비공유 공간 데이터베이스 클러스터에서 고가용성을 위한 병렬 회복 기법

유병섭*, 장용일*, 이순조**, 배해영*
*인하대학교 전자계산공학과, **서원대학교 컴퓨터 교육과
e-mail : subi@dblab.inha.ac.kr

The Parallel Recovery Method for High Availability in Shared-Nothing Spatial Database Cluster

Byeong-Seob You*, Yong-Il Jang*, Sun-Jo Lee**, Hae-Young Bae*
*Dept. of Computer Science and Engineering, Inha University
**Dept. of Computer Education, Seowon University

요 약

최근 인터넷과 모바일 시스템이 급속히 발달함에 따라 이를 통하여 지리정보와 같은 공간데이터를 제공하는 서비스가 증가하였다. 이는 대용량 데이터에 대한 관리 및 빠른 처리와 급증하는 사용자에 대한 높은 동시처리량 및 높은 안정성을 요구하였고, 이를 해결하기 위하여 비공유 공간 데이터베이스 클러스터가 개발되었다. 비공유 공간 데이터베이스 클러스터는 고가용성을 위한 구조로서 문제가 발생할 경우 다른 백업노드가 대신하여 서비스를 지속시킨다. 그러나 기존의 비공유 공간 데이터베이스 클러스터는 클러스터 구성에 대한 회복을 위하여 로그를 계속 유지하므로 로그를 남기기 위해 보통의 질의처리 성능이 저하되었으며 로그 유지를 위한 비용이 증가하였다. 또한 노드단위의 로그를 갖기 때문에 클러스터 구성에 대한 회복이 직렬적으로 이루어져 고가용성을 위한 빠른 회복이 불가능 하였다.

따라서 본 논문에서는 비공유 공간 데이터베이스 클러스터에서 고가용성을 위한 병렬 회복 기법을 제안한다. 이를 위해 클러스터 구성에 대한 회복을 위한 클러스터 로그를 정의한다. 정의된 클러스터 로그는 마스터 테이블이 존재하는 노드에서 그룹내 다른 노드가 정지된 것을 감지할 때 남기기 시작한다. 정지된 노드는 자체회복을 마친 후 클러스터 구성에 대한 회복을 하는 단계에서 존재하는 복제본 테이블 각각에 대한 클러스터 로그를 병렬적으로 받아 회복을 한다. 따라서 정지된 노드가 발생할 경우에만 클러스터 로그를 남기므로 보통의 질의처리의 성능 저하가 없고 클러스터 로그 유지 비용이 적으며, 클러스터 구성에 대한 회복시 테이블단위의 병렬적인 회복으로 대용량 데이터인 공간데이터에 대해 빠르게 회복할 수 있어 가용성을 향상시킨다.

1. 서론

최근 PDA와 같은 모바일 장치가 발달하여 널리 보급되고 인터넷이 대중화됨에 따라 이를 통하여 지리정보와 같은 공간데이터를 제공하는 서비스가 증가하였다. 그러나 공간데이터의 특성상 대용량의 데이터를 관리하고 빠르게 처리할 수 있는 데이터베이스가 요구되었으며, 또한 이러한 서비스를 이용하는 사용자의 급증에 따른 높은 동시처리량과 안정성이 요구되었다[1]. 비공유 공간 데이터베이스 클러스터는 독립적으로 질의를 처리할 수 있는 노드들을 고속의 네트워크로 연결하여 구성한 것으로 여러 노드에서 동시에 질의를 처리함으로써 동시처리량이 증가하였고, 대용량의 공간데이터를 분할하여 관리하는 분할정책을 제공함으로써 대용량의 데이터를 효율적으로 관리하고 빠르게 처리할 수 있게 되었으며, 분할된 데이터에 대하여 복제본을 갖는 복제정책을 제공함으로써 서비스를 하는 노드가 정지되어도 복제본을 갖는 노드를 통해 지속적인 서비스가 가능한 안정성을 제공하게 되었다[2]. 특히 복제정책은 하나의 노드가 정지할 경우 복제본을 갖는 노드에서 서비스를 지속하며 그동안 정지한 노드는 회복을 하는 고가용성을 제공하게 되었다[3,4]. 그러나 기존의 비공유 공간 데이터베이스 클러스터는 클러스터 구성에 대한 회복을 위하여 로그를 계속 유지하므로 로그를 남기기 위해 보통의 질의처리 성능이 저하되었으며 로그 유지를 위한 비용이 증가하였다. 또한 노드단위의 로그를 갖기 때문에 클러스터 구성에 대한 회복이 직렬적으로 이루어져 고가용성을 위한 빠른 회복이 불가능 하였다.

