제18회 한국정보처리학회 추계학술발표대회 논문집 제9권 제2호 (2002.11)

비공유 데이터베이스 클러스터에서 온-라인 확장을 위한 데이터 분할 기법의 분석 및 평가

°장용일* 이충호* 이재동** 배해영*
*인하대학교 전자계산공학과
**단국대학교 전산과
himalia@dblab.inha.ac.kr

Analysis and Evaluation of Data Partitioning Methods for On-line Scaling in a Shared Nothing Database Cluster

°Yong-Il Jang* Jae Dong Lee** Hae-Young Bae*
*Dept. of Computer Science and Engineering, Inha University
**Dept. of Computer Science, Dankook University

요 약

비공유 데이터베이스 클러스터는 그 구조의 특성 상 동적인 질의 패턴의 변화, 특정 데이터에 대한 질의 집중에 의한 부하 불균형 및 집중, 사용자 증가에 의한 처리량 한계 등의 문제가 발생한다. 이러한 문제를 해결하기 위해 데이터베이스 클러스터는 최근에 제안된 온-라인 확장 기법을 사용하며, 이 기법은 데이터베이스의 확장성에 의해 큰 영향을 받는다. 일반적으로 클러스터 시스템에서 사용되는 데이터 분할 기법에는 키 값의 순서대로 분할하는 라운드-로빈 분할 기법, 해쉬 함수를 이용해 데이터를 분할하는 해쉬 분할 기법, 범위에 따라 각 노드에 데이터를 분할하는 범위 분할 기법, 그리고 조건식에 따라 데이터를 분할하는 조건식 분할 기법이 있다.

본 논문에서는 이 네 가지 분할 기법의 특성을 정리하고, 비공유 데이터베이스 클러스터에서 확장성에 있어서 우수한 분할 기법을 각 분할 기법의 성능평가를 통해 얻는다. 성능평가에서는 각각의 분할 기법을 평가하기 위해 확장 시 발생되는 이동 데이터의 크기, 질의 처리에 대한 영향, CPU 사용률, 그리고 온-라인 확장 기법의 수행 시 발생되는 특성에 대한 영향을 분석하며, 얻어진 결과를 토대로 비공유 데이터베이스 클러스터에서 가장 적합하면서도 온-라인 확장 기법 적용을 위해 확장성이 우수한 데이터 분할 기법을 찾는다.

1. 서론

비공유(Shared-Nothing) 데이터베이스 클러스터는 각각의 노드가 독립적으로 데이터를 관리하고 각 노드가 고속의 네트워크로 연결되는 비용대비 성능의 효율을 높인 하드웨어 플랫폼으로 구성된다[1]. 이러한 구조의 특성 때문에 비공유 데이터베이스 클러스터는 다음의 세 가지 문제들이 가장 큰 해결 과제로 인식되고 있다.

첫째, 대부분의 데이터베이스 정보는 관리자에 의해 그 저장 구조가 지정된다. 이 때 관리자는 사용자들의 요구 사항과 서비스 내용을 분석하여 최적의 성능을 낼 수 있도록 데이터를 분할(Fragmentation)하여 저장한다. 그러나, 오늘날 대부분의 서비스들은 사용자들의 요구가 시시각각 변하는 특징을 갖는데 이럴 경우 기존에 관리자에 의해 분할된 저장 구조는 오히려 데이터베이스의 성능을 떨어뜨리는 요인으로 작용한다. 둘째, 비공유 구조를 사용한 데이터베이스의 경우 모든 데이터들이 각 노드에서 독립적으로 처리된다. 이러한 구조는 특정 데이터에 대한 질의 집중이 발생할 수 있으며 해당 노드의 처리 속도 저하로 인해 전체적인 성능저하를 발생시키는 요인이 된다. 마지막으로 사용자의 증가로 인해 서버 처리 능력의 한계를 벗어나 서비스가 중단되는 경우이다. 이 때, 데이터베이스 클러스터는 서비스를 온-라인 상태로 지속적으로 유지시키면서도 전체적인 처리량을 확장시킬 수 있어야 한다.

