공간 데이터 웨어하우스에서 GML 데이터의 효율적인 적재를 위한 데이터 통합 기법

전병윤*, 이동욱*, 유병섭*, 배해영*
*인하대학교 컴퓨터 정보 공학과
e-mail : mysummit@dblab.inha.ac.kr

GML Data Integration Method for Load Processing of Spatial Data Warehouse

Byung-Yun Jeon*, Dong-Wook Lee*, Byeong-Seob You*, Hae-Young Bae*
*Dept. of Computer Science & Information Engineering, Inha University

요 약

GIS 분야에서 데이터 교환의 표준으로 OGC(Open Geospatial Consortium)에서 GML(Geography Markup Language)이 제안되어 웹 어플리케이션이나 공간 데이터 교환에서 사용이 일반화 되어가고 있다. 또한, 공간 데이터를 효과적으로 수집하여 의사결정을 지원하기 위한 시스템인 공간 데이터 웨어하우스에서도 GML 데이터를 추출하여 소스 데이터로 활용하는 것이 요구되고 있다. 하지만 GML 은 반구조형식(semi-structured)의 데이터 형식을 가진다. 따라서 기존 구조적인 데이터와는 추출하는 방식이 다르므로 GML 의 특징에 맞는 공간 데이터 추출이 수행되어야 한다.

본 논문에서는 공간 데이터 웨어하우스에서 GML 기반의 공간 데이터 소스를 추출할 때, 중복되는 공간 객체를 하나의 표현으로 통합하여 효율적으로 적재하는 기법을 제안한다. 이는 GQuery 를 이용하여 GML 데이터를 추출한 후, GML 스키마를 메타데이터에서 관리하는 스키마 정보와 비교하여 공간 데이터 웨어하우스에 통합된 공간 데이터를 제공하는 기법이다. 성능평가에서는 기존의 GML 데이터를 추출하는 기법과 제안기법과의 비교를 통하여 제안 기법의 기존 기법에 비해 평균적으로 약 9.95%의 성능향상을 보였다.

1. 서론

웹의 발전과 기업의 의사결정시스템의 발전으로 GIS 분야에서는 기존의 단일 사용자 환경에서 다양한 지리 정보 데이터 형식이 지원 가능한 상호운용성(interoperability)에 관심을 가지게 되었다[1].

이런 요구사항으로 인해 웹에서 지리정보 서비스를 지원하고, 지리 정보 데이터의 상호운용성을 높이기 위하여 지리정보 데이터 교환의 표준으로 OGC 에 의해 GML 이 제안되었다[2]. 또한, 공간 데이터를 포함하는 의사결정과 효과적인 OLAP 를 지원하기 위하여 공간 데이터 웨어하우스 시스템에서도 GML 데이터의 분석기능을 가지고 있어야 한다. 따라서 공간 데이터 웨어하우스에서 GML 데이터의 효율적인 추출 및 적재 방법을 고려해야 한다.

본 논문에서는 공간 데이터 웨어하우스에서 GML 데이터의 효율적인 적재를 위해 여러 GML 데이터 소스로부터 GML Query Language 인 GQuery 를 이용하여 공간 데이터를 추출하고 변환하고 적재하는 과정에서 GML Mediator 를 이용하여 GML 데이터를 Relational Data 로 효율적이고 통합된 형태의 적재 기법을 제안한다[3].

본 논문의 구성은 다음과 같다. 제 2 장 관련 연구에서는 본 연구의 기반이 되는 기존연구에 대해 살펴본다. 제 3 장에서는 본 논문에서 제안하는 공간 데이터 웨어하우스에서 효율적인 GML 데이터의 통합 기법에 대해서 서술한다. 제 4 장에서는 본 기법의 성능평가를 한 뒤, 마지막 제 5 장에서는 결론 및 향후 연

*본 연구는 정보통신부 및 정보통신연구진흥원의 대학 IT 연구센터 육성.지원사업의 연구결과로 수행되었음.

구를 논한다.

