

통신비용을 최소화하는 복제 웹컨텐츠 분배나무 구성을 위한 개선된 알고리즘*

홍성필** · 이동권***

Improved Algorithms for Minimum Cost Replicated Web Contents Distribution Tree*

Sung-Pil Hong** · Dong-Gwon Lee***

■ Abstract ■

Recently, Tang and Chanson proposed a minimum cost distribution model for replicated Web contents subject to an expiration-based consistency management. Their model is a progress in that it can consider multiple replicas via the network of servers located on the Web. The proposed greedy heuristic, however, has an undesirable feature that the solution tends to converge a local optimum at an early stage of the algorithm. In this paper, we propose an algorithm based on a simple idea of preventing the early local convergence. The new algorithm provides solutions whose cost are, on the average, 27% lower than in the previous algorithm.

Keyword : Consistency Management of Replicated Contents, Distribution Tree Configuration, Forest-growing Heuristics, Computational Experiment

1. 서 론

멀티미디어 컨텐츠 서비스 산업이 빠른 속도로 증가함에 따라, 인터넷을 통한 정보의 양과 규모가

급속하게 증가하여 서버와 네트워크의 부담이 가중되고 있다. 웹의 기본 모델은 클라이언트-서버이기 때문에 특정 서버에 집중된 부하는 트래픽을 증가시키고, 클라이언트에게 많은 대기 시간을 요구

논문접수일 : 2005년 7월 17일 논문게재확정일 : 2005년 10월 25일

* 이 논문은 2002년도 중앙대학교 학술연구비 지원에 의한 것임.

** 서울대학교 산업공학과(교신저자)

*** 중앙대학교 정보시스템학과

하여 통신비용을 증가시키게 된다[4]. 이에 대한 해결책으로 캐싱과 미러링 같은 기술이 사용되어 왔으며[6], 최근에는 이러한 기술을 한 단계 발전시켜 전체 네트워크 상에 동일한 콘텐츠 내용을 복제하고 분산시켜 클라이언트들의 액세스를 용이하도록 하는 CDN(Content Delivery Network) 기술이 주목 받고 있다[10].

이러한 기술을 활용하는데 중요한 사항 중에는 하나는 콘텐츠의 원본과 복제본 사이의 일치성을 유지시켜 나가는 것인데[6, 11, 13], 이를 위해 복제 콘텐츠의 유효시간, 즉 TTL(time-to-live)을 정해주는 방식이 가장 널리 사용된다. 일반적으로 TTL이란, 웹상에 존재하는 특정 객체가 유지되는 시간을 의미하며, 본 연구의 경우는 복제본이 제공된 후 해당 서버에서 얼마나 오랫동안 유효한지를 의미한다. 따라서 정해진 유효시간의 값이 작을수록 클라이언트가 새로운 콘텐츠를 가져갈 확률이 높아진다[4, 7].

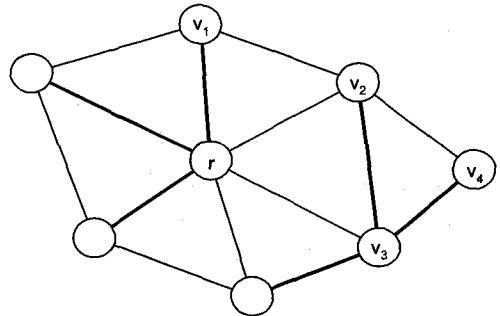
이러한 원-복제본의 일치성 유지에 소요되는 전체 통신비용을 최소화하도록, 원-복제 콘텐츠 서버의 네트워크를 구성하는 문제가 발생하게 된다. 최근 Tang and Chanson[10]은 최초로 복수의 복사본까지 고려할 수 있는, 현실성이 보다 높은 모형을 제시하였다. 그러나 이 모형은 계산적으로 NP-hard 문제가 되며, 지나친 계산시간을 소비할 수 없는 문제의 특성에 따라, Tang and Chanson은 다항시간 발견적 기법(heuristic)을 제안하였다. 이 알고리즘은 서버를 중심으로 현재까지 구성된 분배나무에 저렴한 통신비용으로 추가할 수 있는 마디를 추가하는, 'greedy' 유형의 나무 키우기(tree-growing) 알고리즘이다. 본 논문에서는 비교적 간단한 아이디어를 사용하여 다항성을 유지하며 알고리즘이 개선될 수 있음을 보이려고 한다. 즉, 나무를 키워나가면 생성되는 해가 초기부터 특정한 국지해(local optima)에 수렴하는 경향이 커지기 때문에, 이를 방지하기 위해 나무 대신 숲을 키워가는(forests-growing) 것이 본 논문이 제시하려는 알고리즘의 원리이다.

