구조적 유사성을 이용한 마이크로어레이 데이터의 효율적인 저장을 위한 기법의 설계 및 구현

윤종한○ 신동규 신동일
세종대학교 컴퓨터 공학과
jhyoon@gce.sejong.ac.kr, shindk@sejong.ac.kr, dshin@sejong.ac.kr

Design And Realization For Efficiently Storing Microarray Data Using Structural Similarity

Jonghan Yun○ Dongkyu Shin Dongil Shin
Dept. of Computer Engineering Sejong University

요 약

생명정보 대량 획득기술의 하나인 마이크로어레이(microarray)는 DNA와 각종 유적자 연구에 사용되는 도구로서 확립되면서, 생명정보학(bioinformatics)분야의 발전에 크게 기여하였다. 그러나 마이크로어레이는 생명정보학분야의 핵심기술 중 하나로 발전하였음에도 불구하고 마이크로어레이 실험으로 생성되는 데이터는 형태가 다양하고 매우 복잡한 형태를 갖기 때문에 데이터의 공유나 저장에서 많은 어려움을 겪는 것이 사실이다. 따라서 마이크로어레이 실험결과 분석을 위한 최소한의 컨텐츠가 정의되고 표준화 되었다. MIAME 데이터, MAGE-OM/ML과 같은 표준화된 공개 저장소는 전문 생물학 연구단체에게 과거부터 지금까지 주요 관심사가 되어왔다. 하지만 많은 공개저장소의 설립되었지만 마이크로어레이 데이터의 구조적 특징을 고려하여 효과적인 설계를 하지 않은 것이 사실이다. 본 논문은 표준을 따르는 동시에 마이크로어레이 데이터의 구조적 빈발 패턴이 반복되는 계층적 특징을 반영하는 전략을 제안한다. 이를 통하여 복잡한 데이터의 구조를 객체들의 빈발 패턴을 파악하여 그 계층을 줄임으로서 복잡도를 줄일 수 있었다. 이 과정에서 관계형 데이터베이스 기반의 공개저장소의 성능에 영향을 주는 관계 테이블(join-table)의 숫자는 줄어든다. 이에 따라, 성능은 개선된다. 이 전략을 통하여 생성된 테이블의 숫자는 원본 데이터를 단순 매핑시켜 저장하는 방법에 비하여 약 31%줄어든다. 결국 MAGE-ML 데이터의 저장과 로딩 시간은 이 논문에서 제시하는 전략을 적용하지 않은 방법에 비해 60%에서 65%를 줄일 수 있었다.

1. 서 론

생명정보 대량 획득기술들 중 하나인 마이크로어레이(microarray)실험으로 생성이 되는, 다양하고 복잡한 구조의 마이크로레이 테이터에 효과적으로 접근하고 처리하는 방법은 필수적이다. 마이크로어레이 데이터는 분석하고 해석하는 방식이 다양하기 때문에 데이터를 저장하는 형식이 다르다. 따라서 상호간에 원활한 데이터 교환을 위한 기반으로 XML형식의 마이크로어레이 유전자 표현 데이터(MGED: Microarray Gene Expression Data) 표준을 따르는 추세이다. 따라서 어레이데이터를 효과적으로 접근하고 처리할 수 XML 데이터 저장에 관련된 많은 연구들이 진행되어 왔다[1], [2], [3]. 이러한 연구들은 테이블간의 결합을 줄임으로써 데이터베이스의 성능을 개선시킬 수 있는 방법들을 제안하였다. 이 방법들은 엘리먼트(element)와 속성(attribute)이 정의 되어있는 DTD 또는 XML 스키마(XML schema)로부터의 테이블을 추출하고 최적화하는데 매우 효과적이다. 그러나 이러한 방법들은 마이크로어레이 같은 생물학적 정보를 위한 XML스키마에는 효과적이지 않다. 그 이유는, 마이크로어레이 데이터는 복잡한 계층을 가지고 있지만, 그 형태를 들여다보면 단지 몇 개의 마이크로어레이 데이터의 핵심 값들만이 반복적으로 나타나기 때문이다. 바이오인포매틱스에 효과적인 접근을 위하여 새로운 접근방식이 필요하다. 본 논문에서는 XML문서에서 구조적 유사성 기반의 빈발패턴(Frequent Pattern)을 마이닝할 것이다. 그리고 XML 문서의 객체들을 관계형 데이터베이스로 직접 매핑 시켰던 기존의 연구와는 다르게 한정된 범위에서 마이닝된 빈발패턴들을 구조적 유사성에 기반하여

