이질 웹 서비스 환경에서 차별화 서비스를 위한 부하 분산 기법

황미선[0*] 박기진[**] 남궁미정[*]

안양대학교 문리과학대학 컴퓨터학과[*]
아주대학교 공과대학 산업정보시스템공학부[**]
snow0123@anyang.ac.kr[0*] kiejin@ajou.ac.kr[**] mjmj201@anyang.ac.kr[*]

A Load Balancing Mechanism for Differentiated Service in Heterogeneous Web Service

Misun Hwang[0*] Kiejin Park[**] Mijung Namgung[*]

Department of Computer Engineering, Anyang University[*]
Division of Industrial & Information Systems Engineering, Ajou University[**]

요 약

인터넷 사용자들의 증가로 인해 신뢰성 있는 차별화된 고품질의 서비스를 제공하기 위해서는, 사용자와 서비스 제공자 간의 SLA(Service Level Agreement)를 고려한 웹 서버 클러스터의 부하 분산 기능은 필수적이다. 본 논문에서는 이질 웹 서비스 환경에서 SLA 를 만족시켜주는 동적 부하분산 기법을 연구하였으며, 시뮬레이션 결과를 통해 기존의 정적 기법 보다 웹 서버의 응답시간(Response Time)성능이 개선되는 것을 확인하였다.

1. 서 론

인터넷 사용자의 증가와 멀티미디어 데이터를 포함한 웹 서비스의 팽창으로 인해, 웹 서버의 성능과 가용성(Availability)을 개선하려는 시도가 활발하다. 웹 서버의 성능을 향상시키는 방법으로 웹 서비스 시스템을 구성하는 웹 서버를 클러스터링 하는 기법이 주로 채택되고 있으며, 사용자 요청에 대한 응답시간을 최소화하기 위해서는, 클러스터링된 웹 서버 노드간의 부하 분산이 필수적이다.

한편, 웹 서버는 인터넷 사용자에게 기본적인 서비스(Best Effort) 제공은 물론, 고품질의 차별화된 서비스를 제공해야 할 필요성이 증가하고 있으며, 현재까지는 주로 차별화된 서비스를 제공하기 위한 네트워크 계층을 대상으로 한 QoS(Quality of Services) 기법이 활발히 연구되었으나, 최근에는 사용자와 서비스 제공자간의 서비스 품질을 보장해 주는 정책(SLA: Service Level Agreement)을 고려한 보다 상위계층의 QoS에 관한 연구가 활발하다.

기존의 웹 서버는 사용자의 요청을 선입선출(FCFS) 방식으로 처리하기 때문에, 과부하 상황에서 우선 순위가 높은 사용자의 요청이 거절되는 경우를 발생시킬 수 있으며, 이를 해결하기 위해 사용자의 요청을 우선 순위로 구별한 후, 차별화된 서비스를 제공하는 방식에 대한 연구가 있었다[1]. 서비스 차별화(Service Classification) 개념은, 웹 서버에 들어오는 사용자 요청을 분류하여 계층별 큐(Queue)로 보내고, 웹 서버의 과부하 발생시 상위 계층의 사용자 요청을 받아들이고 하위 계층의 사용자 요청을 거절(Admission Control)하여 차별화된 서비스를 제공하는 기본적 방식을 의미한다. 서버의 부하 상태와 사용자 요청의 도착률의 변동을 고려하지 않은 클러스터 부하 분배는 자원 낭비가 발생하기 때문에, 사용자 계층별 요청 도착률에 따라 계층별 요청을 처리하는 서버 노드를 할당하는 성능 분리(Performance Isolation)개념을 도입할 경우, 효율적인 서버 노드 자원 활용이 가능하다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구를 언급하였고, 3장에서는 차별화된 서비스를 위한 알고리즘을 제시하였으며, 4장에서는 제시한 알고리즘의 성능 평가를 수행하고, 5장에서는 결론을 내렸다.

2. 관련 연구

웹 서버에서 SLA를 보장하기 위해, 시스템 구조를 통한 방법과 차별화된 서비스를 통한 방법이 연구되고 있으며, 전자는 시스템 관리자에 의한 서버 노드의 추가 또는 제거 작업이 이루어짐으로 인해, 구조의 변화가 쉽지 않아서 효율적이지 못하다. 후자의 방법들 중 DDSD(Demand-driven Service Differentiation) 기법[2]은 사용자의 요청에 따라 CPU와 디스크 I/O의 용량을 할당하며, Dynamic Part 기법[3]은 서버 부하 량에 따라 서버 노드를 동적으로 분할하여 차별화된 서비스를 제공하지만, 이러한 기법들은 일정한 시간 간격으로 서버 노드를 분할하기 때문에 갑작스런 과부하 상태를 처리하기에는 부적절하며, 특히 Dynamic Part 기법은 정적 요청을 전혀 고려하지 않고 동적 요청만을 고려하였다. 반면, LARD(Locality-Aware Request Distribution)기법[4]은 내용 기반 요청 분배 방식으로 사용자의 요청 내용에 따라 분배하며 서비스 노드의 메인 메모리 캐쉬 적중률을 향상시켰으나 정적

본 연구는 한국과학재단 목적기초연구(R05-2003-000-10345-0) 지원으로 수행되었음.