본 논문의 2장에서는 Tang and Chanson의 모형을 간략히 정리한다. 3장에서는 기존 알고리즘을 살펴보고 이를 개선하는 알고리즘을 제안한다. 4장에서는 전산실험을 통해 제안 알고리즘을 평가한다. 5장에서는 향후 연구과제를 간단히 제시한다.

2. Tang and Chanson의 모형

2.1 시스템 모델

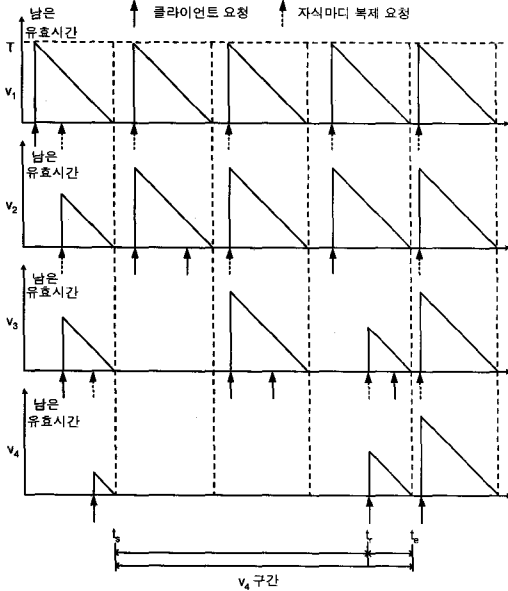
원본 서버 r 과 모든 복사본 서버, v_1, v_2, \dots, v_{n-1} 으로 이루어진 마디 집합을 V 라고 하자. 또한 서버들 사이의 물리적 또는 논리적인 링크들을 나타내는 간선의 집합을 $E \subseteq V \times V$ 로 표기하자. 링크는 양방향으로 대칭이라고 가정하자. 그러면 본 연구의 문제는 무향네트워크 $G=(V, E)$ 상에서, 뒤에서 정의하게 될 전체 통신비용을 최소화하는 분배 네트워크 D 를 찾는 것이다. 이 때 분배 네트워크는 r 을 뿌리로 하는 나무 형태로 가정한다. 예를 들면 <그림 1>과 같다(G 의 간선은 실선으로 분배나무 D 의 간선은 굵은 선으로 표시하였다).



<그림 1> 서버네트워크 G 와 분배나무 D

분배나무의 작동 원리는 다음과 같다. 클라이언트들은 분배나무의 마디로 표시된 서버들에 콘텐츠를 요청하게 된다. 예를 들어, 클라이언트가 서버 $v=v_4$ 로 콘텐츠를 요청하였다고 하자. v_4 가 가지고 있는 해당 복사본의 유효시간이 초과하지 않았다면 유효하므로 제공된다. 반대로 유효시간이

초과 하였다면, 서버 v_4 는 <그림 1>의 분배나무 D 에서 자신의 부모마디(parent node), $p(v_4, D)$ 인 v_3 에 유효한 복사본을 요청하게 된다. 이 때, v_3 에서는 v_4 와 같은 원리로 반복하게 된다. 즉 v_4 는 v_3 의 클라이언트가 되는 것이다.



<그림 2> 각 마디의 클라이언트 도착 및 복제본 유효시간 변화의 예

이를 각 서버에서 복제본 요청 과정과 유효시간 소멸 과정을 예시한 <그림 2>를 통해 자세히 보자. v_4 에 첫 번째 클라이언트가 도착한 경우, v_4 는 자신이 유효 복사본을 소유하고 있지 않으므로, 부모마디 $p(v_4, D)$ 인 v_3 에게 요구하여 받아오게 된다. 이 때 받아 온 복사본의 유효시간은 v_3 에서 남은 유효시간 만큼 물려받게 된다. <그림 2>의 마디 v_3 의 첫 번째 클라이언트 요청을 또 다른 예로 살펴보자. 이 경우, 요청은 $p(v_3, D)$ 인 v_2 를 거쳐, 레벨-1 마디(D 에서 r 의 자식 마디)인 v_1 에서 해결 되었으며, 따라서 해당 복사본은 v_1 에서 남은 유효시간을 물려받은 것을 알 수 있다.

