메인 메모리 데이터베이스 시스템에서의 그룹 완료 방식에 관한 연구

이 인선
신구대학 컴퓨터 정보 처리과
inseon@mail.shingu-c.ac.kr

A Study of Group Commit Policy in Main Memory Database System

In-Seon Lee
Computer Information Processing Dept. of Shingu College

요 약

트랜잭션 로그 정보를 안전한 저장장치에 저장하는 것은 트랜잭션의 완료를 위한 필수 불가결한 요소이지만 시스템 성능에 커다란 저해 요인이 되고 있다. 이에 대한 개선책으로 선-완료(pre-commit), 그룹 완료(group commit)와 같은 방식들의 제안이 있었으나, 이 방식들을 시스템에 적용했을 경우, 특히 로깅이 평상시 트랜잭션 수행시간의 거의 대부분을 차지하는 메인 메모리 데이터베이스(MMDB) 시스템과 같은 환경에서의 구체적이고 다양한 논의가 아직 이루어지지 않고 있다. 본 논문에서는 MMDB시스템에 그룹 완료를 적용할 때 발생할 수 있는 교착 상태에 대한 설명과 해결책을 제시하였고, 다각적인 모의실험을 실시하여 이전 논문들의 우려와는 달리 그룹 완료 방식이 전체 시스템 성능뿐만 아니라 트랜잭션 수행시간 단축을 위해서도 매우 우수한 해결 방식이며, 선-완료 방식은 예상과 달리 그룹 완료 방식의 보조수단으로 병행될 때에만 시스템 성능에 추가적인 도움을 준다는 것을 보였다.

1. 서론

데이터베이스 시스템에서는 완료된 트랜잭션의 지속성을 유지하기 위하여 갱신에 관한 재수행 로그를 반드시 안전한 저장 장치에 옮겨 놓은 후 트랜잭션이 완료되었음을 사용자에게 알린다. [Ros95]에서 알 수 있듯이 전체 트랜잭션의 수행시간중 대부분을 차지하는 것이 디스크에 대한 입출력작업으로 그중 필수적인 로그 정보의 디스크 출력에 대한 해결책으로 하드웨어부착, 메인메모리의 비휘발화등 여러 가지 해결책들이 시도되고 있다. 이중 로깅 정보의 디스크 출력 과정의 성능 개선책으로는 선-완료(pre-commit)방식과 그룹 완료(group commit)방식을 들 수 있다. 선-완료 방식은 갱신에 관한 로그 정보가 디스크로 출력되기를 기다리지 않고 연산 작업이 모두 끝난 직후에 바로 그 트랜잭션이 보유한 로크(lock)들을 해제하는 것으로 선-완료를 수행하는 트랜잭션의 응답시간에는 변화가 없지만 이 트랜잭션이 보유한 로크를 기다리는 다른 트랜잭션의 로크 대기 시간을 감소시킴으로써 전체적으로 시스템 성능을 향상시킬 수 있다[Gar92]. 그룹 완료 방식은 개별적으로 트랜잭션의 로그 정보를 디스크로 출력하지 않고 메인 메모리의 시스템 로그 버퍼에 여러 트랜잭션들의 로그 정보를 축적했다가, 예를 들어 시스템 로그 버퍼가 가득 찬 경우 이를 한꺼번에 디스크에 출력함으로써 시스템 전체적으로 디스크 출력 횟수를 줄이고자하는 방식이다[Gar92]. 이러한 완료 과정 개선책들은 로깅 정보를 디스크에 저장하는 것이 평상시 트랜잭션 수행시간의 대부분을 차지하는 경우에 매우 효율적으로 작동될 수 있다. 이러한 시스템으로 메인 메모리 데이터베이스 시스템을 들 수 있다.

