네트워크 경로에 기초한 웹 캐쉬 알고리즘

민 경 훈† · 장 혁 수††

요 약

기존 대부분의 웹 캐쉬 구조가 정적인 형태로 이루어져 있기 때문에 웹 문서 사용자들의 역동적인 요구의 변화를 원활히 지원하지 못하고 있다. 현재의 윈도우 환경 하에서 웹 문서 사용자들의 일반적 경향은 여러 윈도우를 열어 놓고 나잉한 종류의 URL에 접속하고 있으며 비교적 짧은 시간 내에 또 다른 URL로 바꾸어 가면서 사용하고 있다. 본 논문에서는 동일한 캐쉬 용량으로도 적중률이 높고 웹 문서 사용자들의 급격한 URL 변화에도 적중률 저하를 어느 정도 차단할 수 있는 캐쉬 알고리즘을 제시한다. 제안 알고리즘은 요청된 URL이 속한 네트워크 경로에 따른 캐쉬 방법으로 서로 다른 경로에 대해서 서로 다른 서버로 캐쉬하고 각 캐쉬 서버들은 논리적으로 연결하여 캐쉬 시스템(캐쉬 농장)을 구축하고 있다. 제안 알고리즘을 실제 사이트에 적용하여 얻은 검사 결과를 분석하며 보면 기존 알고리즘보다 적중률 및 응답 시간 면에서 현저하게 뛰어남을 알 수 있다.

A Network-path based Web Cache Algorithm

Kyong-Hoon Min† · Hyuk-Soo Jang††

ABSTRACT

Since most of the existing web cache structures are static, they cannot support the dynamic request change of the current WWW users well. Users are generally using multiple programs in several different windows with rapid preference change within a relatively short period of time. We develop a network-path based algorithm. It organizes a cache according to the network path of the requested URLs and builds a network cache farm where caches are logically connected with each other and each cache has its own preference over certain network paths. The algorithm has been implemented and tested in a real site. The performance results show that the new algorithm outperforms the existing static algorithms in the hit ratio and response time dramatically.

1. 서론 및 기존 알고리즘

인터넷 이용자 수, 호스트 수 및 인터넷 서비스 공급자(ISP : Internet Service Provider) 수의 증가는 국내뿐만 아니라 전 세계적으로 폭발적이며 이러한 추세는 전자 상거래 활성화와 더불어 더욱 가속화되고 있다. 인터넷 사용자가 요구하는 정보 및 데이터는 우선적으로 웹 캐쉬(cache)를 검색하여 가져오게 되는데 캐쉬에 원하는 정보가 없으면 원래 요구된 서버 또는 이웃 캐쉬 서버로부터 필요한 정보를 받게 된다. 따라서 원하는 정보가 캐쉬에 없으면 없을수록 인터넷망에 걸리는 트래픽 양도 많아지고 데이터의 이동 거리도 길어지게 되어 망 전체의 성능을 저하시키며 응답 시간이 길어진다[1, 2, 3]. 대부분의 웹 서버가 국외, 특히 미국에 있는 현실을 고려하면 국외로 연결된 대륙 간 회선에 병목 현상을 초래하고 외국 통신 업체에게 회선 사용료를 과다 지불하게 된다.

현재 사용중인 웹 캐쉬 형태는 주로 "CERN"[2, 4]과 "Harvest/Squid"[4] 방식으로 정적인 캐쉬 계층 구조에

※ 본 결과는 정보통신부의 정보통신 우수시범학교의 지원사업에 의하여 수행된 것입니다.
† 준 회 원 : 명지대학교 정보통신공학과 대학원
†† 정 회 원 : 명지대학교 전자 정보통신 공학부 교수
논문접수 2000년 4월 20일, 심사완료 2000년 6월 20일

기반을 두고 있다. 이러한 구조는 사용자 또는 호스트별로 해당 캐쉬 서버를 미리 지정하도록 되어 있으며 계층적 캐쉬 상호간 하드 와이어로 연결되어있다. 따라서 캐쉬 서버에 원하는 정보가 없으면 이 캐쉬 서버와 연결된 또 다른 캐쉬 서버로 요구가 넘어간다. 서버 중 더 이상 연결되어 있지 않은 최종 부모(parent) 서버의 캐쉬에 원하는 정보가 없으면 원래의 웹 서버에서 정보를 가져오게 한다. "Harvest/Squid" 방식의 특징은 프락시(proxy) 서버가 동일 망 내부에서는 몇 개의 동료(sibling) 서버에 하드와이어로 연결되어 있고 망 외부로는 하나의 부모 서버에 연결되어 있다. 검색 속도를 빠르게 하기 위해 부모 및 동료 서버에게 필요로 하는 정보를 동시에 멀티 캐스트하고 해결이 안된 요청(request)은 상위 서버로 전달한다. 현재 "Harvest" 방식이 가장 많이 쓰이고 있으나 이러한 방식은 엄청난 양의 트래픽을 양산 할 가능성이 있어 캐쉬 설치 본래의 목적에 위배 될 수 있다. "CERN" 및 "Harvest" 방식 모두 정적 계층 구조로 인해 프락시 하나의 결함이 프락시 전체의 결함으로 오 동작할 가능성이 있다. 아울러 망 구성 및 트래픽 양의 변화에 프락시 구성을 자유롭게 바꾸기 어렵고 다른 캐쉬에 중복된 정보의 복사본을 가지고 있을 가능성이 있어 자료간 불일치를 초래 할 수 있으며 결과적으로 기억 장소의 낭비로 이어질 수 있다. 최근에는 캐쉬 서버들 간의 통신에 관한 연구[2, 5, 6] 및 분산된 캐쉬 서버간 동적 구조와 운영 방법[7]등에 관한 연구가 활발히 진행되고 있다. 하지만 이러한 연구들의 취약점은 실제 인터넷 사용자의 웹 문서 요구 형태를 반영시키지 않고 어느 조직이든 일률적인 웹 캐쉬 구조 및 방식을 적용하고 있다는 사실이다.