최대 유효시간이 T 이므로 레벨-1 마디가 원본

서버 r 로부터 시점 t 에 복제본을 받는 경우 유효성이 소멸되는 시점은 $t + T$ 가 된다. 레벨-1 마디인 v_1 의 자식 마디 v_2 를 비롯한 하위 레벨 마디에서 복사본의 유효시간은 T 를 넘을 수 없을 것이다.

다시 정리하면, 어떤 마디에 클라이언트의 요청이 도착하면, 그 마디가 가지고 있는 복사본의 유효시간이 경과 했는지를 확인한 후, 유효시간이 지났다면, 부모마디에게 해당 복사본 및 유효 시간을 물려받고, 그렇지 않은 경우는 해당 마디에서 요청이 처리된다. 전자의 경우는 해당 마디 v 와 부모노드 $p(v, D)$ 간의 통신비용이 발생하며, 후자의 경우는 그렇지 않다.

이러한 점들을 종합하여 보면, 총 통신비용은 첫째로 분배나무 D 를 어떻게 구성 하는가와 둘째로 각 마디에 도착하는 클라이언트 요청의 분포로 결정되는 것을 알 수 있다. 우리는 원본 서버 r 을 포함한 모든 서버 v 의 클라이언트의 요청이 $\lambda(v)$ 를 갖는 포아송 과정(Poisson process)을 따른다고 가정한다.

2.2 통신비용의 분석 요약

앞 절에서 제시한 시스템 모델을 바탕으로 Tang and Chanson은 주어진 분배나무 D 의 통신비용을 다음과 같이 분석하였다.

우선 D 의 각 링크는 v 와 D 에서 v 의 부모마디인 $p(v, D)$ 로 이루어진다. 링크 $(v, p(v, D))$ 에서 발생하는 비용은 서버 v 와 $p(v, D)$ 간에 복제 요청이 일회 발생할 때 소요 비용 $d(v, p(v, D))$ 에 두 마디 사이에 발생하는 단위시간 당 요청 회수를 곱한 것으로 정의한다(이 때, 비용 $d(v, p(v, D))$ 는 회선 지연시간 등, 경우에 맞게 다양하게 정의할 수 있다). 즉,

$$\text{링크 } (v, p(v, D)) \text{의 비용} = d(v, p(v, D)) \times v \text{와 } p(v, D) \text{간의 단위시간 요청 회수.} \quad (1)$$

모든 링크의 비용을 합하면 분배나무 D 의 통신비용이 된다. 이 비용을 계산하기 위해, 주어진 나무 D 의 뿌리인 r 에서 어떤 잎(leaf) 마디까지의 경

로를 생각하자. 예를 들어, <그림 1>에서 $r-v_1-v_2-v_3-v_4$. 경로의 각 마디 v 에서 발생하는 두 개의 연속한 소멸 시점은 한 주기를 이루는데, 이를 v -구간(v -interval)이라고 부르자. <그림 2>의 맨 아래에는 v_4 -구간이 예시되어 있다. 한 개의 v -구간에 한 번씩, v 로부터 부모마디 $p(v, D)$ 로의 복제요청이 발생하게 된다. 따라서 v -구간의 길이의 평균을 구하여 역수를 취하면, 단위 시간 당 링크 $(v, p(v, D))$ 에서 발생하는 복제요청의 평균 회수가 된다.

<그림 2>에서와 같이 한 개의 v -구간은 시작점 t_s 와 끝점 t_e 의 구간 (t_s, t_e) 로 표시할 수 있으며, (t_s, t_e) 는 다시 복제 요청이 발생하는 시점 t_r 을 사용하여 두 개의 구간 (t_s, t_r) 과 (t_r, t_e) 으로 나눌 수 있다.