따라서 본 논문[1]에서는 비공유 공간 데이터베이스 클러스터에서 고가용성을 위한 병렬 회복 기법을 제안한다. 이를 위해 클러스터 구성에 대한 회복을 위한 클러스터 로그를 정의한다. 정의된 클러스터 로그는 마스터 테이블이 존재하는 노드에서 그룹내 다른 노드가 정지된 것을 감지할 때 남기기 시작한다. 따라서, 보통의 질의처리에서는 클러스터 로그 기록을 위한 부하가 없으며, 각 테이블에 대하여 마스터 테이블이 독립적으로 클러스터 로그를 유지하므로 특정 노드에서 모든 테이블에 대해 로그를 유지해

1) 본 연구는 정보통신부의 대학 S/W 연구센터 지원사업의 연구 결과임

야 하는 부담을 덜게 된다. 또한 제안 기법은 정지된 노드가 클러스터 구성에 대한 회복을 하는 단계에서 존재하는 복제본 테이블 각각에 대한 클러스터 로그를 각 테이블의 마스터가 존재하는 그룹내 다른 노드로부터 병렬적으로 받아 회복 한다. 따라서 각 테이블이 독립적으로 회복하는 병렬 회복이 가능하여 대용량 데이터인 공간데이터에 대해서도 빠르게 회복할 수 있어 가용성을 향상시킨다.

본 논문의 구성은 다음과 같다. 2장에서는 관련연구를 다루며, 3장에서는 제안기법의 기반이 되는 시스템을 설명한다. 4장에서는 제안기법을 설명하며, 5장에서 결론을 맺는다.

2. 관련연구

ClustRa는 메인메모리 기반의 데이터베이스 클러스터로 비공간 데이터에 대해 클러스터 구성으로 서비스한다. 이는 독립적인 질의처리가 가능한 노드들을 고속의 네트워크로 연결하고, 각 노드들은 2개가 하나의 그룹이 되어 같은 데이터베이스를 유지한다. 따라서 하나의 마스터에 하나의 백업을 갖게 되고, 최악의 경우 두 대가 모두 정지되었을 경우 서비스가 불가능하게 된다[5].

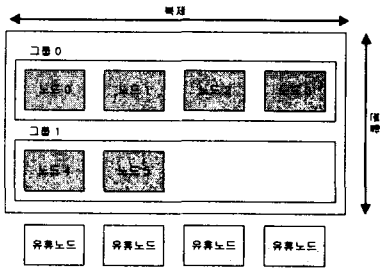
ClustRa는 노드 자체의 회복을 위한 로그 이외에 백업 전파를 위한 글로벌로그(Global Log)를 갖는다. 백업 전파는 글로벌 로그를 백업 노드에 전송함으로써 이루어지며, 마스터에서는 글로벌로그를 백업 노드에 전송하기 위하여 하나의 큐를 유지한다. 즉, 같은 노드내에 존재하는 모든 마스터 테이블들은 갱신질의 처리시 하나의 순차적인 번호를 갖는 글로벌로그를 생성하고, 이를 큐에 넣게 된다. 따라서 모든 마스터 테이블들이 글로벌로그를 위한 순차번호를 생성할 때 동시성제어가 필요하며 이를 위해 락(Lock)을 사용하므로 성능이 저하된다[6,7].

ClustRa에서는 백업 노드가 정지되었을 경우 마스터 노드에서는 글로벌로그를 계속 큐에 유지하게 된다. 그리고 백업 노드가 자체 회복을 마치고 나면 큐에 유지한 글로벌로그를 전송함으로써 클러스터 구성에 대한 회복을 하게 된다. 그러나 백업 노드의 자체회복이 오래 걸릴 경우 마스터 노드의 큐의 크기가 증가하게 되고, 클러스터 구성에 대한 회복시 큐에 유지한 로그를 순차적으로 보내게 되므로 회복시간이 느려지게 된다[6,7].

