A Replication Design for Main Memory Database Systems
In-Seon Lee
Computer Information Processing Dept. of Shingung College

1. 서론
메인 메모리 데이터베이스(Main Memory Data Base: MMDB)시스템은 데이터베이스 시스템의 대부분 또는 모든 부분이 메인 메모리에 상주하는 데이터베이스 시스템으로, 트랜잭션 수행중 자료 읽어들임을 위한 디스크 액세스를 유발하지 않으므로 전체 시스템 성능을 크게 향상시킬 수 있다. 80대 중반에 일본의 2002년에 이 시스템이 소개된 후 현재까지 많은 연구가 이루어지고 있으며, 최근에는 빅 데이터의 가치가 높아지고, 비용화되면서 데이터의 실시간 조화, 고정된 금융, 증권, 통신 등의 분야에서 MMDB 시스템의 활용도가 가시화되고 있다. MMDB 시스템의 사용자들은 현재 대다수 트랜잭션 처리를 수행하고, 시스템 고장 등으로 인한 서비스 중단과 같은 걸함이 일어나지 않는 시스템에 대한 요구가 커져가고 있다. 이 요구들은 연산시키는 가장 적절한 해결책으로 이중화 시스템(replication system)를 들 수 있다. 그러나, 기존의 트랜잭션 기반 분산 데이터베이스 시스템을 위한 이중화 기법을 대상으로 MMDB에 적용하기 위해서는 많은 제한과 문제점이 있으며, 본 논문에서는 먼저 MMDB 시스템의 구조에 대해 고찰하고, 디스크 기반 분산 데이터베이스 시스템에 적용하고 있는 이중화 기법들에 대한 정리와 분석한다.

2.  메인 메모리 데이터 베이스 시스템에 대한 연구
● 동시성 제어
MMDB 시스템에서 한계 트랜잭션의 수행과정을 살펴보면 트랜잭션 선언시점에 필요한 모든 데이터가 메인 메모리에 상주하므로 메인 메모리 데이터베이스 시스템(Main Memory Database System: MMDB System)은 트랜잭션 수행중 자료 읽어들임을 위한 디스크 액세스를 유발하지 않으므로 전체 시스템 성능을 크게 향상시킬 수 있다. 80대 중반에 일본의 2002년에 이 시스템이 소개된 후 현재까지 많은 연구가 이루어지고 있으며, 최근에는 빅 데이터의 가치가 높아지고, 비용화되면서 데이터의 실시간 조화, 고정된 금융, 증권, 통신 등의 분야에서 MMDB 시스템의 활용도가 가시화되고 있다. MMDB 시스템의 사용자들은 현재 대다수 트랜잭션 처리를 수행하고, 시스템 고장 등으로 인한 서비스 중단과 같은 걸함이 일어나지 않는 시스템에 대한 요구도 커져가고 있다. 이 요구들은 연산시키는 가장 적절한 해결책으로 이중화 시스템(replication system)를 들 수 있다. 그러나, 기존의 트랜잭션 기반 분산 데이터베이스 시스템을 위한 이중화 기법을 대상으로 MMDB에 적용하기 위해서는 많은 제한과 문제점이 있으며, 본 논문에서는 먼저 MMDB 시스템의 구조에 대해 고찰하고, 디스크 기반 분산 데이터베이스 시스템에 적용하고 있는 이중화 기법들에 대한 정리와 분석한다.

본 논문에서는 먼저 MMDB 시스템의 구조에 대해 고찰하고, 디스크 기반 분산 데이터베이스 시스템에 적용하고 있는 이중화 기법들에 대한 정리와 분석한다. 이 분석을 통해 마치 MMDB시스템에 이중화 기법을 적용하기 위해서 고려해야 할 점들을 정리하고, 제시한 고려 사항들을 모두 만족하는 MMDB 이중화 시스템을 설계하였다.
3. 디스크 기반 분산 데이터베이스 시스템에서의 이중화 기법

