저속 네트워크 환경에서 데이터 변화 탐지를 위해 타임스탬프 트리를 이용하는 B2B 시스템 구축

손 세 일[†] · 김 흥 준[††]

요 약

본 논문에서는 저속 네트워크 사용자를 지원하기 위해 기존의 웹 기반 B2B 시스템을 확장하였다. 클라이언트와 서버 사이에 공유된 데이터의 일관성을 보장하기 위해 타임스탬프 트리를 이용한 데이터 변화 탐지 방법을 제안하고, 시뮬레이션을 통해 제안된 방법의 성능을 분석하였다. 타임스탬프 트리의 단말 노드들이 일양 분포로 변경되는 최악의 조건에서, 시뮬레이션 결과는 데이터 갱신율이 15% 이하일 때 제안된 방법이 순차 탐지보다 효율적임을 보였다. A사의 웹 기반 건설 MRO B2B 시스템을 2004년 4월부터 2004년 8월까지 관찰한 결과에 따르면, 월 평균 데이터 갱신율은 7% 이하였다. 따라서 제안된 방법은 실질적으로 데이터 변화 탐지 성능을 향상시켰다. 또한 제안된 방법은 서버가 클라이언트들이 복제한 데이터를 저장할 필요가 없기 때문에 서버의 저장 공간 사용이 줄었다.

키워드 : 전자상거래, 비투비, 저속 네트워크, 데이터 변화 탐지, 타임스탬프

Building B2B system using timestamp tree for data change detection in low speed network environment

Sei-Il Son[†] · Heung-Jun Kim[††]

ABSTRACT

In this paper we expanded a existing web based B2B system to support users in low speed network. To guarantee shared data consistency between clients and a server, we proposed a method of data change detection by using a timestamp tree and the performance analysis of the proposed method was proved by a simulation. Under the worst condition that leaf nodes of a timestamp tree were changed uniform distribution, the simulation result showed that the proposed method was more efficient than a sequential detection until the percentage of changed nodes were below 15%. According to our observation, the monthly average of data change was below 7% on a web-based construction MRO B2B system of a company A from April 2004 to August 2004. Therefore the proposed method improved performance of data change detection in practice. The proposed method also reduced storage consumption in a server because it didn't require a server to store replicated data for every client.

Key Words : E-commerce, B2B, Low Speed Network, Data Change Detection, Timestamp

1. 서 론

초고속 네트워크의 보급은 대량의 데이터 교환을 가능하게 하여 정보 시스템이 여러 분야에 응용되는 것을 촉진하며, 기업 간의 거래에서도 B2B 시스템의 이용이 증가하고 있다. 웹 기반 B2B 시스템들은 충분한 네트워크 대역폭을 기반으로 상품의 이미지를 포함한 많은 정보를 사용자에게 제공하여 구매 결정을 돕는다. B2B 시스템을 이용하는 사용자들이 늘어나면서 이들의 네트워크 환경 또한 다양해져서 이에 대한 지원의 필요성이 증대되고 있다. 국내의 경우 도시를 중심으로 초고속 네트워크가 보급되고 있지만, 거주 인구가 적은 지역의 사용자들은 여전히 전화선과 모뎀을 이용한 저속 네트워크를 사용하고 있다[2]. 도로, 댐, 항만 공사 현장의 경우도 해당 지역의 네트워크 기반 시설에 따라 초고속 네트워크를 이용할 수 없는 경우가 있다. 대부분의 웹 기반 B2B 시스템들은 사용자가 초고속 네트워크를 이용하는 것을 가정하여 구축되었기 때문에 많은 이미지와 사운드 등이 웹 페이지에 내포되어 있어 전송되는 데이터의 양이 크다. 이 같은 웹 기반 B2B 시스템을 저속 네트워크 내의 사용자가 이용하려면 많은 인내심이 필요하다.

본 논문에서는 기존의 웹 기반 B2B 시스템을 재구축하지 않고 저속 네트워크를 이용하는 사용자를 지원하기 위해 중계 서버를 도입하였다. 또한 별도의 클라이언트 프로그램을

† 정 회 원 : 단국대학교 정보컴퓨터학부 강의전임강사
†† 종신회원 : 진주산업대학교 컴퓨터공학부 부교수
논문접수 : 2005년 5월 20일, 심사완료 : 2005년 10월 6일

