韓國 컴퓨터情報學會 論文誌
第11卷 第6號, 2006. 12

2006-11-6-4-4

이동 객체의 실시간 연속 질의를 위한 모바일 클라이언트-서버 시스템

주 해 종**, 최 창 훈*, 박 영 배**

Mobile Client-Server System for Realtime Continuous Query of Moving Objects

Hae-Jong Joo**, Chang-Hoon Choi*, Young-Bae Park**

요 약

무선 네트워크(Wireless Network)의 약한 연결성 및 접속단절, 모바일 클라이언트의 이동성, 모바일 클라이언트의 휴대성으로 인해 발생하는 모바일 데이터베이스 시스템(Mobile Database System) 관련 이슈들과 이 문제들을 해결하기 위한 연구들이 한창이다. 이동 컴퓨팅은 언제 어디서나 원하는 모든 정보를 이용할 수 있는 사용자의 편의성이나 성능 면에서의 요구를 만족시키고 있지만, 데이터 관리 측면에서는 해결되어야만 하는 많은 문제점들을 안고 있다. 본 논문은 모바일 클라이언트-서버(Mobile Client-Server) 환경에서 모바일 데이터베이스 시스템 특성상 가질 수 있는 무선 네트워크의 약한 연결성 및 접속성 단절로 인한 데이터베이스 비축(Database Hoarding)과 관련된 문제, 공유 데이터(Shared Data)의 일관성(Consistency)유지 문제, 그리고 로그 최적화 문제를 해결하기 위한 모바일 연속 질의 처리 시스템(MCQPS : Mobile Continuous Query Processing System)을 포함하는 새로운 모바일 클라이언트-서버 시스템을 설계하는데 목적이 있다. 또한, MCQPS의 효율성 증명을 위해 C-I-S(Client-Intercept -Server) 모델과의 성능비교를 통해 제안한 시스템이 우수하다는 것을 입증한다. 그리고 실시간 연속 질의를 위해 제안한 색인 구조와 기법의 효율성을 입증하기 위해 다양한 실험을 수행한다.

Abstract

Many researches are going on with regard to issues and problems related to mobile database systems, which are caused by the weak connectivity of wireless networks, the mobility and the portability of mobile clients. Mobile computing satisfies user's demands for convenience and performance to use information at any time and in any place, but it has many problems to be solved in the aspect of data management. The purpose of our study is to design Mobile Continuous Query Processing System(MCQPS) to solve problems related to database hoarding, the maintenance of shared data consistency and the optimization of logging, which are caused by the weak connectivity and disconnection of wireless networks inherent in mobile database systems under mobile client server environments. We proved the superiority of the proposed MCQPS by comparing its performance to the C I S(Client-Intercept-Srever) model. In

• 제1저자 : 주해종, 교신저자 : 최창훈
• 접수일 : 2006.12.12, 심사일 : 2006.12.23, 심사완료일 : 2006. 12.26
* 상주대학교 소프트웨어공학과 ** 명지대학교 컴퓨터공학과

Addition, we experiment on proposed index structure and methodology in various methods.

▸ Keyword : Mobile Client-Server System, MCQPS(Mobile Continuous Query Processing System), Realtime Continuous Query

Ⅰ. 서론

무선 네트워크(Wireless Network)의 약한 연결성 및 접속단절, 모바일 클라이언트의 이동성, 모바일 클라이언트의 휴대성으로 인해 발생하는 모바일 데이터베이스 시스템(Mobile Database System) 관련 이슈들과 이 문제들을 해결하기 위한 연구들도 한창이다[1,2,3,7]. 이동 컴퓨팅은 언제 어디서나 원하는 모든 정보를 이용할 수 있는 사용자의 편의성이나 성능 면에서의 요구를 만족시키고 있지만, 데이터 관리 측면에서는 해결되어야만 하는 많은 문제점들을 안고 있다[1,3].

이러한 문제점을 해결하기 위해, 본 논문은 모바일 클라이언트-서버 환경에서 모바일 데이터베이스 시스템 특성상 가질 수 있는 무선 네트워크의 약한 연결성 및 접속성 단절로 인한 데이터베이스 비축(Database Hoarding)과 관련된 문제, 공유 데이터(Shared Data)의 일관성(Consistency) 유지 문제, 그리고 로그(Log) 최적화 문제를 해결하는 새로운 모바일 클라이언트-서버 시스템의 구성을 제안하는데 목적이 있다. 특히, 이동 객체가 위치 이동을 하면서 인접한 위치의 정보를 얻기 위한 연속 질의를 요구할 시 이동 객체에게 실시간으로 변하는 위치 정보에 반응하고 연속적인 질의에 실시간으로 최적의 결과를 서비스하기 위한 인덱스 구조와 처리 기능을 가진 MCQPS(Mobile Continuous Query Processing System)를 포함하는 새로운 모바일 클라이언트-서버 시스템을 제안한다. 이 논문의 구성은 다음과 같다. 2장에서는 본 논문의 이론적 고찰을 위해 모바일 클라이언트-서버 구조와 모바일 데이터베이스 시스템 특성에 따른 문제 정의를 살펴본다. 3장에서는 모바일 데이터베이스 환경에 문제를 해결하기 위한 모바일 연속 질의 처리 시스템(MCQPS)을 포함하는 시스템의 요소 정의, 설계, 그리고 신뢰성 보장기법을 설명한다. 4장에서는 MCQPS의 효율성 증명을 위해 C-I-S(Client-Intercept -Server) 모델과의 성능비교를 통해 제안한 시스템이 우수하다는 것을 입증한다. 그리고 실시간 연속 질의를 위해 제안한 색인 구조와 기법의 효율성을 입증하기 위해 다양한 실험을 수행한다. 마지막으로 5장에서는 결론을 맺는다.