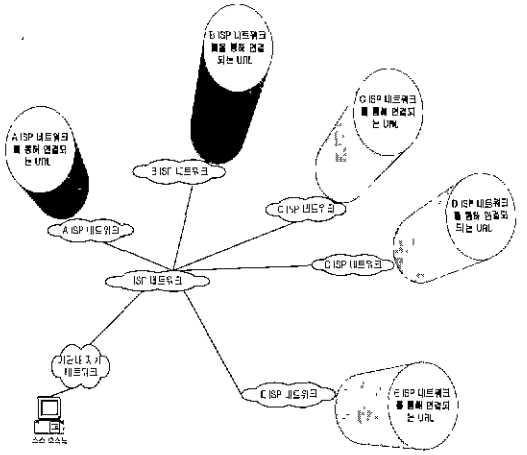
현재의 캐쉬 서버 운영 형태는 사용자 요구(request)에 의한 캐쉬으로 최근에 요구율이 높지 않은 정보는 저장 능력의 한계 때문에 캐쉬 서버에서 교체된다. 다양한 웹 문서의 폭발적 증가와 한 명의 사용자가 윈도우를 여럿 열어 놓고 동시에 다양한 웹 문서를 이용하는 지금의 상황에 최근 요구율이 높은 웹 문서만을 캐쉬하는 방법은 적절한 캐쉬 방안이 되지 못할 것으로 생각된다. 또한 각 사이트의 요구율이 비슷할 경우 전반적으로는 요구율 차이가 없음에도 불구하고 사이트 요구 순서의 역동적 변화로 인해 우선 순위에서 밀려 제거된 웹 문서에 대해 다시 요구하게 되면 원래 서버로 가서 데이터를 가져 와야 하는 상황이 빌생하게 된다. 이렇듯 사용자의 URL(Uniform Resource Locator) 선호도 변화를 반영하는 웹 캐쉬 서버 구조를 구축하는데 어려움이 있었다. 본 논문에서는 인터넷 사용자의 웹 문서 요구 형태에 대한 데이터를 수집 분석하여 요구되는 URL을 네트워크 경로에 기초하여 분류하고 서로 다른 캐쉬 서버에 중복되지 않게 저장하며 웹 문서 요구를 라우터가 해당 캐쉬 서버로 연결시키는 방법을 사용하여 동일한 용량의 캐쉬로도 운영 방법의 개선을 통하여 적중률을 높일 수 있는 알고리즘을 제시 하고자 한다.

2절에서는 본 논문에서 제안하는 웹 캐쉬 알고리즘을 설명하고 제 3절에서는 제안된 캐쉬 알고리즘을 실제 사이트에 적용한 실례를 보여 주며 제 4절에서는 기존 방식과의 성능 분석 및 결과 비교를 시행하고 제 5절에서 결론을 맺는다.

2. 웹 캐쉬 알고리즘 제안

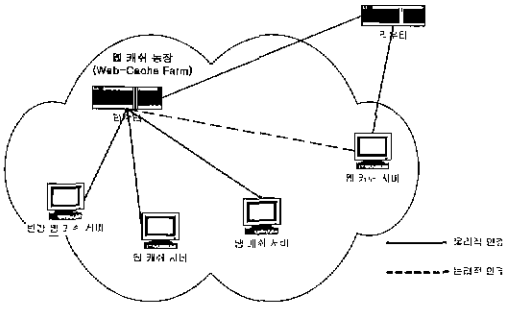
웹 캐쉬를 두는 목적이 캐쉬의 적중률을 높여 원거리 원래(original) 웹 서버까지 가야 할 확률을 줄이고 이를 통한 네트워크 트래픽 양을 감소시켜 통신비용을 절감 하고자 하는데 있다. 대부분의 웹 캐쉬는 정적인 계층 구조를 가지고 있으며 원하는 데이터가 없을 때 물리적으로 연결된 캐쉬 서버에게 요청을 넘겨주는 형태를 취하고 있다. 캐쉬에 있을 확률을 높이기 위해서는 캐쉬에 있는 정보의 내용을 체계적으로 분류하여 적중률이 가장 높은 캐쉬로 요청을 라우팅 해주는 캐쉬 시스템 구축이 필요하다 하지만 인터넷에 존재하는 URL과 그에 따른 객체의 종류 및 수가 워낙 많아 현실적인 분류 방법을 찾기에 어렵다. 이러한 문제를 해결하는 방법으로 조직 또는 기관의 자치 영역(AS : Autonomous System)의 임의의 호스트에서 출발하여 목적지 URL에 이르는 네트워크 경로를 조사해 보면 대략 어느 시점까지 몇 개의 네트워크 경로를 공통으로 이용한다는 사실을 알 수 있다 이는 원래 서버에 있는 문서의 속성이나 물리적인 위치 정보를 알 수 없는 현재의 인터넷 상황에서 간단한 명령어, "traceroute"로 쉽게 파악 가능한 네트워크 경로를 바탕으로 사용자의 인터넷 이용 현황을 분석한 결과이다 특히 국내에서는 (그림 1)에 나타났듯이 네트워크 경로별 특성이 대략 ISP별로 구분 할 수도 있는데 이는 국내 ISP 수가 미국등 선진국에 비해 상대적으로 많지 않은 데서 기인하지 않나 생각한다. 또한 인터넷 사용자 한

