일관성 제어를 지원하는 클라이언트 공간뷰의 설계 및 구현

임 덕성[O*], 반 재훈[*], 문 상호[**], 홍 봉희[*]

[*]부산대학교 컴퓨터공학과, [**]위덕대학교 컴퓨터공학과

Design and Implementation of Client Spatial Views to Support Consistency Control

Duk-Sung Lim[O*], Chae-Hoon Ban[*], Sang-Ho Moon[**], Bong-Hee Hong[*]

[*]Dept of Computer Engineering, Pusan National University [**]Dept of Computer Engineering, Uiduk University

요 약

클라이언트/서버 환경에서 서버의 공간 데이터를 검색, 접근하기 위한 공간 질의는 대용량의 공간 객체에 대해 복잡한 공간 연산을 수행한다. 그러므로 클라이언트가 자주 이용하는 공간 질의를 매번 서버에서 처리하는 경우에 서버의 부하가 증가하며 질의 응답시간도 길어지게 된다. 따라서 이러한 공간 질의를 뷰로 정의하고 클라이언트에 실체화하면 질의를 효율적으로 처리할 수 있다. 이 경우에 공간뷰를 유도한 서버의 소스 객체의 변경에 따라 클라이언트의 실체화된 뷰 객체의 일관성을 유지해야 한다.

본 논문에서는 클라이언트/서버 환경에서 공간뷰 개념을 확장한 클라이언트 공간뷰를 정의하고 일관성 유지를 위한 알고리즘을 제시한다. 그리고 상용 지리정보시스템인 고딕에서 클라이언트 공간뷰 시스템을 설계 및 구현한다. 마지막으로 질의 재수행과 본 논문에서 제시한 클라이언트 공간뷰의 성능을 실제 데이터를 이용하여 비교 평가한다.

1. 서론

최근 LAN, WAN 환경에서 네트워크 속도의 향상과 빠른 처리 속도를 가진 PC의 등장으로 클라이언트/서버 컴퓨팅 환경이 보편화되고 있다[3]. 특히 지리정보시스템(GIS)에서는 대용량의 공간 데이터를 취급하므로 서버에 공간 데이터를 저장 및 관리하고 다수의 클라이언트들이 검색하는 클라이언트/서버 컴퓨팅 환경이 효율적이다. 그러나 이 환경에서는 클라이언트들의 수와 질의 횟수가 증가함에 따라 서버의 부하가 증가하고 질의 응답 시간이 길어지는 문제점이 발생한다. 따라서 클라이언트가 서버의 공간 데이터를 효율적으로 접근하는 방법의 제시가 필요하다.

기존 연구에서는 클라이언트/서버 환경에서 질의 수행 속도를 향상시키기 위해 캐쉬 또는 중복 질의 방법이 제시되었다[1,2,3]. 그러나 클라이언트에 저장된 캐쉬는 데이터의 영속성을 보장하지 못하며, 중복 질의는 서버의 데이터가 변경되면 클라이언트에 저장된 데이터의 일관성을 보장하지 못하는 단점이 있다.

클라이언트 공간뷰는 클라이언트/서버 GIS환경에서 효율적인 공간 질의 처리를 위하여 자주 사용하는 질의를 뷰로 정의한 것이다. 그리고 클라이언트 공간뷰를 이용하여 빠른 질의 처리를 수행하기 위해서는 공간뷰를 실체화하여 클라이언트에 저장하는 것이 필요하다. 따라서 동일한 질의 수행시에 실체화된 클라이언트 공간뷰를 사용함으로써 질의 처리 수행을 위한 서버의 부하를 감소시키며 서버와 클라이언트간의 메시지 전송량을 줄이므로 효율적이다.

본 논문에서는 클라이언트 공간뷰의 정의, 실체화 및 일관성 유지 알고리즘을 제시하고 상용 GIS 서버인 GOTHIC[4]을 사용하여 구현한다. 본 논문의 구성은 다음과 같다. 먼저 2장에서는 클라이언트 공간뷰의 개념을 설명하고 3장에서는 클라이언트 공간뷰 시스템 구조에 대하여 설명한다. 4장에서는 시스템의 세부 알고리즘에 대하여 설명하며, 5장에서는 실험을 통해 성능을 평가한다. 마지막으로 6장에서는 결론 및 향후 연구를 기술한다.