일반적으로 데이터베이스 시스템은 성능향상을 위해 이중화기법을 사용하며, [GHO+96]에서 데이터베이스 이중화 기법을 [1]과 같이 두 가지 요소를 가지고 분류하였다. 첫 번째 요소는 이중화가 이루어지는 시기에 "Eager 기법"이 복잡하고 모든 데이터들의 복제가 하나의 트랜잭션 전체로 이루어진다. 두 번째로 "Lazy 기법"은 트랜잭션 수행 중에는 한 지역의 데이터만 갱신하고, 이 갱신 트랜잭션이 완료된 후 다른 지역들에 있는 복사 데이터의 갱신이 이루어진다. 전자는 일관성이 확실하게 기여하지만, 부수적인 복잡성이 생기고 트랜잭션 동합시간이 길어지는 단점이 있다. 후자는 다양한 성능 향상을 기질 수 있으며, 여러 지역에 있는 복사 데이터간의 복합 처리가 발생한다. 이중화 분류 두 번째 요소는 갱신의 이동 방향을 "프로세스"기법은 "primary"라고 불리는 데이터베이스 멤버가 먼저 갱신되며, "primary"가 있는 서버가 고정되어 전각 시스템이 중앙이고, 이 지역의 성능에 영향을 끼치는 단점이 있다. "Update Everywhere"기법은 복사되어 있는 데이터 갱신의 대상이 될 수 있어 메모리 속도가 빨라지며, 복사 데이터간의 조정 작업을 줄일 수 있다.[WPS+00]

![update propagation](image)

### Eager Update Everywhere Replication with Distributed Lock

<table>
<thead>
<tr>
<th>Phase 1:</th>
<th>Client Request to any replica</th>
<th>Phase 2:</th>
<th>Server Coordination Distributed Lock</th>
<th>Phase 3:</th>
<th>Execution all update</th>
<th>Phase 4:</th>
<th>Agreement Coordination 2PC</th>
<th>Phase 5:</th>
<th>Client Response from local</th>
</tr>
</thead>
</table>

### Lazy Primary Copy

<table>
<thead>
<tr>
<th>Phase 1:</th>
<th>Client Request to Primary</th>
<th>Phase 2:</th>
<th>Server Coordination Primary Copy</th>
<th>Phase 3:</th>
<th>Execution Primary update</th>
<th>Phase 4:</th>
<th>Agreement Coordination from primary</th>
<th>Phase 5:</th>
<th>Client Response from other</th>
</tr>
</thead>
</table>

### Lazy Update Everywhere

<table>
<thead>
<tr>
<th>Phase 1:</th>
<th>Client Request to any replica</th>
<th>Phase 2:</th>
<th>Execution 1 o a l update</th>
<th>Phase 3:</th>
<th>Client Response from local</th>
<th>Phase 4:</th>
<th>Agreement Coordination apply other replicas &amp; reconciliation</th>
<th>Phase 5:</th>
<th>Agreement Coordination from primary</th>
</tr>
</thead>
</table>

4. MMDB를 위한 이중화 시스템의 설계

4.1 단일 MMDB시스템의 구조

MMDB시스템은 대표적으로 많은 트랜잭션들이 발생하지만 많은 갱신이 생겨나고, 대부분 갱신의 주의 트랜잭션들로 구분되며, 복제구성과 신속한 트랜잭션 동합시간을 요구하는 시스템에 많이 사용되고 있으며, 이 요구사항에 맞추기 위하여 아래의 구조를 갖추는 것이다.

- 동상성 방식: 모든 트랜잭션이 DB 전체에 대해 공유(share/exclusive)로 로크를 가지고 순차수행을 하며, 디스크 로크전에 로크를 해제하는 "사용완료" 방식을 한다.
- 로그의 효율성: 디스크 촉력 지속도에 맞추어 그룹화를 수행하는 "디스크 그룹 완료"방식을 제공하여 디스크 활동을 최소화하고 한다.
- 체크포인팅 방식: 트랜잭션 성능에 영향을 미치지 않으며, 신속한 복구를 위해 "평행-갱신"을 하는 "지키 체크 포인팅"방식을 사용한다.

4.2 디스크기반 데이터베이스 시스템에서의 이중화 기법

- "기계임" DB는 디스크에 있어 효과가 낮기 때문에 하나의 갱신 트랜잭션 병합 안에서 모든 복사버전의 로그를 이루어내야하기 때문에 DBMS에 큰 2PC(Two-Phase Commit 방식)을 이용하여 처리안정성을 지킨다.
- "Eager update everywhere"방식은 로그를 사용하는 것을 복제하고 있는 복제 상태를 유발시키며, 전체 시스템 성능을 크게 저하시키며, Atomic Broadcasting은 "전체 손실(total-ordering)"을 구현하여야 한다.
- "Lazy"방식은 복제되어 있는 데이터의 원본을 복제하지 못한다.
4.3 MMDB이중화 시스템 설계