명에게 주어진 환경 및 취향을 조사해 보면 다수의 원도우를 통하여 성격이 전혀 다른 URL들을 수시로 바꿔가며 사용하고 있어 캐쉬 시스템에 저장하는 URL 종류를 다양화하는 것이 캐쉬 적중률을 높일 수 있을 것으로 생각된다. 본 논문은 URL이 몰리는 대표적인 몇 개의 네트워크 경로별로 캐싱하는 알고리즘을 제안하며 캐쉬별 저장 정보를 라우터가 알 수 있도록 함으로써 사용자의 URL 요청에 대해 적중률이 가장 높은 캐쉬로 라우팅 하는 캐쉬 시스템을 구축하도록 한다. 캐쉬별 저장 정보는 다양화 및 특성화되어 사용자 취향의 급격한 변화에도 적중률의 급격한 저하를 차단하는 효과를 얻을 수 있다.

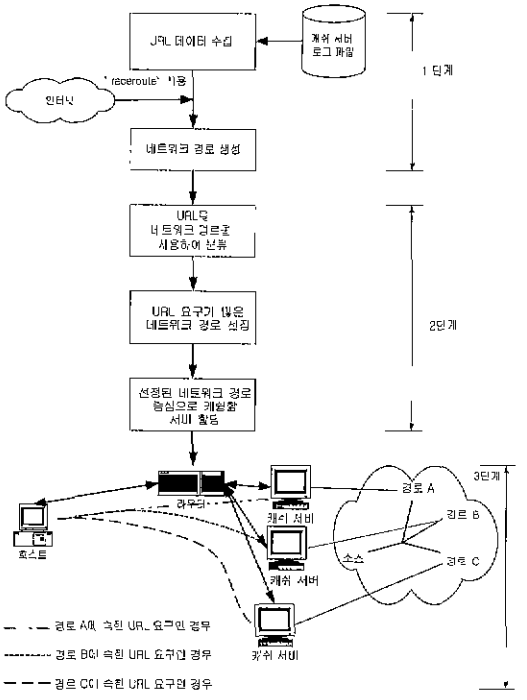


(그림 1) 네트워크 경로에 의한 분류 방법

제안 알고리즘은 분류된 URL을 캐싱하는 서버와 웹 문서 사용자를 연결 시켜 주기 위해 (그림 2)와 같이 하나의 라우터와 여러 개의 캐쉬서버를 엮는 웹 캐쉬 농장(Web-Cache Farm)[8] 형태의 캐쉬 시스템을 이용한다. 웹 캐쉬 농장 구조는 반장 웹 캐쉬(designated web-cache)가 모든 연결 설정(redirect) 및 트래픽 분산 방법을 관리하며 각 캐쉬 서버가 가지고 있는 정보를 라우터에게 넘겨주어 라우터가 이 정보를 이용하여 웹 문서 요청을 해당 웹 캐쉬 서버로 배달하게 한다. 웹 캐쉬 농장의 캐쉬들이 저장하는 정보를 네트워크 경로별로 특성화하여 동일한 정보가 여러 캐쉬에 중복해 존재하는 것을 막고 캐쉬농장에 다양한 종류의 정보를 저장하게 하여 사용자 취향이 급격히 변해도 웹 캐쉬 농장내의 특정 캐쉬에 원하는 정보가 있을 확률이 저하되지 않도록 시도하였다.