온-라인 확장 기법은 이러한 문제들을 해결하기 위해 최근에 제안된 기법이다[2]. 이 기법은 각각의 문제점들을 해결하기 위해 항상 여분의 유휴 노드(Spare Node)를 두고 문제가 발생할 경우 이 유휴 노드로 데이터를 확장시키는 기법이다. 이러한 처리의 효율을 위해 온-라인 확장 기법은 실시간 처리되는 트랜잭션에 대한 영향을 최소로 하며, 확장에 소요되는 시간을 단축시키기 위해 이동되는 데이터의 크기를 축소시켜야 하는 동시에 가능한 동시성 유지를 해야 한다.

데이터베이스 클러스터에 있어서 확장성이란 데이터베이스를 구성하는 데이터를 기존에 구성되어 있는 노드 또는 새로운 노드로 확장시키는데 있어서 시스템에 적용되는 부하의 크기와 데이터 이동 크기, 그리고 처리 시간의 길이 등의 요소들에 영향을 받으며, 그 크기가 작을수록 확장성이 좋다[3].

데이터베이스 클러스터의 데이터 분할 기법은 크게 네 가지가 존재한다. 먼저 키 값의 순서대로 분할하는 라운드-로빈 분할 기법, 해쉬 함수를 이용해 데이터를 분할하는 해쉬 분할 기법, 범위에 따라 각 노드에 데이터를 분할하는 범위 분할 기법, 그리고 조건식에 따라 데이터를 분할하는 조건식 분할 기법이 있다[4]. 각각의 기법들은 분할 방식의 차이로 인해 서로 상이한 특징들을 갖는다. 마찬가지로 각 분할 기법을 어떻게 적용하느냐에 따라서 데이터베이스의 확장성도 크게 좌우된다.

본 논문에서는 데이터베이스의 온-라인 확장 기법에 초점을 맞추어 이 기법을 적용하였을 때 확장성에서 최적의 성능을 보이는 분할 기법을 찾는다. 이를 위해 각 분할 기법의 특징과 성능을 분석하고 용도에 따른 활용 여부를 가늠하고, 다시 각 기법들을 데이터베이스의 확장성을 기준으로 성능을 평가한다. 성능평가의 항목으로는 확장 시 발생되는 이동 데이터의 크기, 질의 처리에 대한 영향, CPU 사용률, 그리고 온-라인 확장 기법의 수행 시 입력되는 실시간 트랜잭션 처리에 대한 영향 평가가 있다. 각 성능평가를 통해 확장성이 우수한 데이터 분할 기법을 찾는다.

본 논문의 내용 구성은 다음과 같다. 먼저 2 장에서 관련연구 내용으로 데이터베이스 클러스터 구조와 확장성을 다룬 후 온-

라인 확장 기법과 데이터베이스 클러스터에 사용되는 각 분할 기법에 대해 설명하고, 3 장에서 각 분할 기법에 따른 데이터 이동성을 계산한다. 그리고 4 장에서는 각 기법에 따른 확장성 성능평가를 수행하며, 5 장에서 평가 내용을 분석하고 결론을 내린다.

2. 관련연구

본 장에서는 데이터베이스 클러스터의 종류와 확장성, 온-라인 확장 기법, 그리고 데이터 분할 기법에 대해 살펴본다.

2.1 데이터베이스 클러스터와 확장성

- 데이터베이스 클러스터

데이터베이스 클러스터는 둘 이상의 데이터베이스 시스템에서 데이터를 분할과 복제(Replication)기법을 사용해 관리하며 사용자에게는 하나의 시스템에서 동작하게 보이도록 투명성(Transparency)을 보장하는 시스템이다. 데이터베이스 클러스터는 기존의 중앙집중식 데이터베이스에 비해 고가용성을 보장하며 우수한 처리성능을 보인다. 데이터베이스 클러스터에는 비용대비 성능이 우수한 비공유 구조가 있다. 이 시스템은 높은 대역폭과 네트워크 지연이 거의 없는 고속의 랜(LAN) 망을 통해 지리적으로 가까운 여러 개의 노드가 서로 연결되어 클러스터를 구성하고 각 노드는 독립적인 주기억장치와 메모리, 저장장치를 갖는 워크스테이션으로 구성된다. 다른 종류로는 메모리 또는 디스크를 공유하는 형태인 메모리 공유 구조와 디스크 공유 구조가 있다.