● 기존 동작 원리
MMDBS시스템을 적용하는 응용분야에서 이중화기법의 적용을 목적으로
온 또는 서버가 시스템 동작 중임으로 중단되지 않는 한 계속 진행하게 하고, 대용량을 발생하게 하기 위해, 현재는 트랜잭션
의 정확도를 위해, 응용에 동작 시간을 보장하기 위해서는, 또한 개선으로
온의 전체 시스템 성능이 저하되면서도 안된다. 한편, 유의어 복제의
표현가 가능하게 하기 위해, "Eager""기술을 사용하면서, 개선 성능을
발전할 수 있는 기술을 개발하고 있다. 이는 기존 DB시스템에서는
하의 트랜잭션 안에서 모든 복제의 정해진 로그가 이루어지게 하기
위해 2PC를 사용하였으나, MMDB의 경우 DB가 메인메모리에 상
주하게 생성이 시작에 지역으로 로깅하고 다른 지역으로부터는
"Lock Ack"를 받으면 바로 트랜잭션을 완료한다. 다른 접근은 생성
을 위해 기존에 "독점 로그"를 획득하기 위해 "Lock Ack"를 보고,
로그는 이후에 진행된다. 이러한 새로운 "Eager""기법과 "정격전
Eager"방식과 같은 다른 지역에서 생성의 정확한 값을 구할 수 있
다. "Primary"기법에는 기존이 하나의 서버에서 발생하게 시스
템 성능 복제가 발생하지나, 이 "primary"서버가 고장이 난 경우 시
스템 전체에서 생성이 이루어질 수 없다는 단점이 있다. 그러나,
"Update everywhere"기법은 이런 복제본도 생성할 수 있게 하여
"Eager"인 경우 오류 상태가 발생하게 되고, "Lazy"인 경우 비정상
(horizontal - non - serializability)이 발생하게 되어 이로 유발하는 "외부 그래프
(external graph)"가 발생해 관점 자체에 문제가 발생한다. "primary"방식의
외에 같은 단점을 초래 기기 때문에 데이터의 "primary"를 따르게 하여
생성 트랜잭션을 로징하고, 데이터의 "primary"에 대한 데이터를 변경
하려면 데이터를 위치가 처리하는 간접 트랜잭션에 있어 서버"primary"
로 가진 데이터의 생성만 있기 한다.

DB전체 "공유 로크"를 얻고 조작 작업은 수행한다. 모든 연산자들이 낮은 판단을 시작한다. ("공유 로크"로 해제
하고, 실행되는 간접 트랜잭션과의 간섭성을 기기 위해 "Mutex Array"
"로크를 순차적으로 해제한 후 해제한다)

● 시스템 예제 조각 및 복무 후 과정
하의 하드웨어는 라이선스-에도 예제 어느 로컬 DB가 고장나왔다를 알게 되면 자신들이 "Primary"로 동작하게 하기 위해 데이터에 대해 이후 발생한 간접 트랜잭션을 수행하고, 고장이 난 서버가 복구되면, 그
 데이터에 대해서 다시 "Secondary"로 동작한다.

5. 결론
모든 자료가 메인 메모리에 상주하여 확장적인 트랜잭션 성능을 가
장하는 MMDB시스템에서도 더 나은 성능과 무결점성을 위해 이중화
기법이 도입되고 있다. MMDB시스템에 적용한 이중화 기법을 설계
하기 전에 MMDB시스템이 적용하고 있는 분야의 트랜잭션 특성과
함께 이중화 설계에 영향을 미칠 수 있는 MMDB시스템 구조에서
지금까지 이루어진 연구를 구현 시험 결과를 살펴보았다. 그리고, 지금
까지 SAN 기반 분산 데이터베이스 시스템에 적용되고 있는 이중화
기술들의 종류와 그 동작 과정, 장단점을 함께 해석하였다짐을 정
의하였다. 이러한 고장은 트랜잭션 없는 현재 MMDB시스템이 적용되
고 있는 분야의 트랜잭션 처리에 어떠한 영향이나 수치가 가지지 않
으면서 트랜잭션 처리 능력을 배가시킬 수 있는 이중화 시스템의 기
본 동작 원리와 트랜잭션 종류별 사용방법도 설계하였다. 한편에는
설계한 이중화 기법들을 실제로 MMDB시스템에 구현하여 어느 정도
성능 향상을 가져다주는지 구체적으로 다양한 설명을 할 계획이다.

참고문헌