메인 메모리 데이터베이스(Main Memory Data Base: MMDB)시스템은 데이터의 대부분 또는 모든 부분이 메인 메모리에 상주하는 데이터베이스 시스템으로 트랜잭션의 연산 수행시 디스크 입출력이 발생하지 않으므로 시스템 성능이 매우 향상되어 기존의 디스크를 기반으로 하는 데이터베이스보다 더 빠른 응답시간과 더 높은 트랜잭션 처리능력이 장점이 된다. 이러한 MMDB 시스템의 성능에 걸림돌이 되는 부분이 트랜잭션의 완료를 위한 로깅 정보의 디스크 출력이다. 이의 해결책으로 MMDB가 논의되던 초창기 논문들에서는 특별한 하드웨어의 부착을 가정하여 로깅 정보를 이 하드웨어에 기록하는 것으로 문제를 해결했다[Dew84][Gru91]. 이후 특별한 하드웨어를 가정하지 않을 경우선-완료 방식을 많은 시스템에서 채택하였으며[Dew84][Hec93], 그룹 완료방식에 대해서는 구체적인 적용기준, 성능에 대한 아무런 선행연구도 없이 막연히 성능에 도움을 줄 것이라는 추측으로 가정하거나[Dew84][ken90][Hec93], 또는 그룹 완료 방식이 시스템 전체적인 성능엔 도움을 줄 수 있으나, 개별 트랜잭션들의 응답시간을 늦추는 역효과가 있어 비용 측면에서 좋지 않을 것이라고 언급하기만 하였을 뿐[Wee98] 구체적이고 다각적인 성능 분석이 아직까지 이루어지지 않고 있다.

이 시점에서 디스크 출력이 전체 시스템에 과부하가 되는 MMDB시스템에 여러 완료방식들을 적용함으로써 시스템의 성능에 미치는 영향과 개별 트랜잭션 수행 시간의 변화에 대해 구체적인 성능 분석과 각 방식의 적용 시 어떤 문제점이 있는 지 발생 가능한 문제점들에 대한 상세한 고찰이 이루어지는 것이 타당할 것이다.

본 논문의 2장에서 그룹 완료 방식의 동작 원리 및 교착 상태가 발생하는 경우 및 이의 해결책에 대해 설명한다. 3장에서는 선-완료 방식의 적용시 선결사항에 대해 기술하고, 4장에서 다각적이고 구체적인 모의실험을 실시하여 여러 완료 방식들의 성능을 분석하며 5장에서 결론을 맺는다.

2. 그룹 완료 방식(Group Commit)

그룹 완료 방식은 하나의 트랜잭션이 완료를 시작하면서 그 트랜잭션의 로그를 바로 디스크에 출력하는 대신 여러 개 트랜잭션의 로그들을 메인 메모리에 있는 시스템 로그 버퍼에 계속 쌓아 두었다가, 정해진 기준에 다다르면 시스템 로그 버퍼에 쌓인 로그들을 하나의 디스크 출력 연산으로 디스크에 저장하는 방식이다[Gar92]. 그룹 완료 방식은 IMS FastPath에서 처음으로 구현되었으며[Gaw85], MMDB분야에서는 [Dew84]에서 고안했다. [Rob87]에서는 그룹 완료 방식을 파일 시스템에 적용하면서 1/2초를 그룹 완료 시행주기로 하였으나 이는 일반적이지 않고, 통상적으로 메인 메모리에 있는 시스템 로그 버퍼가 가득 찼을 때를 그룹 완료의 시행기준으로 한다[Ken90], 이 경우 아래의 예와 같이 교착상태가 발생할 수 있다.

(예 1) MPL=5, 시스템 로그 버퍼 크기 = 100바이트인 경우 터미널 3, 1에서의 트랜잭션은 연산수행이 끝나 이미 그룹 완료를 기다리는 '완료 그룹 목록'에 있는 상태이고, 시스템 로그에는 48바이트의 로그가 기록된 상태에서

터미널 2 : 읽기 1543페이지
터미널 2 : 쓰기 1371페이지
터미널 2: 완료 요청

위의 연산과정으로 터미널 2 역시 '완료그룹목록'에 삽입이 되나, 아직 시스템 로그 버퍼에 찬 로그가 80바이트로 그룹 완료가 시행되지 못하고, 또 다시 완료 요청하는 다른 트랜잭션을 기다리게 된다.

터미널 4: 쓰기 1543 페이지
터미널 0 : 읽기 1371페이지

위의 두 연산은 완료를 기다리는 터미널 2가 가지고 있는 페이지들의 로크를 요청하는 것으로 터미널 4, 0에서 수행 중이던 두 개의 트랜잭션은 '로크 대기 리스트'에 들어가게 된다. 이로써 더 이상 연산작업이 끝내고 시스템 로그 버퍼를 채울 트랜잭션이 없으므로 교착상태가 발생하게 된다.