2. 클라이언트 공간뷰

기존의 공간뷰와 마찬가지로 클라이언트 공간뷰에서도 정의, 시멘틱을 포함한 뷰 모델, 실체화 방법, 실체화된 뷰의 일관성 유지 방법이 제시되어야 한다. 먼저 클라이언트 공간뷰 모델은 기존의 공간뷰 모델과 동일하며 단지 클라이언트에서 자주 사용하는 질의를 공간뷰로 저장한다. 실체화는 정의한 공간뷰의 객체를 생성하여 클라이언트에 저장한다. 마지막으로 일관성 유지에서는 서버 객체의 변경에 따라 영향받는 클라이언트의 공간뷰 객체를 점진적으로 변경한다. 이를 위하여 추가정보를 이용한다.

본 논문의 이전 연구에서 클라이언트 공간뷰는 앞에서 언급한 사항들에 따라 4가지 유형으로 분류하였다[6]. 본 논문에서는 정의, 실체화, 일관성 유지를 서버와 클라이언트가 분담하는 유형 4의 클라이언트 공간뷰에 대하여 다룬다.

2.1 공간뷰 정의

클라이언트 공간뷰는 클라이언트에서 자주 사용하는 질의를 공간뷰로 정의하는 것이다. 따라서 클라이언트 공간뷰는 공간 질의에 따라 다양하게 표현되고 사용자 관점을 정의할 수 있다. 클라이언트는 서버로부터 공간뷰를 유도하는 소스 클래스(들)의 스키마 정보를 전송받아 공간뷰를 정의한다. 그리고 서버는 정의된 공간뷰와 소스 클래스들 간의 관련성 정보를 생성한다.

2.2 실체화

기존 연구에서 뷰의 실체화 방법은 식별자 유지 방법과 값-복사에

의한 방법이 있다[5] 식별자 유지 방법은 뷰 객체를 유도하는 소스 객체(들)의 식별자만을 저장하여 실체화하는 방법이다 반면에 값-복사 방법은 뷰 객체가 공간 속성과 비공간 속성 값을 직접 저장하여 실체화하는 방법이다 본 논문에서는 질의 수행 속도를 향상시키고 네트워크의 부하를 줄이기 위해 값-복사 방법을 이용한다.

클라이언트 공간뷰의 실체화 작업은 크게 서버와 클라이언트 작업으로 분류된다 서버는 클라이언트에서 요청하는 뷰-정의 질의를 수행하여 선별된 후보 객체들을 클라이언트에 전송하는 작업을 담당한다 클라이언트 작업은 다시 3가지로 분류된다 첫째 공간뷰를 실체화하기 위해 서버에 전달할 뷰-정의 질의를 생성한다 둘째 서버에서 전송된 후보 객체를 이용하여 뷰 객체를 생성하고 저장한다 마지막으로 소스 객체 변경시 실체화된 공간뷰 객체의 일관성 유지를 위해 뷰 유도 관련성을 생성한다.

2.3 추가정보

일관성 유지를 위한 추가 정보로서 유도 관련성(derivation relationship)과 변경 로그(update log)를 이용한다.

유도 관련성은 소스 클래스와 공간뷰 클래스 간의 클래스 유도 관련성(CDR, Class Derivation Relationship)과 소스 객체와 뷰 객체간의 뷰 유도 관련성(VDR· View Derivation Relationship)이다[5]

변경 로그는 서버에서 변경된 객체에 대한 정보를 클라이언트에게 제공하기 위한 추가 정보로서 <Update-type, Oid, Timestamp> 의 구조를 가진다[6] Update-type은 변경된 소스 객체의 변경 형태로서 삽입, 삭제, 수정의 3가지이다 Oid는 변경된 소스 객체의 식별자를 나타내며 Timestamp는 객체가 변경된 시간을 알려주는 정보이다.

2.4 일관성 유지

뷰 객체를 유도하는 서버의 소스 객체가 변경되는 경우에는 대응되는 뷰 객체의 일관성을 유지해야 한다 뷰의 일관성 유지 방법은 크게 변경 방법에 따라 재계산(Recomputation)과 점진적 변경(Incremental Update)으로 분류하며, 변경 시간에 따라 즉시 변경(Immediate Update)과 차후 변경(Deferred Update)으로 분류한다[2].

