Differentiated Services for Web Servers," In World Wide Web Journal, Volume 3, Issue 2, PP.133-142, 1999.

[6] A. Fedlman, R. Caceres, F. Douglis, G.Glass and M.Rabinovich, "Performance of Web proxy caching in heterogeneous bandwidth environments," In Proceedings of the IEEE INFOCOM' 99 Conference, 1999.

[7] H. Frystrk Nielsen, J. Gettys, A. Baird-Smith, E.Prud' hommeaux, H.W. Lie and C.Lilley, "Network Performance effects of HTTP/1.1, CSS1 and PNG," In Proceeding of the ACM SIGCOMM' 97 Conference, Cannes, France, August 1997.

[8] H.Kaplan and E.Cohen, "Reducing user-perceived latency by perfeching connection and per-warming servers," http://www.research.att.com/~edith/publications.html, 1999.

[9] Z.Wang and P.Cao, "Persistent connection behavior of popular browsers," http://www.cs.wisc.edu/cao/paper/persistent-connection.html.

[10] Edith Cohen, Balachander Krishnamurthy, and Jennifer Rexford, "Evaluating Server-Assisted Cache Replacement in the Web," Revision of an ESA' 98.