첫째로 v 의 복제본 유효성이 소멸하면 분배나무 D 에서 v 의 모든 후손 마디의 유효성이 소멸하는 것을 알 수 있다. 둘째로 v 에서 부모마디로 복제요청을 하는 경우는 v 에 직접 도착하는 클라이언트 뿐 아니라, v 의 모든 후손(descendants)들의 집합 $D(v)$ (v 의 후손은 v 를 포함)에 도착하는 클라이언트 요청이 발생하는 경우이다. 즉 $(v, p(v, D))$ 에서 발생하는 복제 요청은 $D(v)$ 의 모든 클라이언트의 도착들이 만드는 포아송 과정의 총합이 된다. 따라서, 포아송 과정의 합은 다시 포아송 과정이 되므로, 구간 (t_s, t_r) 의 길이의 기대값은 $D(v)$ 에 도착하는 클라이언트의 요청율의 총합 $\Lambda(v, D) = \sum_{u \in D(v)} \lambda(u)$ 의 역수로 주어진다.

$$\begin{aligned} &v\text{-구간 중 } (t_s, t_r)\text{의 길이의} \\ &\text{기대값} = 1/\Lambda(v, D). \end{aligned} \quad (2)$$

v -구간 중 (t_r, t_e) 는 부모마디에 새로 요청하여 물려받은 복제본 유효시간이 된다. 따라서, 만약 $q(v, D)$ 를 어떤 마디 v 를 후손 마디로 갖는 레벨-1 마디라고 하면, $q(v, D)$ 는 r 로부터 유효시간을 물려 받으므로 $q(v, D)$ -구간의 (t_r, t_e) 의 길이는

최대 유효시간인 T 가 된다. 따라서 $q(v, D)$ -구간의 전체 길이의 기대값은, 식 (2)를 사용하면, $1/\Lambda(q(v, D), D) + T$ 가 된다. 각 $q(v, D)$ -구간 마디, $q(v, D)$ 에서 r 로 일회의 복제요청을 하게 되므로, 단위시간당 요청회수의 평균은 $q(v, D)$ -구간 길이의 기대값의 역수인 다음으로 주어진다.

$$\frac{1}{1/\Lambda(q(v, D), D) + T} \quad (3)$$

이 중에서 후손 마디인 v 와 $p(v, D)$ 간에 복제요청 때문에 발생한 부분의 비율을 계산하면 다음과 같다(본 논문에서는 증명 과정은 생략한다. 자세한 사항은 [10]을 참조).

$$\begin{aligned} \rho(v, D) = \\ 1 - \frac{\Lambda(q(v, D), D) - \Lambda(v, D)}{\Lambda(q(v, D), D)} e^{-T\Lambda(v, D)} \end{aligned} \quad (4)$$

식 (1), 식 (3), 그리고 식 (4)를 사용하면, 링크 $(v, p(v, D))$ 에서 발생하는 통신비용은,

$$\begin{aligned} c(v, p(v, D)) \\ = d(v, p(v, D)) \times \frac{1}{1/\Lambda(q(v, D), D) + T} \\ \times \left(1 - \frac{\Lambda(q(v, D), D) - \Lambda(v, D)}{\Lambda(q(v, D), D)} e^{-T\Lambda(v, D)}\right). \end{aligned} \quad (5)$$

따라서 모든 링크의 비용을 합하여 분배나무 D 의 총 통신비용을 계산하면 다음과 같다.

$$c(D) = \sum_{v \in V-(r)} c(v, p(v, D)). \quad (6)$$

3. 알고리즘

앞에서 언급한바와 같이, 본 연구의 문제는 모든 가능한 분배나무 중에서 식 (6)의 비용을 최소화하는 분배나무 D 를 찾는 것이다. 그러나, 이 문제는 매우 어려운 문제이다. 다항시간에 최적해를 구하는 것이 불가능하다고 믿는 NP-hard임을 보일 수 있다 [10]. 특히 각 간선의 비용 식 (5)가 비선형이며, 나무 D 에 따라 간선 하나의 비용이 따라 변하

기 때문에 보통의 조합최적화문제와 다른 매우 어려운 최적화문제가 된다.

본 연구의 문제는 실시간에 해를 생성해야할 필요까지는 없지만, 클라이언트의 도착율과 서버들의 개수나 위치 같은 네트워크의 환경이 고정된 것이 아니므로, 수시로 D 를 변화시켜 통신비용을 최소화하는 것이 필요하다. 따라서 분지한계법과 같이 해의 질을 추구할 수 있지만, 알고리즘의 수행시간의 상한이 보장되지 않은 해법 보다는, 해의 품질을 실용적 수준으로 만족하면서 대신 수행시간을 문제입력 크기의 다항시간으로 제한하는 발견적 기법을 택하는 것이 현실적인 대안이라고 하겠다.