작성하고 데이터베이스 서버의 일부를 클라이언트의 지역 데이터베이스에 저장하였다. 클라이언트와 서버 사이에 공유된 데이터의 일관성을 보장하기 위해 데이터 동기화가 필요하며, 갱신된 데이터를 탐지하는 것은 동기화 과정에서 매우 중요하다. 이것은 저장 장치의 입출력을 위주로 하기 때문에 서버에 많은 부하를 준다. 본 논문에서는 타임스탬프 트리를 이용한 데이터 변화 탐지 방법을 제안하고, 구현하였다. 그 결과, 기존의 순차 접근을 이용한 데이터 변화 탐지 방법들과 비교하여 저장 장치 입출력 횟수가 줄어 성능이 향상되었다. 또한 스냅샷을 이용한 동기화 방법과는 달리 추가적인 저장 공간이 필요하지 않기 때문에 저장 공간의 소모가 줄었다. 본 논문에서는 동기화 과정에서 발생하는 네트워크 트래픽 부하는 고려하지 않았지만, 기존 동기화 방법과 비교하여 추가로 요구되는 트래픽은 없다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구를 살펴보고, 3장에서는 네트워크 연결 상태에 따른 클라이언트의 상태 변화와 중계 서버를 포함한 확장된 B2B 시스템의 구조를 알아본다. 4장에서는 데이터 동기화 과정과 타임스탬프 트리를 이용한 데이터 변화 탐지에 대해 살펴본다. 5장에서는 구축된 시스템에 대해 알아보고, 시뮬레이션을 통해 제안된 타임스탬프 트리를 이용한 데이터 변화 탐지 방법의 성능을 평가한다. 6장에서는 결론과 향후 연구 과제에 대해 기술한다.

2. 관련 연구

복제된 데이터의 동기화를 위해 타임스탬프를 이용한 기존 연구들을 살펴보면[1, 9], 데이터 갱신 전파, 트랜잭션 처리, 일관성 등의 문제를 다루고 있다.

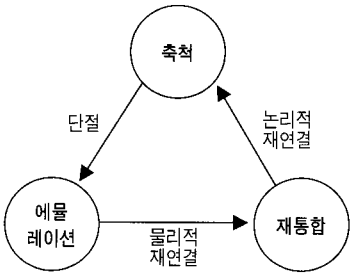
[1]은 부분 중복 환경에서 모든 사이트에 갱신 결과를 지연 갱신 전파하는 과정에서 발생할 수 있는 충돌에 따른 비직렬성 문제를 타임스탬프와 보상 트랜잭션을 이용해 해결하였다. 이 논문은 갱신을 전파하는 과정에서 발생하는 네트워크 트래픽을 줄이기 위해 주사본과 복사본이 존재하는 사이트들을 계층 구조로 연결하였다. [9]는 약한 일관성을 갖는 복제들 사이의 갱신 전파 성능을 향상하기 위해 계층적 행렬 타임스탬프(hierarchical matrix timestamp)를 이용한 알고리즘은 제안하였다. 복제를 갖는 사이트들을 도메인이라 불리는 집합으로 나누고 도메인들을 계층화함으로써 N개 사이들이 관리하는 행렬 타임스탬프의 크기를 $O(N^2)$에서 $O(N)$으로 줄였다.현재 상용화된 데이터 동기화 시스템들로 Sybase의 SQL Remote[10, 11], Oracle의 iConnect [5], IBM의 Sync Server[6] 등이 있다.

타임스탬프를 이용하는 동기화 방법은 최종 동기화 이후 갱신된 데이터만을 처리한다. 이것은 데이터 변경이 발생할 때마다 타임스탬프 값도 변경해야 하므로 데이터 변경 작업의 처리 시간이 증가한다. 또한 동기화 될 모든 데이터들의 타임스탬프 값을 확인해야하는 부담이 있다. 스냅샷(sanpshot)을 이용한 방법은 서버가 클라이언트가 복제한 데이터의 이미지를 저장한 후, 동기화 시에 클라이언트와 서버가 소유한 이미지를 비교하여 갱신된 데이터를 탐지하고, 이들을 교환한다. 스냅샷을 이용한 방법은 서버가 각 클라이언트들이 복제한 데이터의 이미지를 저장해야 하므로 서버의 저장 공간 소모가 매우 크다.

3. 기본 모델

3.1 클라이언트 상태 모델

클라이언트와 서버의 접속이 지속적으로 유지되지 않는 네트워크 환경에서 클라이언트는 서버의 연결 상태에 따라 처리할 작업을 구분하기 위해 연결 상태 변화에 대한 정의가 필요하다. 본 논문에서 클라이언트는 네트워크의 물리적 연결 상태에 따라 축척, 에뮬레이션, 재통합 중 하나의 상태를 갖는다[7, 8]. (그림 1)은 네트워크 연결 상태에 따른 클라이언트의 상태 전이를 보여준다.



(그림 1) 클라이언트 상태 전이

축척(hoarding) 상태는 클라이언트와 서버가 연결된 상태이다. 이 상태에서 클라이언트는 단절을 대비해서 필요한 데이터를 서버로부터 복제한다. 복제될 데이터는 서버의 사용자 프로파일에 기술된다.