- 확장성

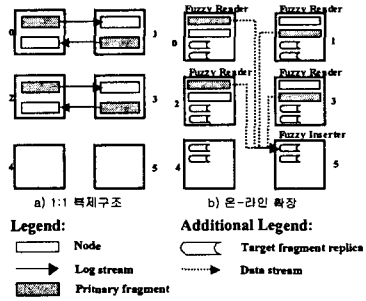
확장성이란 데이터베이스 클러스터에서 노드의 추가 또는 데이터가 보다 많은 노드로 분할되는데 따른 질의 처리 시 비용 평가에 의한 성능을 의미한다. 비공유 구조를 적용하는 환경에서 발생되는 문제점들을 해결키 위해 확장성은 반드시 필요한 요소이다. 고확장성을 위해 연구되어지는 분야는 크게 질의 처리, 인덱스, 복제 프로토콜, 데이터 분할 기법으로 나뉜다. 본 연구에서는 분할 기법에 따른 데이터베이스 클러스터의 확장성을 다룬다. 이 중 온-라인 재조직 기법은 보편적으로 연구되는 기법이며, 온-라인 확장 기법이 최근에 제안되어 그 활용도를 높이고 있다[5].

2.2 온-라인 확장 기법

온-라인 확장 기법은 물리적으로 현 시스템의 활성 노드를 추가하여 부하 분산 및 전체적인 처리량을 늘리는 기법이다. 본 기법은 확장 시 소요되는 처리 시간을 줄이고, 실시간 트랜잭션에 대한 영향을 최소화해야 하며, CPU·네트워크 부하를 최소화하며, 데이터베이스의 확장성을 향상시키는 동시에 확장 후 질의 처리 성능을 향상시킬 수 있어야 한다. 온-라인 확장 과정은 크게 세 단계로 나뉜다.

1) 준비과정
   a) 새로운 카탈로그 정보 생성
   b) 이동 데이터의 카탈로그 정보에 로크 설정
   c) 로그 기록 시작
   d) 대상 복사본 생성
2) 데이터 전송과정(블록 별 다단계 전송)
   a) 읽기 로크
   b) 데이터 전송
   c) 로크 해제
   d) 해당 복사본에 데이터 저장
   e) 실시간 트랜잭션 로그 전송 및 반영
3) 완료 과정
   a) 질의 입력 중지
   b) 최종 로그 전송 및 반영
   c) 새로운 카탈로그 정보 반영
   d) 질의 입력 시작
   e) 카탈로그 로크 해제

클러스터를 구성하고 있는 여러 노드들을 역할에 따라서 두 가지 타입으로 나누는데 실시간 클라이언트 질의를 수행하고 있는 노드를 활동 노드(Active Node) 라 하고 시스템의 확장을 위해서 사전에 등록되어 있는 노드를 유휴 노드(Spare Node)라 한다. 활동 노드 중에서 특정 노드가 고장을 일으킨 경우나 부하가 집중되는 경우, 유휴 노드를 활성화 상태로 만들고 유휴 노드에 기존의 분할 및 복제되어 있던 데이터를 온-라인 확장을 수행한다. 이로써 전체 시스템의 가용성을 높이고 병목현상을 제거하여 전 체 트랜잭션 처리량을 증가시키고 평균 응답시간을 줄이도록 한다.



[그림 1] 온-라인 확장 기법

온-라인 확장은 [그림 1]와 같은 복제 기법에서 퍼지 리더(Fuzzy Reader)와 퍼지 인서터(Fuzzy Inserter)에 의한 데이터 이동 및 확장이 이루어진다.