Ⅱ. 관련 연구

2.1 모바일 클라이언트-서버 모델 구조

그림 1의 모바일 환경에서의 일반적인 명령전달 구조는 응용 클라이언트(Application Client)가 무선망을 통해 유선망상의 서버로 요청(Request)하는 모델을 나타내며, Coda[1]에 의해 소개된 C-CA-S 모델은 소규모 UNIX 파일 시스템을 가진 서버와 대규모 클라이언트에 사용되며, 이 클라이언트 에이전트(CA)는 서버에 의해서 정상적으로 수행된 파일 시스템 연산들을 모바일 클라이언트를 대신에서 수행한다. 또한 C-SA-S 모델에서 서버 에이전트(SA)는 서버와의 인터페이스 작업을 수행하는 유선망에 존재하는 프록시(Proxy)와 같으며, 그러한 프록시를 서버 에이전트(SA)라고 부른다. 이와 같이 기존의 클라이언트-서버 모델의 기능을 향상시키기 위해서 클라이언트 또는 서버 측에 에이전트를 사용하였지만, 이동하는 동안 클라이언트 에이전트(CA)와 동작을 할 수 있도록 서버 또는 클라이언트 응용을 수정하기에 항상 수월한 것만 아니다. 이러한 난제를 해결하기 위해 클라이언트와 서버 양쪽에 에이전트를 두어, 쌍방의 에이전트간에 무선망과 유선망의 모든 통신을 담당하도록 한 것이 바로 C-I-S 모델이다.

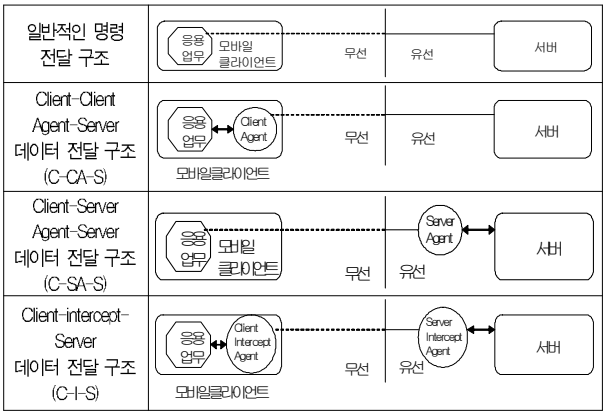


그림 1. 기존 모바일 클라이언트-서버 모델 구조
Fig. 1 The Existing Mobile Client-Server Model

2.2 기존 모바일 데이터베이스 시스템 동기화

모바일 데이터베이스 업체들은 중앙의 데이터베이스와 각 모바일 데이터베이스 사이에 동기화 서버가 존재하여 전체의 동기화를 이루는 방식을 취하고 있다. 모바일 데이터베이스 사이의 동기화는 고려되어 있지 않거나 동기화를 위하여 모바일 데이터베이스 사이에 새로운 동기화 서버를 각각 위치시키는 방식을 취하고 있다[5]. 대부분의 모바일 데이터베이스 업체들은 모바일 데이터베이스를 기존의 데이터베이스의 확장 개념으로 접근하고 있으며 중앙의 정적 데이터베이스와 일관성 유지를 위하여 하나의 중앙 싱크서버를 사용한다[5].

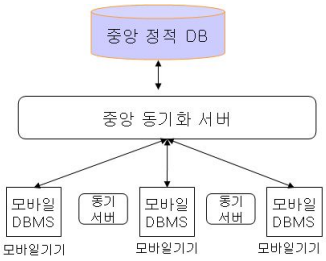


그림 2. 기존 모바일 데이터베이스 시스템 동기화
Fig. 2 The Existing Synchronization of Mobile Database System

2.3 모바일 데이터베이스 시스템 특성

그림 3과 같이 모바일 데이터베이스 환경에 적합한 데이터베이스 서비스를 설계할 때, 세 가지 요소를 고려해야 한다[1,7].

즉 첫째, 본질적으로 모바일 기기가 지니고 있는 무선망에서의 한계를 극복해야 하며 둘째, 특정 데이터베이스 응용에서 해당 데이터의 이용가능성 또는 현재성이 보장되어야 하고 셋째, 모바일 기기에 표시되는 데이터는 데이터 서버 또는 데이터 웨어하우즈로 부터 사용되는 데이터 일관성을 유지해야 한다는 사실이다.

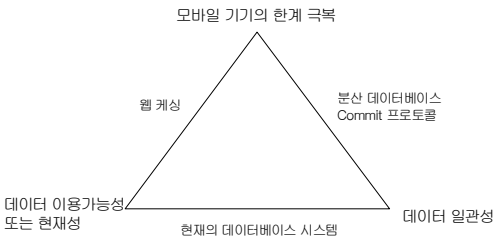


그림 3. 모바일 데이터베이스 환경 세 가지 고려 요소
Fig. 3 Various Consideration of Mobile Database Environments

2.4 이동 객체를 위한 연속적인 최근접 질의

이동 객체는 시간에 따라 위치 정보가 계속 변경되는 공간 객체를 의미한다. 이러한 이동 객체의 질의는 위치에 따라 결과는 유효하지 않을 수 있으므로 위치가 변경될 때마다 계속해서 새로운 결과를 요청해야 하기 때문에 위치 변경에 따라 연속적인 질의가 발생하며, 결과의 유효성을 검증하는 방법이 필요하다.[9][10]

연속적인 최근접 질의를 효율적으로 처리하기 위한 기법으로 R-tree에 정적인 객체 집합들의 Voronoi cell를 미리 계산하여 저장하는 기법[9]을 제안 했다. Voronoi cell이란 전체 정적 객체 집합 P = {P1,P2,...Pn}에서 pi와 인접한 객체들의 수직 이등분선들이 형성하는 영역이고, 이 영역 안에 있는 이동 객체의 최근접(Nearest Neighbor) 객체는 항상 pi다.