이런 교착상태는 자료 경쟁 수준에 비해 큰 시스템 로그 버퍼를 가질 때 생기는 것으로 트랜잭션이 완료 요청을 할 때와 트랜잭션이 '로크 대기 리스트'에 들어갈 때마다 '완료 그룹 목록'에 있는 트랜잭션의 수와 '로크 대기 리스트'에 있는 트랜잭션의 수를 합하여 이 값이 현재 수행중인 모든 트랜잭션의 개수와 동일하면 강제로 그룹 완료를 시행함으로써 해결할 수 있다.

3. 선-완료 방식(Pre-Commit)

선-완료 방식은 트랜잭션이 완료 작업에 들어갈 때 가지고 있던 모든 로크를 해제하여 트랜잭션이 로크를 가지고 있는 시간을 감소시킴으로써 다른 트랜잭션의 로크 대기시간을 줄여 전체적으로 시스템 성능을 향상시키고자하는 방식으로 메인 메모리 데이터베이스 시스템뿐만 아니라 많은 논문에서 이 방식의 사용을 가정하고 있다[Dew84][Hec93][Gaw85]. 이 방식을 사용하기 위한 전제조건으로 완료작업을 시작한 이후에는 트랜잭션간의 완료 순서가 바뀌지 않는 동시성 제어 방식을 사용해야 한다. 스트릭 두단계 동시성 제어 방식을 적용하면 선-완료를 적용하더라도 자신이 의존하는 트랜잭션보다 먼저 완료되는 일은 발생하지 않아 트랜잭션간의 의존성은 유지되며, 시스템의 고장으로 인한 회복성까지도 보장된다[Elj95]. 또한 시스템 로그 버퍼를 기준으로 그룹 완료를 수행할 때 발생할 수 있는 교착상태를 선-완료방식을 병행하면 예방할 수 있다. 그러나, 좀 더 나은 성능을 위해 로그 디스크가 여러 개인 경우 선-완료 순서대로 로깅 정보가 디스크에 기록되도록 디스크 출력 순서를 제어하거나 부가적인 정보를 추가해야 하는 단점이 있다[Dew84].

4. 모의 실험

4.1 MMDB 시스템 모델의 가정사항

● 비휘발성 메모리의 장착은 가정하지 않는다.
● 선-완료 방식의 적용을 위해 스트릭 두 단계 동시성 제어방식을 가정한다.
● 데이터베이스는 페이지로 구성되며, 트랜잭션은 지수 함수 분포에 의해 발생한다.
● 트랜잭션은 사용자나 시스템의 고장에 의한 철회는 없고 단지 교착상태 해결을 위한 철회만이 있다고 가정한다.
●트랜잭션은 "READ", "WRITE"와 같은 자료에 대한 연산과 "BEGIN", "END"와 같은 트랜잭션 관리연산들로만 구성된다.
●트랜잭션의 연산은 워드단위이며, 로크는 페이지 단위로 이루어지고, 하나의 트랜잭션에 같은 페이지에 대한 한 번 이상의 연산은 없다고 가정한다.
●한 워드를 읽고 쓰는 경우 트랜잭션에는 "WRITE" 연산만 있고, 한번의 "WRITE" 연산은 "해당 워드 찾기+읽기+쓰기"에 해당하는 연산수행시간이 걸리는 것으로 한다.
●터미널에서 생성된 트랜잭션은 트랜잭션 관리기(Transaction Manager:TM)에 의해 수행되며, 만약 수행이 철회되면 실제의 그 트랜잭션이 완료될 때까지 트랜잭션관리기에 의해 반복 수행된다.
●트랜잭션의 갱신은 "즉각적인 갱신"방식으로 이루어지며, 모든 트랜잭션은 시작되면서 개별적인 '취소로그(undo log)', '재수행 로그(redo log)'를 가진다.
● 트랜잭션이 연산작업을 모두 마치면 TM에 의해 트랜잭션의 개별 '재수행 로그'가 메인 메모리의 전역 로그로 옮겨지고, 주어진 기준에 따라 전역 로그에 있는 '재수 행로그들이 디스크로 쓰여진 후 트랜잭션 작업은 완료된다.
● 체크포인팅작업은 완료 방식의 성능 분석을 하는 이 모의실험에서는 제외한다.
●모의 실험중 시스템 고장은 발생하지 않는 것으로 가정했으므로 시스템 고장으로 인한 MMDB시스템의 재적재(reload)는 발생하지 않는 것으로 한다.
●로깅 작업만의 정확한 성능 측정을 위해 로그 전용 디스크를 가정한다