2.3 데이터 분할 기법

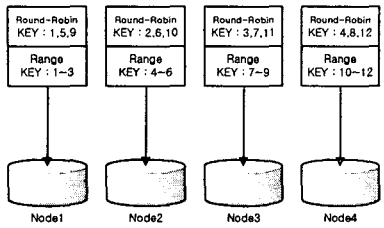
데이터베이스 클러스터는 그 효율성을 고려해 수직 분할을 제외한 수평 분할만을 사용한다. 기본적으로 분할기법에는 다음과 같은 4 가지 기법이 있다.

- 라운드-로빈(Round-robin) 분할 기법
  - 가장 간단한 분할 기법
  - 각 레코드의 키 순서대로 데이터 저장
  - 해쉬 분할 기법에 포함됨
- 해쉬(Hash) 함수 분할 기법
  - 해쉬 함수에 의한 레코드 재배열 방식
  - 부하 분산에 효율적인 기법
- 범위(Range) 분할 기법
  - 키 값의 범위에 의한 분할 방식
  - 범위 검색 연산에 효율적인 기법
- 조건식(Expression) 분할 기법
  - 기본키 값에 근거한 조건별 분할 방식
  - 데이터 형태에 따른 유연한 분할 기법

각각의 기법은 모두 기본키를 기반으로 분할이 되며, 각각 분할 방식에 따라서 부하 분산, 영역 질의 처리 효율, 공간 활용, 확장성 등의 특성에 장·단점을 갖는다. 라운드-로빈 분할 기법은 각 분할 데이터가 레코드를 균등 분할하여 부하 분산에 강한 장점을 갖는다. 대신 대부분의 질의 처리가 모든 노드로부터 이루어져야 하기에 항상 네트워크 비용을 고려하여야 한다. 반면에 범위 분할 기법과 조건식 분할 기법은 각 데이터 별 접근 빈도에 따라 비 균등 분할이 가능하다. 적절한 분할이 이루어진 경우 불필요한 네트워크 비용을 절감할 수 있지만 질의 패턴의 변경에 의해 부하의 집중 현상이 일어나는 문제점이 있다.

조건식 분할 기법은 데이터의 사용 유형별 분할이 가능하여 다른 분할 기법에 비해 유연한 장점을 갖는다. 반면 조건식이 복잡할 경우 질의 수행 시 조건 판별의 추가적인 오버헤드가

작용할 우려가 있다.



[그림 2] 데이터 분할 기법

3. 데이터의 이동성

본 장에서는 그 성능에 대한 객관적 판단이 가능한 라운드-로빈, 해쉬, 범위 분할에 대한 확장 시 각 분할 기법에 따른 데이터 이동성의 계산 및 각 기법의 비교 평가를 한다.

3.1 라운드-로빈 분할 기법

라운드-로빈 기법에서 확장 연산을 수행할 경우 해당되는 모든 노드에서 데이터 이동이 일어나게 되어 확장 연산에 의한 부하가 커진다. 라운드-로빈 분할 기법에서 확장 시 이동되는 데이터 크기를 수식으로 표현하면 다음과 같다. 먼저 전체 노드 수를 n, 노드 순서를 i, 확장 노드 수를 k, 전체 데이터를 X로 정의하였을 때, 각 노드에 저장되는 데이터 X_i는 다음 수식으로 표현된다.

$$X_i = \{x \mid x \bmod n \subseteq i, \forall x \in X\}$$

이 때, k만큼 확장된 노드에 의해 기존 데이터 x_i가 k를 포함한 확장된 동일 위치에서의 데이터는 다음과 같다.

$$X_i = \{x \mid x \bmod (n+k) \subseteq i, \forall x \in X\}$$

위의 수식을 통해 확장된 기법에서 위치가 변경되지 않고 각 노드에 저장되는 데이터는 n과 $n+k$의 최소공배수(L.C.M)로 X 값을 mod 연산을 취한 경우이다.