Voronoi cell을 기반으로 질의 결과의 유효성을 검증하여 연속적인 질의의 발생빈도를 감소시킴으로써 네트워크 부하를 줄일 수 있다.

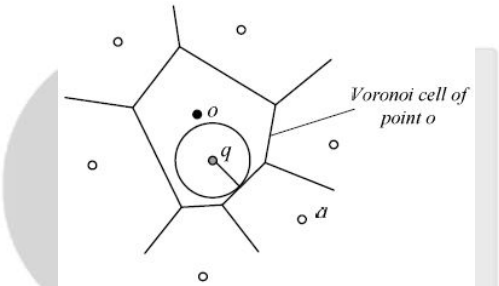


그림 4. Voronoi 다이아그램
Fig. 4 Voronoi Diagram

III. MCQPS 클라이언트-서버 시스템 구성

이 논문에서 제시하는 새로운 모바일 클라이언트-서버 시스템 구조는 클라이언트와 서버 양쪽에 에이전트를 가지는 시스템(C-I-S)이외에 모바일 클라이언트 지향적인 데이터 웨어하우즈 기능을 가진 추가적인 개체로 모바일 연속 질의 처리 시스템(MCQPS)으로 구성된다. 에이전트와 서버는 유선 네트워크를 통해 연결되어 있고, 에이전트는 모바일 클라이언트와 무선 네트워크를 통해 통신한다. 처리속도가 상대적으로 느리고 대용량 데이터 처리에 한계가 있는 모바

일 클라이언트와 무선 네트워크를 통한 통신 부하를 줄이기 위한 구조이다.

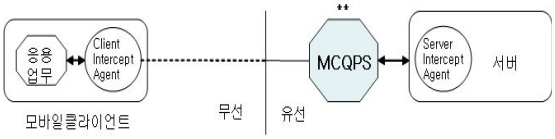


그림 5. MCQPS를 포함하는 모바일 클라이언트-서버 구조
Fig. 5 New Mobile Client-Server Model with MCQPS

3.1 시스템 구성요소

제안 시스템은 모바일 클라이언트, 모바일 에이전트와 MCQPS 서버로 구성되며, 그림 6과 같은 구조를 갖는다. 모바일 컴퓨팅 환경은 모바일 클라이언트(MC)라고 부르는 모바일 컴퓨터들과 컴퓨터들의 유선 네트워크들로 구성된다. 모바일 클라이언트는 모바일 지원 스테이션(MSS)이라고 하는 컴퓨터를 통해 컴퓨터의 유선 네트워크와 통신을 수행한다.

각 MSS는 자신이 지원 가능한 지리적 영역, 즉 셀 내부의 모바일 클라이언트들을 관리하며, 이벤트 중심의 실시간 모바일 네트워크 인터페이스를 담당한다. 모바일 에이전트는 실시간으로 모바일 클라이언트의 요청에 관한 질의를 요청하고 그 결과를 확인 할 수 있다.

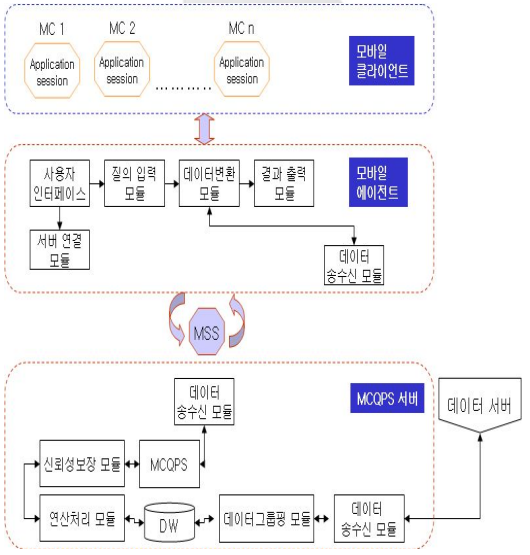


그림 6. MCQPS 클라이언트-서버 시스템 구조
Fig. 6 New Mobile Client-Server Framework with MCQPS

모바일 클라이언트의 구성은 사용자 인터페이스, 서버 연결 모듈, 질의 입력 모듈, 데이터 변환 모듈, 결과 출력 모듈, 데이터 송수신 모듈로 되어 있다. MCQPS 서버는 데이터 송수신 모듈, 데이터 그룹핑 모듈, 데이터웨어하우즈(DW), 연산처리 모듈, 데이터의 동기화와 일관성을 유지하는 신뢰성 보장 모듈, 그리고 MCQPS로 구성되어있다.

3.2 MCQPS 설계

3.2.1 내부구조

MCQPS는 접속단절이 발생하면 사용하였던 뷰에서 발생한 데이터 변화를 보관하고, 뷰의 복잡한 버전을 유지 관리하도록 다음과 같은 구성요소를 가지고 있어야 한다. 이러한 요소들을 포함하는 MCQPS의 내부구조는 그림 7과 같다.

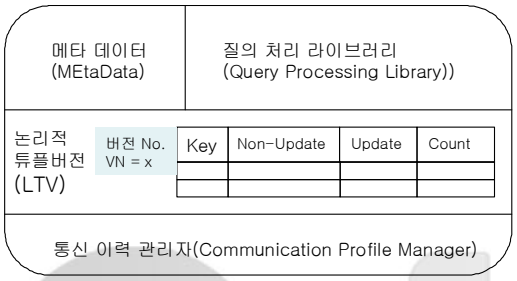


그림 7. 설계된 MCQPS의 내부 구조
Fig. 7 Design of MCQPS Structure

3.2.2 연속 질의 처리

(1) 이동 객체의 인접 영역을 최적화하는 방법

이동 객체의 연속질의시 인접 영역을 이용하여 질의 결과의 유효성을 검증함으로써 연속적인 질의에 대한 네트워크 부하를 줄일 수 있으며, 특정한 도메인에 대한 질의 처리에서 대상 객체의 정적 속성의 지배관계를 고려하여 최적화된 이동 객체의 인접 영역을 미리 계산하여 효율적으로 처리할 수 있다. 최적화된 이동 객체의 인접 영역(ONRi : Optimal Near -est Region)이란 대상 객체 p_i가 정적 속성에 대하여 지배하는 객체와 정적 속성에 대하여 p_i를 지배하는 객체 보다 이동 객체의 위치와 가까운 영역이다.