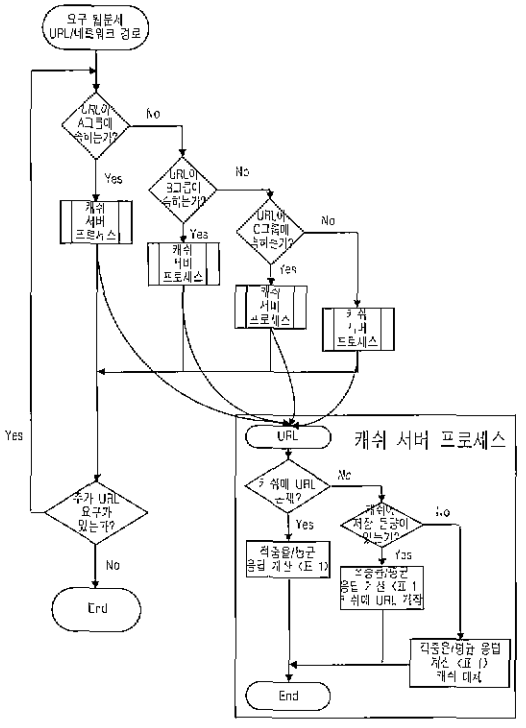
(그림 2) 웹 캐쉬 농장



(그림 3) 제안 웹 캐쉬 알고리즘

본 논문에서 제안하는 웹 캐쉬 알고리즘은 (그림 3)과 같다. 알고리즘 1 단계는 정보 수집 단계로 기관내 사용자들이 사용하는 URL을 알기 위해 캐쉬 서버의 로그 파일을 수집하고, 이를 "traceroute"를 사용하여 각 URL이 위치한 네트워크 경로를 생성한다. 2 단계는 정보 분석 단계로 생성된 네트워크 경로를 사용하여 URL을 분류하는데 이를 위해 각 URL이 지나는 공통경로를 찾고 공통경로 및 공통경로 이후 경로별 특성을 분류 기준으로 사용한다. 이 과정을 통하면 각

URL을 특정 네트워크 경로로 나타낼 수 있다 다음으로 네트워크 경로 중 요청이 많은 네트워크 경로를 선정한다. 이는 웹 문서 사용자의 URL 요청이 많다는 것으로 주 캐쉬 대상이 되어야 하기 때문이다. 몇 개의 경로가 선정이 되면 이를 캐쉬할 서버에 할당하고 선정된 경로와 이에 해당하는 서버 정보를 라우터로 보낸다. 3단계는 운영 단계로 사용자가 URL을 요청하면 라우터가 요청 URL을 보고 해당하는 캐쉬 서버로 연결을 바꾸어주는 것이다 캐쉬 서버는 특정 경로에 대해서 캐쉬를 하게 되며 사용자의 URL 요청은 URL이 속한 경로에 따라 해당 서버로 보내지게 된다. 이때 요청 URL중 URL이 속한 경로에 해당하는 캐쉬 서버가 없는 경우는 디폴트 캐쉬 서버로 보내진다 (그림 4)는 편의상 4개 서버를 통한 전체 알고리즘 중 제 3단계 과정의 논리 흐름을 보여 주고 있으며 <표 2>와 <표 3>은 논리 흐름을 기능 별로 기술한 것이다 URL 요청이 들어오면 웹 캐쉬 농장 내 라우터가 URL이 해당하는 경로를 판단하여 해당 캐쉬 서버로 보내며 캐쉬 서버는 저장된 URL과 요청 URL을 비교하여 정보가 있을 경우 요청에 응답을 하고 없을 경우



(그림 4) 제안 알고리즘의 논리 흐름도

원래 서버로 요청을 보낸 후 받은 응답을 캐쉬 서버에 일단 저장한 후 요청한 사용자에게 보낸다. 제 4절에 설명될 성능 평가를 위해 <표 1>과 같은 계산 방법이 사용되었다 적중이라는 것은 요청한 URL이 해당 서버에 있을 경우를 말하며 적중률은 전체 요청 URL중 적중한 URL의 비율이다. 응답 시간은 적중할 경우는 캐쉬 서버까지 갔다온 시간이 되고 실패했을 경우는 캐쉬 서버까지 간 시간에 원래 서버까지 갔다온 시간, 다시 캐쉬 서버에서 사용자까지 간 시간의 합이 된다. 본 본문에서는 응답 시간을 편의상 홉(hop) 수로 사용하였다

〈표 1〉 적중률과 평균 응답 시간 계산 방법

적중률(%)	= 적중된 URL 개수 / 전체 요구 URL 개수×100
평균 응답 시간 (홉 수)	= (적중되었을 때 전체 홉 수 합 + 실패했을 경우 전체 홉 수 합)/ 전체 요구 URL 개수

〈표 2〉 라우터 알고리즘

while	URL 요구가 있는 동안
if	요구한 URL이 속한 경로가 A 그룹에 속하는가
	A 서버로 리우팅 변경 A 서버에서 캐쉬 프로세스 <표 3> 참조
else if	B 그룹에 속하는가
	B 서버로 리우팅 변경 B 서버에서 캐쉬 프로세스 <표 3> 참조
else if	C 그룹에 속하는가
	C 서버로 리우팅 변경 C 서버에서 캐쉬 프로세스 <표 3> 참조
else	디폴트 서버로 라우팅 변경 디폴트 서버에서 캐쉬 프로세스 <표 3> 참조
end	

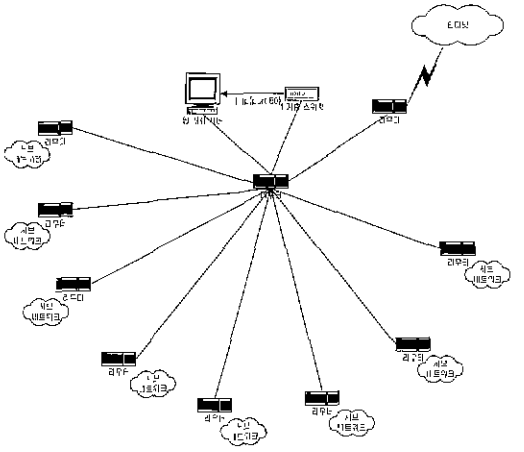
〈표 3〉 캐쉬 서버 프로세스

if	캐쉬에 요구 URL이 있는가
적중	
else	적중 실패 원래 서버로부터 데이터를 가져와서 최근에 사용된 지가 가장 오래된 URL 정보와 대체