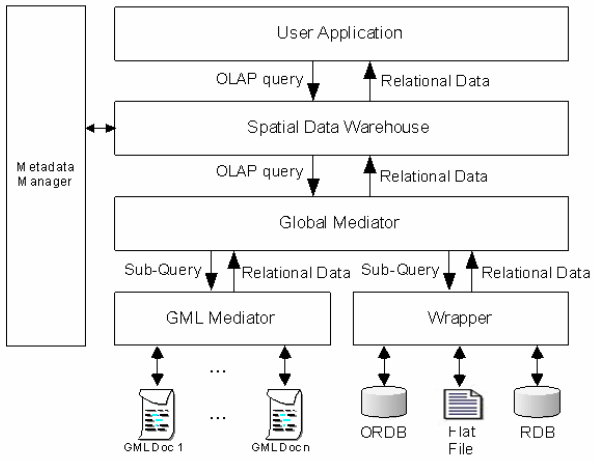
2. 관련 연구

기존연구는 XML 데이터를 데이터 웨어하우스에 통합기법이 있다. 이 기법은 XQuery 를 이용하여 여러 소스데이터로부터 XML 추출하여 통합된 XML 데이터를 생성하는 기법이다[4].

GML 은 XML 의 응용 언어이기 때문에 GML 데이터 통합에서도 XQuery 를 이용하는 것이 가능하다. 하지만 XQuery 를 사용하게 되면, 공간 데이터 앨리먼트도 다른 앨리먼트와 동등한 처리를 수행하게 된다. 즉, 공간 데이터의 특성을 충분히 고려하지 못하게 된다.

따라서 GML 의 공간 데이터 특성을 고려하여 GML 간의 통합 처리가 가능하다면, 공간 데이터 통합에 보다 효과적인 처리가 가능할 것이다. 이런 문제를 보완하기 위해서 XQuery Specification 의 사용자 정의 함수를 정의하는 기능을 이용하여 공간 연산자를 추가한 GQuery 가 있다. GQuery 를 사용하면, GML 의 데이터의 공간 앨리먼트를 보다 효과적으로 처리할 수 있다.

3. 공간 데이터 웨어하우스의 GML 데이터 통합

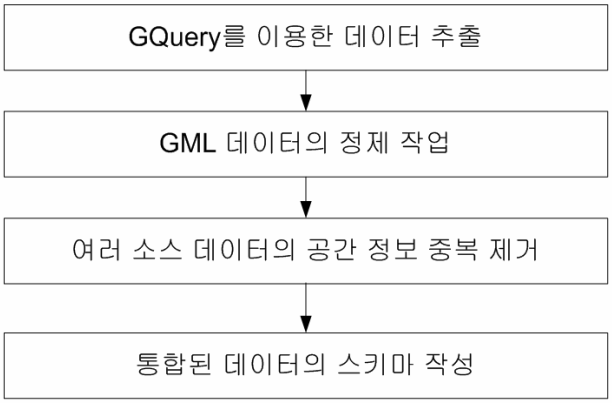


(그림 1) 공간 데이터 웨어하우스의 시스템 구조

위의 (그림 1)은 공간 데이터 웨어하우스의 시스템 구조이다. 공간 데이터 웨어하우스 시스템의 GML 데이터 통합은 GML Mediator 에서 수행된다.

소스 데이터 중 GML 데이터가 필요하다고 요청되면, GML Mediator 에서는 메타데이터의 소스 데이터를 참조하여 GML 의 소스 데이터에 알맞게 질의를 분할한다. 분할된 질의를 통하여 데이터를 추출하고 데이터에 대한 효율적인 통합을 수행한다. GML Mediator 는 Global Mediator 의 부하를 줄여주고, GML 데이터를 중복 없이 통합하고 적재하는 것이 가능하다.

다음 (그림 2)는 GML Mediator 에서의 GML 의 통합 단계의 과정을 설명한 것이다.



(그림 2) GML Mediator 의 GML 의 통합 단계

제안하는 GML 데이터의 통합 단계는 전체적으로 4 단계를 흐름을 거친다.

처음 단계는 메타정보의 소스정보를 이용하여 GQuery 를 분할하고, 그 질의를 이용하여 소스데이터를 추출한다.

두 번째 단계는 추출된 GML 문서를 맵핑 테이블과 여러 가지 정제 알고리즘을 통하여 메타데이터의 전역적인 표현으로 변환된다. 그리고 정제 작업에서 생기는 문서에 대한 스키마를 작성한다.

위의 두 단계를 통한 데이터들을 통하여 얻어진 GML 데이터는 공간 객체의 중복을 판단을 하여 공간 데이터의 중복제거 작업을 수행하는 것이 세 번째 단계이다.

마지막 단계에서는 통합된 GML 문서의 스키마를 통합하여 통합된 GML 문서의 유효성을 보장한다.

본 논문에서 제안하는 기법은 세 번째 단계와 네 번째 단계에서 통합하는 기법을 서술하는 것이다. 3.1 절은 제안기법의 통합과정에 대해서 서술한다.