$$ONR_i = \left\{ q(x,y) \mid \begin{matrix} (dist(q,p_i) < dist(q,p_j),\ \forall p_j < p_i) \\ \wedge\ (dist(q,p_i) < dist(q,p_j),\ \forall p_i < p_j) \end{matrix} \right\}$$

즉, p_i의 최적화된 인접 영역 ONR_i은 p_i의 인접 영역을 형성하는 수직 이등분선들과 정적 속성에 대하여 p_i가 지배하는 대상 객체들과 생성하는 수직 이등분선들

이 형성하는 영역이다. 최적화된 인접 영역 영역을 형성하는 수직 이등분선은 이전에 기술한 인접 영역 결정 기법을 이용하여 결정할 수 있다.

이동 객체의 위치가 ONR_i에 포함 되는 경우, 이동 객체는 정적 속성에 대하여 p_I를 지배하는 객체 보다 p_I에 가까움으로 항상 질의 결과에 포함되고, 정적 속성에 대하여 p_i가 지배하는 객체들보다 가까움으로 p_I가 지배하는 객체는 항상 질의 결과에 포함되지 않는다.

(2) 이동 객체의 인접 영역 결정 방법

1단계 : 정적 속성에 대하여 p_I를 지배하는 객체들과 수직이등분선 및 부등식 영역 계산

2단계 : SR_I을 형성하는 수직이등분선들의 모든 교차점 계산.

3단계 : 연립부등식을 만족하는 교차점 계산 (영역의 꼭지점)

4단계 : 꼭지점 2개를 지나지 않는 선분 제거

(3) 최적화된 인접 영역의 데이터 구조

이동 객체의 인접 영역을 형성하는 수직이등분선은 $b \circ y = a \circ x + c$ 형태의 직선의 방정식으로 표현하여 방정식의 a, b, c계수 값과, 부등식 영역 검사를 위한 선분에 대한 p_I의 위치정보(Location), 영역의 객체 id와 지배하는 객체의 id를 저장한다. 그리고 질의 위치와 영역을 형성하는 선분과의 최단거리를 계산하기 위하여 직선 위의 두 꼭지점의 좌표 값(Xmin, Ymin, Xmax, Ymax)을 저장한다.

Line ID	Dominatee ID	Dominator ID	A	B	C	Location
Xmin	Ymin	Xmax	Ymax			

그림 8. 수직이등분선의 데이터 구조
Fig. 8 Data Structure of Vertical Bisector

최적화된 이동 객체의 인접 영역 대상 객체의 id와 최적화된 이동 객체의 인접 영역을 형성하는 수직이등분선의 id 리스트와 그리고 효율적인 영역검사를 위하여 최적화된 이동 객체의 인접 영역을 완전히 포함하는 객체의 id 리스트와 최적화된 이동 객체의 인접 영역에 겹치는 인접 영역의 수직이등분선의 id 리스트를 저장한다.

Object ID	Line_List	Enclosing Region List	Acrossing Line List

그림 9. 최적화된 인접 영역의 데이터 구조
Fig. 9 Optimal Data Structure of Nearest Region

(4) 최적화된 인접 영역기반의 인덱스 구조

대상 객체의 위치 좌표를 이용하여 R-tree에 색인하고 단말노드는 대상 객체의 위치 좌표와 최적화된 인접 영역 정보를 저장한다. R-tree에 대한 최근접 질의(Nearest Neighbor)를 이용하여 현재 위치와 가장 가까운 객체 p_I를 검색하고, 검색된 p_I의 최적화된 인접 영역 정보를 이용하여 이동 객체에 대한 인접 영역을 계산한다.

그림 10은 최적화된 인접 영역을 기반으로 검색대상의 위치 정보를 이용하여 R-tree에 의해 색인하는 R-트리 구성을 보여준다.

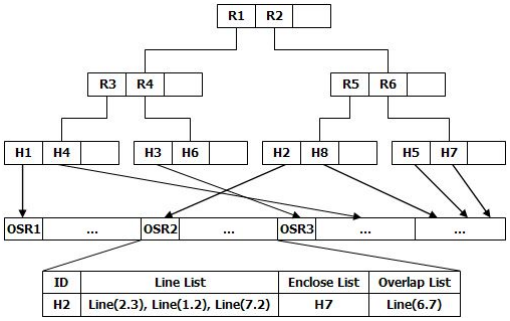


그림 10. 최적화된 인접 영역기반의 R-tree 구성
Fig. 10 Optimal R-tree Structure of Nearest Region

(5) 최적화된 인접 영역기반의 연속적인 질의 처리기법

이동 객체가 질의 후 계속해서 위치를 변경하면 어떤 한 시점에서 계산된 결과는 이후 다른 시점에서 유효하지 않을 수 있으므로, 위치 변경에 따라 연속적인 질의가 발생한다. 연속적인 질의는 네트워크 부하의 증가와 빈번한 재 계산의 문제점이 발생한다. 연속적인 질의의 발생 빈도를 최소화하기 위하여 질의 요청 위치좌표와 가장 가까운 수직이등분선과의 거리를 이용할 수 있다. 서버는 질의 요청에 대한 응답으로 질의 결과 집합, 질의 요청 위치좌표, 가장 가까운 수직이등분선과의 거리를 클라이언트에 전송한다. 클라이언트는 위치 변경에 따라 이전 결과의 질의 요청 위치 좌표와 변경된 위치 좌표간의 거리를 계산하여 가장 가까운 수직

이등분선과의 거리보다 작을 경우 이전 결과의 유효성을 보장할 수 있으므로 질의의 재요청을 하지 않을 수 있다.