에뮬레이션(emulation) 상태는 클라이언트와 서버가 단절된 상태이다. 이 상태에서 클라이언트는 지역 데이터베이스에 복제된 데이터를 이용하여 사용자 요청을 처리한다. 사용자가 데이터 갱신을 요구하면 데이터와 타임스탬프가 함께 기록되며, 이후 동기화 과정에서 이용되게 된다.

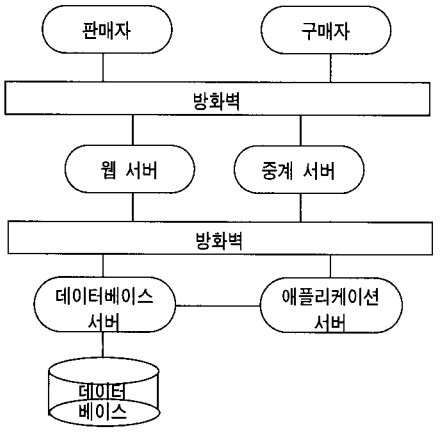
재통합(reintegration) 상태는 클라이언트와 서버가 재연결 시 단절 기간동안 갱신된 데이터를 상호교환 하여 데이터 일관성을 보장하는 상태이다. 클라이언트에서 갱신된 데이터는 타임스탬프를 참조하여 하나의 파일로 생성되고, 중계 서버에 전달된다. 중계 서버는 클라이언트로부터 수신된 파일의 데이터를 서버의 데이터베이스에 반영한다. 또한 서버는 타임스탬프를 이용하여 클라이언트가 마지막 동기화를 수행한 이후 갱신된 데이터를 추출하여 클라이언트에 전달함으로써 클라이언트와 서버 사이의 데이터 일관성을 보장한다.

3.2 제안된 B2B 시스템 구조

본 논문에서는 e-마켓플레이스 모델의 웹 기반 B2B 시스템을 확장하였다. e-마켓플레이스 모델은 거래 당사자 혹은

제3자가 구매자와 판매자를 가상공간에서 만나 거래할 수 있도록 돕는다[3, 4]. 운영 주체는 판매자, 구매자, B2B 서비스 제공업체로써 일부 사용자들의 필요에 의해 시스템을 재구축하는 것이 참여자들 간의 이해관계 때문에 어려울 뿐만 아니라, 설사 재구축이 진행되어도 시간과 비용 측면에서 많은 부담이 있다. B2B에는 여러 기업들이 참여하기 때문에 지정된 인터페이스를 통하지 않고 참여 기업들의 정보 시스템을 직접 접근할 수 없다.

본 논문에서는 기존 웹 기반 B2B 시스템을 재구축하지 않고 저속의 네트워크를 이용하는 사용자를 지원하기 위해 (그림 2)와 같이 별도의 중계 서버를 도입하였다.



(그림 2) 확장된 B2B 시스템 구성도

웹 서버는 판매자와 구매자에게 필요한 정보를 제공하고, 사용되는 데이터는 데이터베이스 서버에 저장된다. 애플리케이션 서버는 비즈니스 프로세스에서 각종 함수 및 프로시저 모듈을 가지고 있다. 중계 서버는 웹을 이용하지 않고 업무를 처리하기 위해 별도의 클라이언트 프로그램을 사용할 때, 클라이언트와 데이터베이스 서버 사이의 데이터 교환이 이루어진다. 웹 서버, 중계 서버를 보호하기 위해 외부 방화벽이 설치되어 있으며, 데이터베이스 서버와 애플리케이션 서버를 사용자가 직접 접근할 수 없도록 내부 방화벽을 두었다.

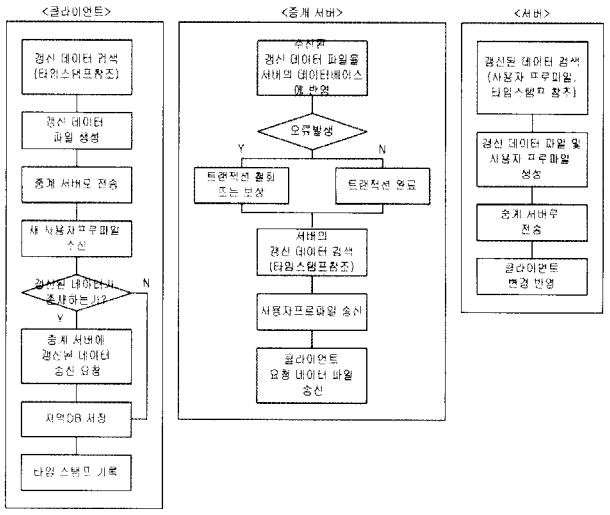
본 논문에서 중계 서버를 도입한 이유는 기존의 웹 기반 B2B 시스템의 구조 변경에 따른 비용 발생을 줄이고, 보안을 이유로 클라이언트가 직접 서버의 데이터베이스를 접근하지 못하기 때문이다. 기존의 동기화 방법들은 클라이언트와 서버의 데이터베이스들을 직접 접근하지만, 본 논문에서는 파일 시스템을 기반으로 한 중계 서버가 도입되었기 때문에 데이터 동기화 과정의 차이가 있다.

