▨ 연구논문

데이터베이스에서의 시간 시스템에 관한 연구

(A study of Time Management System in Data Base)

최 진탁*
Choi, Jin Tag

Abstract

A new algorithm is proposed in this paper which efficiently performs join in the temporal database. The main idea is to sort the smaller relation and to partition the larger relation, and the proposed algorithm reduces the cost of sorting the larger relation. To show the usefulness of the algorithm, the cost is analyzed with respect to the number of accesses to secondary storage and compared with that of Sort-Merge algorithm. Through the comparisons, we present and verify the conditions under which the proposed algorithm always outperforms the Sort-Merge algorithm. The comparisons show that the proposed algorithm achieves 10~30% gain under those conditions.

1. 서 론

기존의 데이터베이스 시스템은 현실세계의 현시점에 대한 정확한 정보만을 관리한다. 따라서 한 데이터 항목의 값을 새로운 값으로 변경하면 데이터베이스로부터 이전의 데이터 값은 삭제되고 새로운 값만이 데이터베이스에 존재하게 된다. 이러한 특성 때문에 기존의 데이터베이스 시스템은 현재시점의 정보뿐만 아니라 과거의 이력정보(historical infomation)까지도 필요로 하는 응용환경에는 부적합하다. 과거의 이력정보까지도 요구하는 응용환경을 지원하기 위해서는 현재 시점의 데이터 값뿐만 아니라 과거의 데이터 값들도 관리하는 시간적 데이터베이스 시스템(temporal database system)을 필요로 한다.

시간지원 데이터베이스는 기존의 데이터베이스에 시간의 개념을 추가한 것으로 현실세계에서 발생된 사실에 대하여 현재 시점뿐만 아니라 과거와 미래 시점에 대하여 효율적인 데이터 관리를 제공하여 다양한 분야에 응용되고 있다. 이러한 시간지원 데이터베이스는 시간지원 데이터 모델, 시간지원 질의어, 시간지원 질의어 처리 등 다양한 분야에서 연구되고 있다[1, 2, 3]. 특히 질의어 처리에 관한 연구는 연산자의 복잡성과 높은 연산비용 때문에 최근 들어 많은 연구가 진행되고 있다[6, 7]. 본 논문은 시간지원 데이터베이스에서 질의어 처리의 한 분야인 시간지원 조인에 대한 연구이다.

조인은 데이터베이스 정규화 작업으로 인하여 분리된 릴레이션으로부터 유용한 정보의 추출을 위하여 여러 릴레이션을 결합하는 연산자를 말한다. 시간지원 데이터베이스에서의 조인은 특히 연산자의 복잡성과 잦은 사용 때문에 많은 연구가 진행 중에 있으며, 이러한 연구의 초점은 연산에 필요한 보조 기억 장치의 액세스 수를 최소화하는 관점에서 진행되어 왔다.

시간지원 조인은 크게 두 분야에서 연구되어 왔다. 하나는 조인을 위한 어떠한 보조 정보 등이 제공되지 않는 상태에서 조인 방법[7, 8]이며 다른 하나는 인덱스와 같은 보조 정보 또는 특정 애트리뷰트 값에 대하여 정렬된 정보를 이용하는 조인 방법[4, 6]이다. 전자의 경우 중첩루프 방법과 정렬병합 방법이 잘 알려져 있다. 중첩루프 방법은 데이터 값의 분포에 관계없이 주어진 두 릴레이션과 버퍼 페이지의 크기에 따라 일정한 비용으로 연산을 수행하나 주어진

인천대학교

버퍼 페이지가 작은 경우 매우 큰비용을 요구한다[6, 10]. 정렬병합 방법은 두 릴레이션을 먼저 정렬한 후 조인을 수행하는 것으로 조인 애트리뷰트 값의 분포에 따라 부분적으로 중첩루프 방법을 적용해야 하므로 조인 비용이 유동적이다[6]. 일반적으로 버퍼 페이지의 크기가 두 릴레이션중 작은 릴레이션의 크기보다 크거나 조인 애트리뷰트 값의 분포가 많이 중첩되어 있는 경우 중첩루프 방식이 우수하며 그렇지 않거나 조인의 결과가 조인 애트리뷰트 값으로 정렬되어야 하는 경우에는 정렬병합 방법이 좋은 것으로 알려져 있다[10]. 인덱스와 같은 보조 정보가 있어 이를 이용한 시간지원 조인에 대한 연구[4, 6]도 많이 되어 왔다.