3.1 Tang and Chanson의 알고리즘

Tang and Chanson은 다음과 같이 r 을 중심으로 분배나무를 키워나가는 ‘greedy’ 유형의 알고리즘을 제안하였다. 현재까지 생성된 분배나무를 $D = (V(D), E(D))$ 라고 하자. 그리고 식 (6)에서 정의한 $c(D)$ 를 알고리즘의 설명을 위해 마디와 간선까지 표기하여 $cost(V(D), E(D))$ 로 쓰자 ($V(D)$ 와 $E(D)$ 는 D 의 마디와 간선의 집합).

D 의 초기해로는, 마디는 원본 서버 r 그리고 간선은 없는 나무를 택한다. 즉, $(V(D), E(D)) = (\{r\}, \emptyset)$.

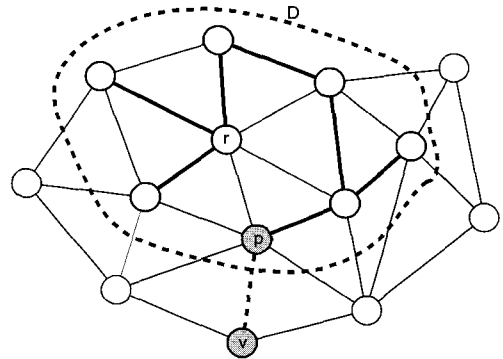
각 반복단계마다, 현재 $V(D)$ 에 포함되어 있지 않은 마디 v 그리고 v 와 이웃한 $V(D)$ 의 마디 p 중에서 다음 값을 최소화하는 것을 선택하여 D 에 추가한다.

$$\frac{cost(V(D) \cup \{v\}, E(D) \cup \{(v, p)\}) - cost(V(D), E(D))}{\lambda(v)}$$

즉, 특정 마디와 이를 연결하는 간선이 추가될 경우 통신비용의 증가 값을, 해당 마디의 클라이언트 요청 율로 나누어, 그 값이 가장 적은 마디와 간선을 선택하는 것이다. 이러한 과정을 모든 마디를 포함할 때까지 반복한다. <그림 3>에는 한 반복단계가 도시되어 있다.

이 알고리즘은 최대 $|V|$ 의 반복단계를 거친다. 각 반복단계에서, $V(D)$ 에 속하지 않은 마디를 끝

마디로 가지는 간선에 대해 비용을 계산하게 되는데 최대 $|E|$ 개가 된다. 각 비용을 계산하는데 최대 $|E|$ 개의 간선의 비용을 합해야 한다. 따라서 알고리즘의 계산시간은 $O(|V|^2|E|)$ 가 되어 다항시간 알고리즘이 된다.



<그림 3> Tang and Chanson의 알고리즘

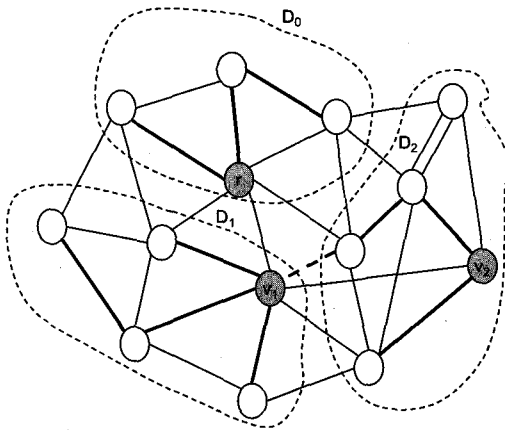
Tang and Chanson의 알고리즘은 나무를 키워가는 알고리즘이다. 현재까지 구성된 나무는 원본 서버를 뿌리로 하여 그 상대적인 위치가 최종해까지 고정되면서도 변한다. 이는 알고리즘의 비교적 초기에 생성되는 해가 국지적 최적 해를 결정될 수 있다는 것을 의미한다.

3.2 제안 알고리즘

본 절에서는 나무 대신 숲 F 를 키워가는 알고리즘을 제안하려고 한다. 현재 F 에 속한 나무들의 집합을 D_0, D_1, \dots, D_l 이라고 표기하자. 각 나무의 뿌리를 v_0, v_1, \dots, v_l 이라고 하자. 우선 초기에는 모든 마디들을 하나의 분배나무로 생각한다. 즉 $l=|V|$ 가 되며 각 마디는 각 나무의 뿌리가 된다.