요청만을 고려한 점과 작업 부하에 가중치를 설정해야 하는 문제점이 있다.

일반적으로 클러스터링된 웹 서버 노드들을 기능적 도메인(Function Domains)에 따라 구분할 수 있다[2]. 예를 들어 e-비즈니스를 처리하는 사이트는 메일, 문서, VOD, 사용자 정보관리 및 결제 등의 기능적 도메인으로 나뉘며, 각각의 도메인은 사이트 관리기에 의해 실질적인 특정 서버로 맵핑된다. 기존의 부하분산 기법들은 클러스터 웹 서버의 모든 노드가 단일 도메인으로 이루어진 동질(Homogeneous) 웹 서비스 환경을 가정한 방식이며, 서로 다른 기능적 도메인을 갖는 이질(Heterogeneous) 웹 서비스 환경에 대한 고려는 없었다. 본 논문에서는 사용자의 정적·동적 요청을 고려한 이질 웹 서비스 환경에서 SLA를 보장하는 동적 서버분할 알고리즘을 다루었다.

3. 차별화된 서비스를 위한 웹 서버 구조

본 논문에서는 내용 기반 분배가 가능한 Layer-7 스위치를 이용하여, 사용자의 요청을 계층별로 분류하여 처리하며, 처리된 결과는 서버에서 사용자에게 직접 전달되는 One-way 구조를 채택하였다[5](그림 1 참조). One-way 구조에서는 들어오는(Inbound) 패킷은 웹 스위치를 거쳐 웹 서버로 전달되는 반면에 나가는(Outbound) 패킷은 웹 스위치를 거치지 않고 직접적으로 사용자에게 전달된다. 반면에 Two-way 구조는 클러스터 안의 각각의 서버는 유일한 하나의 IP 주소로 설정되어 들어오는 패킷과 나가는 패킷은 웹 스위치에 의해 재작성되기 때문에 One-way 구조에 비해 스위치의 오버헤드가 크다.

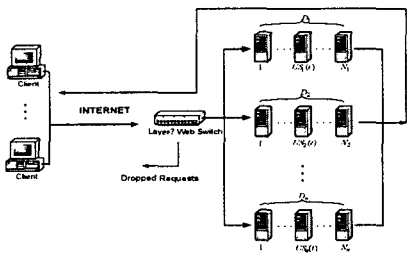

그림 1 Layer-7 스위치 기반 One-way 웹 서버 구조

웹 서버 구조에서 차별화된 서비스를 제공하는 기법 중 클러스터링된 서버 노드의 정적·동적 분할을 통해 효율적인 서버 노드의 사용 별 차별화된 계층별 요청을 처리할 수 있다. 정적 분할은 계층별 사용자 요청 수준에 따라 고정된 서버 노드를 할당하는 방법으로 사용자 요청의 도착률과 서버 노드의 적재 상태에 따라 서버 노드를 변동할 수 없으며, 한 계층에 할당된 서버 노드가 과부하 상태이더라도 다른 계층의 서버는 휴지 상태로 서버 노드의 자원 낭비가 일어나는 문제점이 있다. 반면에 동적 분할 방식은 사용자 요청의 수나 필요에 따라 동적으로 서버 노드를 할당하여 휴지 서버 노드를 줄여줌으로써 높은 자원 활용을 제공한다.

3.1 동적 서버분할

차별화된 서비스를 제공하기 위해서 사용자 계층을 서비스의 종류와 SLA를 고려한 우선 순위를 적용하여, 동적 서비스나 유료 사용자를 High Class로 정적 서비스 또는 무료 사용자를 Low Class로 구분하였다. 사용자 계층별 부하 량에 따라 동적으로 서버 노드를 할당하여 특정 계층의 노드가 과부하 상태일 때, 더 낮은 계층의 서버를 추가적으로 이용하는 방식으로 서버 노드의 자원을 효율적으로 사용할 수 있는 이점이 있다. i번째 기능적 도메인 웹 서버($D_i = \{1,2,...,N_i\}$)에 사용자의 요청이 들어오면, 그 요청의 우선 순위를 확인한 후에 그에 적합하게 서버를 분할하여 처리한다. 이질 웹 서비스 환경에서의 서버 할당 방식을 수식으로 나타내면 식 (1)과 같다.

