

디지털 문서의 메타데이터 관리를 위한 2 버전 래치 기법

좌 은희, 박 석
서강대학교 컴퓨터학과
arumi@dblab.sogang.ac.kr, spark@dblab.sogang.ac.kr

Two Version Latch Algorithm for the Metadata Management in Digital Documents

Eun-Hee Chwa, Seok Park
Dept. of Computer Science, Sogang University

요 약

최근 메타데이터의 주요 논쟁점으로 메타데이터의 표준화 문제가 등장하고 있다. 새로운 표준화 방향으로 인한 메타데이터의 확장성은 기존 메타데이터 관리 기법의 변화를 요구하게 되었다. 즉, 사용자의 질의에 대한 신속한 정보 제공과 동적인 자료의 일관성 있는 저장과 유지방안을 필요로 한다.

이에, 본 논문에서는 디지털 라이브러리 환경에서 메타데이터 속성을 정의하고, 이러한 요구사항을 만족하는 병행수행 제어 기법인 2VL(Two Version using the Latch)을 제안한다. 2VL은 래치를 사용하여 2버전을 유지함으로써 판독과 기록 연산간의 충돌로 인한 지연을 최소화하며 판독 연산에 있어서의 빠른 응답시간과 높은 최근성 반영율을 보인다.

1. 서론

메타데이터란 원문 검색을 위해 제공되는 데이터를 말한다. 최근 메타데이터의 주요한 논쟁점으로 메타데이터의 표준화 문제를 들 수 있다. 다양한 형태로 존재하는 메타데이터를 하나의 프레임워크로 통합함으로써 상호운영성을 증대시키고, 웹 환경을 기반으로 한 전자상거래의 개념을 어떻게 메타데이터에 도입하는가가 그 초점이 되고 있다. 이에 대한 대표적인 프로젝트로 INDECS[4][7]와 DC[5]에서 메타데이터 표준화 방안을 마련하고 있다. 표준화 내용의 특징으로는 기존의 메타데이터가 단순히 서지 정보의 정적인 기록만을 의미했던 데 반해 새로운 표준화 내용은 관리적(지적 재산권 내용 포함), 상업적 내용을 포함하는 동적인 형태의 정보로 확장되고 있다는 점이다. 표준화 방향으로 인한 메타데이터의 확장성은 기존의 메타데이터 관리 기법의 변화 또한 요구하게 되었다. 즉 사용자의 질의에 대한 신속한 정보 제공과 함께 동적인 자료의 일관성 있는 저장과 유지방안을 필요로 한다. 이러한 메타데이터 관리는 기존 관리 방안 [2][6]인 데이터에 대한 삽입과 삭제 연산만으로 처리하기가 어렵다. 이는 처음 입력된 데이터에 대해 갱신 연산이 거의 없는 환경을 가정하였기 때문이다.

이에 본 논문에서는 메타데이터 관리시 요구되는 속성들을 정

의하고 이를 고려한 2VL(2 Version using the Latch) 알고리즘을 제안한다. 2VL은 일관된 데이터의 판독을 보장하고 최근성 반영율을 높이며 짧은 응답 시간을 갖도록 하여, 기존 메타데이터 관리 기법이 동적인 데이터를 관리할 때 발생할 수 있는 문제를 해결한다.

2. 메타데이터 관리 요구사항

메타데이터는 그 응용분야에 따라 다른 성질을 가지게 된다. 이에, 디지털 라이브러리의 메타데이터 관리에 요구되는 트랜잭션을 분류하고 메타데이터 속성을 정의한다.

2.1. 메타데이터 관리에 요구되는 트랜잭션의 분류

메타데이터 관리에 요구되는 작업을 크게 두 가지 형태인 갱신 트랜잭션과 판독전용 트랜잭션으로 분류한다. [표 1]은 트랜잭션 분류에 따른 세부 내용이다.