4.2 모의 실험 수행 횟수 선정

모의 실험은 CSIM 모의 실험기[Sch92]를 사용하며, 완료방식의 성능 평가 기준치는 트랜잭션 평균 수행시간으로 한다. 모의실험 초기 값들은 정확한 자료 경쟁수준에 도달하지 못한 값들이므로 총 55,000개의 트랜잭션을 수행시키고, 초기 5,000개의 트랜잭션 수행 결과는 모의실험 결과치에서 제외시켰다.

4.3 모의 실험 매개 변수

매개변수는 불변 매개 변수와 가변 매개 변수로 나뉘며, 각각 [표 1]과 [표 2]에 그에 대한 의미와 값이 나타나 있다. 불변 매개변수들은 시스템을 대표하는 값들로 고정된 값을 가지며, 가변 매개 변수들은 다양한 자료경쟁수준에서의 성능을 측정하기 위해 주어진 범위에서 변하는 값들을 가지며, 이 매개변수는 [Mar97][Gru91]값들을 참조로 작성하였다.

매개 변수명	의미	값
DataBase	메인 메모리 데이터베이스 크기	1800 pages
MM_ACCESS	메인 메모리에서 하나의 워드를 접근하는 데 걸리는 시간	0.0001
MM_SEARCH	메인메모리에서 한 페이지를 찾는데 걸리는 시간(주소변환 시간)	0.0003
LOCK_TM	로크를 얻는 데 걸리는 시간	0.25 ms
UNLK_TM	로크를 해제하는 데 걸리는 시간	0.25 ms
LOG_ACCESS_TIME	로그버퍼에 로그 레코드를 기록하는 데 걸리는 시간	0.00011 ms
LOGPAGE_WRITE	로그 버퍼를 디스크에 쓰는 데 걸리는 시간	1ms
LATENCY	평균 latency	4 ms
BT_SIZE	BT 로그 레코드 크기	4 bytes
ET_SIZE	ET 로그 레코드 크기	4 bytes
UNDO_SIZE	undo 로그 레코드 크기	8 bytes
REDO_SIZE	redo 로그 레코드 크기	8 bytes
ArrivalRate	트랜잭션 도착 간격 (tps)	300

[표 1] 불변 매개 변수

매개 변수 명	의미	값의 범위(디폴트값)
MPL	동시프로그래밍 정도	5,10,15,20,25 (10)
TranSize	트랜잭션 하나의 크기	5,10, 15, 20 (10)
WriteProb	쓰기 연산의 비율	0.2, 0.4, 0.6, 0.8, 1.0 (0.2)
TranDiv	트랜잭션 크기의 편차	0, 0.5 (0)

[표 2] 가변 매개 변수

4.4 모의 실험 결과 분석

●모의 실험 1 : 그룹 완료 방식의 성능 평가

그룹 완료의 성능과 전역 로그 버퍼 크기가 미치는 영향을 분석하기 위해 그룹 완료를 시행하지 않고 개별적으로 완료작업을 하는 경우(A)와 자료경쟁수준에 맞는 전역 로그 버퍼 크기를 구하여 그룹 완료를 시행하는 경우(B), 일반적인 디스크 입출력 단위(512바이트)를 전역 로그 버퍼 크기로 하여 그룹 완료를 시행하는 경우(C)의 모의실험을 실시하였다. C의 경우에는 완료작업중인 트랜잭션들과 연산작업중인 트랜잭션간의 의존성이 생기면서 로크 대기 상태가 되는 트랜잭션이 생기고, 이후 더 이상 작업을 진행시켜 전역 로그 버퍼를 채워줄 트랜잭션이 없게되는 교착 상태가 발생하게 되는 경우에만

'강제그룹완료'를 시행하고, 그렇지 않을 경우 512바이트의 전역 로그 버퍼가 가득 찰 때만 그룹완료를 시행하도록 한다. 모의 실험 결과가 [표 3]에 나타나 있다.