그림 11에서 서버는 R-tree에 최근접 질의(Nearest Neighbor Query)를 이용하여 Q1과 가장 가까운 대상 객체 H2를 검색하고 H2의 최적화된 인접 영역 정보를 읽는다. H2의 최적화된 인접 영역 정보를 이용하여 현재 위치의 인접 영역을 계산하고, 질의 결과 집합 {H1, H2, H4, H6, H7, H8}과 질의 요청 위치좌표(10,9), 그리고 가장 가까운 수직이등분선과의 거리 r을 클라이언트에게 전송한다. 클라이언트는 변경된 위치가 반지름이 r인 원의 내부에 포함 되지 않을 때만 새로운 결과를 요청하여 질의의 발생 빈도를 감소시킬 수 있다.

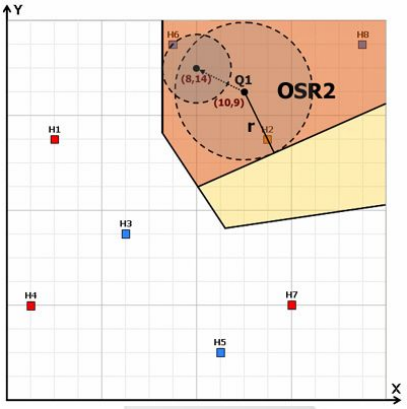


그림 11. 최적화된 인접 영역의 유효 영역
Fig. 11 Optimal Valid Region of Nearest Region

3.3 신뢰성 보장 기법

3.3.1 모바일 데이터베이스 시스템 3단계 동기화

기존 모바일 시스템의 동기화는 정적 데이터베이스의 데이터 일부를 사용하는 모바일 데이터베이스들을 대상으로 하며, 종적 데이터베이스와 모바일 데이터베이스 사이의 데이터 동기화는 주기적으로 발생한다고 가정한다. 그림 12는 기존 모바일 데이터베이스 동기화 시스템 전체에 대한 개요도 이다. 정적 데이터베이스는 단일 혹은 분산 데이터베이스로서 하나의 통합된 정보를 모바일 데이터베이스에게 제공한다. 또한 주기적인 동기화뿐만 아니라 필요에 따라 시스템 요청 에이전트를 이용하여 모바일 그룹간의 동기화를 이루다. 각 모바일 그룹은 그룹안에서 동기화를 유지하면서 동기화가 이루어진 다른 그룹과 질의 에이전트를 통하여 필요한 데이터를 처리하고 전체 시스템의 동기화를 처리한다. 모바일 데이터베이스 시스템 전체의 동기화를 위해 모바일 그룹내에서의 동기화, 모바일 그룹간의 동기화, 그리고 중앙의 정적 데이터베이스 동기화 3가지 단계를 거친다.

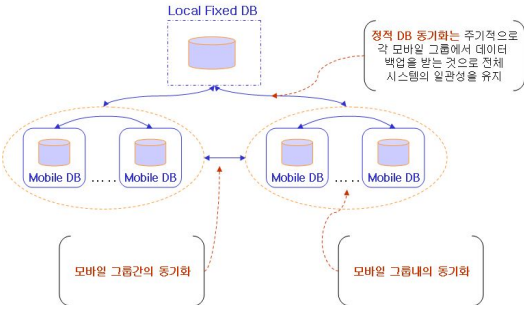


그림 12. 모바일 데이터베이스 시스템의 3단계 동기화
Fig. 12 Three-Level Synchronization of Mobile Database System

3.3.2 모바일 데이터베이스 그룹핑

모바일 데이터베이스 환경의 성능이 향상되고는 있지만 기존의 정적인 데이터베이스와 비교하여 작은 데이터 용량과 처리 능력을 가진 것은 확실하다. 이러한 환경에서 모바일 데이터베이스는 정적 데이터베이스의 선택적인 일부만을 복사하여 사용하게 된다[5]. 이는 각 모바일 데이터베이스가 자신들이 주로 사용하는 데이터의 종류를 구별할 수 있다는 것이다. 이러한 사실에 근거하여 각 모바일 데이터베이스를 각각의 사용에 따라 분류하여 그룹핑한다. 분류된 모바일 데이터베이스는 해당 모바일 그룹안에 배치되며 그룹별로 데이터의 일관성 유지와 현실성 유지 정책이 적용된다.

3.3.3 모바일 데이터베이스 일관성 유지

MCQPS의 또 다른 기능 설계 중에 하나는 MC의 사용 데이터의 일관성을 보장하기 위하여 트랜잭션이 발생한 후에 뷰와 데이터 소스를 사상(Mapping)하는 작업이 필요하다. 이를 위해서 본 논문에서는 다음과 같은 (정의 1)하에 MC의 사용 데이터 일관성 보장을 유지하는 방법으로 사용한다.

(정의 1) 데이터 일관성 보장 정의

데이터 소스 상태 벡터 ssv_k = [$ds1_k$, $ds2_k$, ..., dsm_k]는 주어진 트랜잭션 T_k가 실행된 후에 모든 데이터 소스의 상태를 포함한다고 하자. 그러면 T_k 트랜잭션을 수행한 후에 뷰에서 데이터 소스 상태 m까지의 사상(Mapping)이 다음과 같이 존재한다 :

1. 각 뷰의 상태 vs_j는 임의의 k에 대하여 타당한 상태 $m(vs_j) = ssv_k$를 반영한다.
2. 만약 $vs_j = vs_i$ 이면, $m(vs_j) < m(vs_i)$이다.