표 1 트랜잭션의 분류

성격	분류	갱신 트랜잭션	판독 전용 트랜잭션
상대적 수행시간		장기간	단기간
접근하는 데이터량		소량	다양한 범위
주요 연산 종류		판독과 기록	판독
데이터 최근성 요구 정도		절대적	절대적이지 않음
동시수행되는 트랜잭션수		적음	많음
연산 내용		추가, 갱신, 삽입, 삭제	질의 처리

갱신 트랜잭션은 기록 연산과 판독 연산간의 충돌을 최

소화하고자 각 연산내용에 따라 세분화시켰으며 이 중 삽입 연산은 다른 연산과의 충돌이 없으므로 별도의 로크가 요구되지 않는다.

2.2. 메타데이터의 속성

디지털 라이브러리의 메타데이터는 일반 데이터와는 다른 속성을 갖는다. 이에 메타데이터의 4가지 속성을 정의한다.

[메타데이터 속성 정의 1]

메타데이터 상호간에는 의존성이 존재하지 않는다.

[메타데이터 속성 정의 2]

어떤 메타데이터에 대해 완료된 연산 결과가 다른 메타데이터에 영향을 미치지 않는다.

[메타데이터 속성 정의 3]

판독연산은 일정 시간동안 판독한 데이터에 대해 재판독 하지 않는다.

[메타데이터 속성 정의 4]

하나의 메타데이터 문서 안에서 요소들간의 의존성은 존재한다. 즉, 다중 버전인 경우 버전간 의존성이 존재한다.

2.3. 판독 전용 트랜잭션의 일관성 기준

메타데이터에 있어 중요 평가 사항 중 하나는 최근성 반영률에 있다. 최근성 반영률은 데이터베이스가 유지해야 하는 일관성 기준의 완화로써 획득될 수 있다. 갱신 트랜잭션과는 달리 판독전용 트랜잭션은 데이터베이스 상태에는 아무런 영향을 미치지 않는다. 따라서 판독전용 트랜잭션이 유지해야 하는 논리적 일관성은 완화될 수 있다. 판독전용 트랜잭션에 대해 적용될 수 있는 일관성 기준[1]중 2VL은 갱신 일관성 기준을 만족한다. [그림 1]은 2VL이 생성하는 스케줄로 갱신일관성 기준을 만족한다. 트랜잭션 Q3는 7번 시점에서 트랜잭션 T2의 결과 값을 판독함으로써 메타데이터 관리 요구 사항인 최근성을 반영할 수 있게 된다. 트랜잭션 Q3가 판독하는 데이터의 일관성은 갱신 일관성 기준을 만족함을 [정리 1]에서 증명하였다.

	1	2	3	4	5	6	7	8
T1	R(W0)		R(Z0)	W(Z1)	C			
T2					R(W0)	W(W1)	C	
Q3		R(Z0)					R(W1)	C

그림 1 2VL이 허용하는 스케줄

[정리 1] 2VL의 판독 전용 트랜잭션은 갱신 일관성 기준을 만족한다.

· [메타데이터 속성 정의 1]에 의해 판독 연산은 판독에 영향을 미치는 갱신 트랜잭션 연산만을 고려한다.

· [메타데이터 속성 정의 3]에 의해 판독 전용 트랜잭션은 재판독 하지 않는다.

· 기록 충돌 연산간에는 배타적 래치의 적용으로 각 연산간 직렬 순서를 보장한다.

3. 관련 연구

기존 메타데이터 관리 기법[6]에서는 단일 버전을 사용하여 삽입과 삭제 연산을 번갈아 수행함으로써 모든 연산을 처리하였다. 그러나, 이러한 기법은 기록 연산이 빈번해질 경우 판독 연산에 대한 응답시간의 지연과 단일 버전 훼손시 발생될 수 있는 문제점들을 갖는다. 반면 다중 버전 알고리즘[1][8]을 고려해 볼 수 있는데 이는 저장되는 데이터의 방대함으로 부적합하다. 이에, 최소 수의 버전을 갖고 판독 연산과 기록 연산간의 충돌을 해결하고자 제안된 알고리즘인 2VQL[3]을 고려해 볼 수 있다. 그러나 2VQL은 Strict 2PL에 기반한 판독 전용 트랜잭션의 로크 기법으로 트랜잭션 완료 시점까지 불필요한 로크를 소유하게 함으로써 리프레쉬를 지연시켜 응답시간과 함께 최근성 반영률을 낮게 한다. 이는 [메타데이터 속성 정의 1] 즉, 상호간의 의존성이 존재하지 않는 서로 다른 데이터 항목에 대한 불필요한 판독 로크를 소유하도록 함으로써 갱신 트랜잭션의 리프레쉬를 상당시간 지연시킨다.