저장하는 전략을 보인다.

2. 관련연구

반-구조적 데이터(semi-structured data)의 관리를 위하여 상용 RDBMS(관계형 데이터베이스 관리 시스템)들은 객체 관계 매핑 방법 [1], [2], [3]들을 제공한다. 일반적으로 이 방법들은 XML 문서를 객체 트리의 형태로 다루고, 객체를 여러 관계형 테이블들로 변환한다. 이것은 많은 결합을 요구하는 XML 데이터 집합의 쿼리를 만든다. 결국 이러한 방법을 채택한 데이터베이스의 성능은 비효율적이게 된다. 이러한 문제를 극복하기 위한 대안으로 여러 연구들이 XML 저장 매핑 기술 제안하였다: DTD 기반 [4], Edge 기반 [5], 그리고 데이터 마이닝 기반 [6]. Edge 기반[5]의 기술은 그래프와 edge를 단일 edge테이블에 저장한다. 그리고 그 단일 edge 테이블은 XML 데이터의 모든 그래프와 edge 정보를 처리한다. 데이터 마이닝 기반[6]의 방식은 알고리즘[7]을 사용하는 저장 시스템을 제안하였다. 이 마이닝 알고리즘은 XML 데이터로부터 관계들을 추출해내고 그것들을 관계형 데이터베이스 스키마(schema)로 변환한다. 접근 방식 [5], [6]은 데이터베이스의 성능을 확연하게 향상 시킨다. 그러나 이러한 접근방식은 하나의 구조만을 처리할 수 있다. 왜냐하면 이러한 접근들은 변환 과정에서 오직 XML 데이터 중 하나의 인스턴스(instance)만을 요구하기 때문이다. XML 인스턴스의 구조적 변화를 관리하는데 DTD를 간단히 하는 것이 기본적인 방법이었다. 연구[4]는 객체 관계 매핑에 앞서 DTD의 단순화에 초점을 맞춘 인라인 테크닉(in-line technique)을 제안하였다. 이 기술은 DTD로 부터 생략할 수 있는 엘리먼트를 배제하였다. 여기서 생략할 수 있는 엘리먼트란 조상 엘리먼트와 후손 엘리먼트 사이에 연결고리의 역할만을 하는 엘리먼트를 말한다. 이러한 엘리먼트를 제거함으로써 조상 엘리먼트는 직접적으로 후손 엘리먼트에 연결될 수 있다. 많은 MGED(Microarray Gene Expression Data) 표준을 따르는 저장소는 안정성, 편리한 관리, 그리고 뛰어난 성능 [8], [9]과 같은 장점을 가진 RDBMS를 채택하였다. 이러한 연구들의 최대 관심사는 MGED 표준을 따르는 저장 도구였다. 비록 연구[8]에서 객체 모델의 로컬 변화를 통하여 일반 쿼리의 성능을 개선시켰지만, 그러한 변화를 통해 효과적인 데이터베이스 설계 스키마를 기대하긴 어려웠다. 그것은 그 데이터베이스 설계 스키마가 바로 직접적 매핑 기술을 따르기 때문었다. 이와 비슷하게 마이크로 어레이 데이터베이스에 대한 연구 The Stanford Microarray Database[9]는 트리 기반의 XML 문서가 표준화된 관계형 데이터베이스 테이블에 저장되어질 때 발생하는, 객체-관계 불일치의 문제점[10]에 대한 명확한 해결책을 보이지 않았다. 연구[11]이 MAGE(Microarray Gene Expression)객체를 XML 문서로 변환하는 구체적인 매핑 규칙을 발표 하였지만, 그것은 마이크로어레이 데이터를 위한 관계형 데이터베이스의 효과적인 설계를 통하여 이점들을 취하려는 의도가 아니라 단지 관계형 테이블로부터 XML문서를 가져오는데 초점을 맞추었을 뿐이었다.