본 논문에서는 조인을 위하여 어떠한 보조 정보가 없는 경우에 대한 시간지원 조인 알고리즘을 제안한다. 기존이 정렬병합 방법이 버퍼 페이지가 크지 않거나 조인스캔율[4, 6]이 작은 경우 좋은 성능을 발휘하나 두 릴레이션을 정렬하는 것이 필요하며 이러한 정렬에도 불구하고 부분적으로 중첩루프 방법을 적용해야 한다는 단점이 있다. 정렬병합 방법의 이러한 단점을 보완하기 위하여 두 릴레이션 모두를 정렬하는 대신에 크기가 작은 릴레이션은 정렬하고 큰 릴레이션은 분할하여 하나의 조인을 여러개의 독립된 부 조인으로 분할한 후 마지막 단계인 조인에서 중첩루프 방법을 도입하는 새로운 시간지원 조인 알고리즘을 제시하고자 한다.

시간지원 데이터베이스 관리시스템은 기존의 관계형 데이터베이스 관리 시스템이 보유하고 있는 기능이외에도 이력 데이터를 효율적으로 처리할 수 있는 시스템의 구조 및 기능이 필요하게 된다. 즉, 시간 지원 데이터베이스 관리 시스템은 한 시점을 기준으로 유효한 데이터뿐만 아니라 과거의 이력 데이터의 내용도 모두 관리하게 되므로 데이터베이스 관리 시스템이 저장, 관리해야하는 데이터베이스의 양이 종래의 데이터베이스 관리 시스템이 관리해야 하는 데이터베이스의 양보다 훨씬 많게 된다. 따라서 시간 지원 데이터베이스 관리 시스템은 많은 양의 데이터베이스를 효율적으로 저장 관리할 수 있는 능력을 가지고 있어야 한다.

2. 시간지원 데이터 모델

2.1 시간 영역의 개념

시간지원 데이터 모델에서는 기존의 데이터 모델에서 취급하지 않은 시간 의미를 추가시켜야 한다. 시간 의미를 표현하기 위해서는 시간에 대한 기본 기간 개체를 선택하고 시간의 순서를 나타내는 모델과 그에 적절한 타임스탬프를 선택해야 한다.

① 시점(time point) : 가장 기본적인 시간을 표현하기 위해서 시간을 하나의 점으로 표현하기 위해서 시간을 하나의 점으로 나타낸다. 이 경우 시간 간격은 간격의 시작점과 끝점의 쌍으로 표현할 수 있다.

② 시간 간격(time interval) : 지속시간을 주로 처리해야 하는 스케쥴링 시스템에서 많이 사용하는 방법으로 가장 기본적인 시간 개체로 시간 간격을 사용한다. 이 경우 점 시간은 매우 작은 시간 간격으로 표현된다.

③ 시간의 순서에 대한 선형 : 가장 많이 사용되는 방법으로 시간을 일직선으로 표현하며 현재, 미래와 과거에 관계없이 모든 시간들이 일직선상에 선후에 관계를 가진다. 그러므로, 미래의 가능성은 하나만이 존재한다.

④시간의 순서에 의한 분기형 : 소프트웨어 공학등에서 사용되는 방법으로 시간지원 시스템 내의 데이터는 한 기원이되는 시간 점에 의해 분기될 수 있다. 점 시간들 사이의 순서는 분기를 기준으로 부분적으로 행해진다. 미래에 발생할 수 있는 사건에 대한 여러 가지 가능성에 대한 정의가 가능하다.

⑤시간의 순서에 의한 원형 : 시간의 흐름이 원형을 이룬다. 반복적인 사건이나 처리에 대한

모델링에 사용된다.

2.2 타임스탬프의 구조 정의

시간지원 데이터 모델에서는 시간을 기본 단위 표현을 위해 타임스탬프의 구조를 선택할 것인지를 결정해야 하며 다음과 같은 모델이 존재한다.

① 유리수(dense model) : 시간을 유리수 Q의 집합으로 표현한다.

② 실수(continuous model) : 시간을 실수 R의 집합으로 표현하며 연속적인 시간의 표현이 필요한 경우 사용된다.

③ 정수(discrete model) : 시간을 정수 Z의 집합으로 표현하며 시간을 불연속한 것으로 간주한다.