4. 데이터 동기화

4.1 데이터 동기화

데이터 동기화란 클라이언트와 서버 사이의 공유된 데이터에 대해 일관성을 유지하기 위한 처리를 말한다. 데이터 동기화는 클라이언트와 서버 양방향으로 진행된다. 클라이언트에서 갱신된 데이터는 서버로 전달되고, 서버에서 갱신된 데이터는 클라이언트로 전달된다. 확장된 B2B 시스템에서 저속의 네트워크 환경에 위치한 클라이언트는 중계 서버를 통해서만 데이터베이스 서버에 접근할 수 있다.

(그림 3)은 데이터 동기화 과정을 보여주고 있다. 데이터 동기화는 일반적으로 클라이언트 프로그램의 시작 직후와 종료 직전에 이루어진다. 하지만 서버 장애나 네트워크 장애 등을 이유로 동기화 시점은 연기될 수 있다. 클라이언트는 타임스탬프를 이용하여 마지막 동기화가 이루어진 이후에 갱신된 데이터를 탐지한다. 갱신된 데이터는 중계 서버로 전송되기 위해 파일로 저장되며, 파일명은 중계 서버 내에서 다른 클라이언트들로부터 전송된 파일들과 충돌하지 않도록 유일해야 한다. 중계 서버는 클라이언트의 동기화 요청을 수신한 후 갱신된 데이터 파일을 전송 받아 서버의 데이터베이스에 반영한다. 이 과정에서 충돌이 발생될 수 있지만[5, 6, 7, 8, 11], B2B 시스템에서 클라이언트의 지역 데이터베이스에 복제된 데이터는 대부분 읽기 전용의 속성을 가지며, 신규 데이터의 생성과 신규 데이터의 변경만 실행되기 때문에 충돌은 일어나지 않는다. 서버는 사용자 프로파일과 서버 데이터베이스의 타임스탬프를 참조하여 단절 기간동안 갱신된 데이터를 탐지하고, 이를 파일로 생성하여 중계 서버에 전달한다. 중계 서버는 이 파일을 클라이언트에 전달함으로써 데이터 동기화가 완료된다.



(그림 3) 데이터 동기화 흐름도

4.2 타임스탬프 트리를 이용한 데이터 변화 탐지

클라이언트와 서버 사이에 공유된 데이터의 일관성을 보장하기 위해서는 단절 기간동안 갱신된 데이터의 탐지가 반드시 필요하다. 기존의 데이터 동기화를 지원하는 시스템들은 단절 기간동안 클라이언트와 서버의 갱신된 데이터의 정보를 로그에 기록하거나 서버로부터 클라이언트에 복제된 데이터들의 스냅샷을 저장한 후, 재연결시 저장된 스냅샷과 현재 스냅샷을 비교하여 갱신된 데이터를 발견하였다.

본 논문에서는 m-원 트리 구조를 갖는 타임스탬프 트리를

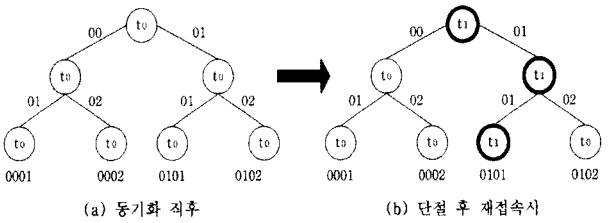
이용한 데이터 변화를 탐지하는 방법을 제안하고 구현하였다. 이 타임스탬프 트리의 각 레벨은 코드의 의미적 분류를 따른다.

제안된 방법에서 서버로부터 클라이언트로 데이터가 복제될 때, 대응되는 타임스탬프 테이블이 서버 내에 생성된다. 이 테이블은 클라이언트에 복제된 데이터에 대해 $<k_i, t_i>$와 같이 데이터 D_i의 키 k_i와 최종 갱신 시간 t_i가 기록된다. 클라이언트와 서버가 단절된 후 데이터 D_i가 갱신되면, 타임스탬프 트리의 키 k_i와 일치하는 레코드를 찾아 최종 갱신 시간 t_i를 변경한다. 이것은 단절 기간 동안 서버와 클라이언트 모두에서 이루어진다. 그 결과 단절 기간동안 클라이언트에 복제된 데이터의 갱신이 발생하면, 클라이언트에 복제된 데이터 D'_i의 최종 갱신 시간 t'_i와 서버에 존재하는 원본 데이터 D_i의 최종 갱신 시간 t_i의 값이 상이하게 된다.

