제20회 한국정보처리학회 추계학술발표대회 논문집 제10권 제2호 (2003. 11)

부분 중복 데이터베이스에서 사본 트리를 이용한 중복 제어

배미숙 *, 황부현**
* **전남대학교 전산학과
e-mail:msbae@sunny.chonnam.ac.kr

The Replication Control using the Replica Tree in the Partially Replicated Databases

*Misook Bae , **Buhyun Hwang
* **Dept of Computer Science, Chonnam National University

요 약

데이터의 중복은 데이터의 가용성과 시스템의 성능을 향상시키기 위해 사용한다. 대규모 부분 중복 데이터베이스에서 효율적인 중복 관리를 위해서는 정확한 수행을 보장하기 위한 정확성 검증방법과 효율적인 갱신 전파 방법이 필요하다. 이 논문에서는 부분 중복 환경에서 각 주사본 사이트의 중복 데이터에 대한 트리 구조를 기반으로 한 갱신 전파 방법을 제시하며, 갱신 지연 전파로 인해 갱신 전파 도중에 발생할 수 있는 전파 트랜잭션과 주 트랜잭션의 충돌로 인한 비직렬성 문제를 타임스탬프와 상태 데이터베이스를 이용하여 해결한다. 이것은 판독 가용성을 증가시키면서 비직렬성으로 인한 재수행을 회피하게 되어 트랜잭션의 완료율을 증가시킨다.

1. 서 론

중복은 데이터의 가용성과 규모의 확장성을 위해 사용하며 또한, 이동 환경에서 이동 노드가 단절 상태에서도 데이터베이스를 접근할 수 있도록 하기 위해서도 사용되어진다. 분산 중복 시스템은 일관성을 유지하면서 높은 가용성을 제공해야한다. 중복 데이터베이스 시스템은 데이터의 중복 정도에 따라 데이터의 전체를 중복시키는 완전 중복과 부분적으로 필요한 데이터에 대해서만 중복시키는 부분 중복으로 나눌 수 있다. 완전 중복에 비해 부분 중복은 사용되는 데이터에 대해서만 일관성을 유지함으로써 계산부하를 줄이기 때문에 매우 효율적이며 중복 비용과 디스크공간을 절약할 수 있으며 중복의 일관성을 유지하기 위한 조정 (reconciliation) 시간을 줄일 수 있다. 이러한 부분 중복의 사용 예는 이동 환경에서 단절 연산을 수행하기 위해 그 사이트에서 필요할 것 같은 데이터만을 캐쉬에 저장하는 경우를 들 수 있다 [1,2]. 이러한 장점에 힘입어 부분중복의 응용이 현재 현격히 확대되고 있다.

제안하는 방법의 특징을 간략히 몇 가지로 요약하면 다음과 같다. 첫째, 이 논문은 부분 중복 환경에서의 중복 관리 방법을 제안한다. 완전 중복 환경에서 관리 방법을 부분 중복 환경에 적용하면 모든 사이트에 갱신 결과를 전파시킴으로써 불필요한 메시지 전송이 발생하고, 전파 시간 또한 길어진다. 이를 방지하기 위하여 부분적으로 중복된 데이터와 사이트를 갖는 구조에서 중복된 데이터가 있는 사이트만을 계층화하여 갱신이 발생할 때 불필요한 메시지가 발생하지 않고 빠른 시간내에 갱신을 전파할 수 있는 메카니즘을 제안한다. 둘째, 이 논문은 갱신 전파를 위해 지연갱신 전파 방법을 사용하여 데이터의 위치정보를 갖는 사본트리(RT: Replicas Tree)를 이용한다. 세째, 일반적인 지연 갱신 전파 방법에서는 트랜잭션이 갱신을 수행한 후 전파되는 시점과 임의의 사이트에서 트랜잭션 연산의 완료 시점의 차이로 인하여 비직렬성 문제가 발생할 수 있다. 이 논문에서는 이러한 비직렬성 문제를 타임스탬프와 상태 데이터베이스 (SDB: Status DataBase)를 이용하여 해결한다. 넷째, 이 논문은 갱신이 드문 환경을 가정하며 단일 마스터 갱신 기법을 사용한다.

%본 논문은 한국과학재단 지역대학우수과학자지원연구(R05-2003-000-10532-0)에 의하여 지원되었음.