보통 실세계의 시간은 일반적으로 실수의 형태로 연속적인 값을 갖는다. 그러나, 인간이 인식할 수 있으며, 측정 가능한 가장 최소의 시간 단위인 크로논이라는 비연속적인 단위 값을 갖게 된다.

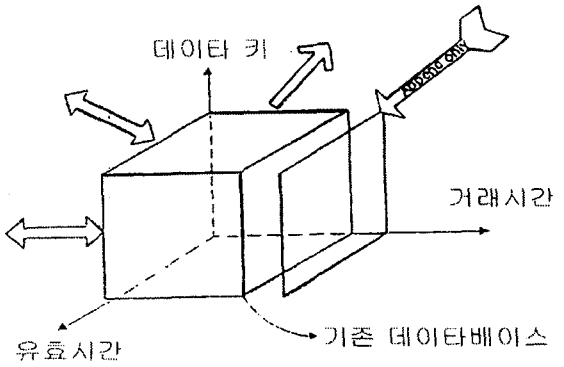
2.3 타임스탬프의 종류

시간 값은 의미적으로 세 가지 종류로 분류할 수 있다. 첫째, 단일 크로논인 시점과 연속적인 크로논의 집합인 간격 그리고 크로논의 임의의 집합인 시간 요소가 있다.

단일 사건(event)은 단일 크로논 값에 의해서 표현하며 시간 간격(interval)은 원자 속성의 시간 값의 쌍에 의해 표현된다. 시간의 흐름에 따라 정보의 변화를 데이터 모델에서 표현하기 위해서는 시간 의미(temporal semantic)를 데이터 모델에 포함시켜야 한다.

3. 시간지원 데이터베이스 시스템의 특성

시간지원 데이터베이스는 기존의 데이터베이스와는 달리 현재시점의 데이터뿐만 아니라 과거의 이력 정보까지도 포함하고 있다. 시간지원 데이터베이스 시스템에서 현재시점의 데이터 집합에서 특정 데이터를 구분하기 위해서는 데이터 키값을 사용하며 과거의 이력 데이터를 데이터베이스 시스템의 시간을 기준으로 구분하기 위해서는 거래시간 값을 이용하며 과거의 이력 데이터를 현실세계의 시간기준으로 구분하기 위해서는 유효시간 값을 이용한다. 이러한 시간지원 데이터베이스는 [그림3-1]과 같이 3차원 공간으로 표현될 수 있다.



[그림 3-1] 시간지원 데이터베이스의 3차원 표현

기존의 데이터베이스에서는 여러개의 키값에 대한 최근에 유효한 데이터항목만을 관리하고 있을 뿐 데이터 항목의 과거 이력정보는 관리하고 있지 않다. 즉, 그림[3-1]에서 굵은 선으로 표시된 부분이 기존의 데이터베이스를 표시한 것이다. 시간지원 데이터베이스에서는 유효시간 뿐만 아니라 거래시간 개념까지도 지원하고 있으므로 시간지원 데이터베이스를 [그림3-1]에서와 같이 <데이터 키값, 유효시간, 거래시간>으로 구성된 3차원 공간으로 모델링할 수 있다. 따라서 시간 지원 데이터 베이스 관리 시스템이 관리해야하는 데이터 항목의 수는 기존의 데이터베이스 관리 시스템이 관리해야 하는 데이터의 양보다 훨씬 많다.

기존의 데이터베이스 시스템에서와 마찬가지로 시간지원 데이터베이스 시스템에서도 사용자는 트랜잭션을 한 단위로 하여 데이터베이스를 액세스한다. 이러한 사용자트랜잭션에는 시간지원 데이터베이스를 액세스하는 연산을 1개이상 포함하고 있다. 시간지원 데이터베이스를 액세스하는 연산으로는 삽입연산, 삭제연산, 수정연산, 검색연산으로 구분할 수 있다. 그러나 수정연산은 삭제연산과 삽입연산의 조합으로 구현 가능하므로 시간지원 데이터베이스에 대한 연산으로 삽입연산, 삭제연산, 그리고 검색연산만을 고려한다.