본 논문에서는 MAGE 표준을 따르는 마이크로어레이 데이터의 효과적인 저장 구조에 초점을 맞출 것이다. 그리고 그 과정에서 MAGE 데이터에서 유사한 구조를 찾는 의사결정트리 알고리즘을 사용할 것이다. 그리고 이에 앞서 MAGE와 의사결정트리 알고리즘을 간략히 설명할 것이다.

3. 마이크로어레이 데이터와 MAGE

마이크로어레이는 방대한 양의 유전자 정보 추출을 용이하게 하는 주된 방법이다. 이 정보는 유전자 표현 데이터뿐만 아니라 환경, 기술, 원료와 같은 실험과 관련된 전반적인 내용을 포함 하는 마이크로어레이 실험으로부터 생성 되었다. 연구 집단 사이에 이 정보를 공유함으로써, 각 연구 집단들은 다른 연구 집단에서 사용하는 다양한 기술들을 습득할 수 있고, 자신들의 생물학적 의문을 해결하기 위해 이것들을 사용할 수 있다. 연구 집단들 사이에 마이크로어레이 데이터를 공유하기 위해서는, 유전자 표현 데이터를 분석하는데 필수적인 정보가 분류 되어야 한다. 게다가 분류되는 정보는 다른 연구 집단들과 의 공유에서 충분한 이식성을 가져야 한다. 이러한 이식성의 필요에 의해 마이크로어레이 유전자 표현 데이터(MGED: Microarray Gene Expression Data) 집단은 Microarray Experiment(MIAME)를 따르는 데이터의 설명을 위한 최소한의 정보를 담기 위한 상호교환 모델을 설계하였다. 그 모델의 형태는 마이크로어레이 데이터의 상호교환을 위한 표준으로 발표 되었다:MAGE-OM/ML(Object Model/Markup Language)[12]. MAGE-OM은 데이터의 상호교환을 위해 XML로 표현되는 MAGE-ML을 사용한 데이터 교환을 위한 형태로 변환되어야 한다. MAGE-ML 기반의 바이오인포매틱스(Bioinformatics) 데이터는 매우 크다. 이것은 교환과 저장에서 많은 어려움을 동반한다. 이러한 단점을 가진 MAGE-ML 데이터는 모든 연구자들에게 공개적인 DTD 파일로써 열려있다.

4. 의사결정트리 알고리즘을 사용한 핵심

이번 장에서는 반-구조적 데이터로부터 구조적 유사성을 가지고 있는 XML 엘리먼트(element)를 분류할 것이다. 그림 1은 알고리즘을 의사결정트리를 사용하여 도식화한 것이다.

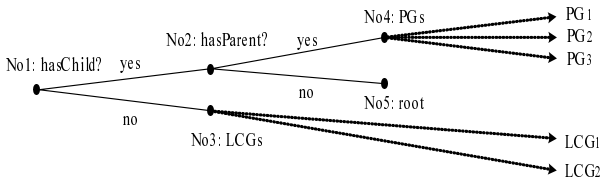

그림 1 엘리먼트의 분류를 위한 의사결정트리

특정한 형태의 엘리먼트 ele_x 가 *complexType*을 가지고 있고, ele_x의 각 구성 엘리먼트 집합은 다음과 같이 구성되어 있다고 한다면,

$$complexType(elex) = \{SE_{elex}, SA_{elex}\}$$
$$SE_{elex} = \{e_1, e_2, \cdots, e_n\}$$
$$SA_{elex} = \{Ae_1, Ae_2, \cdots, Ae_n\}$$

일정한 엘리먼트 ele_x의 *complexType*에 있는 SE_{elex} 또는 SA_{elex}가 다른 엘리먼트 ele_y의 *complexType*에 존재하는 SE_{eley} 또는 SA_{eley}의 구성형태와 일치할 때, ele_x와 ele_y는 같은 *complexType*을 가지고 있다고 말한다.