여기서, $HS_i(t)$ 와 $LS_i(t)$ 는 임의의 시간 t 에서 각 도메인 i 에서 우선 순위에 따라 할당되는 서버의 수이며, $US_i(t)$ 는 임의의 시간 t 에서 도메인 i 에 속하는 사용자 요청을 처리하는 서버를 구분하는 경계 값이다.

$$
\begin{aligned}
&D_1 : HS_1(t) = \{1,...,US_1(t)\}, LS_1(t) = \{US_1(t)+1,...,N_1\} \\
&D_2 : HS_2(t) = \{1,...,US_2(t)\}, LS_2(t) = \{US_2(t)+1,...,N_2\} \\
&\quad \Lambda \\
&D_n : HS_n(t) = \{1,...,US_n(t)\}, LS_n(t) = \{US_n(t)+1,...,N_n\} \\
&\text{전체 서버 수} = \sum_{i=1}^{n} N_i
\end{aligned} \tag{1}
$$

식 (2)는 i번째 도메인을 서비스하는 서버 경계 값을 나타내며 여기서 ρ_i 는 전체 요청 수와 High Class에 속하는 사용자의 요청 수의 비율이고, T_{N_i} 은 i번째 도메인의 서버 N_i 대에서 허용 가능한 최대 지연 시간이다. 또한, $MaxConn(T_{N_i})$ 은 지연 시간 T_{N_i} 을 만족시키는 최대 접속 수이며, T_H 는 High Class의 SLA를 고려한 임계 시간(Threshold Time: $T_H < T_{N_i}$)이다.

각 도메인의 High Class의 서버수를 구하기 위해서 ρ_i 와 도메인 i 의 전체 서버수를 곱하며, T_{N_i} 에서의 이상적인 최대 접속수를 T_H 에서의 실제 최대 접속수로 나누어 구한, 최대 접속 비율을 곱하여 정확한 값을 이끌어낸다.

$$
US_i(t) = \lceil \rho_i N_i \rceil \cdot \frac{MaxConn(T_{N_i})}{MaxConn(T_H)} \tag{2}
$$

식 (3)은 동적 서버분할 방식을 나타내며, $SumLoad_{HS_i}(t)$ 는 $HS_i(t-1)$ 에서의 서버 부하(Load)의 합이며, 사용자의 최대 접속수($MaxConn$)를 의미한다. 특정 계층의 부하합이 작업 능력을 넘어서면, 과부하 상태로 낮은 계층의 서버를 가져오고, 부하량이 적어질 경우 가져온 서버를 반납하는 방식을 따른다. 임의의 시간 t 에서 $US_i(t)$ 가 갖을 수 있는 최소의 서버 수는 $t=0$ 일 때 서버 수와 같아야 하며, 최대의 서버 수는 $N_i - 1$ 이다.

$$
\begin{aligned}
&if\left(SumLoad_{HS_i}(t) > US_i(t-1) \cdot MaxConn(T_H)\right) \\
&\quad US_i(t) = US_i(t-1) + 1; \\
&else\ if\left(SumLoad_{HS_i}(t) < (US_i(t-1)-1) \cdot MaxConn(T_H)\right) \\
&\quad US_i(t) = US_i(t-1) - 1;
\end{aligned} \tag{3}
$$

4. 성능 평가

이질 웹 서비스 환경에서 사용자 계층에 따른 동적 서버분할 기법의 성능을 분석하기 위해, 웹 서버 클러스터 시스템을

구성하는 서버 대수, 사용자 계층, 작업 수행시간 등 다양한 입력 파라미터 변화에 따른 응답시간의 관계에 대한 그래프를 작성하였으며, 성능 평가를 위한 각 파라미터에 대한 정의는 표 1, 2와 같다[3].

표 1 기능적 도메인 부하 모델

기능적 도메인(D_i)	서비스 시간(S_i)	빈도수
D_1 : Low Intensive (e.g. Static Text)	100 msec.	0.1
D_2 : Medium Intensive (e.g. Dynamic Text)	300 msec.	0.65
D_3 : High Intensive (e.g. VOD)	600 msec.	0.25

표 2 시스템 파라미터 (단위: millisecond)

서비스 요청 도착 시간 간격	Exp(평균서비스시간)
도메인 D_i 의 서버 대 수	$Max\left(1, \left\lfloor N \times \frac{S_i}{Sum(S_i)} \right\rfloor\right)$
사용자요청비율 $[class_H, class_L]$	[0.2,0.8]
시간(t)	500

기능적 도메인 D_i 는 Low, Medium, High Intensive로 각각 100, 300, 600 msec. 의 서비스 시간을 갖고, 서비스 요청 도착 시간 간격은 서비스 시간(S_i)의 평균을 모수로 하는 지수분포를 사용하여 나타내며, 사용자 요청 비율은 High Class는 0.2, Low Class는 0.8로 나뉘고, 도메인(D_i)의 서버 대 수는 전체 서버 수에 서비스 시간 비율을 곱한 후 버림(FLOOR)하여 구한다. 도메인은 최소 1대이상의 서버 대 수를 가져야 하며, 버림함수를 사용하기 때문에 남는 서버가 발생하게 된다. 즉, $N-\sum_{i=1}^{n} Max\left(1, \left\lfloor N \times \frac{S_i}{Sum(S_i)} \right\rfloor\right)$ 가 양수일 경우 여분의 서버를 최상위 도메인(e.g. High Intensive)에 강제로 서버를 할당한다.