$$X_i' = \{x \mid x \bmod L.C.M(n, n+k) \subseteq i, \forall x \in X\}$$

전체 데이터 X 에서 옮겨지지 않는 데이터의 비율은 X_i'/X_i에 해당하며, 이는 다시 전체 노드에서 이동되지 않는 데이터의 비율 $X_{notmove}$으로 표현되며 이는 다시 전체 노드 수에 대해 확장 노드 수의 비율로 다음과 같다.

$$X_{notmove} = \frac{n}{L.C.M(n,n+k)}$$

마지막으로 이동 데이터의 비율 X_{move}은 다음과 같다.

$$X_{move} = 1 - \frac{n}{L.C.M(n,n+k)} = \frac{L.C.M(n,n+k)-n}{L.C.M(n,n+k)}$$

위 공식으로 얻어지는 데이터 이동 비율은 확장 노드 수에 의한 최소공배수 값에 따라 달라진다.

3.2 해쉬 분할 기법

이 기법은 적용되는 해쉬 함수의 유형에 따라 달라지므로 어떤 계산식을 유도할 수 없다. 대신 해쉬 함수의 비순차적 분할 특성에 따라서 라운드-로빈 분할 기법과 마찬가지로 확장 시 대부분의 데이터가 이동에 참여하게 된다.

3.3 범위 분할 기법

범위 분할은 균등 및 비균등 분할로 나뉜다. 균등 분할은 각 노드에 분할되는 데이터의 분포를 균등하게 조절하지만 비균등 범위 분할은 각 노드에 서로 다른 범위의 데이터가 할당된다. 또한, 균등 범위 분할을 처리하기 위해서 노드의확장이 발생되면 기존의 각 노드에 할당되던 범위는 확장 노드에 맞게 축소된다. 확장됨에 따라 축소되는 범위의 크기를 l이라 한다. 이때, 첫 노드에서 l개의 데이터가 이동되면 다음 두 번째 노드에서는 첫 노드에서 이동된 l개의 데이터와 함께 $2l$개의 데이터가 다음 노드로 이동이 되고 세 번째 노드는 총 $3l$개의 데이터를 이동시킨다. 이 경우 이동되는 데이터의 전체적인 크기 X_{range}는 다음과 같다.

$$X_{range} = l + 2l + \ldots + nl = \sum_{1}^{n} l = \frac{nl(n+1)}{2} \quad (l \geq 1, n \geq 1)$$

그러나, 의 크기가 한 노드상에 들어간 데이터 범위 X/n 보다 커지는 경우가 발생하면 그 이후 이동되는 데이터의 크기는 X/n 가 된다. 따라서 임의의 i, 즉 il<X/n 인 위치까지 위의 식을 적용하며 그 이후의 영역 n-i 부분은 전체 데이터가 이동된다. 이때, l의 크기는확장 후에도 각 노드의 범위가 균등해야 하므로 다음과 같이 계산이 된다.

$$l = \frac{X}{n} - \frac{X}{n+k} = \frac{X(n+k)}{n(n+k)} - \frac{nX}{n(n+k)} = \frac{kX}{n(n+k)}$$

그리고, i 는 정수이며 그 범위는 다음 수식에 의해 결정된다.

$$il \leq \frac{X}{n} \rightarrow \frac{ikX}{n(n+k)} \leq \frac{X}{n} \rightarrow \frac{ik}{n+k} \leq 1 \rightarrow i \leq \frac{n+k}{n}$$

결과적으로 균등 범위 분할 기법에서의 전체 데이터 이동 크기는 다음과 같다.

$$X_{range} = \frac{kX}{n(n+k)} \times \frac{i(i+1)}{2} + \frac{X(n-i)}{n} = \frac{ikX(i+1)}{2n(n+k)} + \frac{X(n-i)}{n}$$

이를 바탕으로 이동되는 데이터의 비율을 계산하면 다음과 같다.

$$X_{move} = \frac{X_{range}}{X} = \frac{ikX(i+1)}{2nX(n+k)} + \frac{X(n-i)}{nX} = \frac{ik(i+1)}{2n(n+k)} + \frac{n-i}{n}$$

반면에 비균등 범위 분할 기법은 해당 확장 노드에 대해 데이터의 이분을 통한 확장을 수행한다. 이때 다른 노드에 미치는 영향은 없다.