ClustRa는 위의 락에 의한 성능 저하와 백업노드의 오랜 자체 회복에 따른 큐의 증가에 관한 문제를 디스크기반 보다 훨씬 빠른 메인메모리 기반의 시스템으로 해결하였다. 그러나 공간 데이터의 경우 비공간 데이터와 달리 데이터의 크기가 크고, 검색에 대한 연산이 빈번히 일어나게 되므로 디스크를 기반으로 하며, 하나 이상의 복제본을 유지할 수 있는 클러스터 시스템을 필요로 한다. 따라서 공간 데이터를 관리하기 위한 디스크 기반의 비공유 공간 데이터베이스 클러스터가 필요하며, 백업 전파와 회복에 대해 디스크 기반에 알맞은 방법이 필요하다.

3. 시스템 개요

제안 기법의 기반이 되는 시스템은 비공유 공간 데이터베이스 클러스터인 GMS/Cluster이다. GMS/Cluster의 클러스터 구성은 [그림 1]와 같다.



[그림 1] GMS/Cluster의 클러스터 구성

각 노드는 독립적인 질의 처리가 가능한 GMS/Cluster 시스템으로 구성되며, 2개 이상의 노드들이 하나의 그룹을 구성한다. 그룹 내에 존재하는 각 테이블은 하나의 노드에 마스터가 존재하고 나머지 노드들에 해당 테이블의 복제본들이 존재하게 된다. 또한, 하나의 그룹에는 여러 개의 테이블이 존재할 수 있고, 각 테이블에 대한 마스터 노드의 위치는 테이블 생성시 사용자가 지정할 수 있으며, 분할된 마스터들은 그룹간 구분을 위한 고유번호를 갖는다. 분할정책은 그룹 사이에 적용되며, 복제정책은 그룹 내에 적용된다. 각 노드들은 클러스터 구성을 위해 확장된 메타정보를 유지하며, 노드간 정보 전송을 위해 고속의 네트워크로 연결되어 있다. GMS/Cluster는 급증하는 사용자 질의에 대하여 유연하게 대처하여 안정적인 서비스가 가능한 온-라인 확장을 제공하며, 이를 위해 서비스를 하지 않는 유휴 노드들이 존재한다[2].

4. 클러스터 로그를 이용한 병렬 회복 기법

4.1 클러스터 로그

클러스터 로그는 클러스터 구성에 대한 회복을 위해 존재하는 것으로 백업 전파에 대한 내용을 갖는 논리적 구조를 갖는다. 클러스터 로그의 구조는 [그림 2]와 같다.

로그 번호	전송내용

[그림 2] 클러스터 로그의 구조

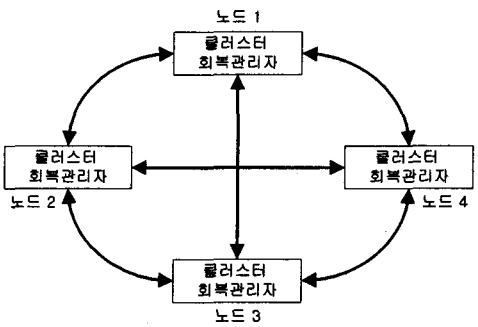
로그번호는 로그에 대한 순차적인 번호로써 유일성이 보장된다. 전송내용은 백업에 전파하는 내용을 말하며, 공간 데이터의 경우 크기가 커서 변경된 데이터를 전송할 경우 전송할 데이터량이 커지고 이에 따라 성능이 저하되기 때문에 SQL 구문으로 이루어진다.

클러스터 로그는 노드에 존재하는 마스터 테이블마다 하나씩 독립적으로 관리된다. 즉, 노드에 3개의 마스터 테이블이 존재할 경우 3개의 클러스터 로그가 존재하게 된다. 따라서 노드에 존재하는 모든 마스터 테이블들이 유일한 로그번호를 각각 독립적으로 관리하게 되므로 기존 기법처럼 락을 필요로 하지 않으므로 성능이 향상된다.

클러스터 로그는 기존의 로그와는 달리 클러스터 구성에 대한 회복에만 필요하므로 계속적인 유지가 필요없다. 따라서 제안 기법에서는 그룹내의 한 노드가 정지되었을때 나머지 노드들이 이를 감지한 시점부터 클러스터 로그를 파일에 남기게 된다. 따라서 정지된 노드가 없을 경우 클러스터 로그를 남기지 않으므로 질의처리에 대한 성능이 향상된다.

4.2 클러스터 회복관리자

클러스터 회복관리자는 클러스터 구성에 대한 회복을 담당하는 것으로 노드 자체의 회복을 위한 회복관리자와 구별된다. 클러스터 회복관리자의 구성은 [그림 3]과 같으며, 주요 역할은 다음과 같다.