(그림 4)의 (a)는 데이터가 동기화된 직후 타임스탬프 테이블에 기록된 값을 트리 구조로 보여주고 있다. 각 노드는 최종 갱신 시간을 나타내며, 간선은 키의 부분 값을 나타낸다. (그림 4)의 (b)는 단절 기간동안 발생한 데이터 변경에 의해 최종 갱신 시간이 바뀌었음을 보여준다. 만약 단절 기간 동안 아무런 갱신이 없었다면, 루트 노드에 저장된 타임스탬프 값은 동일할 것이다. <표 1>은 (그림 4)에 대응되는 내용을 타임스탬프 테이블로 표현한 것이다.

데이터 변화 탐지는 타임스탬프 트리를 루트 노드로부터 깊이-우선 방식으로 순회하면서, 노드의 타임스탬프 값이 갱신된 경우만 하위 서브 트리를 탐색하여 갱신된 데이터를 발견한다.

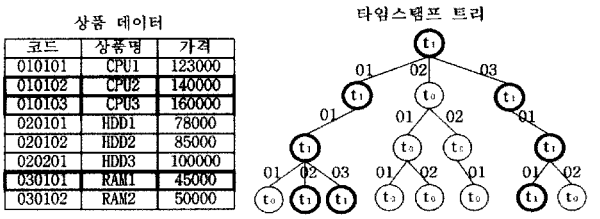
(그림 5)는 상품 코드 "010102", "010103", "030101"을 갖는 세 가지 상품의 가격이 수정되면서 대응되는 타임스탬프 트리의 노드 값이 t_0에서 t_1으로 변경된 예를 보여준다.



(그림 4) 타임스탬프 트리를 이용한 데이터 변화 탐지

〈표 1〉 (그림 4)를 테이블에 $\langle k_i, t_i \rangle$로 표현

(그림 4)의 (a)		(그림 4)의 (b)	
키	최종 갱신 시간	키	최종 갱신 시간
0000	t_0	0000	t_0
0001	t_0	0001	t_0
0002	t_0	0002	t_0
0100	t_0	0100	t_1
0101	t_0	0101	t_1
0102	t_0	0102	t_0



(그림 5) 상품 데이터가 수정된 경우의 예

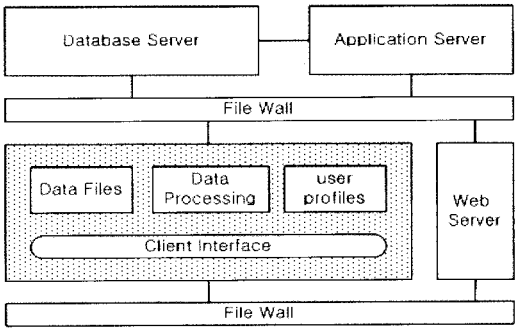
제안된 방법은 갱신 데이터를 추출하고 동기화를 수행하기 위해 기존 데이터베이스의 구조를 변경할 필요가 없다. 또한 스냅샷을 이용한 데이터 동기화와 같이 각 클라이언트들에 대해 복제된 데이터의 이미지를 서버가 저장할 필요가 없기 때문에 저장 공간의 사용을 줄이는 장점이 있다. 타임스탬프 테이블이 서버와 클라이언트에 모두에 생성되지만, 이것은 스냅샷을 이용한 방법에 비하여 요구되는 저장 공간의 크기가 매우 작다.

5. 시스템 구현과 성능 평가

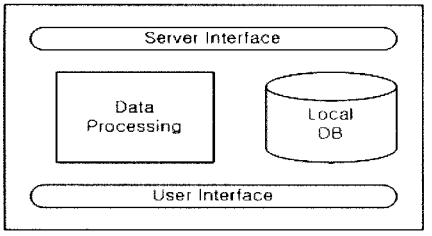
5.1 서버와 클라이언트의 구성 요소

서버의 구성은 (그림 6)과 같다. 서버에는 중계 서버, 웹 서버, 애플리케이션 서버, 데이터베이스 서버가 있다. (그림 6)에서 음영으로 표시된 중계 서버는 클라이언트로 복제할 데이터가 무엇인지가 기술되어 있는 사용자 프로파일이 있다. 클라이언트와 서버의 데이터 교환은 파일을 이용하며 이것을 데이터 파일이라 부른다. 파일명은 중계 서버 내에서 유일해야 다른 클라이언트들과 데이터 교환 시 충돌이 발생하지 않는다. 데이터 프로세싱 모듈은 데이터 동기화와 트랜잭션 처리에 사용되는 프로시저들을 포함한다.