이 초기해를 시작으로 각 반복 단계에서는 다음의 기준에 의해 선택한, 한 개의 간선을 숲 F 에 추가한다. 현재의 어떤 나무와도 회로(circuit)를 만들지 않는 간선 하나를 추가하면 F 의 나무 두 그루 D 와 D' 을 연결하여 한 개의 나무로 키우게 된다. 이 때, 원래 두 그루의 각 뿌리 v 와 v' 중 하나

가 새로 커진 나무의 뿌리가 되는데, 만약 v 와 v' 중에 r 이 있다면 이 것이 새나무의 뿌리가 된다. 그렇지 않다면, 각각이 새나무의 뿌리가 되었을 때 비용이 적게 드는 쪽을 택한다. 이런 단계를 모든 간선에 적용하여, 가장 비용이 적은 간선을 택한다. 이로서 한 반복단계가 끝나게 되며, F 가 나무가 될 때 까지 반복한다.



<그림 4> 제안 알고리즘의 한 반복단계

<그림 4>에는 제안하는 알고리즘의 한 반복단계가 도시되어 있다. 현재까지의 숲 F 는 세 그루의 나무 D_0, D_1, D_2 로 이루어져 있으며, 새로운 반복 단계에서는 D_1 과 D_2 가 선택된 간선(파선으로 그려진 간선)에 의하여 한 그루의 나무가 되며, D_1 의 뿌리였던 v_1 이나 D_2 의 뿌리였던 v_2 중에서 비용을 적게 발생시키는 것이 합친 새나무의 뿌리가 된다.

제안된 알고리즘의 특징은 분배나무를 숲으로 분산하여 균형 있게 키워나가는 것이다. 따라서 초기부터 특정 국지해로 수렴하는 것을 방지하는데 요점이 있다. 알고리즘의 수행시간은 역시 $O(|V|^2 |E|)$ 가 되는 것을 쉽게 알 수 있다. 그러나, 각 비용을 계산하는 시간이 두 배로 늘어나게 된다.

마지막으로 기존 알고리즘이나 새로 제안한 알고리즘 모두에 사용할 수 있는 나무 개선 알고리즘을 제안한다.

1. 알고리즘이 생성한 분배나무 D 에 속하지 않은 간선 하나를 D 에 추가하면 회로가 형성된다.
2. 이 회로 중에 기존 간선을 삭제하면 새로운 분배나무가 되는데 비용이 가장 많이 감소하는 것을 선택한다.
3. 이러한 방법으로 추가 했을 때, 나무비용이 가장 많이 감소하는 간선을 고른다.
4. 비용을 감소시키는 간선이 더 이상 없을 때, 종료한다.

본 논문에서 제안하는 알고리즘은 엄격히 말하면 중심형(centralized) 알고리즘이다. 즉, 어떤 노드가 관련 서버 네트워크(G)의 토폴로지 정보를 모두 알고 있다는 가정이 만족되어야 한다. 이것은 인터넷에서는 비현실적일 수 있다. 그러나, 알고리즘이 생성하는 숲의 각 요소인 나무들이 자신의 토폴로지에 대한 정보를 적절하게 주고받으면 본 알고리즘은 분산형(decentralized) 알고리즘으로 전환할 수 있을 것이다. 즉 두 개의 나무가 합쳐 한 개의 나무가 될 때, 새로운 나무의 토폴로지로 수정하는 것은 어렵지 않기 때문이다. 자세한 내용은 본 논문의 주제를 넘으므로 생략하기로 한다.

4. 성능분석

네트워크 G 는 마디의 수를 $|V|= 20, 30, 40, 50$ 으로 하였고, 각 경우 10개씩의 예를 풀어보았다. 간선은 최소 $2|V|-1$ 개의 성긴(sparse) 네트워크와 최대 $|V|(|V|-1)/2$ 개의 밀집된(dense) 네트워크를 인접행렬을 사용, 무작위로 생성하였다[4, 11]. 각 간선의 비용은 구간 $[1, |V|]$ 에서 무작위 생성하였고, 각 마디 클라이언트 요청 도착을 λ 는 구간 $[0.1, 1.5]$ 에서, 최대 유효시간은 구간 $[0.8, 1.2]$

<표 1> 실험환경(P4 : 1.13GHz, RAM : 256, C언어)에서의 기존 알고리즘과 제안 알고리즘의 수행시간

N	평균간선의 수	기존 알고리즘의 평균 수행시간(sec)	제안 알고리즘의 평균 수행시간(sec)
20	78	21.72	42.57
30	215.5	43.41	108.96
40	323	59.33	176.80
50	718.6	203.72	660.05