[그림 3] 클러스터 회복관리자의 구성

첫째, 그룹내 노드의 정지를 감지한다. 클러스터 회복관리자는 각 노드에 존재하며, 그룹내 노드들 사이에 망형 연결을 구축하여 서로 다른 시스템의 상태를 감시한다.
예를 들어, [그림 3]에서 노드 3이 정지된다면 나머지 노드들은 노드 3과 연결된 네트워크가 단절되어 노드 3이 정지되었음을 알게 된다.

둘째, 클러스터 로그를 유지한다. 클러스터 회복관리자는 그룹 내 특정 노드가 정지된 것을 감지할 경우 자신의 노드에 존재하는 모든 마스터 테이블에 대하여 각각 독립적인 클러스터 로그를 생성하고 이를 유지한다.

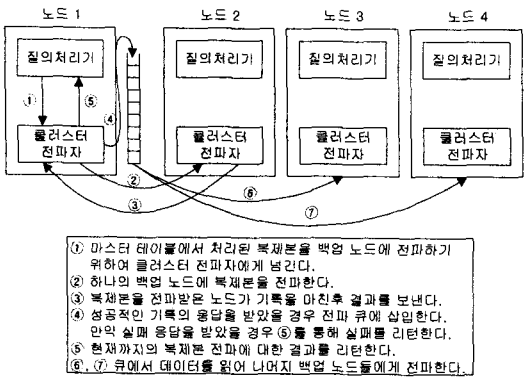
셋째, 클러스터 구성의 회복을 담당한다. 클러스터 회복관리자는 정지된 노드가 자체회복을 마친 후 클러스터 구성의 회복을 담당한다. 이를 위해 회복하는 노드는 그룹내 다른 노드의 클러스터 회복관리자에 연결을 시도하며, 연결이 되면 각 노드로부터 클러스터 로그를 전송받아 클러스터 구성에 대한 회복을 한다.

4.3 클러스터 전파자

클러스터 전파자는 클러스터 구성에서 지연전송(Lazy Master)을 이용한 복사본 전파를 담당하는 것으로 각 노드에 존재한다. 복제본의 전파방법에는 2PC기법과 지연전송기법이 존재하는데, 2PC 기법은 모든 노드에 복제본을 완벽하게 전파한 후 결과를 보내는 것으로 안정적이지만 응답시간이 느려지고, 지연전송기법은 하나의 복제본에만 전파하면 결과를 보내고 이후에 나머지 복제본을 전파하는 것으로 느슨한 일관성을 갖지만 응답시간이 빠르다. 따라서 사용자의 질의 처리 요구를 신속하게 처리하려는 비공유 공간 데이터베이스 클러스터에서는 지연전송기법을 사용한다.

클러스터 전파자는 지연전송을 위하여 전파 큐(Propa -gation Queue)를 유지한다. 이는 복제본 전파를 위한 큐로써 하나의 복제본을 전파한 후 해당 내용을 전파 큐에 넣고 결과를 보낸다. 이를 위하여 하나의 노드는 완벽한 복제본 유지를 보장하는 하나의 백업 노드를 갖는다. 클러스터 전파자에는 큐에 담긴 내용을 읽어 나머지 노드들에게 전송하는 것을 담당하는 컴포넌트가 존재한다. 따라서 큐에 담긴 내용은 이 컴포넌트를 통하여 바로 나머지 노드들에게 전송된다.

[그림 4]는 클러스터 구성에서 질의를 처리하는 과정을 나타낸 것이다.