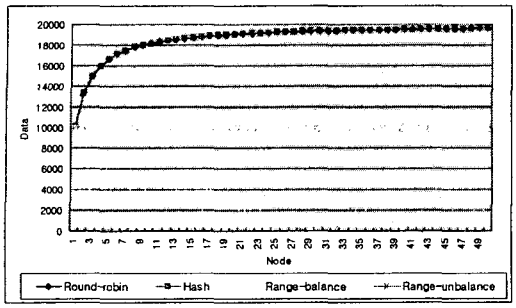
4. 성능평가

본 장에서는 계산식에 의해 정리된 각 분할 기법의 이동되는 데이터의 크기에 대한 자료를 바탕으로 이동되는 데이터 크기를 비교하며, CSIM 을 통해 구현된 가상 환경에서 온-라인 확장 기법에 의해 발생되는 영향을 분석한다[6].

[그림 3]은 노드의 수를 1 대부터 50 대까지 늘려가면서 한 노드에 대한 확장 시 이동되는 데이터 크기를 분석한 그래프이다. 라운드-로빈 분할 기법과 해쉬 분할 기법은 키를 분배하는 방식이 서로 다르지만 확장 시 이동되는 데이터 크기는 거의 비슷하다. 그리고, 균등 범위 분할 기 법은 정확히 반 수에 해당하는 데이터가 확장 시 이동된다. 비균등 범위 분할 기법은 이동되는 데이터의 크기가 최고일 때 전체 데이터의 반 수이며, 최저일 때 거의 0 에 가깝다. 그러나 평균적인 이동 데이터의 크기는 가장 낮다.

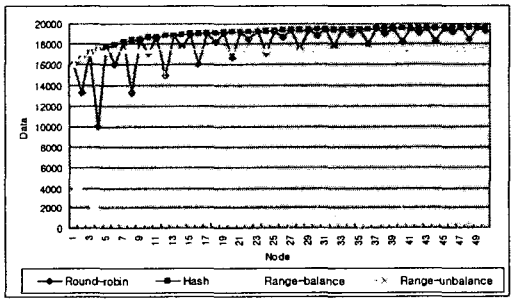
<표 1> 실험환경

시스템 속성	
네트워크 속도	1000 M bits/s
노드 수	1 ~ 50
시스템 사양	Pentium III 700 CPU, 512MB Memory
데이터베이스 속성	
레코드의 수	20000
레코드 크기	128Byte



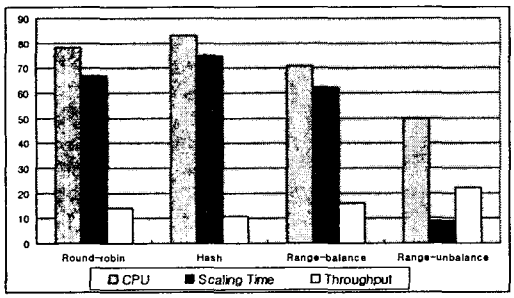
[그림 3] 이동되는 데이터의 크기(확장 노드: 1)

[그림 4]는 확장되는 노드의 수를 네 대로 늘린 결과이다. 라운드-로빈 분할 기법은 최소공배수의 특성에 따라 규칙적인 상하 진동 그래프를 그렸으며, [그림 3]에 비해 낮은 이동 데이터 크기를 갖는다. 반면에 해쉬 분할 기법과 균등 범위 분할 기법은 오히려 이동 데이터의 크기가 커졌다. 그리고, 비균등 범위 분할 기법의 이동 크기가 현저히 낮아졌다. 이를 통해 비균등 범위 분할 기법은 확장 노드의 수가 많을수록 이동 크기가 작음을 알 수 있다.