3. 제안 알고리즘 적용 방법

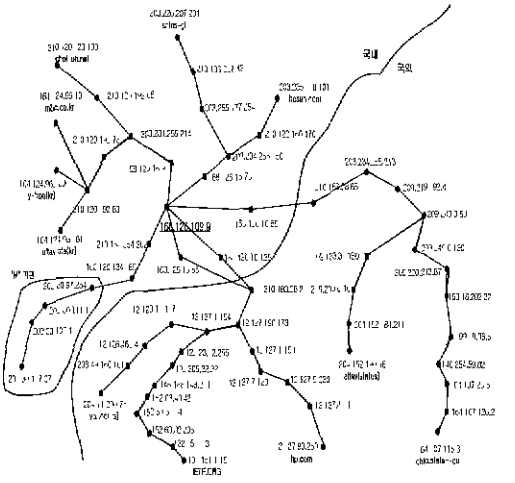
제안 알고리즘 적용 대상 기관은 기존에 많이 사용되고 있는 캐쉬 서버의 연결 형태인 (그림 5)와 같은 정적 구조를 가지고 있으며, 원래 서버(orginal server)의 URL, 이름(도메인 이름) 또는 IP 주소, 문서 이름

및 내용을 캐쉬에 저장하고 이 URL에 대한 요구가 다시 발생할 경우 저장되어 있는 정보를 응답으로 보내주고, 만약 데이터가 있지 않는 경우는 원래 서버로 가서 데이터를 가지와 저장을 하고 응답을 하게 된다



(그림 5) 알고리즘 적용 대상 기관의 웹 캐쉬 서버 구성

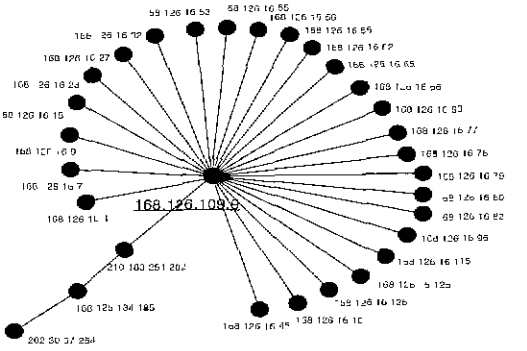
(그림 6)은 알고리즘 적용 대상인 국내 "M" 기관에서 사용자가 주로 찾아가는 웹 문서 URL 기준으로 네트워크 경로를 조사한 그림이다.



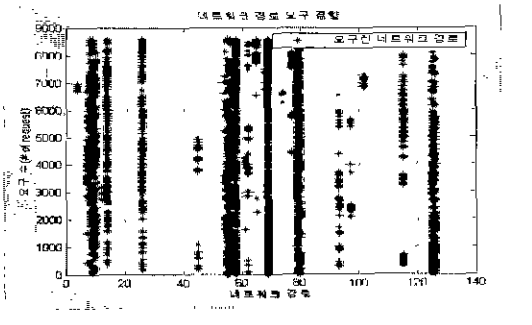
(그림 6) "M" 기관에서 웹 문서들의 네트워크 경로

(그림 6)에서 볼 수 있듯이 크게 국내와 국외로 구분이 가능하며, 특히 거의 같은 경로를 이용하던 트래픽들이 외부 ISP 라우터 "168.126.109.9"에서부터 여러 경로로 나누어진다는 것을 알 수 있다

따라서 이 라우터를 기준으로 연결된 네트워크들을 분류 대상으로 이용을 할 수 있다. (그림 7)는 (그림 6)에서 판단한 네트워크 경로 분기점 "168.126.109.9"를 기준으로 사용하여, 실제 "M" 기관 사용자들의 웹 문서 URL 리스트를 가져와 분류한 경로를 나타낸 것이다. 보다 자세한 분류가 필요하다면 (그림 7)의 끝점에서 이어지는 경로를 따라 보다 더 많은 홉 수를 이용하면 더 자세하게 분류 할 수 있다. 우리가 실험한 실제 사이트에서, "M" 기관 경로 유형을 조사해 보면 주 관심 노드 "168.126.109.9"까지는 거의 같은 경로를 따라가고 그 이후에 한 홉 정도 떨어진 서로 다른 노드로 가고 그 이후에는 다양한 경로로 갈라짐을 알 수 있다. 이를 통해서 주 관심 노드에서 한 홉 정도 떨어진 노드까지의 경로의 수를 조사해 보면 26개로 압축됨을 알 수 있다.