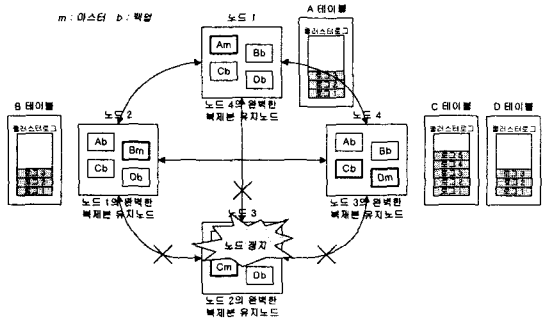


[그림 4] 클러스터 구성에서의 질의처리 과정

4.4 노드 정지시 서비스

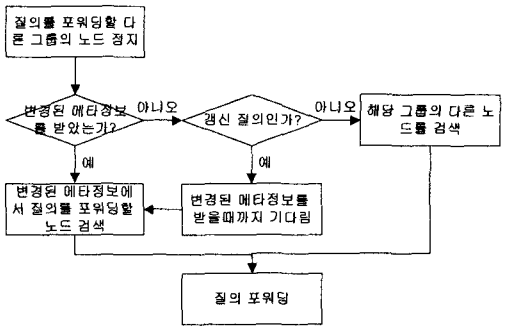
비공유 공간 데이터베이스 클러스터는 하나의 노드가 중지되어도 계속적인 서비스가 가능한 시스템이다. 여기서는 노드가 정지되었을 경우 서비스의 유지와 나머지 노드들의 동작에 대해 알아본다.

클러스터의 구성 중 하나의 특정 노드가 정지되었을 경우 그룹 내 모든 노드들이 클러스터 회복관리자를 통해 노드가 정지되었음을 알게 된다. 이때 클러스터 회복관리자는 메타정보를 통하여 정지된 노드에 존재하는 마스터 테이블을 검색하게 된다. 만약 마스터 테이블이 존재한다면 정지된 노드의 완벽한 복제본을 보장하는 백업 노드에서 해당 테이블의 복제본이 마스터로 대체되어 서비스를 계속 하게 된다. 이때 변경된 사항에 대하여 모든 그룹의 모든 노드에게 변경된 메타정보를 전송한다. 대체 마스터가 결정되었을 경우 그룹내 나머지 노드들은 각 노드에 존재하는 마스터 테이블에 대한 클러스터 로그를 유지하게 된다. 따라서 나머지 노드들은 마스터 테이블이 질의를 처리한 후 하나의 복제본을 완벽히 전파한 후 해당 질의에 대해 전파 큐에 넣고 클러스터 로그를 남기게 된다. 이는 정지된 노드가 없을 경우 클러스터 로그를 기록하지 않으므로 빠른 질의처리를 할 수 있다. [그림 5]는 이를 그림으로 나타낸 것이다.



[그림 5] 그룹내 노드 정지시 다른 노드들의 구성

클러스터 회복관리자의 연결은 각 그룹내의 노드간의 망형 연결을 구축하므로 해당 다른 그룹의 노드가 정지되었을 경우 이를 바로 알지 못한다. 이를 알기 위해서는 질의 전파시 감지하는 방법과 변경된 메타정보로 감지하는 방법이 있다. 전자는 다른 그룹의 노드들이 노드가 정지된 시점에서 정지된 노드에 대한 변경된 메타정보 받는 시점 사이에 질의를 포워딩(Forwarding)할 경우 발생한다. 이때 해당 질의가 갱신 질의일 경우 마스터 테이블에 전송해야 하므로 변경된 메타정보를 받을때까지 기다리게 된다. 그렇지 않고 검색 질의일 경우 해당 그룹의 다른 노드를 선택하여 질의를 포워딩하게 된다. 후자는 정지된 노드가 존재하는 그룹의 클러스터 회복관리자가 정지된 노드에 속하는 마스터 테이블들의 대체 마스터 노드를 결정한 후 메타정보를 변경하고 이를 전파할 경우 발생한다. 이때에는 변경된 메타정보를 이용하므로 질의의 포워딩이 정상적으로 이루어질 수 있다. 이를 순서도로 나타내면 [순서도 1]과 같다.



[순서도 1] 다른 그룹의 노드 정지시 질의 포워딩

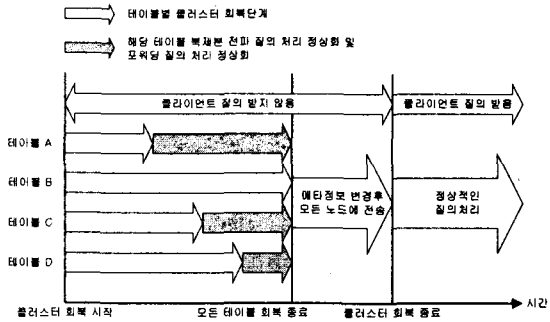
4.5 병렬 회복 기법

노드가 정지된 이후 정지된 노드에 대한 회복이 있게 되는데 여기에는 두 가지가 존재하게 된다. 첫 번째는 정지된 노드가 회복되어 다시 그룹에 참여하는 경우 이고, 두 번째는 유휴노드가 정지된 노드 대신 그룹에 참여하는 경우 이다.