다수 사용자가 시간지원 데이터베이스를 공유하는 환경에서 시간지원 데이터베이스 관리 시스템은 시간지원 데이터베이스의 일관성(consistency)을 유지하기 위해 트랜잭션에 대한 동시성 제어(concurrency control)를 실행해야 한다. 시간지원 데이터베이스를 액세스하는 사용자 트랜잭션은 현재시점의 정보뿐만 아니라 과거의 이력 정보까지도 요구하기 때문에 기존의 관계형 데이터베이스를 액세스하는 트랜잭션이 액세스해야 하는 데이터 양보다 훨씬 많은 양의 데이터 정보를 액세스한다. 따라서 시간지원 데이터베이스 관리 시스템에서 사용되는 효율적으로 관리하는 특성을 지녀야 한다. 시간지원 데이터베이스 시스템에서 사용되는 효율적인 동시성 제어 기법은 동시성 제어의 비용을 최소로 유지할 수 있어야 한다. 또한, 동시성 제어의 비용을 최소로 하기 위해 불필요하게 동시에 수행되는 트랜잭션의 수를 제약해서는 안 된다. 즉, 시간지원 데이터베이스관리 시스템에서 사용하는 동시성 제어 기법은 동시성 제어의 비용을 최소로 유지하면서도 동시에 허용하는 트랜잭션의 동시성 정도를 최대로 유지하는 특성을 가지고 있어야 한다.

3.1 시간 데이터 모델의 종류

시간지원 데이터 모델은 지원되는 시간에 따라 모델을 분류할 수 있다. 기존의 릴레이션 모델과 같이 시간 개념이 없는 모델을 스냅 데이터 모델이라 하고 실세계에서 사건이 발생한 시간을 나타내는 유효시간 데이터 모델과 발생된 사건에 대한 정보를 데이터베이스를 수록한 시간을 지원하는 거래시간 데이터 모델이 있다.

시간지원 데이터 모델은 시간과 연관된 자료를 처리하기 위해 기존의 데이터 모델에 시간 속성을 추가한 데이터 모델을 사용한다. 대부분의 모델은 관계형 데이터 모델을 확장하여 사용하나 일부 객체 지향 데이터 모델과 EER(Enhanced Entity Relationship)모델 등에 시간 속성을 추가하여 확장된 모델을 사용하기도 한다. 시간지원 데이터 모델의 분류는 지원되는 시간 여부에 따라 유효시간 데이터 모델, 거래시간 데이터 모델, 이원시간 데이터 모델로 나눌 수 있다.

① 유효시간 데이터 모델

유효시간 데이터 모델에 대한 릴레이션의 예로 (a)는 Sarda 모델에 의한 릴레이션으로 튜플 타임스탬프 형식이고, 시간 표현은 시간 간격을 사용하였으며 (b)는 Clifford 데이터 모델로서 속성 타임스탬프 형식을 사용하며 튜플의 생명주기를 시간 요소로 표현하였고 (c)는 Tansel의 모델로 속성 타임스탬프와 시간 간격으로 표현하였다.

② 거래시간 데이터 모델

거래시간을 지원하는 데이터 모델은 비삭제정책을 사용한다는 것이 유효시간 데이터 모델과의 차

이가 되며 데이터가 데이터베이스에 수록된 모델이며, 거래시간을 나타내는 시간을 나타낸다.

③ 이원시간 데이터 모델

유효시간과 거래시간이 모두 지원되는 데이터 모델이다. 이원시간 데이터 모델은 유효 시작시간과 종료시간, 거래 시작시간과 종료시간을 나타내는 속성이 있으며, 시간 간격과 튜플 타임스탬프로 표현하고 있다.

④ 시간지원 데이터 모델 비교

시간지원 데이터 모델은 아래의 4가지 조건에 의해 각 모델을 비교할 수 있다.

· 유효시간을 표현하는 방법은?
· 거래시간을 표현하는 방법은?
· 속성에 대한 동형성(homogeneous)의 표현 여부?
· 속성에 대한 결합(coalescing)을 표현할 것인지?

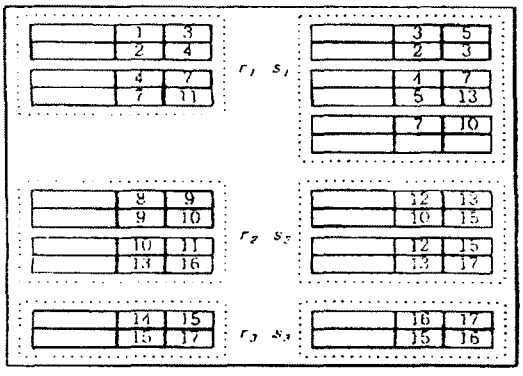
⑤ 유효시간

시간지원 데이터 모델에서 유효시간을 표현할 경우 두 가지 사항을 고려해야 한다. 첫째, 시간 표현 방법을 결정해야 한다. 시간 표현 방법에는 싯점 타임스탬프의 크로논으로 표현하는 방법, 간격 타임스탬프로 표현하는 방법과 튜플들의 집합에 시간을 결합하는 유효시간 요소(valid-time element)로 표현하는 방법이 있다. 둘째는 유효시간의 결합을 속성 타임스탬프로 표현할 것인지 튜플 단위의 시간 단위로 표현할 것인가를 결정해야 한다.