그림 2는 사용자 도착률에 따른 동질·이질 웹 서비스 구조의 응답시간 변화를 표시하고 있다. 사용자 요청 수가 증가함에 따라 이질 웹 서비스의 응답시간이 동질 웹 서비스일 경우보다 빠르게 나타나는 것을 알 수 있다. 이는 이질 웹 서비스 구조의 사용자 요청이 각 기능적 도메인 서버로 이동하여 기능적 도메인 간의 도착시간에 관계없이 작업을 동시에 수행하는 반면에, 동질 웹 서비스 구조는 사용자 요청이 도착한 순서대로 작업을 수행하여 먼저 도착한 작업이 끝난 후 다음 요청을 수행하기 때문이라 판단된다.

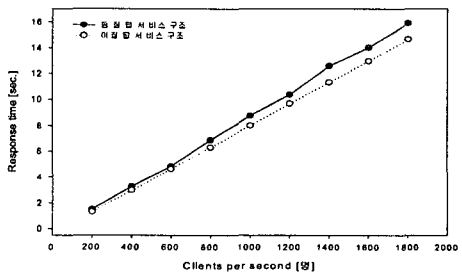

그림 2 사용자 도착률에 따른 동질·이질 웹 서비스 구조의 응답시간비교

그림 3은 이질 웹 서비스 환경에서 사용자 도착률에 따른 정적·동적 서버분할 기법간의 95-percentile 응답 시간 차이를 나타낸다. 95-percentile은 SLA를 고려하기 위해 사용된 척도로써, 들어온 요청중 95%이상은 서비스 제공자와 사용자 간의 계약된 시간 안에 응답시간을 만족하는 것을 의미한다. 사용자 요청 수가 증가함에 따라 동적 분할 방식이 정적 분할보다 응답시간이 빠르게 나타나며, 이는 고정으로 서버를 할당하는 기법보다는 사용자의 요청을 고려하여 서버 노드를 동적으로 할당하는 동적 서버분할 기법을 사용하였기 때문이라 판단된다.

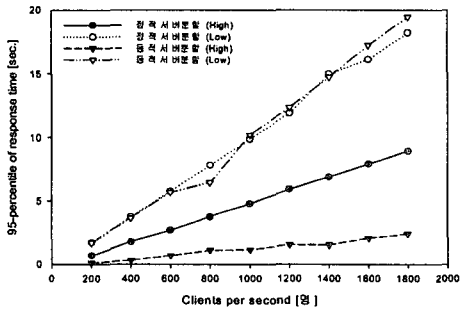

그림 3 사용자 도착률에 따른 정적·동적 서버분할 기법의 95-percentile 응답시간비교

5. 결 론

본 연구에서는 이질 웹 서비스 환경에서 사용자 계층에 따라 효과적인 서버 노드의 동적 분할로 사용자들의 SLA를 보장한 차별화된 서비스를 제공하고 웹 서버의 응답시간 성능을 향상시켰다. 추후에는 이질 웹 서비스 환경에서 사용자 계층을 Multi Class로 확장한 서버 분할 기법 및 승인제어에 대하여 연구할 계획이다.

참고문헌

[1] N. Bhatti and R. Friedrich. "Web server support for tiered services." IEEE Network, 13(5):64-71, Sept./Oct. 1999.

[2] H. Zhu, H. Tang, and T. Yang, "Demand-Driven Service Differentiation in Cluster-based Network Servers," Proceedings of IEEE Infocom, pp. 679-688, Apr. 2001.

[3] V. Cardellini, E. Casalicchio, M. Colajanni, and M. Mambelli. "Web switch support for differentiated services." ACM Performance Evaluation Review, 29, 2001

[4] V.S. Pai, M. Aron, G. Banga, M. Svendsen, P. Druschel W. Zwaenepoel and E. Nahum, "Locality-aware Request DIstribution in Cluster-based Network Servers," In Proceedings of 8th ACM Conference on Architecture Support for Programming Languages, Oct. 1998.

[5] Valeria Cardellini , Emiliano Casalicchio , Michele Colajanni , Philip S. Yu, The state of the art in locally distributed Web-server systems, ACM Computing Surveys (CSUR), v.34 n.2, p.263-311, June 2002