4.5.1 정지된 노드가 회복되어 다시 그룹에 참여하는 경우

정지된 노드가 회복되는 경우 먼저 자체회복을 하게 된다. 이때 노드가 가지는 자체로그를 이용하여 회복을 하게 된다. 자체회복이 끝나게 되면 노드의 클러스터 회복관리자가 그룹내 다른 노드의 클러스터 회복관리자에게 연결한다. 연결 후 클러스터 로그의 전송을 요청하고, 전송받은 로그를 이용하여 클러스터 구성의 회복을 하게 된다. 이때 클러스터 로그는 각 테이블 별로 병렬적으로 전송 받으며, 모든 회복이 끝난 테이블은 포워딩 질의를 처리할 수 있게 된다. 병렬적인 클러스터 구성의 회복에 대하여 모든 테이블이 회복을 마치면 노드가 정지되기 이전의 메타정

보로 바꾸며 이를 전파한다. 그리고 클라이언트의 질의를 받으며 정상적인 노드로 되게 된다. 한편 정지된 노드의 연결을 받은 다른 노드의 클러스터 회복관리자는 연결 후 클러스터 로그의 요청에 대하여 자신이 갖는 클러스터 로그를 전송한다. 이때 클러스터 로그가 둘 이상일 경우 각각의 로그를 독립적으로 병렬로 전송하게 된다. 이를 그림으로 나타내면 [그림 6]과 같다.



[그림 6] 테이블별 회복에 따른 처리 과정

클러스터 구성의 회복은 각 테이블별로 독립적인 회복이 병렬적으로 이루어진다. 즉, 회복노드가 속하는 그룹에 테이블이 n개가 존재할 경우 n개의 클러스터 로그가 존재하며 각각 동시에 회복노드로 전송되어 클러스터 구성의 회복을 하게 된다. 이때 회복하는 동안에 복제본 전파 질의가 계속 들어오기 때문에 클러스터 로그가 계속 증가되어 회복이 느려질 수 있게 된다. 이러한 문제를 해결하기 위하여 회복 시작 후에는 복제본 전파 질의를 클러스터 로그로 남기지 않고 회복노드로 보내게 된다. 그러면 회복노드는 해당 테이블의 회복이 끝날때까지 질의를 대기한 후 회복이 끝나면 수행하게 된다. 이를 알고리즘으로 표현하면 다음 [알고리즘 1]과 같다.

1. 정지된 노드의 회복을 감지한다.
2. 복제본 전파 질의에 대하여 클러스터 로그를 남기지 않고 해당 노드로 질의를 전송한다.
3. 회복노드에게 클러스터 로그의 전송 시작을 알린다.
4. 회복노드에게 클러스터 로그를 전송한다.
5. 클러스터 로그의 전송이 모두 끝나면 로그 전송이 끝남을 알린다.
6. 클러스터 로그를 삭제하고 정지되었던 노드에 대해 정상적인 질의 처리를 시작한다.

a) 클러스터 로그를 갖는 노드의 동작

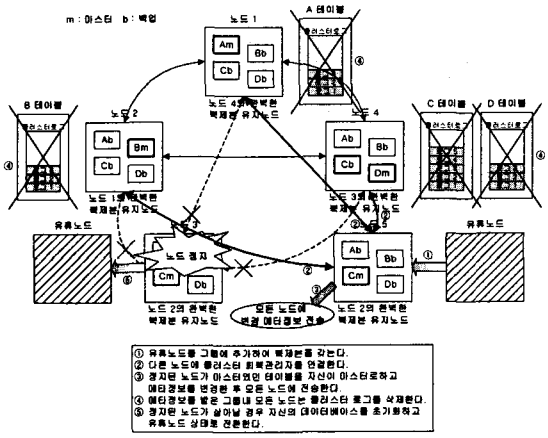
1. 다른 노드에게 클러스터 로그를 요청한다.
2. 전송받는 복제본 질의에 대하여 해당 테이블의 회복이 끝날때까지 처리를 중지한다.
3. 클러스터 로그의 시작 메시지를 받는다.
4. 클러스터 로그를 이용하여 해당 테이블의 클러스터 구성의 회복을 한다.
5. 클러스터 로그의 끝 메시지를 받는다.
6. 중지하였던 복제본 질의에 대하여 처리를 시작한다.
7. 해당 테이블에 대하여 정상적인 질의처리를 시작한다.