2. 관련 연구

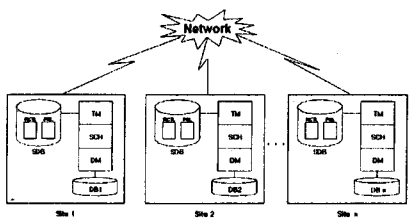
데이터가 여러 사이트에 중복 저장된 경우에는 데이터베이스 일관성뿐만 아니라 중복 일관성도 유지해야 한다. 중복 일관성 유지를 위해 갱신 전파 방법이 필요하다. 이러한 갱신을 전파하는 방법으로는 즉시갱신 전파 기법과 지연갱신 전파 기법이 있으며, 갱신을 제어하는 방법으로 갱신이 어느 곳에서나 발생할 수 있는 그룹 (group)갱신 방식과 주사본이 있는 곳에서만 갱신이 가능한 마스터 (master)갱신 방식으로 분류하였다 [3,4,5,6]. 그룹 갱신 방식은 마스터 갱신 방식에 비해 갱신 충돌의 가능성이 더 많다. 마스터 갱신방식은 시스템의 엔지니어링 비용이 적거나 서비스가 대부분 읽기 위주의 시스템에 적합하며 그룹 갱신 방식에 비해 단순하며 충돌이 희박하고 규모 확장면에서 우월하다.
[3,7,8,9]에서는 지연 마스터 전파 중복 데이터베이스 (lazy master replication database)에서 지연 전파로 인하여 중복 일관성 (replication consistency)이 위배될 수도 있다는 것을 지적하고 있다. 사본의 정보 흐름을 표시하는 사본들의 토폴로지 형태는 성능과 복잡성이 서로 다르다. Bengal 시스템에서도 많은 사본을 포함하는 경우 트리 토폴로지의 우수성을 지적하였다 [5,10,11]. 일반적으로 트리 구조는 임의의 데이터 노드에 대한 효율적인 접근과 갱신이 모두 가능하며, 네트웍상에서의 모든 사이트를 트리의 노드로 대응시켜 모든 노드를 순회 (traverse)하는데도 효율적이다.
낙관적인 중복 시스템에서는 충돌갱신을 자동적으로 해결하기 위한 특정 응용 라이브러리를 제공하며, 일관성을 유지하기 위해 조정을 수행할 때 완전 낙관 중복과 선택(부분) 낙관 중복 방법이 있다. 이러한 두 방법에 대해 요구되는 운영시간을 비교하여 선택 중복에서 많은 오버헤드를 낮출 수 있음을 보였다 [1,2,6,11].

3. 제안한 충돌 회피 방법

3.1 시스템 모델의 구조 및 가정

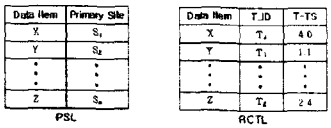
이 논문에서는 주사본 사이트를 탐색하는 오버헤드를 줄이기 위해 주사본 사이트의 위치 정보를 따로 저장한다. 이를 위해 상태 데이터베이스라는 자료구조를 사용한다. 또한, 모든 사이트에 갱신 메시지를 보내는 비효율성을 제거하기 위해 사본들의 위치를 트리로 구성한 사본 트리를 사용한다.