클라이언트의 구성은 (그림 7)과 같다. 클라이언트는 사용자 인터페이스, 데이터 프로세싱 모듈, 지역 데이터베이스, 서버 인터페이스 등으로 구성된다. 사용자 인터페이스는 사용자의 요청을 입력받고, 실행 결과를 출력하는 부분이다. 데이터 프로세싱 모듈은 입력된 사용자 요청을 지역 데이터베이스나 서버 데이터베이스에 접근하여 처리하는 부분이다. 지역 데이터베이스는 서버 데이터의 일부를 복제하고 있으며, 데이터와 타임스탬프를 함께 갱신하여 이후의 데이터 동기화에 대비한다.



(그림 6) 서버 구성도



(그림 7) 클라이언트 구성도

5.2 성능평가

타임스탬프 트리를 이용한 데이터 변화 탐지 방법의 성능을 평가하기 위해 시뮬레이션을 수행하였다. 성능 평가의 척도는 데이터 변화 탐지를 위해 타임스탬프 트리의 노드를 접근한 횟수이다. 시뮬레이션은 여섯 자리의 키를 갖는 데이터를 이용했으며, 대·중·소 분류를 나타내는 세 개의 두 자리 코드로 구성되었다. 이 타임스탬프 트리의 높이는 4가 된다. 시뮬레이션에서 사용된 데이터의 키는 일양 분포를 갖는 난수를 이용하여 생성했고, 타임스탬프와 함께 트리에 저장되었다. 시뮬레이션은 데이터의 수를 1000, 5000, 10000 개로 증가시키면서 수행하였다. 갱신된 데이터가 단말 노드 전체에 균일하게 퍼져있는 최악의 경우를 실현하기 위해 일양 분포의 난수를 사용하였다. 시뮬레이션은 10회 반복하여 실행하고, 평균값을 결과로 이용하였다.

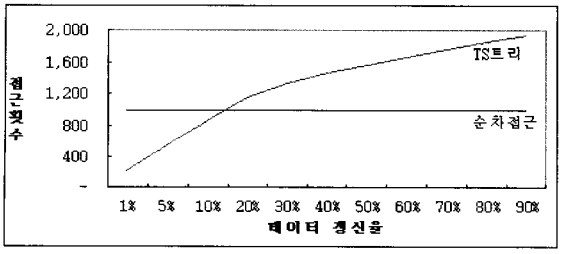
(그림 8)은 시뮬레이션의 수행 결과를 보여준다. (그림 8) (a)는 데이터의 수가 1,000인 경우 데이터 갱신율에 따라 접근한 타임스탬프 트리의 노드 수를 보여준다. 이 경우 트리의 총 노드 수는 평균 2,052개이다. 갱신된 데이터를 탐지하기 위해 타임스탬프 트리의 노드를 접근한 횟수를 보면, 데이터의 갱신율이 1%인 경우 218개, 5%인 경우 530개, 10%인 경우 866개의 노드를 접근해서 순차 접근을 이용한 변화 탐지 방법보다 우수한 성능을 보였다. 하지만 (그림 8) (a)의 그림에서 보듯 갱신율이 20%인 경우 1,169개의 노드를 접근하여 순차 접근 방법과 비교하여 많은 노드를 접근하였다. 타임스탬프 트리를 이용한 데이터 변화 탐지 방법은 데이터 갱신율과 비례하여 접근하는 노드의 수가 증가하며, 갱신율이 15%이상인 경우 순차 접근을 이용한 변화 탐지보다 많은 노드를 접근함을 알 수 있다.

(그림 8) (b)는 데이터의 수가 5,000인 경우를 시뮬레이션한 결과를 보여준다. 타임스탬프 트리의 총 노드 수는 평균 9,024개이며, 갱신된 데이터를 탐지하기 위해서 갱신율이 1%인 경우 1,760개의 노드를 접근했고, 갱신율이 5%인 경우 4,004개의 노드를 접근했으며, 갱신율이 10%인 경우 4,720개의 노드를 접근하였다. 또한 갱신율이 15%이상인 경우 순차 접근을 이용했을 때 보다 많은 노드들을 접근하였다.

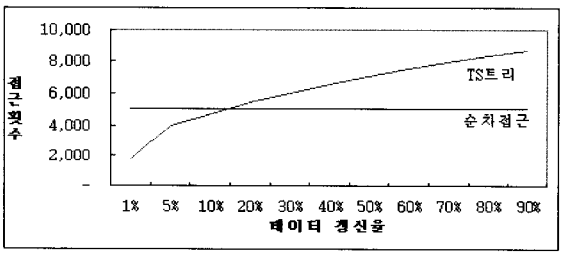
(그림 8) (c)는 데이터의 수가 10,000인 경우를 시뮬레이션한 결과로 타임스탬프 트리의 총 노드 수는 16,380개이다. 갱신율이 1%인 경우 4,179개의 노드를 접근했고, 갱신율이 5%인 경우 4,735개의 노드를 접근했으며, 10%인 경우에는 8,265개의 노드를 접근했다. 갱신율이 20%인 경우 9,841개의 노드를 방문하여 순차 접근을 이용했을 때보다 적은 접근 횟수를 갖는다.