실험 결과 변수	A	B	C
트랜잭션 평균수행시간 (연산작업+로그디스크출력)	41.28798	9.329702	13.884037
단위시간생산량	0.2241198	0.789853	0.580894
연산작업	4.0395	1.267731	2.017691
로그디스크출력	37.24829	8.061926	11.865979
그룹완료간격	-	6.707532	17.361244
그룹완료횟수	-	9438	4958
강제그룹완료횟수	-	1	4958

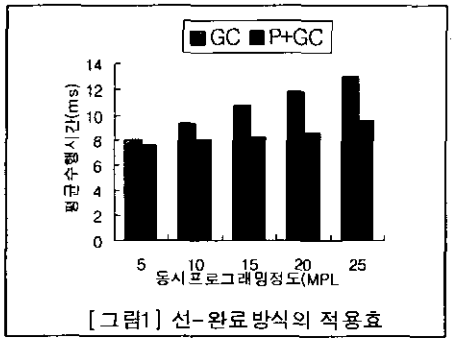
A:그룹완료 미시행, B:그룹완료시행, 전역로그버퍼=110바이트, C:그룹완료시행, 전역로그버퍼=512바이트

[표3] 그룹 완료 방식의 성능 분석(단위: ms)

A의 경우 연산작업중인 트랜잭션들은 모든 자료가 메인 메모리에 있으므로 연산작업을 매우 신속하게 진행시켜 끊임없이 로그의 디스크 출력 요청을 디스크 큐에 삽입시키나, 디스크 컨트롤러가 이를 하나씩 수행하기 때문에 이 작업이 병목이 되어 "로그디스크출력"시간이 급증하게 된다. 이로 인해 트랜잭션이 완료될 때까지 로크를 가지고 있는 시간도 증가하여 결국 "연산작업"시간도 B에 비해 증가하게 된다. B의 경우 "로그디스크출력"시간이 최고 4.5배 감소되어 시스템 전체의 성능과 개별적인 트랜잭션의 수행시간이 효과적으로 개선된다.

자료경쟁수준을 감안하지 않은 대형 크기의 로그 버퍼를 가지는 C의 경우 '강제' 그룹 완료만이 100%을 차지한다. 이는 "그룹완료간격"을 디스크 출력주기(5ms)보다 길게 하여 디스크 컨트롤러가 대기상태가 되는 시간을 많게 한다. 이는 B보다 못한 "로그디스크출력"시간을 가지는 것으로 보아 병목이 되는 디스크 컨트롤러를 최대한 바쁘게 작업하게 하는 것이 시스템 성능을 최상으로 만든다는 것을 알 수 있다. 하지만 최악의 경우 적절하지 못한 전역 로그 버퍼크기(또는 그룹완료 주기)를 가지더라도 그룹 완료방식을 적용하는 것이 훨씬 향상된 시스템 성능을 가짐을 알 수 있다.

●모의 실험 2 : 선-완료 방식의 효과



[그림1] 선-완료방식의 적용효

먼저 *MMDB*시스템에 선-완료 방식만을 적용했을 경우 성능향상이 있는 지 모의 실험을 실시하였다. 그 결과 아무런 성능의 변화가 없었는데 그 이유는 선-완료를 적용하면 연산작업은 이전보다 더욱 신속하게 이루어지나, 이는 결국 병목이 되는 디스크 출력 큐의 길이만 길게 하는 역효과를 가져와 성능향상으로 이어지지 못하기 때문이다. 그러나, 선-완료 방식을 그룹 완료방식과 병행할 경우 이는 로킹을 먼저 해제한 상태에서 디스크 출력을 그룹으로 하게 되어 시스템 성능에 상승작용을 하게 되고, 이러한 상승작용은 자료경쟁수준이 클수록 더 효과가 있음을 [그림1]에서 확인할 수 있다.