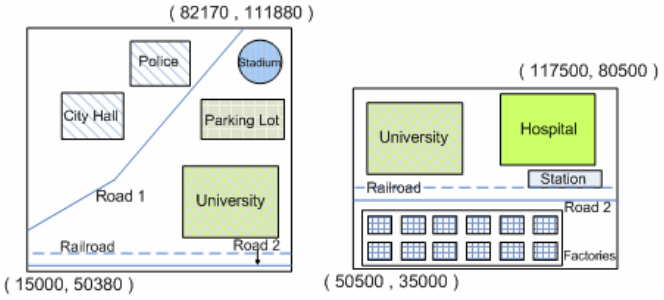
3. 1. 소스 데이터들의 공간 데이터 중복 제거 기법

여러 소스로부터 데이터가 추출되면, 그 데이터들간에는 공간 데이터의 중복이 발생한다. 이런 공간 객체의 중복은 다음과 같은 문제점이 발생한다.

첫 번째로는 저장공간의 낭비가 발생한다. 공간 데이터는 상대적으로 표현이 복잡하기 때문에 중복되어 데이터를 저장하는 것은 시스템의 많은 오버헤드를 발생시키고 많은 비용이 들 것이다.

또한 데이터의 중복은 의사결정을 지원하기 위한 정보를 생성하는 데에 있어서 효과적이지 못하다. 동일한 객체를 다른 객체로 인식을 하게 될 것이기 때문이다.

아래의 (그림 3)는 GML 공간 앨리먼트를 가지는 MapA 와 MapB 라는 GML 데이터의 예이다. 두 그림에서 University, RailRoad, Road2 가 소스 데이터에 동시에 표현이 되어 있다. 이 두 데이터들이 통합이 이루어지지 않는다면 공간 앨리먼트와 데이터와 더불어 공간에 대한 설명이 포함되는 비공간 앨리먼트의 중복이 발생하게 되므로, 공간에 대한 낭비가 발생한다. 또한 데이터 분석에서 다른 객체로 판단을 하게 되어 정확한 데이터 분석을 도출하지 못하게 된다.



(a) MapA의 지리정보 데이터 (b) MapB의 지리정보 데이터

(그림 3) GML 파일 MapA 와 MapB 의 공간 데이터

일반적으로 이런 문제를 해결하기 위해 Global Mediator 가 공간 데이터 웨어하우스에서 처리가능한 형태로 데이터를 변환하고, 중복된 데이터를 제거한다. 하지만 GML 데이터 통합은 Global Mediator 에게 많은 부하가 예상되므로 GML 데이터의 중복 문제를 해결하고 Global Mediator 의 부하를 줄이기 위하여 GML 소스는 GML Mediator 에서 통합을 수행한다.

아래의 [알고리즘 1]은 여러 GML 데이터 소스에서 공간 객체가 중복되었을 때, 중복된 GML 데이터를 하나의 객체로 통합을 하기 위한 알고리즘이다.

[알고리즘 1] GML Integration 알고리즘

Input
GML1, GML2 : 비교되어질 GML형식의 데이터
Variables
olRect : 두 MBR 사이의 Overlap이 되어지는 영역
sub1, sub2 : GML 형식의 데이터의 Path
PathExpr1, PathExpr2 : GML Element의 Path Expression
result : 앨리먼트의 통합 결과

```
Algorithm GMLIntegraion (GML1, GML2)
Begin Algorithm
01 : if( Disjoint(GML1, GML2) = TRUE )
02 :    return;
03 : else
04 :    olRect ← GetOverlapMBR(GML1, GML2);
05 :    sub1 ← SendGQuery(GML1, olRect);
06 :    sub2 ← SendGQuery(GML2, olRect);
07 :    SweepPos1 ← 0;
08 :    while(!(PathExpr1←ScanObject(sub1, olRect, &SweepPos1)))
09 :      SweepPos2 ← 0;
10 :      while(!(PathExpr2←ScanObject(sub2, olRect, &SweepPos2)))
11 :        if(IsEqual(PathExpr1, PathExpr2) = TRUE)
12 :          if(CompareAttr(PathExpr1, PathExpr2) = TRUE)
13 :            result ← IntegrateElement(PathExpr1, PathExpr2);
14 :            IntegrateSchema(PathExpr1, PathExpr2, result);
15 :          end if
16 :        end if
17 :        ScanObject(sub2, olRect, &SweepPos2);
18 :      end while
19 :      ScanObject(sub1, olRect, &SweepPos1);
20 :    end while
21 : end if
End Algorithm
```

위의 [알고리즘 1]에서는 추출과 정제 작업을 마친 데이터를 Storage Manager에서 임시 저장하므로 질의를 Storage Manager에게 API를 통하여 전송한다. 또한 일정한 영역r에 포함되는 공간 객체를 추출하는 기능이 필요하다. 다음과 같은 기능은 [알고리즘 2]이 수행한다.