3.3.4 모바일 데이터베이스 현실성 유지

MCQPS의 다른 기능 설계 중에 하나는 MC의 현실 데이터의 이용가능성을 보장하기 위하여 서버측의 DBMS 에이전트와 연동되어 모니터링 뷰를 생성하여 현실성을 재계산하는 방법이다. 이를 위해 데이터 서버 DS1, DS2, DS3 각각의 r1(a,b), r2(b,c), r3 (c,d) 테이블이 있고, 이들 테이블을 조인하여 데이터의 현실성을 측정하기 위한모니터링 뷰가 생성되어 있다고 가정하자. 그림 13은 DBMS 에이전트와 모니터링 뷰가 연동되어 데이터의 현실성 평가를 하는 알고리즘이다.

```
public class Coordinating DBMS-agent() {

// Query DS2.r2
   SELECT DS2.r2.b, DS2.r2.c ;
   FROM DS2.r2 ;

   B = b and C = c ;
   local variable version number v = 0 ;

   Start view recomputation: {
      Supply query trip plan (query tree)
            with Query DS2.r2 results.
      Launch view evaluation DBMS-agent
        with query trip plan and version number v = 0
   } //end recomputation

   Begin Monitoring: {
   Every (Monitor_time) {
      SELECT DS2.r2.b, DS2.r2.c
      FROM DS2.r2

      if (B  ≠ b or C ≠ c) {
        version number v = v + 1 ;
        Start view recomputation: {
            Supply query trip plan (query tree)
                with Query DS2.r2 results ;

            Launch view evaluation DBMS-agent
                with query trip plan
                and current version number v ;
        B = b and C = c ;
        } //end recomputation
      } //end if
   } //end Every
   } //end Monitoring
   }
```

그림 13. 데이터 현실성 평가 알고리즘
Fig. 13 Algorithm for Data Currency Estimate

3.3.5 로그 최적화

현재의 모바일 환경에서는 MC에서 발생한 변경 내용을 MC에 로그로 기록하였다가 MC의 단절 후 재통합 시 사용하기 위해 로그를 사용하고 있다. 이는 또한 무선 통신 환경의 특성상 물리적 또는 시스템적 요소로 인한 잦은 고장이 발생할 때에도 활용가능하다[5]. 이러한 상황에 대처할 수 있는 고장허용의 관리 능력은 신뢰성 있는 시스템을 구축하기 위한 가장 중요한 요소 중의 하나이다. 아울러 로그의 크기를 최소화하는 것은 MC의 로컬 메모리 절약, 재통합 동안 무선 대역폭 전송시간 단축에도 유리한 조건이 된다.

Ⅳ. 실험 및 성능평가

이 논문에서 제안한 이동 객체의 실시간 연속질의 인접 영역 및 최적화된 인접 영역을 이용한 색인과 유효 영역(Safe Region)을 이용한 질의 처리기법의 성능을 평가하기 위해 대상 객체와 이동 객체의 모의 데이터를 생성하여 실험하였다. 최적화된 이동 객체의 인접 영역기반의 질의 처리 기법의 성능을 평가하기 위하여 기존의 C-I-S 모델과 비교 실험 하였다. 또 유효 영역을 이용한 질의 처리 기법의 성능을 평가하기 위하여 대상 객체의 수와 이동 객체의 속력에 따른 질의 요청 발생 빈도를 측정 하였다.

4.1 실험 환경

제안한 MCQPS 기법과 연속적인 실시간 질의를 효율적으로 처리하기 위한 유효 영역 기법은 Pentium-IV 2.8GHz 프로세서와 512MB의 메인 메모리, 80GB의 하드 디스크를 가진 Window XP 운영체제의 PC에서 자바언어(JSDK 1.5)를 이용하여 구현하였다.

성능 평가에 사용된 데이터 집합은 GSTD(Genera- tor of spatio Temporal Datasets)[9]을 통해 대상 객체 및 이동 객체의 위치 좌표를 5000x5000m의 2차원 영역에 균등(Uniform) 분포된 100, 200, 300개의 대상 객체와 1000개의 이동 객체를 생성하였다. 이동 객체의 속력은 랜덤 함수를 이용하여 가우시안(Gaussian) 분포로 생성하였고, 이동 객체가 가지는 속력은 1, 3, 5, 10, 15, 20m/s로 가정하였다.

이 논문에서 제안한 MCQPS기반의 색인 구조와 알고리즘의 성능을 평가하기 위해서 기존 C-I-S 모델의 질의 처리 기법을 적용한 방법과 비교 실험하였다. 표 1은 성능 평가에 사용된 파라미터를 나타낸다.

표 1. 실험 파라미터
Table 1. Experimentation Parameters

실험 파라미터	설명	값
ClientNum	이동 객체의 수	1000
ServiceNum	대상 객체의 수	100, 200, 300
UpperBound_Y	공간 좌표의 y좌표 최대값	5000
LowerBound_Y	공간 좌표의 y좌표 최소값	0
RightBound_X	공간 좌표의 x좌표 최대값	5000
LeftBound_X	공간 좌표의 x좌표 최소값	0
AVG_Speed	이동 객체의 평균 속력	1, 3, 5, 10, 15, 20

4.2 성능 평가

4.2.1 대상 객체 수에 따른 실험

본 논문에서 제안한 MCQPS 구조의 성능 평가를 위해, 2장에서 설명한 C-I-S 구조와 MC에서 요청한 응용 세션에 대한 데이터 신뢰성 보장 서비스를 위한 MCQPS와의 메시지 전달 빈도수와 데이터 전송 또는 메시지의 크기를 비교·분석하였다.