4. 메타데이터 관리를 위한 2 버전 병행수행 제어 기법

제안하는 알고리즘인 2VL은 래치를 사용하여 2 버전을 유지함으로써 2PL에 의한 불필요한 로크의 소유를 막는다. 래치를 사용하는 것은 앞서 정의한 [메타데이터 속성 정의 1]을 고려한다면 불필요한 로크의 소유를 제거했다는 점에서 효율적인 관리 방안이다. 그리고, 래치는 2PL과 같은 로킹 기법에 비해 비용 및 구현 시 실제 오버헤드가 적다는 장점을 갖는다[8]. 또한 2 버전을 유지함으로써 판독과 기록 연산간의 충돌로 인한 지연을 제거함으로써 판독 연산에 있어서의 빠른 응답시간 뿐만 아니라 기록 연산에 있어서도 지연 없는 수행을 함으로써 이후 판독 연산에 한해서는 최근성 반영률이 높게 된다. 2VL의 기본내용은 [그림 2]와 같다.

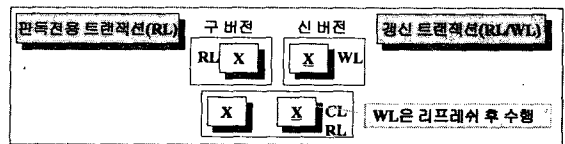


그림 2 2VL 개요

두개의 버전을 유지하며 판독 연산은 [그림 3]의 판독 데이터 테이블에 따라 판독 버전을 결정한다. 기록 연산은 신 버전을 생성하여 리프레쉬 로크를 획득하면 그 시점 이후에 수행되는 판독 연산은 판독 데이터 테이블에 따라 신 버전을 판독하도록 한다. 리프레쉬 수행시 발생하는 기록 연산은 리프레쉬가 완료될 때까지 지연되었다가 수행된다. 그러나, 구 버전에 대한 판독 연산의 로크 기간은 래치 사용으로 단시간이며 신 버전이 생성된 이후 판독 연산은

신버전을 판독하므로 리프레쉬 수행 시간은 짧게 된다. 따라서 대기하는 기록 연산의 시간도 짧다고 할 수 있다. 이러한 2VL 알고리즘은 앞서 소개한 [그림 1]의 스케줄을 허용한다. 트랜잭션 Q3에서 판독 연산의 래치 적용으로 구버전에 대한 판독 로크가 4번 시점과 6번 시점에서 남아있지 않기 때문에 트랜잭션 T1과 T2는 즉시적인 리프레쉬를 할 수 있다. 따라서, 트랜잭션 Q3는 트랜잭션 T2의 결과 값을 판독함으로써 최근성을 반영할 수 있다.

	판독 전용 트랜잭션	갱신 트랜잭션
상태 1	구 버전	구 버전
상태 2	구 버전	구 버전
상태 3	신 버전	신 버전

그림 3 2VL 판독 데이터 테이블

4.1. 판독 전용 트랜잭션 수행

2VL은 [표 1]의 트랜잭션 분류에 따라 판독 전용 트랜잭션과 갱신 트랜잭션 수행으로 나누어 소개한다. 다음은 판독 전용 트랜잭션 수행을 위한 규칙을 정의한다.

[판독 규칙 1]

모든 판독 연산은 판독 데이터 테이블에 따라 판독 데이터 버전을 선택한다.

[판독 규칙 2]

판독 연산에 대한 로크는 단기적 공유 래치를 사용한다.

4.2. 갱신 트랜잭션 수행

갱신 트랜잭션의 수행은 [그림 4]의 세 단계로 나누어 소개한다. 단계 2에서 기록 래치를 획득하고 1 버전임을 나타내는 OV를 RESET시킨다. 단계 3에서 기록 래치는 리프레쉬 래치로 변환되고 리프레쉬 조건이 만족되면 리프레쉬를 수행하여 단계 1로 변환한다. 기록 규칙은 다음과 같다.