본 논문에서는 효율적인 뷰 객체의 변경을 이용하며, 이것을 위하여 추가 정보를 이용한다 그리고 변경 시점을 뷰에 대한 질의 처리 요구가 있는 경우 변경하는 요구시-처리 방법을 이용한다

3. 클라이언트 공간뷰 시스템 구조

본 논문에서 구현한 클라이언트 공간뷰 시스템의 전체 구조는 크게 클라이언트 모듈과 서버 모듈로 구성되며 공간뷰의 정의, 실체화 및 일관성 유지를 수행한다

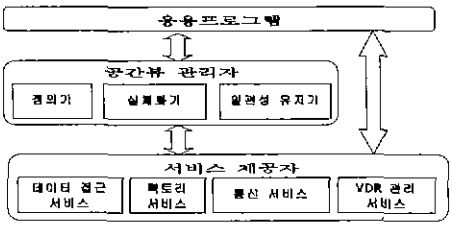

그림 1 클라이언트 구조

클라이언트 모듈은 그림 1과 같이 3계층으로 구성된다 하위 계층인 서비스 제공자는 상위 계층에 공통적인 서비스를 제공한다 서비스 제공자는 실체화된 공간뷰 객체를 접근하기 위한 데이터 접근 서비스, 통신 패킷 및 공간뷰 객체를 생성하기 위한 팩토리 서비스, 서버와의 통신을 담당하는 통신 서비스와 뷰 객체와 소스 객체간의 유도 관련성을 유지하는 VDR 관리 서비스로 구성된다

공간뷰 관리자는 정의기, 실체화기, 일관성 유지기로 구성된다 정의기는 공간뷰를 정의하기 위한 인터페이스를 제공하며, 실체화기는 서버가 전송한 뷰-정의 질의의 결과를 이용하여 공간뷰 객체를 생성하는 작업과 VDR을 생성하는 작업을 수행한다 일관성 유지기는 클라이언트에 저장된 뷰 객체들의 일관성 유지를 위해 서버에 변경 정보를 요청하고, 변경된 정보가 있는 경우에 뷰 객체를 점진적으로 변경하여 일관성을 유지한다

서버 모듈은 그림 2와 같이 3계층으로 구성된다. 하위 계층인 서비스 제공자는 상용 공간 데이터베이스인 GOTHIC의 API를 이용하며, 공간 질의 처리를 위한 질의 서비스, 통신 패킷 생성을 위한 팩토리 서비스, 통신 패킷을 처리하는 통신 서비스와 CDR과 Log를 관리하는 CDR & Log 서비스로 구성된다

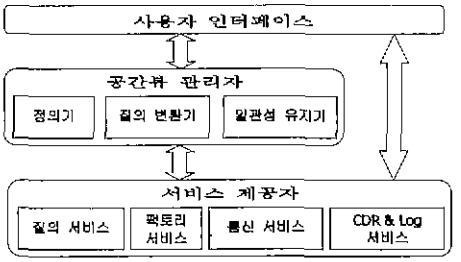

그림 2 서버의 구조

공간뷰 뷰 관리자는 정의기, 질의 변환기, 일관성 유지기로 구성된다 정의기는 클라이언트가 요청한 공간뷰 정의문을 이용하여 소스 클래스와 공간뷰 클래스간의 유도 관련성 정보인 CDR을 생성한다. 질의 변환기는 서비스 제공자의 질의 서비스를 이용하기 위해 뷰-정의 질의의 변환을 담당한다 일관성 유지기는 소스 객체의 변경시 변경 정보를 저장하고, 클라이언트에게 변경된 정보를 전송하는 역할을 수행한다