[알고리즘 2] SendGQuery 알고리즘

Input
GML : GML 형식의 데이터의 경로
olRect : Rectangle 구조체
Variables
QueryStmt : 질의문을 담는 문자열
ResultPath : Storage에 저장된 결과 경로

```
Algorithm SendGQuery (GML, olRect)
Begin Algorithm
      ▶ [Overlap Query]
      ▶ LET $sub IN doc ( & GML)
      ▶ WHERE getOverlap( $sub/MBR, Rect)
      ▶ RETURN <SubGML> $sub </SubGML>
01 :  MakeOverlapQuery (QueryStmt, GML, olRect)
      ▶ Storage Manager에 질의 요청
02 :  ResultPath ← SM_SendQuery (QueryStmt)
03 :  return ResultPath

End Algorithm
```

[알고리즘 3]은 공간 Element 를 통합하고, 기존 앨리먼트는 삭제하는 기능을 수행한다.

[알고리즘 3] IntegrateElement 알고리즘

Input
GML1, GML2 : GML 형식의 데이터의 경로
Variables
Stmt : 질의문을 담는 문자열
ResultPath : Storage에 저장되어있는 경로 정보

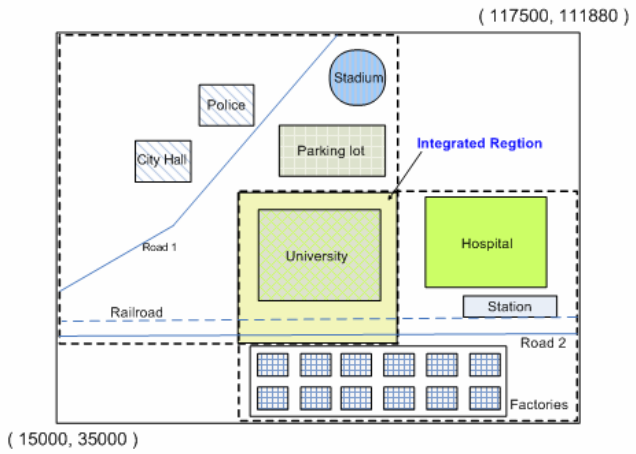
```
Algorithm IntegrateElement
Begin Algorithm
      ▶ [Integration Element]
      ▶ LET $gml1 IN doc ( "&PathExpr1" )
      ▶ RETURN
      ▶    $gml1,
      ▶    FOR $gml2 IN doc ( "&PathExpr2" )/getchild()
      ▶    RETURN $gml2
01 :  MakeIntegrationQuery (Stmt, PathExpr1, PathExpr1)
      ▶ Storage Manager에 질의 요청
02 :  ResultPath← SM_SendQuery (QueryStmt)
03 :  return ResultPath
End Algorithm
```

[알고리즘 4]는 Element 들이 통합과정을 수행 후의 변경된 구조의 유효성을 보장하기 위하여 스키마를 통합해주는 기능을 수행한다.

[알고리즘 4] IntegrateSchema 알고리즘

```
Input
GML1, GML2 : GML 형식의 데이터의 경로
Algorithm IntegrateSchema(PathExpr1, PathExpr2, result)
Begin Algorithm
    ▶ Storage Manager에 질의 요청
01 : SM_IntegrateSchema (PathExpr1, PathExpr2, result)
End Algorithm
```

위의 알고리즘들을 통해 여러 소스 데이터로부터 중복된 앨리먼트가 존재하게 되면, 하나의 앨리먼트로 통합이 된다. 아래의 (그림 4)는 중복된 영역에 공간 데이터가 통합된 모습을 나타낸다.