분류 알고리즘의 수도코드에서 사용되는 용어에 대한 정의는 다음과 같다.

(1) XML 스키마에서 정의된 하나의 엘리먼트(element)
(2) *E*: e의 엘리먼트 집합
(3) *E′*: e의 하위 엘리먼트(sub-elements) 집합
(4) *a*: e의 속성(attribute)
(5) *A*: e의 속성 집합(attribute set)
(6) *A′*: e의 모든 하위 엘리먼트를 위한 속성 집합
(7) *complexType*: E′와 A′로 구성된 구조적 정보
(8) *Lowest child*: 하위 엘리먼트가 없는 엘리먼트
(9) *Lowest parent*: 하나의 최하위 자식 엘리먼트인 하위 엘리먼트가 있는 엘리먼트
(10) *e*:*PG*(parent Group): 최하위 자식의 부모가 될 수 있는 후보 엘리먼트들의 집합
(11) *LPCG*(The Lowest Parent Candidate Group): 최하위 부모가 될 수 있는 후보 집합
(12) *LCG*(The Lowest Child Group): 최하위 자식 엘리먼트들의 집합
(13) *LPG*(The Lowest Parent Group): 최하위 부모 엘리먼트들의 집합(하나이상의 LCG를 포함)
(14) *ULPG*(Upper Level Parent Group): 최하위 부모나 최하위 자식이 아닌 엘리먼트를 포함하는 상위 레벨 부모들의 집합

다음은 의사결정트리에서 엘리먼트들을 형태에 따라 분류하는 분기조건으로 사용되는 네 가지 규칙이다.

규칙 1 - 만일하나의 엘리먼트가 어떤 하위 엘리먼트도 가지고 있지 않다면 그 엘리먼트(예, Figure 1에서의 노드 G, H, I, J)는 *LCG*의 엘리먼트로 분류 된다. 그렇지 않고 하위 엘리먼드를 가지고 있다면 엘리먼트(예, Figure 1에서의 노드 A, B, C, D, E, F)는 *PG*의 엘리먼트로 분류 된다. 규칙 1은 한 엘리먼트가 *LCG* 그룹이나 *LPG* 그룹 어디에 속할 수 있을지를 결정한다. 이 규칙은 그림 2와 같이 표현된다

```
For each ei  E do
   if number of elements in SEi is 0 do
      ei is classified into LCG
   else
   ei is classified into PG
   end if
end for
```

그림 2 규칙 1의 pseudo code

규칙 2 - 규칙1의 *LCG*를 LCG_0라고 하자. LCG_0의 몇 엘리먼트들은 하나 또는 그 이상의 속성들이 정의 되었을 때 같은 *complexType*을 가진 엘리먼트들과 새로운 그룹 LCG_p로 분류 되어 진다($p > 0$). 만일 이미 같은 *complexType*를 가진 LCG_p 그룹이 있다면 그 엘리먼트는 결국 LCG_p 그룹이 된다. 결론적으로 규칙 2는 *LCG*의 여러 집합을 분류한다. 이 규칙은 그림 3과 같이 표현된다.

```
p = 0;
For each ei  LCG0 do
 Flag=0;
If p>0 do
 For q=1 to p
  If complexType(ei) LCGp exsits
        ei is classified into  LCGp
           Flag=1;
  end if
 end for
end if

If Flag is 0 do
   For each ej  LCG0 do
    if complexType(ei) is complexType(ej) do
       p=p+1
       ei and ej are classified into a new group of LCGp
    end if
   end for
end for
```