b) 회복노드의 동작

[알고리즘 1] 클러스터 구성에 대한 각 테이블별 회복

4.5.2 유휴노드가 정지된 노드 대신 그룹에 참여하는 경우

유휴노드가 정지된 노드 대신 그룹에 참여하는 경우는 기존의 그룹내 노드추가를 이용한다[2]. 즉, 정지된 노드가 존재하는 그룹에 유휴노드를 추가하고 해당 노드의 모든 테이블에 대해 복제본을 갖도록 한다. 이후 모든 테이블에 대하여 복제본을 갖게 되면 정지된 노드에서 서비스하던 모든 마스터 테이블을 자신이 하게 된다. 이때 메타정보를 변경하게 되고 변경된 메타정보는 모든 노드에 전송하게 된다. 정지된 노드가 존재하는 그룹의 다른 노드들은 유휴노드가 그룹에 참여하여 정지된 노드의 역할을 대신하게 될 때 자신이 가지고 있는 모든 클러스터 로그를 삭제하게 된다. 또한 정지된 노드가 이후에 살아날 경우 다른 노드의 클러스터 회복관리자에 연결을 시도할 때 이미 다른 노드가 자신의 역할을 대신하였음을 알게 되고 자신의 데이터베이스를 초기화하여 유휴노드 상태로 만든다. 이를 그림으로 나타내면 [그림 7]과 같다.



[그림 7] 유휴노드가 정지된 노드를 대체하는 경우

5. 결론

본 논문에서는 공간 데이터베이스 클러스터에서 고가용성을 위한 병렬 회복 기법에 대하여 제안하였다. 제안 기법은 클러스터 구성에 대한 회복을 위하여 클러스터 로그를 유지하였다. 클러스터 로그는 그룹내에 각 테이블의 마스터가 존재하는 노드에서 테이블마다 독립적으로 유지되며, 노드가 정지되었을 경우에만 로그를 유지하였다. 따라서 보통의 질의처리에서 불필요한 클러스터 로그를 기록하지 않음으로써 질의 처리 속도가 저하되지 않으며, 각 테이블에 대하여 마스터가 독립적인 클러스터 로그를 유지하므로 로그 유지가 훨씬 쉽다. 제안 기법에서는 클러스터 로그를 바탕으로 하여 병렬적인 회복을 하였다. 이는 정지된 노드가 클러스터 구성을 위한 회복을 하는 단계에서 각 테이블마다 클러스터 로그를 병렬적으로 전송받아 회복하므로 테이블 각각이 독립적으로 병렬적인 회복이 가능하다. 따라서 대용량 데이터인 공간데이터에 대해서도 빠르게 회복할 수 있고, 회복이 끝난 테이블은 검색 질의에 대한 포워딩 처리를 함으로써 회복 단계에서 그룹의 부하를 빠르게 줄일 수 있어 가용성을 향상시킨다.

향후 연구로는 둘 이상의 노드가 정지하였을 경우에 대한 회복 기법을 연구할 것이다.

참고문헌

[1] R. H. Guting, and et. al, "A Foundation for Representing and Querying Moving Objects", ACM Transactions on Database Systems, Vol. 25, No. 1, pp. 1-42, 2000

[2] 유병섭, 김명근, 김재홍, 배해영, "GMS/Cluster 설계 및 구현 ", 개방형 지리정보 시스템 학회, p. 225 - 231, 2003

[3] Roel Vandewall, "Database Replication Prototype", Masters thesis, Department of Mathematics and Computer Science, University of Groningen, The Netherlands, 2000

[4] Roger Bamford, Rafiul Ahad, Angelo Pruscino, "A Scalable and Highly Available Networked Database Architecture", Proceedings of the 25th VLDB Conference, Edinburgh, Scotland, 1999

[5] Svein-Olaf Hvasshovd, Svein Erik Bratsberg, Oystein Torbjornsen, "An Ultra Highly Available DBMS", Proceedings of the 26th VLDB Conference, 2000

[6] Svein-Olaf Hvasshovd, Oystein Torbjornsen, Svein Erik Bratsberg, "The ClustRa Telecom Database: High Availability, High Throughput, and Real-Time Response", Proceedings of the 21st VLDB Conference, 1995

[7] Oystein Torbjornsen, Svein-Olaf Hvasshovd, Young-Kuk Kim, "Towards Real-Time Performance in a Scalable, Continuously Available Telecom DBMS", ClustRa, 2001