임의의 데이터 소스를 i 라 하면, 데이터 소스는 $1 \leq i \leq u$ 의 범위에 있다고 하자. 그리고 메시지의 크기를 N 이라 하면, 데이터 소스 i 의 메시지 크기는 Ni가 된다. 또한, ni 는 데이터의 갱신 횟수를 나타낸다. 이때 C-I-S의 메시지 전달 빈도수는 (식1)과 같고, 전송한 메시지의 크기는 (식2)와 같다[8].

$$O(1 + \sum_{j=1}^{u} nj(2u-1)) \quad \cdots\cdots (식1)$$

$$O(1)+\sum_{j=1}^{u} nj\ (O(1)+(\sum_{i=1,}^{u} (O(1)+O(1/Ni)))) \quad \cdots\cdots (식2)$$

제안한 MCQPS 구조에서, 임의의 메시지에 대해 그림 13에서 제시한 알고리즘에 의해 데이터의 정확성을 평가하므로 데이터 소스 u 에 대해 뷰 정확성 검사를 실시한다. 따라서 MCQPS 구조의 메시지 전달 빈도수는 (식3)과 같다.

$$O(u) \quad \cdots\cdots (식3)$$

그리고 MCQPS 구조에서 전송한 메시지 또는 데이터의 크기는, 초기 전송 메시지의 크기가 N1인 메시지를 다음 사이트 DS2로 전송하면 메시지의 크기는 N1+N2가 되며, 이 메시지를 DS3로 전송하게 된다, 그러므로 MCQPS 구조에서 전송한 메시지의 크기는 (식4)와 같다.

$$N1 + (N1+N2) + (N1+N2+N3) + \ldots\ldots$$
$$= Nu + \sum_{j=1}^{u-1} Nj(u-j) \quad \cdots\cdots (식4)$$

모바일 컴퓨팅 환경에서 무선 업무처리를 요하는 사용자가 증가함에 따라 무선망의 약한 연결성과 접속 단절시의 데이터 정확성과 신뢰성 보장 문제는 모바일 데이터베이스 환경에서 중요한 문제가 아닐 수 없다. 따라서 무선 환경의 사용자가 증가하더라도 안정적이고 신뢰성있는 서비스를 제공하기 위하여, 모바일 클라이언트(MC)의 요청 메시지 전달 빈도수를 줄이는 요소와 모바일 기기의 한계성을 극복하기 위해 전송 메시지의 크기를 최소화하는 요소는 모바일 데이터베이스 환경에서 안정된 서비스를 제공하기 위한 중요한 요소이다.

따라서 두 가지 요소를 가지고 C-I-S 구조와 MCQPS 구조의 성능 비교를 하여, 사용자 증가에 따라 메시지 사용 빈도수와 메시지의 크기가 최적화가 무선 컴퓨팅 환경 서비스에 얼마만큼 중요한 영향을 미치는가를 살펴보면 표 2와 같다.

표 2. 사용자 데이터 처리 요청 증가에 따른 두 가지 요소의 비교표
Table. 2 Comparison List of Two Elements

	메시지 전달 빈도수		전달 메시지 크기	
데이터 소스(u)	C-I-S	MCQPS	C-I-S	MCQPS
1	2	1	3	1
10	1046	10	687	175
50	126226	50	67894	20875
100	1004951	100	523151	166750
500	125124751	500	63175600	20833750
1000	1000499501	1000	502873873	166667500

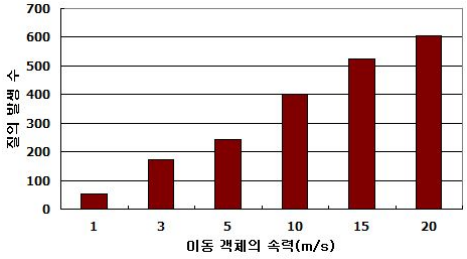
※ 여기서, 초기 메시지 크기 N =1 / 데이터 갱신 횟수 n = 1로 가정한 결과치 임.

표 2는 사용자 요청 수에 따른 기존 C-I-S 구조의 메시지 사용 빈도수와 메시지 크기의 최적화를 제안한 MCQPS 구조의 메시지 사용 빈도수와 메시지 크기 최적화와 비교한 결과를 보여준다. 모바일 컴퓨팅 환경에서 사용자의 증가에 따라, 안정적이고 신뢰성 있는 서비스를 위해 제안한 MCQPS가 C-I-S 구조 보다 우수하다는 것을 확인할 수 있다.

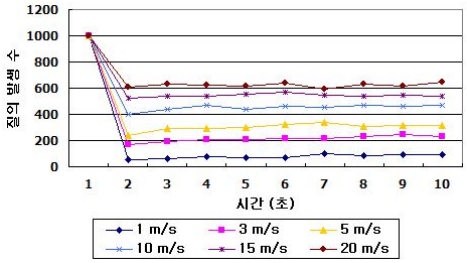
4.2.2 이동 객체의 속력에 따른 실험

제안하는 MCQPS 기반의 최적화된 인접 영역을 이용하여 연속적인 질의에 대한 유효성 검증 기법의 성능을 실험하기 위해 5000x5000m의 2차원 영역에 균등 분포된 100개의 대상 객체를 이용하였다. 대상 객체의 정적인 속성은 1에서 10사이의 균등 분포된 값을 가진다. 균등 분포된 위치 좌표와 정적 속성 값을 가지는 100개의 대상 객체에 대해서 최적화된 인접 영역을 미리 계산하여 색인하고, 1000개의 이동 객체에 대한 평균 속력을 가우시안 분포로 1, 3,

5, 10, 15, 20m/s까지 변경시키면서 질의 발생 빈도를 측정하였다. 그림 14의 (a),(b)에서 질의 발생 횟수를 그래프로 표현하였다.