(그림 7) 네트워크 경로에 따른 분류



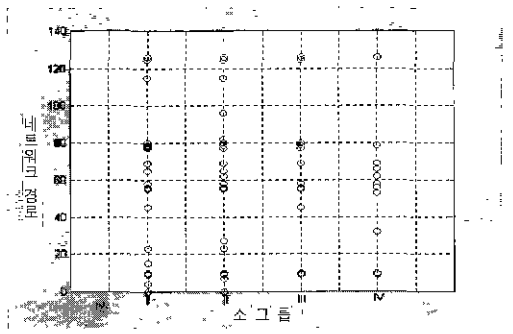
(그림 8) 웹 문서 네트워크 경로별 요구 수

네트워크 경로에 따라 실제 사용자들이 요구한 웹 문서 요구 수를 (그림 8)에 나타내었다. 네트워크 경로는 B형 IP 주소인 "168.126.109.9"를 기준으로 연결이 되는 라우터 IP 주소인 "168.126.16.4"에서 "168.126.16.126"까지 중에서 호스트 주소 부분만을 나타낸 것이다. (그림 8)에서와 같이 사용자의 요구 문서를 찾아가는 경

로가 일정 유형을 보이고 있으며 몇몇의 특정한 경로에 집중되어 있음을 발견할 수 있고 이러한 경로에 집중되는 요구율은 <표 4>와 같으며 전체 요구의 60%가 특정 경로 4개에 집중됨을 알 수 있다.

〈표 4〉 특정 경로에 대한 요구율

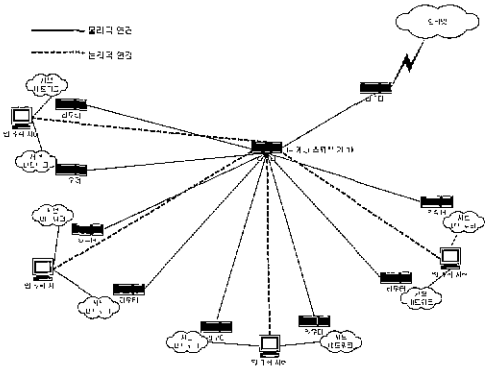
요구가 많은 상위 경로수	3	4	5	6
요구율 (해당 경로 요구수/전체 요구 수, %)	51 %	60 %	67 %	73 %



(그림 9) 그룹별 관심 네트워크 경로

임의의 기관에서 인터넷 사용자의 취향을 파악하기 위해 관련 기관을 몇 개의 소그룹으로 나누고 각 소그룹마다 캐쉬 서버를 둔 형태에 대해 분석하였다. 소그룹 분류는 기관 또는 회사의 성격별, 기관 또는 회사 내의 업무 및 전공 성격 등을 이용하여 분류를 할 수 있는데 가능한 그룹원의 동질성이 어느 정도 캐쉬에 반영되도록 하였다. "M"기관의 실험 데이터에 근거하여 4개의 소그룹으로 나누었을 때 각 그룹별 찾는 웹 문서별 네트워크 경로를 (그림 9)에 나타내었다. 편의상 실험 사이트의 업무 유사성을 고려하여 대략 4개 정도로 분류한 것이다. 우리가 알 수 있는 것은 각 그룹에 관계없이 비슷한 네트워크 경로를 가지고 있고 기존의 정적 캐쉬 구조로는 각 그룹별 캐쉬 서버에 저장된 내용이 유사하여 결국 메모리 낭비 및 캐쉬간 데이터의 일치성 유지에 어려움이 있을 것으로 예견된다. 이 경우 캐쉬 적중률을 높이려면 캐쉬별 저장하는 정보 종류를 다양화 및 특성화해야 한다.

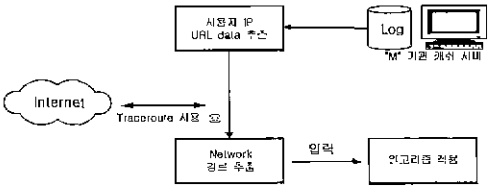
본 논문에서 제안된 알고리즘을 적용한 웹 캐쉬 서버 구성은 (그림 10)과 같으며 이는 4개의 소그룹으로 분류했을 경우의 예이다. 각 캐쉬 서버는 중앙의 라우터와 연결이 되고 4개 캐쉬 서버 중의 임의의 캐쉬 서버가 반장 캐쉬 서버(designated cache server)가 된다. 반장 캐쉬 서버는 각 캐쉬 서버의 데이터를 네트워크 경로 주소를 이용하여 분류하고 이를 이용하여 캐쉬 서버별 특징을 부여한다. 즉, 하나의 캐쉬 서버는 특정 경로에 속한 사이트에 대해서 다른 캐쉬 서버에 비해 우선 순위가 높은 캐쉬을 하고 있다는 것을 캐쉬 서버의 특징으로 부여하고, 이 정보를 라우터에게 주어서 사용자의 웹 문서 요구 발생 시 해당 캐쉬 서버로 라우팅 하게 한다.



(그림 10) 적용된 웹 캐쉬 서버 구성

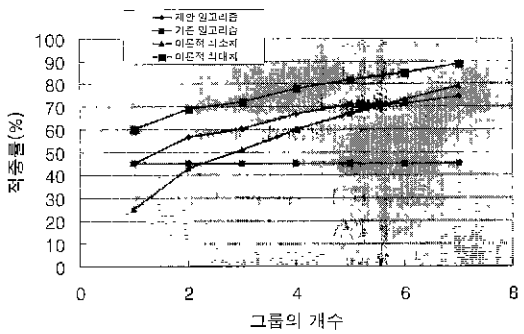
4. 알고리즘 성능 분석

성능 분석을 위해 "M" 기관으로부터 수집한 사용자들의 로그 파일 내용 중 URL 8600개 정도를 입력 데이터로 사용하였다. 이 로그 파일에서 우리가 필요한 사용자 IP, 요구한 웹 문서 URL을 추출하였다. 추출 및 분석 과정은 (그림 11)에 나타나있다. 추출된 데이터를 정적인 구조를 사용하는 기존 알고리즘의 입력 데이터로 사용 할 수 있으며 제안된 알고리즘에 사용하기 위해서는 네트워크 경로를 파악해야 하고 이를 위해서 "traceroute"를 이용하여 네트워크 경로가 추출된 데이터 파일을 만들었다. 제안된 알고리즘과 기존 알고리즘을 입력 파일에 따라 실행시킨 결과 적중률과 응답 시간을 구하였다.