[그림 4] 이동되는 데이터의 크기(확장 노드: 4)

[그림 5]는 각 기법에서 온-라인 확장을 하였을 때 얻은 결과 그래프이다. CSIM을 통해 구현된 가상 환경에서 시연되었으며, 각각 CPU 사용률(%), 확장 시간(ms), 확장 도중 트랜잭션 처리속도(transaction per millisecond)를 측정하였다. 결과 그래프를 분석한 결과 확장 시간에 있어서 데이터 이동 크기는 가장 큰 영향을 미치며 이에 따라서 실시간 트랜잭션 처리량과 CPU 사용률에 미치는 영향이 달라짐을 알 수 있었다.



[그림 5] 온-라인 확장 수행 그래프

5. 평가 분석 및 결론

비공유 데이터베이스 클러스터에서 사용되는 기본적인 데이터 분할 기법에 대해 그 특성과 온-라인 확장에 대한 성능을 평가한 결과, 각 노드에 분할되는 데이터가 다른 노드의 데이터와의 상관관계가 적고 독립성이 강할수록 확장 시 이동되는 데이터의 크기가 작게 나타났다. 키 값의 순서대로 데이터를 분할하는 라운드-로빈 기법과 해쉬 기법의 경우 일반적인 질의 처리에 대해서 병렬성을 최대화하지만 확장성에 있어서는 그 효율이 매우 낮게 나타났다. 균등 범위 분할은 라운드-로빈 기법에 비해서 낮은 데이터 이동 크기를 보였지만 이 역시 확장 노드의 수를 증가시킬 경우 크기가 증가되는 문제점을 보인다.

비균등 범위 분할 기법은 그 특성으로 인해 쉽게 부하가 집중될 수 있는 가능성을 항상 지니고 있다. 그러나 성능 평가를 분석한 결과 부하 집중에 의해 발생되는 문제가 처리 성능에 미치는 영향보다 확장 수행에 의한 부하 분산 효과가 더 큰 것으로 나타난다. 또한, 실제 환경에서 데이터 분할 수가 커질수록 라운드-로빈 분할 기법에서의 데이터 분포 정도와 비슷해져 부하 집중의 가능성이 낮아지는 점을 고려하면 비균등 범위 분할 기법의 단점 또한 보완이 가능하다.

본 논문에서는 비공유 데이터베이스 클러스터에서 온-라인 확장 기법을 위한 분할 기법들을 분석 및 평가하였으며, 평가 결과 비균등 범위 분할 기법이 가장 우수한 것으로 판명되었다. 비균등 범위 분할 기법은 확장성 면에서 가장 뛰어난 성능을 보이며, 확장성 테스트에서도 최적의 성능을 보였다. 온-라인 확장 기법의 처리를 비균등 범위 분할 기법에 기반하여 처리할 경우 처리 속도 및 시스템 자원 효율의 활용도가 높아지며, 아울러 확장 노드에 대한 독립성 또한 높아졌다.

참고문헌

[1] David J. DeWitt, Jim Gray, "Parallel Database Systems : The Future of Database Processing or a Passing Fad?", http://research.microsoft.com/~gray/CacmParallelDB.doc

[2] Svein Erik Bratsberg, Rune Humborstad, "Online Scaling in a Highly Available Database", In Proceedings of the 27th VLDB Conference, Roma, Italy, 2001

[3] Ricardo Jiménez-Peris, Marta Patiño-Martínez, Bettina Kemme, Gustavo Alonso, "Improving the Scalability of Fault-Tolerant Database Clusters", ICDCS 2002, p477-484

[4] Informix, "Informix Extended Parallel Server 8.3", http://www.Informix.com.

[5] Chendong Zou, Betty Salzberg, "On-line reorganization of sparsely-populat-ed b+-trees", In Proceedings of ACM/SIGMOD, p115-124, June 1996.

[6] Mesquite Software, Inc. "CSIM in a Nutshell: A Quick Guide to System Simulation with CSIM18", http://www.mesquite.com