5. 결론

모든 자료가 메인 메모리에 상주하여 획기적인 트랜잭션 성능을 가져다주는 MMDB시스템에서의 좀 더 나은 성능을 위해 많은 연구가 진행중이며, 이중 평상시 성능에 병목이 되는 로깅 시간을 감소시키기 위한 완료 개선 분야에 있어서는 선-완료 방식, 그룹 완료 방식들이 대표적이다. 본 논문에서는 여러 자료경쟁수준에서의 다각적인 모의실험을 실시한 결과 선-완료 방식은 기존의 일반적인 데이터베이스 시스템에서는 성능 향상이 있을 수 있으나, MMDB시스템에서처럼 로깅을 위한 디스크 출력이 전체 수행시간의 90%이상을 차지하는 환경에서는 그 자체만을 적용시키면 신속하게 연산을 마친 트랜잭션들이 더욱 디스크 출력으로 몰리게 되어 결국 로깅의 병목 현상을 가중시켜 시스템 성능 향상으로 이어지지 못하며, 그룹 완료방식을 수행할 때 보조수단으로 병행할 때에만 성능에 도움을 줌을 보였다. 또 다른 개선 방안인 그룹 완료 방식은 자료경쟁수준에 비해 너무 큰 시스템 로그버퍼의 크기를 그룹 완료 시행 기준으로 하는 경우 교착 상태가 발생할 수 있으며 이는 선-완료방식을 병행하거나, 선-완료방식을 적용할 수 없는 경우 본 논문에서 제시한 방안을 적용함으로써 해결할 수 있다. 그리고, 그룹 완료 방식의 성능은 우려와는 달리 모의실험 결과 시스템 성능 향상뿐만 아니라 개별적인 트랜잭션의 수행 시간 또한 획기적으로 감소시키는 아주 효과적인 방안임이 증명되었다. 향후에는 그룹 완료방식을 분산 MMDB시스템에 적용했을 경우의 문제점 파악 및 시스템 성능에 대한 연구가 이루어져야 할 것이다.

6. 참고문헌

[Dew84] DeWitt. D.J.. Katz, R.H., Olken. F., Shapiro, L.D., Stonebraker, M. R., and Wood, D., "Implementation Techniques for Main Memory Database Systems." Proceedings of SIGMOD '84, June 1984, 1-8: also appears in SIGMOD Record Vol. 14. No. 2

[Elj95] Eljas Soisalon-Soininen and Tatu Ylonen, "Partial Strictness in Two-Phase Locking", ICDT 95

[Gar92] Garcia-Molina, Hector, "Main Memory Database Systems: An Overview", IEEE Trans. on Knowledge and Data engineering, Vol. 4, No. 6, pp.509-516, 1992

[Gaw85] Gawlick, Dieter, and David. Kinkade, "Varieties of Concurrency Control in IMS/VS FastPath", IEEE Database *Engineering, June 1985*

[Gru91] Gruenwald, Le., and Eich, Margaret H., "MMDB Reload Algorithms", In proc. of ACM SIGMOD, pp.397-405, 1991.

[Hec93] H. V. Jagadish, Avi Siberschatz, and S. Sudarshan, "Recovering from Main-Memory Lapses", In proc. of 19th *VLDB conference Dublin, Ireland,1993, pp 391-404*

[Ken90] Kenneth Salm, Hector Garcia-Molina, "System M : A transaction Processing Testbed for Memory Resident Data", IEEE Trans. on Knowledge and Data Engineering, Vol. 2, No. 1, pp161-172, March 1990

[Mar97] Margaret H. Dunham, Jun-Lin Lin, and Xi Li, "Fuzzy Checkpointing Alternatives for Main Memory Databases", Recovery Mechanism in Database Systems, edited by vijay kumar, Prentice-Hall, 1997

[Sch92] H.Schewetman, "CSIM user's guide for use with CSIM revision16", MCC, June, 1992

[Rob87] Robert Hagmann, "Reimplementing the Cedar File system Using Logging and Group Commit", Proceedings of 11th Symposium on Operating System Principles, Austin, TX, November 1987, 155-162.Published as Operating Systems Review 21,5 (November 1987)

[Ros95] Rosennblum M, Bugnion E, Herrod SA, Witchel E, Gupta A, "The Impact of Architectural Trends on Operating System Performance, in Proceedings of the 1995 Symposium on Operating Systems Principles, ACCM Press, pp 285-298

[Wee98] Wee Teck Ng, Peter M. Chen, "Integrating reliable memory in databases", International Journal on VLDB, Auguest 1998.