에서 무작위로 생성하였다(이렇게 범위를 택한 이유는, 유효시간이 클라이언트 요청율에 비해 너무 크거나 작으면 의미가 없는 문제되므로 이를 피하기 위해 유효시간이 노드 10개 정도의 요청율로 결정된 도착 사이시간의 7~8배 정도가 되도록 하기

위함이다). 이렇게 무작위로 생성한 40개의 예에 대해, 네트워크 안에서 기존 알고리즘(I)과 제안한 알고리즘(II), 기존 알고리즘에 나무개선 알고리즘을 병행(III), 제안한 알고리즘에 나무개선 알고리즘 병행(IV) 등 네 가지 방법을 적용하여 보았다.

<표 2> 기존알고리즘과 제안 알고리즘의 성능비교

N	E	기존 알고리즘 I	제안 알고리즘 II	기존+ 나무개선III	제안 + 나무개선IV	개선율 I : II	개선율 I : III	개선율 I : IV
20	42	30.19	24.60	28.38	22.10	18%	6%	27%
20	67	22.85	20.65	20.60	19.90	10%	10%	13%
20	100	44.01	39.33	39.88	37.96	11%	9%	14%
20	83	34.07	30.70	31.01	28.41	10%	9%	17%
20	111	54.11	39.99	41.86	38.13	26%	23%	30%
20	112	57.10	42.85	45.24	36.04	25%	21%	37%
20	41	29.96	18.97	26.03	17.16	37%	13%	43%
20	116	22.93	17.45	18.21	16.01	24%	21%	30%
20	45	27.97	25.49	26.28	24.34	9%	6%	13%
20	63	42.04	40.12	36.55	37.35	5%	13%	11%
30	313	51.68	45.36	43.12	39.54	12%	17%	23%
30	88	58.01	53.79	55.16	47.28	7%	5%	18%
30	155	40.37	38.87	37.48	37.15	4%	7%	8%
30	91	71.90	60.84	66.10	51.63	15%	8%	28%
30	250	82.57	71.30	72.58	64.54	14%	12%	22%
30	429	61.51	39.84	49.32	33.28	35%	20%	46%
30	201	48.58	41.40	41.22	32.34	15%	15%	33%
30	67	51.41	43.25	49.46	40.30	16%	4%	22%
30	142	49.55	43.97	41.74	39.22	11%	16%	21%
30	419	100.44	91.49	88.82	70.31	9%	12%	30%
40	311	44.06	40.42	39.83	33.49	8%	10%	24%
40	461	142.46	93.38	112.10	88.81	34%	21%	38%
40	169	105.76	82.48	90.70	78.61	22%	14%	26%
40	386	168.30	121.74	138.81	101.90	28%	18%	39%
40	525	131.20	79.95	121.45	70.17	39%	7%	47%
40	264	99.36	90.32	86.00	81.75	9%	13%	18%
40	352	90.12	77.22	80.79	70.14	14%	10%	22%
40	381	130.91	120.41	110.16	94.04	8%	16%	28%
40	237	84.29	74.47	69.67	66.24	12%	17%	21%
40	144	108.78	99.63	100.79	87.84	8%	7%	19%
50	139	86.82	70.53	80.50	64.72	19%	7%	25%
50	1061	107.15	77.90	88.53	60.94	27%	17%	43%
50	1030	96.58	79.91	88.73	63.02	17%	8%	35%
50	472	99.35	80.10	86.65	73.39	19%	13%	26%
50	1154	125.89	99.51	100.27	79.87	21%	20%	37%
50	218	62.69	55.59	53.41	51.15	11%	15%	18%
50	226	85.08	70.80	82.08	60.51	17%	4%	29%
50	802	68.19	57.31	63.03	51.08	16%	8%	25%
50	976	97.09	67.18	80.55	66.48	31%	17%	32%
50	1108	73.60	50.93	59.51	40.99	31%	19%	44%
평균	334	74.72	60.50	64.82	52.95	17.6%	12.7%	27.0%

계산시간은 표에서 분석한 것과 같이, 기존 알고리즘의 비해 제안된 알고리즘이 2~3배의 시간이 걸렸다. 이는 해의 개선 측면에서 불가피한 결과라 하겠다.