제안하고 있는 알고리즘의 시스템 모델은 (그림 1)과 같다. 사이트들은 네트워크를 통하여 연결되어 있고, 각 사이트는 트랜잭션 관리자 (Transaction Manager)와 스케줄러 (SCHeduler)로 구성된다. 각 데이터베이스의 데이터는 부분 중복되어 있다. 트랜잭션 관리자는 제출된 트랜잭션에 대한 타임스탬프를 부여하고, 조정자가 되며 여러 사이트를 접근하여 수행하는 트랜잭션의 분배를 담당하며, 상태 데이터베이스를 구성하기 위해 각 데이터 항목에 대한 주사본의 정보를 수집한다. 상태 데이터베이스는 갱신 연산이 수행될 주사본 사이트의 위치 (PSL: Primary Site Location) 와 주사본 사이트에서 수행되어 가장 최근에 완료된 트랜잭션의 리스트 (RCTL: the most Recent Committed update-Transaction List)를 유지한다. RCTL은 주사본 사이트에서 갱신 트랜잭션이 완료되는 경우, 전파 트랜잭션이 각 사본 사이트에 도착하기 전, 갱신 정보를 사본들에게 알려주기 위해 사용된다. 주사본 사이트에서는 갱신 트랜잭션이 완료되면 사본이 저장된 모든 사이트에 갱신 정보를 방송하고, 갱신 정보를 받은 사이트에서는 RCTL을 갱신한다. RCTL에는 자신의 사이트가 주사본인 데이터 항목, 최근 완료된 트랜잭션 식별자와 그 트랜잭션의 타임스탬프에 대한 정보를 갖는다. 각 사이트는 상태 데이터베이스를 주기억장치에 둠으로써 잦은 수정으로 인한 접근 시간을 줄이도록 한다.
또한, 각 데이터 항목의 주사본 사이트에서는 사본 트리인 RT를 유지한다. RT는 사본 사이트들에 대한 구조 정보를 갖는 트리이다. 주사본 사이트는 데이터에 대한 접근 횟수가 가장 많은 사이트이며 트리의 루트가 된다. 접근횟수가 비슷하면서 근접한 사이트들은 트리에서 같은 레벨에 위치하도록 구성한다. 이러한 형태는 같은 부모를 갖는 같은 레벨의 사이트들을 그룹으로 분할하여 작업을 가능하게 한다. 각 데이터의 주사본이 위치한 스케줄러는 다른 사본들의 위치를 트리 형태로 유지하여 효율적으로 갱신을 전파한다. 사본 트리를 이용하여 각 사이트들은 [5,12]에서 사용한 방법처럼 갱신을 효율적으로 전파할 수 있도록 사이트 구조가 재구성될 수 있다. 갱신의 결과는 트리의 경로를 따라 전파된다. 트리는 전파 시간을 최소화하기 위해 모든 노드까지의 깊이가 동일한 균형 트리 (balanced tree)로 유지한다. 또한, 제안한 환경은 네트워크가 신뢰성이 있어서 한 사이트에서 다른 사이트로 전달한 메시지는 반드시 전달되며, 각 메시지들은 보내진 순서대로 도착한다는 것과 트랜잭션에 유일한 타임스탬프를 부여하기 위해 각 사이트의 클럭이 동기화되어 있다고 가정한다. 알고리즘 설명을 위해 제출된 사이트에서 실행되는 트랜잭션을 주트랜잭션 (Transaction)이라 하며, 주트랜잭션이 완료된 후 갱신 전파를 위해 사본이 있는 다른 사이트로 보내어져 실행되어지는 트랜잭션을 전파 트랜잭션 (PT: Propagation Transaction)이라 한다.

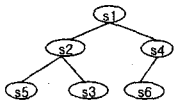


(그림1) 시스템 모델의 구조

아래 (그림 2)는 주사본 사이트에서 유지되는 상태 데이터베이스의 예이며, (그림 3)은 사본 트리의 예이다. 상태 데이터베이스에는 PSL과 RCTL을 유지한다. RT는 주사본 사이트에서 유지하며, 갱신의 전파를 효율적으로 할 수 있도록 사본이 있는 사이트에 대한 정보를 트리 형태로 유지한다. (그림3)은 임의의 데이터 x 에 대한 사본 트리 구조의 예이다. 트리의 루트인 s1은 x에 대한 주사본 사이트를 의미하며, s1을 제외한 나머지 노드들은 x의 사본이 위치한 사이트이다.



(그림2) SDB의 예



(그림3) RT 구조의 예

3.2 RCP-RT (Replication Control Protocol using Tree of Replica Tree)

3.2.1 주트랜잭션의 제출 및 스케줄

트랜잭션은 임의의 사이트의 트랜잭션 관리자에게 제출되며, 트랜잭션이 제출된 사이트가 조정자가 된다. 트랜잭션 관리자는 각 트랜잭션에 유일한 타임스탬프를 부여한다. 트랜잭션의 갱신 연산은 반드시 주사본 사이트로 전달되어 수행된다. 판독 연산은 주사본 사이트에서도 수행을 허용하나, 연산들 사이의 충돌로 인한 지연이 많거나 사본이 존재하지 경우에는 접근 비용이 가장 적거나 부하가 적어서 연산을 빨리 수행할 수 있는 사이트들 중 하나에서 수행하도록 한다.