(그림 4) 통합된 지리정보 IntegrateMapAnB

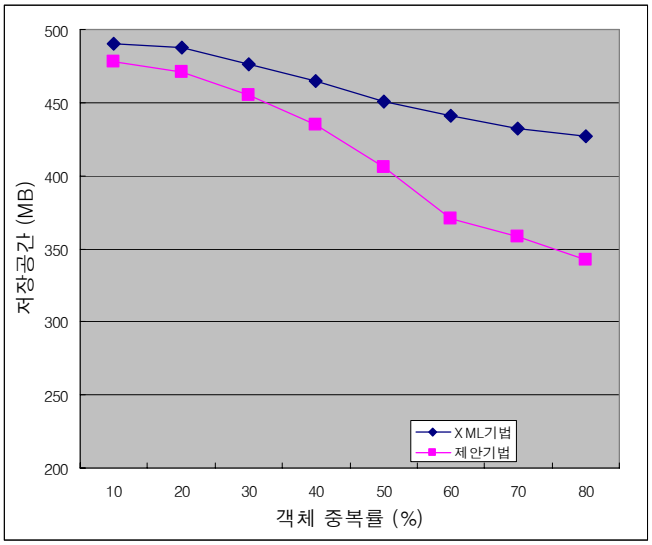
4. 성능평가

본 장에서는 제안 기법의 성능을 평가한다. 평가 방법은 기존의 기법인 XQuery 를 통한 GML 데이터를 추출하고 통합하는 기법과 GQuery 의 공간 연산자를 이용해 추출하고 통합하는 제안기법의 성능 비교를 수행한다. 방법은 데이터 통합을 수행한 후 저장공간의 크기를 비교를 통하여 이루어진다. 실험환경은 <표 2>와 같다.

<표 2> 실험환경

기종	IBM PC 호환
중앙처리장치	Pentium4 2.6 Ghz
주 기억장치	2GB
보조 기억장치	120GB, 7200RPM, ATA 방식
운영체제	Windows XP Professional
개발 언어	C/C++

이 실험의 평가환경은 공간 데이터를 이용한 실험에서 많이 사용되는 TIGER/Line File 을 2 개의 소스에 공간 객체의 중복률을 증가시키며 생성하였다[7]. 생성된 파일은 각각 100,000 개의 레코드에 250MB 의 데이터이다. (그림 5)는 두 통합기법을 공간 객체 중복률에 따라 결과물의 저장공간을 비교한다.



(그림 5) 통합기법에 의한 데이터의 저장 공간 비교

5. 결론 및 향후 연구

본 논문에서는 여러 GML 데이터들간의 공간 데이터 중복이 발생할 때, 통합을 하여 저장공간의 효율성을 높이는 기법을 제안하였다. 공간 데이터는 저장공간을 많이 차지하는 특성을 가지는데 GQuery 의 공간 연산자를 통하여 GML 데이터에서 공간 데이터를 통합하여 저장공간의 효율성이 평균적으로 약 9.95% 향상됨을 볼 수가 있다..

성능평가에서 수행시간은 XML 데이터 통합기법보다는 길지만, 저장공간의 효율성을 높일 수 있었다. 향후 연구는 중복되는 공간 객체 발생시 속성정보를 통한 효과적인 중복제거 기법에 대한 연구가 필요하다.

참고문헌

[1] ESRI, “Spatial Data Standards and GIS Interoperability”, http://www.esri.com/library/whitepapers/pdfs/spatial-data-standards.pdf.

[2] OGC, “Geography Markup Language (GML) Implementation Specification 3.0,” 2003.

[3] 안영수, 박순영, 정원일, 배해영, “GML 문서의 통합 지리 정보 검색을 위한 XQuery 의 확장”한국정보과학회 춘계학술대회, 2003.

[4] *Hélène Gagliardi, Ollivier Haemmerlé, Damiano Migiliori, Nathalie Pernelle, Marie-Christine Rousset, Fatiha Saïs*, “Enriching a Relational Data Warehouse by Integrating XML Data: report on the e.dot project applied to Microbiology”, ISIP 2005.

[5] ROBSON DO NASCMENTO FIDALGO, JOEL DA SILVA, VALERIA C. TIMES, FERNANDO DA F. DE SOUZA, ROBERTO SOUTO m. DE BARROS, “GMLA : A XML Schema for Integration and Exchange of Multidimensional-Geographical data”, GEOINFO, 2003.

[6] Omar Boucelma, François-Marie Colonna. “GQuery: a Query Language for GML. In Proc. 24th Urban Data Management Symposium, Chioggia-Venice, Italy, 2004.

[7] TIGER/Line Files, 2000 Technical Documentation, U.S. Bureau of Census,California,accessiblevia, http://arcdata.esri.com/data/tiger2000/tiger_statelayer.cfm?sfips=06.