그림 3 규칙 2의 pseudo code

규칙 3 – 만일*PG*에 특정한 엘리먼트가 오직 *LCG*에 속한 하위 엘리먼트만을 가지고 있다면 그 엘리먼트는 *LPG*의 엘리먼트로 분류 된다. 그렇지 않다면 그 엘리먼트는 *ULPG*의 엘리먼트로 분류 된다. 이것이 *PG* 엘리먼트를 *LPG*와 *ULPG* 이 두 가지 그룹으로 분리하는 규칙 3이다. 이 규칙은 그림 4와 같이 표현된다.

```
For each e_i PG do
    if SE_ei includes LCG do
        e_i is classified into LPG
    else
        e_i is classified into ULPG
end for
```

그림 4 규칙 3의 pseudo code

규칙 4 – 규칙 3의 *LPG*를 LPG_0 라고 하자. LPG_0의 엘리먼트들이 하나 또는 그 이상의 속성들이 정의되어 있는 *complexType*을 가지고 있다면, 같은 *complexType*을 가지고 있는 그 엘리먼트들은 새로운 LPG_p로 분류 된다.(p>0) 만약 같은 *complexType*을 가지고 있는 LPG_p 그룹이 이미 있다면 그 엘리먼트는 LPG_p의 그룹이 된다. 즉, 규칙 4는 LPG의 여러 집합을 분류한다. 이 규칙은 그림 5와 같이 표현된다.

```
p = 0;
For each ei LPG0 do
  Flag=0;
  If p>0 do
    For q=1 to p do
      If (complexType(ei) = complexType(element in LPGq) do
        ei is classified into LPGq
        Flag=1;
      end if
    end for
  If (Flag==0) do
    For each ej LPG0 do
      if complexType(ei) is complexType(ej) do
        p=p+1
        ei and ej are classified into a new group of LPGp
      end if
    end for
  end if
end for
```

그림 5 규칙 4의 pseudo code

5. 의사결정트리 알고리즘을 이용한 데이터베이스 설계

본 논문에서 제시된 의사결정트리를 사용하여 분류된, MAGE-ML 데이터를 위한 XML 스키마로부터 데이터베이스 테이블을 설계하였다.

분류 알고리즘을 통하여 같은 형태를 갖는 엘리먼트들을 하나의 테이블에 저장할 수 있게 되었다. 이를 통하여 데이터베이스의 테이블의 숫자가 줄어들고, 이에 따라 복잡도가 개선된다. *LCG*와 *LPG*는 N대N의 관계를 갖는다. 본문에서는 *LCG*와 *LPG*가 특수한 데이터이고 반복해서 나타나기 때문에 분류 규칙에서 최하위 자식 집단(*LCG*)과 최하위 부모 집단(*LPG*)에 초점을 맞추었다. 그림 6은 분류 알고리즘 과정을 거쳐 그룹화된 객체들을 데이터베이스에 저장했을 때 데이터베이스의 모습이다. 데이터베이스는 같은 테이블에 저장된 서로 다른 객체들을 원상태로 복원하기 위하여 객체의 형태와 ID를 유지한다.

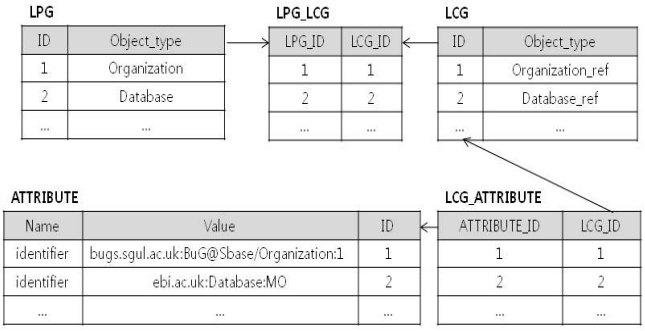

그림 6 그룹화된 객체들의 데이터베이스 저장

5.1 시간 복잡도

데이터베이스 공간의 복잡도 감소를 통하여 저장(storing)과 로딩(loading)이 개선되었다. 본 논문에서 사용된 XML 데이터는 1Mbyte의 크기이고 모든 엘리먼트들은 *LCG*와 *LPG*로 분류된다.