(그림 8) (a), (b), (c)를 보면 데이터의 수가 1,000개인 경우 타임스탬프 트리의 총 노드 수가 약 2배정도 많았으나, 데이터 수가 증가할수록 이 비율은 감소하여 데이터 수가 10,000인 경우 약 1.6배로 그 차이가 줄었다. 이것은 데이터의 수가 적은 경우, 중간 노드의 팬-아웃(fan-out) 역시 적지만, 데이터 수의 늘어날수록 중간 노드의 팬-아웃도 증가한 결과이다.

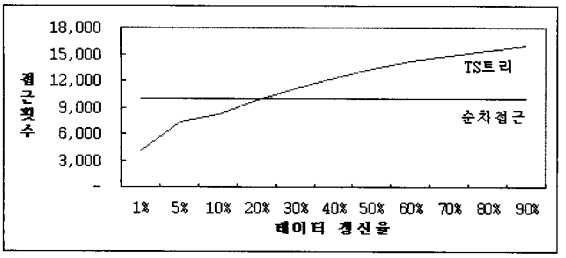
(그림 8) (d)는 데이터 변화 탐지를 위해 제안된 방법과 순차 접근 방법의 상대적 노드 접근 비율을 보여준다. Y축의 값이 100%인 경우 제안된 방법과 순차 접근을 이용한 변화 탐지 방법이 동일한 수의 노드를 접근하는 것을 의미한다. 데이터의 수가 1,000인 경우 갱신율이 15% 이하일 때만 순차 접근과 비교하여 적은 노드 접근 횟수를 갖지만, 데이터의 수가



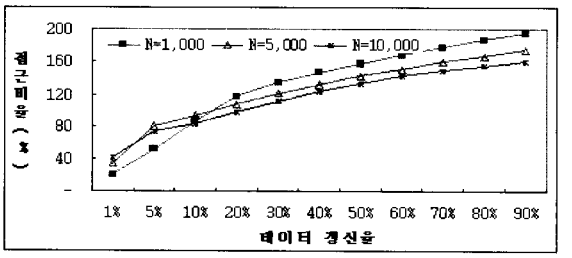
(a) N=1,000 인 경우



(b) N=5,000인 경우



(c) N=10,000인 경우



(d) 순차접근과의 상대적 접근 비율

(그림 8) 시뮬레이션 결과

10,000인 경우 갱신율이 20% 일 때도 순차 접근보다 노드 접근 횟수가 적다. 이것은 타임스탬프 트리의 중간 노드들이 충분한 팬-아웃을 갖을수록 접근하는 노드의 수가 줄기 때문이다.

타임스탬프 트리를 이용한 데이터 변화 탐지의 효율성은 중간 노드들의 팬 아웃과 데이터 갱신율에 따라 결정되는 것을 알 수 있다. 실제로 제안된 방법을 A사의 웹 기반 건설 MRO B2B 시스템에 적용했을 때 성능 향상의 효과가 있었다. 그 이유는 시뮬레이션과 달리 운영중인 B2B 시스템의 데이터 키 값의 분포가 밀집되었기 때문이다. 또한 A사의 웹 기반 건설 MRO B2B 시스템을 2004년 4월부터 2004년 8월까지 관측한 결과에 따르면, 월 평균 데이터 갱신율이 7% 이하였다. 갱신된 데이터를 살펴보면 안전화, 안전모 등과 같이 동일한 분류의 상품들이 집중적으로 변경됨을 알았다. 따라서 데이터 변경이 타임스탬프 트리의 일부 서브트리들에서만 제한적으로 발생하기 때문에 제안된 방법이 기존의 순차 접근을 이용한 변화 탐지보다 우수한 성능을 보였다.

6. 결론 및 향후 연구과제

본 논문은 A사의 웹 기반 건설 MRO B2B 시스템을 확장, 개선하는 프로젝트를 진행하면서 수집된 자료와 결과를 기초로 하고 있다. 기존의 웹 기반 B2B 시스템을 재구축하지 않고 느린 전송 속도를 갖는 네트워크를 이용하는 사용자들에게 원활한 서비스를 제공하기 위해 중계 서버를 도입하였다. 클라이언트와 서버 사이에 복제된 데이터의 일관성을 보장하기 위한 동기화 과정에서 갱신된 데이터를 탐지하는 것은 매우 중요하다. 데이터 변화 탐지는 저장 장치의 입출력을 위주로 동작하기 때문에 서버에 많은 부하를 준다. 본 논문에서는 동시에 다수의 클라이언트들에게 짧은 응답 시간을 갖는 데이터 동기화 시스템을 구축하기 위해 타임스탬프 트리를 이용한 데이터 변화 탐지 방법을 제안하고, 구현하였다.