3.2.2 주 트랜잭션의 완료 및 충돌 회피

분산된 사이트에서 수행된 각 트랜잭션의 조정자는 트랜잭션의 완료를 위해 2PC 프로토콜을 사용한다. 판독 연산을 포함하는 트랜잭션은 완료시 연산을 수행한 사이트가 주사본 사이트가 아닌 경우 접근한 데이터의 주사본 사이트에 완료했음을 알린다. 이 논문에서는 지연 갱신 전파 방법을 사용하고 있는데, 임의의 사이트에서 데이터 x에 대해 전파 트랜잭션 PT_i가 주트랜잭션 T_j가 접근할 판독연산 $r_j(x)$보다 늦게 도착한다면, 트랜잭션 PT_i 와 T_j 의 타임스탬프의 크기가 $ts(PT_i) < ts(T_j)$일 때 직렬가능하지 않은 수행이 발생한다. 따라서 이러한 비직렬성 문제를 해결하기 위하여 판독 연산을 포함하고 있는 주트랜잭션은 완료를 하기 전에 전파트랜잭션과의 직렬성이 위배되는가를 체크할 필요가 있다. 이러한 직렬성 위배여부를 체크하기 위해 이 논문에서는 RCTL의 정보를 이용한다. RCTL에는 각 데이터에 대해 주사본 사이트에서 가장 최근에 완료된 트랜잭션의 리스트를 가지고 있으므로 임의의 사이트에서 주트랜잭션과 전파트랜잭션과의 충돌 관계를 체크하는데 사용될 수 있다. 그러므로 전파트랜잭션 PT_i가 아직 도착하지 않았더라도 RCTL의 정보를 이용하여 주트랜잭션 T_j 과의 타임스탬프를 비교하여 직렬성 여부를 알 수 있다.

3.2.3 갱신의 전파

갱신 트랜잭션이 완료되면 조정자는 그것의 갱신 연산을 주사본 사이트에 제출한다. 조정자는 트랜잭션의 모든 연산을 수행한 후 2PC를 사용하여 완료 여부를 결정하고, 트랜잭션이 데이터를 갱신했으면 주사본 사이트에서 갱신 전파가 수행될 수 있도록 해야하는데 조정자 사이트와 주사본 사이트가 일치하지 않을 경우에는 주사본 사이트에 결과를 알려주어야 한다. 그리고, 이러한 완료 결정을 받은 주사본 사이트는 각 사이트의 상태 데이터베이스가 최신의 정보를 갖도록 하기 위해 완료된 트랜잭션 정보를 각 사이트로 방송한다. 방송 시간은 적어도 갱신 전파 시간에 비해 적거나 같기 때문에 임의의 사이트에서는 전파 트랜잭션의 늦은 도착으로 인한 직렬성이 위배되는 경우를 줄일 수 있다. 갱신연산을 수행한 후, 주사본 사이트는 트리 구조에서 그 주사본 사이트의 자식에게로 갱신 결과를 전파한다. 완료된 트랜잭션의 타임스탬프가 전파되어질 갱신 연산에 할당된다. 판독전용 트랜잭션은 완료하면 자신의 사이트가 주사본 사이트가 아닌 모든 데이터 x에 대해서, x의 주사본 사이트에 완료했음을 알린다.

제안하는 RCP-RT 알고리즘은 다음과 같다.

```
1. 임의의 사이트에 트랜잭션 T가 제출될 때
2. if (T가 주 트랜잭션 Ti 이면) TM은 T에 유일한 타임스탬프 할당;
   else if (T가 전파 트랜잭션 PTi 이면) {
          // primary에서 전파를 시작하기 위한 트랜잭션
        if( 갱신을 전파할 하위 사이트가 있다면 ) 갱신을 전파;
        else PT 수행   // 갱신 반영
      }
3. for (each operation op(x) of T)
     if (op == r)
        if (현 사이트에 부하 많다)  접근비용 적은 사이트로 op(x) 전송;
        else  //현 사이트에서 수행
           { if (w_ts(x) < ts(T)) perform r(x);
             else abort T;
           }
     if (op == w)
           if (현 사이트가 primary site )
              if (w_ts(x) < ts(T) && r_ts(x) < ts(T)) perform w(x);
                        //임시 저장
           else primary site로 전송;
4. for (다른 사이트로부터 전송되어온 각 r(x), w(y)에 대하여)
     if (타임스탬프 규칙을 만족하면) 수행 후 결과를 반환;
     else 철회 결정 전달;
5. if (트랜잭션의 마지막 연산의 수행이 끝나면)
     if (다른 사이트에서 수행된 연산이 존재) 수행한 사이트로 vote-request를 보낸다.      //2PC수행
        if (결과가 모두 yes)
           if (T가 모두 갱신 연산으로 구성) Commit T;
           else Precommit T;   // T가 판독 연산을 포함
        else abort;  // 하나라도 no가 포함
5.1 precommit된 T 에 대하여,
        call CAMP  // Conflict-Avoidance-Method Procedure
```

```
6. if (완료한 트랜잭션 T가 write 연산을 포함)  //조정자 ≠ 주사본 사이트
        완료 결정을 primary 사이트로 전달

Procedure CAMP {
    for each (T 의 모든 판독 연산 r(x)에 대하여)
        if (ts(T) < ts (RCTL(x))  Abort T
        else if (w_ts(x) == ts(RCTL(x)) Commit T  //PT의 도착
        else  { Wait until PT arrives:
                PT가 도착하면 PT수행:
                Commit T:   }
}
```