4. 제안 시간조인 알고리즘

제안 알고리즘에 사용되는 기호는 다음과 같이 정의하여 사용한다.

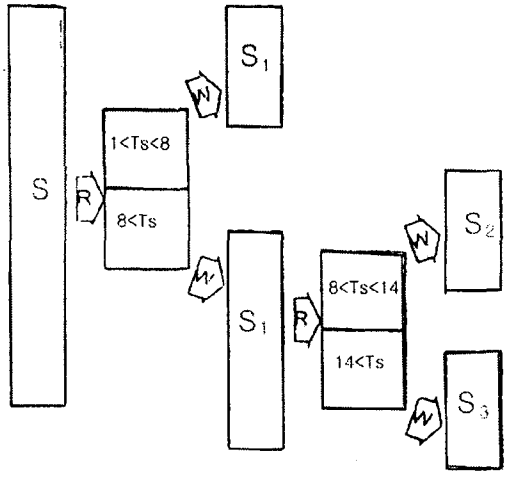
R, S : 조인에 참여하는 두 릴레이션
|R|, |S| : R, S의 보조 기억장치에서의 페이지 수
X : R, S의 페이지 수의 비율, 즉 $\frac{|S|}{|R|}$
B : 조인을 위하여 사용 가능한 버퍼 페이지 수 - 1
N : $\left|\frac{|R|}{|S|}\right|$
a : R, S가 정렬된 상태에서 R이 하나의 튜플에 대하여 조인을 완료하기 위하여 읽어야 하는 S 튜플의 비용



[그림 4-1] 정렬된 R 릴레이션 분할된 S 릴레이션

본 논문에서는 앞으로 R이 S보다 작다고 가정하며, 조인을 위하여 사용 가능한 버퍼페이지 수는 수식 표현의 용이함을 위하여 B+1로 하고 B는 항상 2보다 크다고 가정한다. 제안하는 시간조인 알고리즘은 작은 릴레이션은 Ts 값을 기준으로 정렬을 하고 큰 릴레이션은 Ts 값을 기준으로 분할하자는 것이다. 먼저 작은 릴레이션 R을 Ts 값을 기준으로 외부 정렬 방법을 이용하여 정렬한다. 정렬된 R에 대하여 순차적으로 B 페이지씩 그룹화하여 N개의 독립된 (r1, r2, ..., rn) 버킷을 만든다. 다음 큰 릴레이션 S를 R의 N개의 버킷에 대응하도록 분할 기준값에 의하여 독립된 버킷으로 분할한다. S의 각 버킷에 대한 분할 기준값은 ri 버킷에 대응하도록 분할 기준값에 의하여 독립된 버킷으로 분할한다. S의 각 버킷에 대한 분할 기준값은 r_i 버킷에 대하여 $b_i = Ts_{min}$로 정하여 진다($1 \le i \le N+1$ 이고, Ts_{min}는 r_i버킷의 튜플의 Ts중 최소의 Ts 값이고 b_{N+1}은 ∞ 이다.) S 튜플의 Ts가 분할 기준값 b_i보다 크거나 같고 b_{i+1}보다 작으면 이 튜플은 s_i버킷에 속하도록하여 독립된 N의 버킷으로 분할한다.

마지막 조인 단계는 앞에서 설명한 정렬병합 방법과 유사하다. 먼저 S의 첫 튜플을 Bof(s)라 설정한다. $1 \le i \le N$에 대하여 순차적으로 B개의 버퍼 페이지를 이용하여 R의 r_i버킷을 읽고 나머지 튜플부터 시작하여 s_j 버킷의 마지막 튜플까지 읽어서 조인을 수행한다. 이때 S의 튜플중 b_{i+1}보다 큰 Te가 나오면 새로운 Bof(s)라 설정한다. 여기서 s_j버킷은 r_i버킷의 튜플중 가장 큰 Te보다 큰 Ts들만 포함하는 S의 첫 버킷이다.