시뮬레이션을 통한 성능 평가 결과는 데이터 갱신율이 15% 이하인 경우 제안된 방법이 순차 접근을 통한 데이터 변화 탐지보다 우수한 성능을 보였다. A사의 웹 기반 건설 MRO B2B 시스템을 2004년 4월부터 2004년 8월까지 관찰한 결과에 따르면, 월 평균 데이터 갱신율은 7% 이하이고, 데이터 갱신은 일부 서브트리들에 한정되어 발생했다. 따라서 제안된 방법이 실질적으로 데이터 변화 탐지 성능을 향상시켰다. 또한 제안된 방법은 기존의 스냅샷을 이용한 방법과 비교할 때, 서버가 각 클라이언트들이 복제하고 있는 데이터의 이미지를 저장할 필요가 없기 때문에 서버의 저장 공간 사용을 줄였다.

향후 연구 과제로 본 논문에서 제안한 방법을 다양한 B2B 시스템들에 적용하면서 보다 많은 관측 자료를 분석할 필요가 있다. 그리고 데이터의 키 값이 순차적으로 할당되는 경우에도 타임스탬프 트리를 이용한 데이터 변화 탐지 방법이 유효한지에 대한 연구가 필요하다.

참 고 문 헌

[1] 배미숙, 황부현, "부분 중복 데이터베이스에서 중복 데이터의 트리를 이용한 일관성 유지", 정보처리학회논문지D, 제10-D권 제4호, pp.647-654, 2003. 8.

[2] 서재철 외 4인, "2004년 상반기 정보화실태조사 최종보고서", http://isis.nic.or.kr/sub04/sub04_index.html?sub=00V&id=596, 2004. 8.

[3] 이수철, 변광준, 황인준, "XML 기반 B2B 전자상거래 솔루션 동향", 한국정보과학회지, 제18권, 제7호, pp.21-27, 2000.

[4] 차정은, 김행곤, "전자상거래 시스템 구축을 위한 컴포넌트 아키텍쳐 및 명세 방법 연구", 정보처리논문지, 제7권 제5S호, pp.1629-1637, 2000.

[5] 최윤석, "인터넷 환경을 위한 Oracle8iLite", http://www.oracle.com/kr

[6] J. P. Boone, J. Pederson, "Extending Enterprise Data and Applications to Mobile Device using DB2 Everyplace", IBM's White Paper.

[7] James J. Kistler and M. Satyanarayanan, "Disconnected operation in the coda file system." Proceedings of the thirteenth ACM symposium on operating systems principles, pp.213-225, October, 13-16, 1991, Pacific Grove, California.

[8] L. B. Mummert, M. R. Ebling and M. Satyanarayanan, "Exploiting weak connectivity for mobile file access". Proceedings of the fifteenth ACM symposium on operating systems principles, pp.143-155, December, 3-6, 1995, Copper Mountain, Colorado.

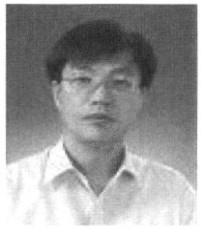
[9] T. Johnson and K. Jeong, "Hierarchical matrix timestamps for scalable update propagation", 10th Workshop on Distributed Algorithms, June, 1996.

[10] Sybase, "Integrating Remote Workgroups & Occasionally Connected Devices with Enterprise", http://www.ianywhere.com/downloads/whitepapers/sql_mob_whitepaper.pdf"

[11] Sybase, "Synchronization Technologies for Mobile and Embedded Computing", http://www.ianywhere.com/downloads/whitepapers/mobilink_sql.pdf

손 세 일

e-mail : eric31@dku.edu
1993년 유한전문대학 전자계산과
1997년 한국방송통신대학교 전자계산학과 (학사)
1999년 단국대학교 대학원 전산통계학과 (석사)
2002년 단국대학교 대학원 전산통계학과 (박사수료)
1993년~1996년 상지전산(주)
2002년~현재 단국대학교 정보컴퓨터학부 강의전임강사
관심분야 : P2P, 유비쿼터스 컴퓨팅, 트래픽 분석, 데이터베이스, 전자상거래



김 흥 준

e-mail : thinkthe@jinju.ac.kr
1989년 단국대학교 전자계산학과(학사)
1993년 단국대학교 대학원 전산통계학과 (석사)
1999년 단국대학교 대학원 전산통계학과 (박사)
1999년~현재 진주산업대학교 컴퓨터공학부 부교수
관심분야 : 컴퓨터구성, 모바일 네트워킹, 전자상거래