4. 결론

최근 데이터베이스의 한 추세는 시스템의 규모가 커지고 있으며, 부분적으로 필요한 데이터에 대해서만 중복시킨다는 것이다. 이 논문에서는 부분 중복 데이터베이스 시스템에서 적용될 수 있는 트랜잭션 수행과 트리 구조에서의 갱신 전파 방법을 제안하였다. 제안한 방법은 타임스탬프와 상태데이터베이스를 사용하여 트랜잭션의 직렬성을 보장한다.
각 주사본 사이트에는 중복 데이터에 대한 균형된 트리를 유지함으로써 갱신 전달 시간을 단축할 수 있다. 각 사이트의 상태 데이터베이스에는 주사본의 위치와 주사본에서 가장 최근에 갱신 완료한 트랜잭션의 정보가 저장된다. 주사본의 위치 정보를 이용하여 주사본에 빨리 갱신 연산을 전달할 수 있으며, 트랜잭션의 정보를 이용하여 전파트랜잭션의 늦은 도착으로 인한 주 트랜잭션과의 충돌을 회피할 수 있었다. 그러므로 제시된 방법은 트랜잭션의 전파시간을 단축시키며, 판독 가용성을 증가시키면서 직렬성을 유지하고, 비직렬성으로 인한 트랜잭션의 재수행을 차단시킴으로써 재수행으로 인한 시스템 자원의 낭비를 막을 수 있으며 트랜잭션의 완료율을 높일 수 있다는 장점을 갖는다.

참 고 문 헌

[1] David H.Ratner, Peter L.Reiher, Gerald J.Popek, and Richard G.Guy, "Peer Replication with Selective Control", 1999.

[2] Peter Reiher, John Heidemann, David Ratner, Greg Skinner and Gerald Popek, "Resolving File Conflicts in the FICUS File System", Proceedings of the Summer USENIX Conference, p183-p195, June 1994.

[3] J.Gary, P.Helland, P.O'Neil and D.Shasha, "The Danger of Replication and a Solution", In *Procs. of ACM SIGMOD International Conf. on* Management of Data, Montreal, Canada, p173-p182, 1996.

[4] T.Anderson, Y.Breitbart, H.F.Korth and A.Wool, "Replication, consistency and practicality: Are these mutually exclusive?", In Procs. of the ACM SIGMOD International Conf. on Management of Data, Seattle, WA, p484-p495, June 1998.

[5] X.Liu, B.Heal and E.Du, "Multiview Access Protocols for Large Scale Replication", ACM Transaction on Database Systems, Vol.23 , No.2, p158-198, June 1998.

[6] Yasushi Saito, Christos Karamanolis, Magnus Karlsson, and Mallik Mahalingam, "Taming Aggressive Replication in the Pangaea Wide-Area File System", 2002.

[7] Y.Breitbart and H.Korth, "Replication and consistency: Being lazy helps sometimes", In Procs. of the ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, Tucson, Arizona, p173-p184, May 1997.

[8] E.Pacitti and E.Simon, "Update Propagation Strategies to Improve Freshness in Lazy Master Replicated Databases", VLDB Journal, p305-p318, 2000 .8.

[9] M.Patino-Martinez, R.Jimenez-Peris, B.Kemme, G.Alonso, "Scalable Replication in Database Clusters", In Proc. of Distributed Computing Conf., DISC'00. Toledo, Spain, volume LNCS1914, p315-p329, October 2000.

[10] Todd Ekenstam., Charles Matheny, Peter Reiher *and Gerald J. Ppopek*, "The Bengal Database Replication System", Distributed and Parallel Databases, p187-p210, 2001.

[11] A. Wang, P.L. Reiher and R. Bagrodia, "A Simulation Evaluation of Optimistic Replicated Filing in Mobile Environments", Proceedings of International Performance, Computing and Communications Scottsdale, AZ, USA, p43-p51, 1999.

[12] T.Johnson and K.Jeong, "Hierarchical Matrix Timestamps for Scalable Update Propagation", 10th International Workshop on Distributed Algorithm, June 1996.