[그림 4-2] B=2일 때 기준값에 따른 S의 분할과정

[그림 4-1]은 제안된 방법에 대한 하나의 예제이다. 릴레이션 R과 S는 공히 각 페이지당 2개의 튜플을 저장하고 있으며 각각은 5, 6페이지로 구성되어 있다. 여기서 B = 2라 가정하였다. [그림 4-1]에서 R은 정렬되어 3개의 버킷(r_1, r_2, r_3)으로 나누어져 있다. S 역시 이미 분할되어 있는데 여기서 S의 분할 기준값은 b_1 = 1, b_2 = 8, b_3 = 14, b_4 = ∞이다. 이러한 기준값에 따른 S의 분할과정은 [그림 4-2]에서 보이고 있다. B가 2이므로 먼저 S를 읽어 s_1과 S_t의 버킷을 생성하고 다시 S_t를 읽어 s_2, s_3 버킷을 생성한다. 따라서 그 비용은 4|S|(=2|S|[logBN] = 2|S|[log23])보다 작다. 마지막 조인단계는 먼저 r_1의 모든 페이지를 2개의 버퍼에 읽은 후 s1의 페이지를 1개의 버퍼를 이용하여 순차적으로 읽으면서 조인하면 된다. 이때 Te가 13인 튜플이 나올 때 이 튜플을 새로운 Bof(s)라 설정하고 s_2버킷의 마지막 튜플까지

읽어서 조인을 수행한다. 다음 r_2 버킷을 2개의 버퍼 페이지에 읽고 새로운 Bof(s)부터 순차적으로 읽으면서 조인을 수행한다. b_2= 8이므로 r_2버킷에 대하여 조인을 완료하기 위해서는 s_3버킷의 마지막 튜플까지 읽는다. 이때 새로운 Bof(s)는 Te가 15인 첫 튜플이다. 마지막으로 r_3버킷을 2개의 버퍼 페이지에 읽고 Te가 15인 첫 튜플부터 s_3버킷의 마지막 튜플까지 읽으면서 조인을 완료한다. 앞의 설명 및 예제를 정리하면 제안된 방법은 다음과 같이 요약할 수 있다.

스텝1 : 작은 릴레이션이 정렬 및 분할 기준값을 설정한다.

① R을 Ts값에 따라 정렬한다.

② 정렬된 R에 대하여 ($b_1, b_2, \ldots, b_N$)의 분할 기준값을 설정한다.

스텝2 : 큰 릴레이션을 분할한다.

① ($b_1, b_2, \ldots, b_N$)에 기준하여 S를 ($s_1, s_2, \ldots, s_N$)으로 분할한다. s_1의 첫 튜플을 Bof(s)로 설정한다.

스텝3 : 모든 r_i에 대하여 다음을 반복한다.

① B개의 버퍼 페이지를 이용하여 r_i를 읽는다. 버퍼내의 튜플들 중 가장 큰 Ts를 Ts_{max}라 하고 가장 큰 Te를 Te_{max}라 한다.

② 하나의 페이지를 이용하여 S의 Bof(s)에서 시작하여 s_j의 마지막 튜플 까지 순차적으로 튜플을 읽어서 조인을 수행한다. 이때 Ts_{max}보다 큰 Ts를 갖는 S의 첫 튜플을 새로운 Bof(s)로 설정한다. 여기서 j는 $j \geq i$ 이고 $Te_{max} < b_j$를 만족하는 S버킷들의 가장 작은 인덱스이다.

스텝1의 ①에서의 작은 릴레이션의 정렬은 일반적인 외부 정렬방법을 이용할 수 있다. 스텝1의 ②는 분할 기준값을 찾는 것으로 알고리즘에 따르면 이러한 기준값을 찾기 위하여 최소한 [|R|/B]만큼의 보조 기억 장치의 액세스가 필요하다.

외부 정렬시 마지막 병합 단계에서 분할 기준값을 찾는 것이며 이 방법을 적용하면 스텝은 다음과 같이 수정할수 있다.

스텝1 : 작은 릴레이션의 정렬 및 분할 기준값을 설정한다.

① R을 Ts값에 따라 정렬한다. 정렬의 마지막 병합단계에서($b_1, b_2, \ldots, b_N$)의 분할 기준값을 설정한다.