[기록 규칙 1]

기록 연산의 로크는 배타적 래치를 사용한다.

[기록 규칙 2]

기록 로크와 다른 로크와의 충돌 관계는 로크 호환 테이블에 따르며 충돌하는 연산간에는 직렬성을 유지하기 위해 이전 연산의 리프레쉬 완료후 다음 연산을 수행한다.

4.3. 래치 관리 테이블

래치는 연산이 수행되는 동안에만 페이지에 걸리는 단기 로크로써, 연산간의 충돌을 방지하기 위한 별도의 관리 방안이 필요하다. 이에 대해 [그림 5]의 래치 테이블을 유지함으로써 연산 수행 순서의 직렬성을 유지한다. 래치관리 테이블은 [메타데이터 속성 정의 1]을 근거로 각 데이터단위별로 현재 로크를 획득한 트랜잭션 아이디들의 리스트를 유지한다.

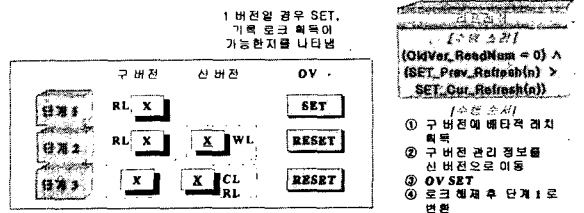


그림 4 갱신 트랜잭션 수행

데이터	OV	구 버전 판독 리스트	신 버전 판독 리스트	기록 리스트	리프레쉬 리스트	대기 리스트
X	RESET		Q3		T2	T5
Y	RESET	Q6		T7		T8
Z	SET	Q9, Q10				

그림 5 래치 관리 테이블

5. 결론

메타데이터베이스는 동적인 정보의 일관성 있는 관리와 사용자의 질의에 대한 빠른 응답 시간과 함께 최근성을 제공할 수 있어야만 한다. 이에, 본 논문에서는 메타데이터의 효율적인 관리를 위해 메타데이터의 속성을 정의하고, 판독전용 트랜잭션의 일관성 기준을 약화시킴으로써 판독 연산에 있어 빠른 응답시간과 최근 정보를 판독할 수 있는 방안을 제시하였다. 제안하는 알고리즘의 성능평가는 각 트랜잭션 수행시간과 리프레쉬 수행시간을 측정 비교하여 2VL이 더 나은 성능을 보임을 제시하였다.

참고문헌

[1]Paul M. Bober and M. J. Carey, "Multiversion Query Locking," in Proc. of the VLDB, pp. 497-510, Aug. 1992.
 [2]Mohan Kamath and Krithi Ramamritham, "Efficient transaction management & query processing in massive digital databases", Tech. report 95-93 Dept. of C.S.,University of Massachusetts, 1995.
 [3]Hoewon Kim and Seog Park, "Two Version Concurrency control Algorithm with Query Locking for Decision Support", Proc. of the International Workshops on Data Warehousing& Data Mining etc. (ER'98),pp153-164, Singapore, 1998.11.
 [4]Rust, Godfrey & Bide, Mark, "The <indec> metadata model", <indec> London conference, 1999.7.5. URL:http://www.indec.org
 [5]Renato iannella, "DC architecture WG proposal",1999. URL:http://archive.dstc.edu.au/RDU/DCAC/arch-wg.html
 [6]이해민, 박석, "디지털 도서관에서 래치에 기초한 메타데이터 관리 기법", 정보과학회 논문지 제27권1호, pp.22- 32, 2000.
 [7]좌은희, 박석, "디지털 정보에 대한 식별자 부여 및 전자 상거래용 메타데이터 모델에 관한 연구", KERIS 연구보고 RR 1999-2, 1999.
 [8]P.A.Bernstein,V.Hadzilacos and N. Goodman. Concurrency Control and Recovery in Database Systems. Reading, Addison-Wesley, 1987.