노드 50개의 실험에서 간선의 수가 크게 증가함에 따라 알고리즘 수행시간이 크게 증가하였다.

전산실험에 나타난 해의 품질은 <표 2>와 같다. 우선 기존 알고리즘보다 제안한 알고리즘의 해는 항상 우수하였다. 최소 4%에서 최대 35%, 평균 17%의 개선을 보였다. 기존의 알고리즘에 나무개선 알고리즘만을 적용하여도 평균 12.7%의 개선효과를 보였다. 그러나 대부분의 경우, 제안 알고리즘의 해의 품질보다는 못하였다. 마지막으로, 당연히 예상되는 결과이지만, 제안한 알고리즘에 나무개선 알고리즘을 병행하는 경우 가장 큰 개선효과를 보였다. 평균 27%의 개선 효과를 보여, 지속적으로 사용하는 웹 콘텐츠 분배 나무의 속성을 생각하면 의미 있는 개선 효과라고 하겠다.

5. 향후 연구 과제

원본 서버가 미리 정해지지 않은 경우 통신비용을 최소화 하는 분배 네트워크 구성문제에 대한 연구와 두 서버 간의 링크가 대칭이 아닌 경우에 대한 연구가 유용하다고 생각한다.

참 고 문 헌

- [1] Arlitt, M.F. and C.L. Williamson, "Internet Web Servers Workload Characterization and Performance Implications," *IEEE/ACM Trans. Networking*, Vol.5, No.5(1997), pp.631-645.
- [2] Calvert, K., M.B. Doar, and E.W. Zegura, "Modeling Internet Topology," *IEEE Comm. Magazine*, Vol.35, No.6(1997), pp.160-163.
- [3] Cardellini, V., E. Casalicchio, M. Colajanni, and P.S. Yu, "The State of the Art in Locally Distributed Web-Server Systems," *ACM Computing Surveys*, Vol.34, No.2(2002).
- [4] Cohen, E. and H. Kaplan, "Aging through Cascaded Caches : Performance Issues in the Distribution of Web Content," *ACM SIG-COMN*, (2001), pp.41-53.
- [5] Dilley, J., "The Effect of Consistency on Cache Response Time," *IEEE Network*, Vol. 14, No.3(2000), pp.24-28.
- [6] Duvvuri, V., P. Shenoy, and R. Tewari, "Adaptive Leases : A Strong Consistency Mechanism for the World Wide Web," *IEEE Trans. Knowledge and Data Eng*, Vol.15, No.5(2003), pp.1266-1276.
- [7] Hou, Y.T., J. Pan, B. Li, X. Tang, and S. Panwar, "Modeling and Analysis of an Expiration-Based Hierarchical Caching System," *Proc. IEEE GLOBECOM*(2002), pp.2468-2472.
- [8] Jamin, S., C. Jin, Y. Jin, D. Raz, Y. Shavitt, and L. Zhang, "On the Placement of Internet Instrumentation," *Proc. IEEE INFOCOM* (2000), pp.295-304.
- [9] Shaikh, A., R. Tewari, and M. Agrawal, "On the Effectiveness of DNS-Based Server Selection," *IEEE INFOCOM*(2001), pp.1801-1810.
- [10] Tang, X. and S.T. Chanson, "The Minimal Cost Distribution Tree Problem for Recursive Expiration-Based Consistency Management," *IEEE Trans. Parallel and Distributed System*, Vol.15, No.3(2004), pp.214-227.
- [11] Uргаonkar, B., A.G. Ninan, M.S. Raunak, P. Shenoy, and K. Ramamritham, "Maintaining Mutual Consistency for Cached Web Objects," *IEEE Int'l Conf. Distributed Computing Systems*(2001), pp.371-380.
- [12] Waxman, B.M., "Routing of Multipoint Connections," *IEEE J.Selected Areas in Comm*, Vol.6, No.9(1988), pp.1617-1622.

- [13] Yin, L., L. Alvisi, M. Dahlin, and C. Lin, "Hierarchical Cache Consistency in a WAN," *Proc. Second USENIX Symp. Internet Technologies and Systems*(1999), pp.13-24.
- [14] Yin, L., L. Alvisi, M. Dahlin, and C. Lin, "Volume Leases for Consistency in Large-Scale Systems," *IEEE Trans. Knowledge and Data Eng*, Vol.11, No.4(1999), pp.563-576.