상기 알고리즘의 스텝3은 정렬방법과 유사하다. 이것은 임의의 r_i버킷의 튜플은 s_i버킷 외의 다른 버킷의 튜플과도 조인이 가능하기 때문이다. 제안된 방법은 S의 분할 비용이 작은 대신 스텝 3에서 부분적인 중첩루프 방법이 필요하다.

5. 결 론

시간지원 데이터 모델은 시간에 따라 변화되는 자료를 처리할 수 있도록 시간 의미를 데이터 모델에 추가하게 된다. 시간지원 데이터 모델은 지원되는 시간에 따라 유효시간 데이터 모델, 거래시간 데이터 모델, 그리고 이원시간 데이터 모델이 있다. 유효시간 데이터 모델은 실세계에서 사건이 발생한 시간을 지원하는 모델이며, 거래 시간 데이터 모델은 사건이 데이터베이스에 수록

된 시간을 지원하는 모델로 유효시간 데이터 모델과 유사한 방법에 의해 데이터 모델을 표현하고, 이원시간 데이터 모델은 거래시간과 유효시간을 모두 지원하는 데이터 모델이다.

본 논문에서는 시간지원 데이터베이스에서 많이 사용되고 있는 시간조인 연산에 대한 새로운 알고리즘을 제안하였다. 제안된 알고리즘은 정적 데이터베이스에서 잘 알려진 Hybrid 방법을 시간조인 연산에 적용한 것으로 주요 개념은 작은 릴레이션은 정렬하고 큰 릴레이션은 정렬된 작은 릴레이션으로부터 얻어진 분할 기준값을 기초로 분할하는 것이다. 기존의 정렬병합 방법과 달리 큰 릴레이션을 분할하므로써 정렬 비용을 줄이고, 일반적으로 일어날 수 있는 다양한 조인 환경에 대하여 제안된 알고리즘이 기존의 정렬병합 방법보다 우수하다는 것을 알 수 있었다.

참 고 문 헌

[1] Shashi K.G., "A Homegeneous Relationsal Model and Query Languages for Temporal Database", ACM Transaction on Database Systems, Vol. 13, No. 4, pp. 418-448,1988

[2] Snodgree R., "The Temporal Query Language TQuel", ACM Transaction on Database Systems, Vol. 12, No. 2, pp. 247-289,1987.

[3] Stanley Y.W. Su, "A Temporal Knowledge Representation Model OSAM*/T and Its Query Language OQL/T", Processdings of the 17th Conference on VLDB, pp, 431-442, 1991.

[4] A.U. Tansel, j.Clifford,Shashi G., et al, "Temporal Databases: Theory, Design and Implementation", The Benjamin/Cummings Publishing Company, 1993

[5] D.Kim, K.Jeon, K.Jeong, K.Kim and K.Ryu, " A Temporal Database Management Main Memory Prototype", IEEE Tencon 94, 1994

[6] 신영호,"시간지원데이터베이스에서 분할기법을 이용한 조인 알고리즘에 관한 연구", 석사학위 논문, 광운대학교, 1995.

[7] 이재문, "시간지원 데이터베이스에서의 분할에 기초한 시간조인 알고리즘",한국정보과학회지, 1997.

[8] 하봉옥, "시간지원 데이터베이스 관리 시스템을 위한 2레벨 단위 로킹 기법의 설계 및 구현", 데이터베이스 연구회지, 1995.

[9] 이인홍, "시간지원 데이터모델 및 집계함수에 관한 연구", 한국정보과학회지,1995.

[10] Himawan G., A. Segev, "Query Processing Algorithms for Temporal Intersection Joins", Proceedings of the 7th International Conference on Data Engineering, pp. 336-344, 1991

[11] G.A Anav,"A Temporally Oriented Data Model,"ACM Transactions on Database System, 11, No.4,pp499-527, Dec. 1996.

[12] J. Clifford, A.U. Trasel, "On an Algebra for Historical Relational Database:Two Views,"in Proc. of AVM SIGMOD, pp.247-265, May 1985.

[13] J. Clifford, A. Croker, "The Historcal Relational Data Model and Algelbra Based on Lifespans," in Proc.of the International Conference on Data Engineering, IEEE Computer Society, pp528-537, Feb, June 1993.

[14] J. Clifford, Tomas Isakowitz, "On The Semantics of Transaction Time and Valid Time in Bitemporal Database,"Proceedings of International Workshop on
an infrastructure for Temporal Database, June 1993.