(a) 질의 발생 횟수



(b) 10초간 질의 발생횟수 비교

그림 14. 이동 객체의 속력에 따른 질의 발생 횟수
Fig. 14 Query Occurrence Counts of Moving Object Speed

실험 결과 그림 14-(a)와 같이 이동 객체의 속력이 증가할수록 이전 질의의 유효영역을 벗어나는 객체가 증가하여 질의 발생 횟수가 증가하며, 속력에 따른 질의 발생 횟수는 그림 14-(b)에서 보는 바와 같이 최소 37.7%에서 최대 92.3%의 감소 효과를 볼 수 있었다. 연속적인 질의에 대해서 유효영역을 고려하지 않을 경우 1000개의 이동 객체가 위치를 변경하였을 경우 1000번의 새로운 질의가 발생하지만, 인접 영역을 이용하여 이전 질의의 유효성을 검증한 경우 평균 369번의 새로운 질의가 발생하였다. 특히 1m/s의 속력을 가지는 느린 이동 객체는 위치 변경에 따른 새로운 질의 발생 횟수는 평균 77번으로 92.3%의 감소율을 보여 큰 성능향상이 있었음을 알 수 있었다. 또한 20m/s의 상대적으로 빠른 이동 객체는 평균 624번의 새로운 질의가 발생하여 37.7%가 감소하였고, 속력에 따른 전체 이동 객체의 질의 발생 횟수는 평균 63.2% 감소하였다. 유효영역을 이용하여 이동 객체에 대한 연속적인 인접 영역 질의를 처리함으로써 이동 객체의 위치 변경에 따라 발생하는 질의 횟수를 감소시켜 네트워크 부하 및 서버의 재 계산 비용을 감소시킬 수 있었다.

V. 결론

본 논문에서는 모바일 데이터베이스 환경에서 무선망의 약한 연결성과 접속단절 시에 모바일 컴퓨팅 과정에서 발생할 수 있는 데이터 처리의 안정성과 사용 데이터의 일관성 유지, 그리고 대용량 데이터 사용시의 효율적인 모바일 기기의 한계성 극복을 위해 MCQPS를 포함하는 새로운 모바일 클라이언트-서버 시스템을 제안하였다.

기존 연구에서, 이동 객체에 대한 질의 처리 기법들은 단순히 동적인 속성인 대상 객체와의 거리만을 고려하고, 이동 객체의 인접 영역 질의 처리 기법들은 질의 위치와는 무관한 정적 속성만을 고려하였기 때문에 이동 객체에 대한 연속적인 인접 영역 질의를 처리할 수 없었다.

이 연구에서는 이동 객체의 속도와 방향에는 무관한 인접 영역(Nearest Region)을 정의하고 영역 결정 방법을 제안하였다. 또한 질의를 효율적으로 처리하기 위하여 최적화된 인접 영역(Optimal Nearest Region) 결정 방법과 확장된 색인구조 및 질의 처리 기법을 제안하고 구현하였다

다양한 실험을 통하여 질의의 빈도수를 최대 92.3%까지 감소시켜 네트워크 및 서버의 부하를 줄일 수 있었다. 이 논문에서 제안한 MCQPS 기반의 이동 객체에 대한 인접 영역 연속 질의 기법은 다양한 위치 기반 서비스 분야에 활용할 수 있다.

참고문헌

[1] 주해종,박영배, “모바일 데이터베이스 환경의 신뢰성 보장 질의처리 시스템 설계”, 정보처리학회 논문지D 12권 4호, pp.521~530, 2005.8

[2] 이재우, “모바일 데이터베이스 응용 사례”, 데이터베이스연구회, 17권 3호, pp.115~118, 2001.9

[3] 최미선, 김영국, “이동(Mobile) 데이터베이스 개요 및 연구 현황”, 데이터베이스연구회, 17권 3호, pp.3~16, 2001.9

[4] Margaret H. Dunham and Vijay Kummer,

"Impact of Mobility on Transaction Management", Proceeding of the International Workshop on Data Engineering for Wireless and Mobile Access, pp.14~21, August 1999

[5] H. Joanne and A. Divyakant, "Planned Disconnections for Mobile Database", Proceedings of IEEE 11th international workshop, 2000

[6] Sanjay Kummar Madria, Bharat K. Bhargava, "A Transaction Model to Improve Data Availability in Mobile Computing", Distributed & Parallel Databases 10(2):127~160, 2001

[7] G. Walborn and P. K. Chrysanthis, "Proceeding in Mobile Database Applications", In Proceeding of the 14th Symposium on the Reliable Distributed Systems, September 1995

[8] S. W. Lauzac, "Utilizing Customized Materialized Views to Create Database Services Suitable for Mobile Database Applications", ph.D thesis, Pittsburgh Univ., 2001

[9] Y. Theodoridis, J. R.O. Silva, and Mario A. Nascimento, On the Generation of Spatiotemporal Datasets, In Proceedings of the 6th Int'l Symposium on Large Spatial Database(SSD), 1999

[10] Zheng B., Lee, D. "Semantic Caching in Location - Dependent Query Processing" SSTD, p.97-116, 2001

[11] Song, Z., Roussopoulos, N. K-Nearest Neighbor Search for Moving Query Point. SSTD, 2001.

[12] Borzsonyi, S, Kossmann, D., Stocker, K. "The Skyline Operator" In ICDE, p.421-430, 2001.

[13] Tan, K., Eng, P. Ooi, B. "Efficient Progressive Skyline Computation" In VLDB, p.301-310, 2001.

[14] D. Kossmann, F. Ramsak, S. Rost, "Shooting Stars in the Sky: an Online Algorithm for Skyline Queries." In VLDB, p.275-286, 2002

저 자 소 개

주해종

1998년 명지대학교 일반대학원 컴퓨터공학과(공학박사)

(현재) 명지대학교 IT교육센터 연구교수

(관심분야) 모바일 DB, 전자지불결제, 멀티미디어 DB

최창훈

1997년 서강대학교 일반대학원 전자계산학과(공학박사)

(현재) 상주대학교 소프트웨어공학과 부교수

(관심분야) 병렬처리, 클라이언트-서버 시스템

박영배

1993년 서울대학교 일반대학원 컴퓨터공학과(공학박사)

(현재) 명지대학교 컴퓨터공학과 교수

(관심분야) 웹 DB, 시계열패턴인식, 다차원 인덱싱