4. 클라이언트 공간뷰 시스템의 세부 알고리즘

이 장에서는 클라이언트 공간뷰 시스템에서 사용한 세부 알고리즘을 제시한다 세부 알고리즘에는 공간뷰 정의, 실체화, 일관성 유지 알고리즘이 있다

4.1 공간뷰 정의 알고리즘

공간뷰 정의 알고리즘은 클라이언트가 공간뷰 정의문을 생성하여 서버로 전송하고, 서버는 전송된 뷰정의문을 이용하여 CDR정보를 수정하는 작업을 한다. 그림 3은 클라이언트에서 공간뷰 정의 요청시 서버에서 수행되는 알고리즘이다

```
SVD    Spatial View Definiton          SA    Source Attribute
CDR    View Derivation Relationship    t     Time

Procedure DefineView(SVD)
begin
        {SA_i} = GetSourceAttribute (SVD)
        t = GetCurrentTime(),
        for each  SA_i  in  {SA_i}
        begin
                if (IsInAttrInCDR(SA_i))   // CDR 정보가 존재할 경우 CDR 수정
                        UpdateCDR (SVD_i,SA_i),
                else                       // CDR 정보가 없는 경우 새로 생성
                        InsertCDR (SVD_i,SA_i),
        end for,
end,
```

그림 3 공간뷰 정의 알고리즘

먼저 공간뷰 정의문에 포함된 속성 정보가 CDR에 존재하는 경우에는 UpdateCDR()을 호출하여 CDR에 공간뷰 정보를 추가한다 반면에 공간뷰에 관한 정보가 CDR에 없는 경우에는 InsertCDR()을

호출하여 소스 클래스와 공간뷰 클래스와의 관련성 정보를 추가한다

4.2 실체화 알고리즘

클라이언트 공간뷰의 실체화 알고리즘은 서버가 공간뷰-정의 질의를 수행하여 후보객체를 생성한 후 전송하고, 클라이언트는 후보 객체를 이용하여 공간뷰 객체와 VDR정보를 생성하는 것이다 그림 4는 클라이언트에서의 실체화 알고리즘이다

```
SVD     Spatial View Definition          t      Time
{C_i}   Candidate Object Set             SVO    Spatial View Object

Procedure Materialize(SVD, t, {C_i})
begin
        for each C_i in {C_i}
        do
                if (FindVDR(C_i))//후보객체로부터 유도된 뷰 객체가 존재하는 경우
                        UpdateVDR(SVD, t, VDR_i, C_i),
                else            //후보객체로부터 유도된 뷰 객체가 없는 경우
                        SVO = CreateSVObject(SVD, C_i),
                        InsertVDR(SVD ,t , SVO, C_i),
                end if,
        end for,
end,
```

그림 4 실체화 알고리즘

먼저 서버에서 전송된 후보 객체 집합 중에서 후보 객체로부터 유도된 공간뷰 객체의 유무를 VDR정보를 이용하여 검색한다. 검색된 VDR정보가 존재하는 경우에는 뷰 유도 관련성중 1:M관계나 N:M관계에 해당하므로 공간뷰 객체를 생성하지 않고 UpdateVDR()을 호출하여 VDR정보만 수정하게 된다. 그러나 검색된 VDR정보가 없는 경우는 CreateSVObject()를 호출하여 새로운 공간뷰 객체를 생성하고 InsertVDR()을 호출하여 뷰 유도관련성 정보를 추가한다

4.3 일관성 유지 알고리즘

실체화된 공간뷰 객체에 대한 일관성 유지는 서버 작업과 클라이언트 작업으로 분류된다 그림 5는 서버에서 수행되는 공간뷰의 일관성 유지 알고리즘이다

```
SVD     Spatial View Definition          log_i   UpdateLog
t_c     the time of materializing in Client  Soid_i  Source Object ID

Procedure ConsistentControl(SVD, t_c)
begin
        if (Not IsConsistent(SVD, t_c))
        begin  // 소스 객체가 변경된 경우
                {log_i} = GetUpdatedLog(SVD, t_c)
                for each  log_i  in  {log_i}
                begin
                    switch (log_i Type)
                        Case Insert
                          { Soid_i } = MakeCandidateObjectForInsert(SVD, log_i),break,
                        Case Delete
                          Soid_i  = GetObjectIDFromLog(log_i), break,
                        Case Modify
                          { Soid_i } = MakeCandidateObjectForModify(SVD, log_i), break,
                    CandidateObjectInfo{} = AddCandidate({ Soid_i }, log_i Type),
                end for,
        end if,
        return CandidateObjectInfo{},
end,
```