(그림 11) 성능 분석 체계도

그룹 분류는 요구율이 많이 몰려 있는 네트워크 경로를 중심으로 하여 그룹 수를 1개에서 7개까지 구분하여 성능 분석을 하였다. 기존 알고리즘이나 제안된 알고리즘 모두에 적용되는 캐쉬의 전체 크기는 동일하게 하였고 사용자 요구 URL 데이터를 입력한 후 각 알고리즘별 적중률과 평균 응답 시간을 알아냈다.



(그림 12) 그룹의 수에 따른 적중률

그룹 개수별 적중률에 대한 성능 분석 결과는 (그림 12)에 나타냈다. 그룹의 개수가 1개인 경우는 기관 전체에 캐쉬 1개를 설치하는 것이며 이때는 제안된 알고리즘과 기존 알고리즘의 구분 없이 동일하고 모두 46% 정도의 적중률을 보인다. 두 개의 그룹으로 분류가 되면 제안 알고리즘의 적중률은 10% 이상 향상된 57%가 된다. 더 많은 그룹으로 분류가 되면 적중률은 계속적으로 향상이 되다가 어느 시점이 지나면 적중률 향상은 둔화를 보인다. 실험 대상 "M" 기관은 4개 또는 5개 그룹으로 분류할 때 가장 최적의 효율을 나타내는 것으로 나타났다. 이 판단의 근거는 실제 적중률과 이론적 계산에 의한 최소/최대치 적중률 비교를 통해 실제 적중률이 이론적 최소치에 근접하거나 오히려 나빠지기 직전 단계의 그룹수가 적정 효율을 나타내는 것으로 판단했다. 이론적 최소치와 최대치는 <표 5>와 같이 표현될 수가 있다.

〈표 5〉 이론적 최소/최대치 계산 공식

이론적 최소치(%)	= 분류된 경로 개수별 요구율
이론적 최대치(%)	= 경로 개수별 요구율 + (100 - 경로 개수별 요구율)× 그룹 분류가 되지 않았을 경우(즉 그룹 수가 1개일 경우) 요구율

해당 경로에 대한 캐쉬를 하는 서버가 있으면 그 경로에 대한 요구를 100% 수용할 수 있다고 가정하면 이론적 적중률의 최소치란 경로별 요구율과 같다. 따라서 <표 2>의 경로 수에 따른 요구율은 각 경로별로 그룹화 하면 그룹 수에 따른 이론적 최소 적중률이라고 할 수 있다. 이론적 최대치란 경로별 그룹화 하였을 때 최대 예상 적중률을 의미하며 이는 경로별 요구는 100% 수용하며 경로에 속하지 않은 요구에 대한 적중률은 경로별로 분리하지 않았을 때의 적중률 정도를 계산하여 구할 수 있다. 즉, 경로별 그룹화에 따른 요구율에 그룹에 속하지 않은 요구에 대한 예상 적중률을 합한 것이 이론적 최대치가 된다. 예를 들어, 그룹수가 3개일 때 전체 요구 중 상위 3개의 그룹에 대한 요구율은 <표 4>에서와 같이 51%이며 나머지 경로에 대한 요구율은 49%가 된다.

이 나머지 49%중에서도 캐쉬에 적중될 확률은 캐쉬를 분류하지 않고도 얻을 수 있는 적중률, 즉 1개 그룹일 때의 적중률 46%를 적용할 수 있으며 따라서 전체적으로는 21%(=49%×46%)정도의 적중률이 된다. 즉 3개의 그룹에 속했을 확률 51%는 100% 적중하고 나머지 중에서도 적중할 확률 21%를 합하면 최대 적중률은 72%가 된다. 이와 같은 근거에 의해 4개 또는 5개 그룹으로 나누어 졌을 경우에 최적의 성능을 보일 수 있다. 성능 분석에서는 7개 그룹까지 고려했고 적중률을 더 높이고자 한다면 최대 분류가 가능한 26개 그룹까지도 가능하겠지만 실제로는 캐쉬의 크기 제한에 의해 그룹 수가 6개를 넘으면 적중률은 향상되지 않는다. 이는 그룹 수가 많아질수록 분류된 캐쉬의 크기는 작아지고 제한적 크기의 캐쉬에 의해 필연적으로 캐쉬 데이터 교체가 자주 일어나게 되고 이로 인해 적중률은 오히려 나빠진 것으로 생각된다.

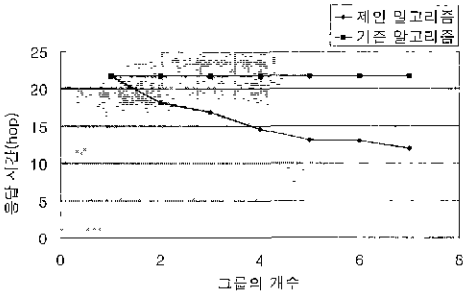
적중률에 대한 이론적인 수치 값과 실험 사이트 데이터를 통해 얻은 실제 값은 <표 6>에 나타나 있다.