그림 5 일관성 유지 알고리즘

서버 작업은 클라이언트의 일관성 유지 요청에 대해 CDR을 이용하여 소스 객체의 변경 유무를 검사하는 IsConsistency()함수를 호출한다 만약 소스 객체가 변경된 경우에는 GetUpdatedLog()를 호출하여 변경된 객체에 대한 로그를 얻는다 로그 정보 내의 변경 유형은 삽입, 삭제, 수정의 3가지 형태로 분류된다 삽입된 소스 객체가 공간뷰-정의 질의에 만족할 경우 후보 객체를 생성하여 변경 유형과 함께 후보객체 집합에 추가한다 삭제된 객체는 식별자와 변경유형을 후보객체 집합에 추가한다 수정은 삽입과 동일하지만 후보 객체만으로는 변경된 소스 객체의 식별이 불가능하므로 수정된 객체의 정보를 추가한다

5. 성능 평가

본 논문에서 구현한 클라이언트 공간뷰 시스템의 성능을 실제 데이터를 이용하여 실험 평가한다 그림 6은 소스 객체의 변경 수에 따라 질의를 재수행한 경우와 클라이언트 공간뷰를 이용한 경우에 대한 수행 성능을 나타낸다

실험 평가에 사용된 질의는 공간 연산자인 CROSSES를 사용한 공간 조인으로서 질의 대상 객체 수는 2160 X 542개이다 공간뷰 생성시에는 클라이언트 공간뷰가 재수행보다 평균 10%정도 느리다 그 이유는 뷰 객체를 실체화하고 일관성 유지를 위해 추가 정보인 VDR을 생성하기 때문이다 그러나 소스 객체가 변경된 경우 동일 질의 수행시에는 클라이언트에서 실체화된 공간뷰의 일관성을 유지하는 경우가 질의 재수행의 경우보다 빠르다

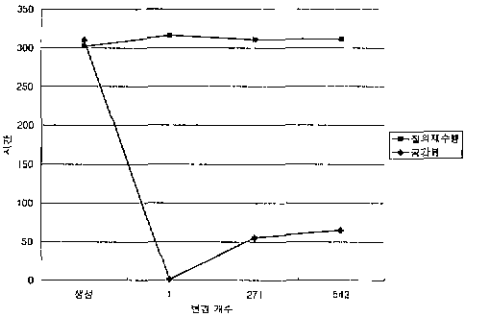

그림 6 클라이언트 공간뷰의 성능평가

6. 결론 및 향후 연구

본 논문에서는 클라이언트/서버 환경에서의 공간뷰를 설계하고 상용 지리정보시스템인 GOTHIC에서 구현하였다 먼저 클라이언트/서버환경에서 공간뷰 개념을 확장한 클라이언트 공간뷰를 정의하고 일관성 유지를 위한 알고리즘을 제시하였다 그리고 구현한 시스템을 실제 데이터를 이용하여 비교 평가하였다

향후 연구 과제는 이질적인 공간 데이터베이스에 접근할 수 있는 미들웨어 기반의 공간뷰 서비스 제공자에 대한 연구와 클라이언트/서버 환경에서 사용자 관점의 변경에 의해 공간뷰가 재정의된 경우에 일관성 유지를 위한 점진적 변경 방법에 대한 연구가 필요하다.

7. 참고 문헌

[1] Nick Roussopoulos , Hyunchul Kang "Principles and Techniques in the Design of ADMS±", IEEE Computer pages 19-25, 1986

[2] Nicholas Roussopoulos, "An incremental access method for ViewCache concept, algorithms, and cost analysis", ACM Trans Database Syst. Vol16, 3, pages 535-563, 1991

[3] Alex. Delis and N Roussopoulos, "Techniques for Update Handling in the Enhanced Client-Server DBMS", IEEE TOKD, Vol 10, 3, pages 458-476, 1998

[4] Laser-Scan Ltd, 'GOTHIC CONCEPTS", Training Course, 1995

[5] Sang-Ho Moon and Bong-Hee Hong, "Incremental Update Algorithms for Materialized Spatial Views by Using View Derivation Relationships', DEXA , pages 539-550, 1997

[6] 임덕성,반재훈,문상호,홍봉희, '클라이언트 공간뷰의 실체화 방법". 정보과학회 봄 학술발표 논문집 Vol 26, No 1, 1999