〈표 6〉 실제 적중률과 이론적 계산값

그룹 수	1	2	3	4	5	6	7
실제 측정치	46%	57%	62%	67%	71%	73%	74%
이론적 최소치	25%	43%	51%	60%	67%	73%	79%
이론적 최대치	60%	69%	72%	78%	82%	85%	89%

그룹의 개수별 응답 시간을 흡 수로 나타낸 것이 (그림 13)에 있다 응답 시간은 적중률과 밀접한 관련이 있는데 적중률이 좋아지면 원래 서버로 가야 하는 요구 수가 줄고 이로 인해 응답 시간은 빨라지게 된

다. 성능 분석 결과도 마찬가지로 적중률에 비례해서 응답 시간이 향상됨을 알 수 있다. 1개 그룹 분류는 기존 알고리즘과 같은 21합 정도이고 2개 그룹 분류는 3합이 줄어든 18합이 되며 이후 계속 2합 정도의 향상이 보이다가 4개의 그룹 분류 이후부터는 둔화를 보이기 시작하며 1합 정도 향상이 된다. 응답 시간 성능 분석에도 알 수 있듯이 4개의 또는 5개의 그룹 분류 이후부터는 뚜렷한 향상이 보이지 않기 때문에 4개 또는 5개의 분류가 최적의 효율을 나타낸다는 것을 알 수 있다.



(그림 13) 그룹의 수에 따른 응답 시간

성능 분석 결과, 실험 대상 "M" 기관은 4개 또는 5개 그룹 분류를 가지는 캐쉬 시스템으로 바뀔 경우 뚜렷한 성능 향상이 기대가 되며 현재의 상황뿐만 아니라 이후 보다 빠르게 변화하는 사용자 경향에도 충분히 적응 할 수 있을 것으로 생각된다.

5. 결 론

여러 윈도우를 동시에 열어 놓고 인터넷 사이트를 동시에 접속하여 다양한 종류의 웹 문서를 요구하며 요구하는 문서 종류 또한 수시로 빠르게 변하는 현재의 인터넷 사용자 취향을 고려하면 기존의 정적 캐쉬 구조 형태의 알고리즘으로는 적중률 및 응답 시간이 나빠지는 결과를 초래한다. 이러한 환경에서도 적중률의 저하를 줄일 수 있는 네트워크 경로에 의거한 웹 캐쉬 알고리즘을 제안하였고 성능 분석을 통하여 같은 용량의 캐쉬로 기존의 알고리즘 보다 적중률은 25%이상 향상시키며 응답 시간은 33%이상 빠르게 할 수 있음을 보였고 사용자 요구 내용이 빠르게 변하는 환경에도 능동적으로 대처 할 수 있음을 보였다. 이런 연구 결과를 바탕으로 캐쉬의 효율성을 보다 높일 방법 연구를 위해 URL, 네트워크 경로, 그리고 사용자 요구 간 연관 관계를 보다 자세히 분석할 것이며 또한 QoS (Quality of Service)가 요구되는 멀티미디어 서비스에 적용할 수 있는 웹 캐쉬 방안에 대해서도 연구를 계속할 예정이다

참 고 문 헌

[1] Lee Breslua, Pei Cao, Li Fan, Graham Phillips, Scott Shenker, "Web Caching and Zipf-like Distributions : Evidence and Implication," IEEE INFOCOM, 1999.
[2] Duane Wessels, K. Claffy, "ICP and the Squid Web Cache," IEEE Journal, Apr. 1998.
[3] H. Sun, X Zang, K S. Trivedi, "The effect of Web caching on the network planning," Elseviser Science B V , 1999.
[4] A Lutonen, H. F. Nielsen, T. Berners-Lee, "Cern httpd." July 1996.
[5] Keith W. Ross. "Hash Routing for Collections of Shared Web Cache," IEEE Network, Nov. 1997.
[6] Li Fan, Pei Cao, Jussara Almeida, "Summary Cache : A Scalable Wide-Area Web Cache Sharing Protocol," ACM, 1998.
[7] Samrat Bhattacharjee, Kenneth L. Calvert, Ellen W. Zegura, "Self-Organizing Wide-Area Network Caches," IEEE INFOCOM, 1998.
[8] M. Cieslak, D. Forster, "Web Cache Coordination protocol V1.0," June 1999.

민 경 훈

e-mail : ncominga@mju.ac.kr
1998년 명지대학교 정보통신공학과(공학사)
1998년~현재 명지대학교 정보통신공학과 석사 과정
관심분야 인터넷, 컴퓨터통신

장 혁 수

e-mail : jang-h@mju.ac.kr
1975년~1983년 서울대학교 산업공학과(공학사)
1986년 미국 Ohio State University 산업 및 시스템 공학석사
1990년 미국 Ohio State University 전산학 박사
1990년~1992년 경북대 전자공학과 교수
1998년~1998년 미국 O.S.U 전산학과 교환 교수
1992년~현재 명지대 전자·정보통신 공학부 교수
관심분야 : 인터넷, 컴퓨터통신, 컴퓨터구조