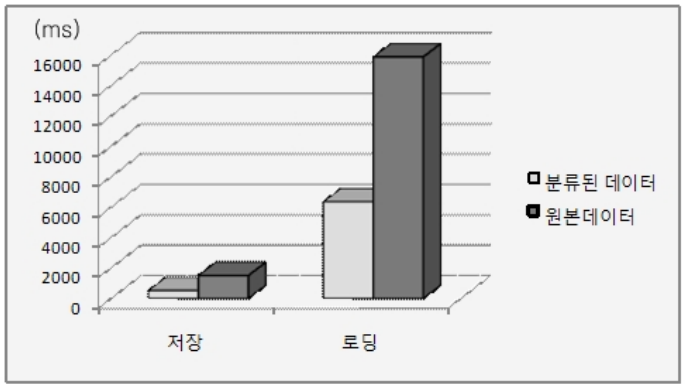

그림 7 저장과 로딩시간의 차이

그림 7은 분류된 스키마와 원본 스키마의 XML 데이터의 저장과 로딩을 위한 처리시간을 보여준다. 이것은 테

이블 결합의 감소가 저장과 로딩의 시간에 중요한 역할을 한다는 것을 의미한다.

5.2 데이터베이스 공간 복잡도

표 1에서 보여지는 MAGE-ML의 개선된 스키마(schema)를 도입함으로써 데이터베이스의 XML 데이터 저장의 복잡도를 개선하였다. 분류 규칙을 따르는 테이블을 만든 이후, 클래스는 테이블에 사상되고 클래스와 테이블의 전체 숫자는 같아졌다.

표 1 데이터베이스 공간 복잡도

	원본 스키마	분류된 스키마
전체 클래스의 숫자	455	314
전체 테이블의 숫자	455	314
전체 레코드의 숫자	2012	160
전체 DB 크기	710 (Kb)	27 (Kb)
전체 테이블 결합 횟수	101	2

5.3 XML 문서의 재구조화

기존의 연구[2]에서 튜플(tuple)들을 복구하기위해 테이블 집합에 접근하고, 그 결과들을 결합한다. 그리고 그 결과들을 XML 문서의 형태로 바꾸기 위해 정렬된 외부결합(outer-join) 메서드를 사용하는 XML 문서의 형태로 정리한다.

이 작업을 간단히 하기 위해 본문의 실험에서는 관계형 데이터베이스로부터 자동으로 XML 문서를 생성하는 도구(tool)를 사용하였다.

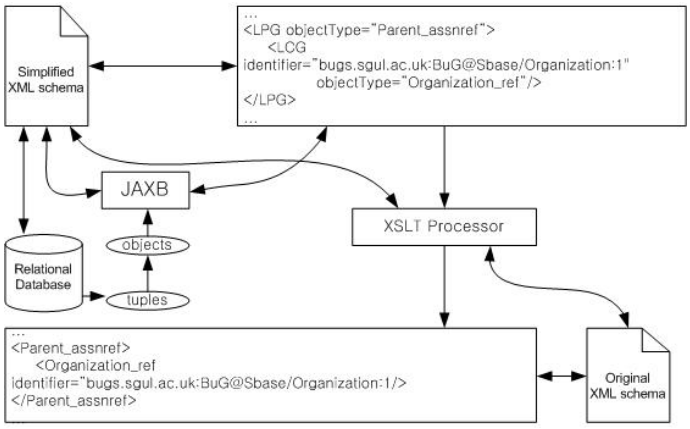

그림 8 XML 문서 재구조화의 흐름

JAXB (Java Architecture for XML Binding)[13]는 객체를 XML 문서로 변환하는 기능과 그에 대한 역 변환 또한 제공한다. 우선, 관계형 데이터베이스로부터 튜플들을 검색하고 결과들을 객체에 사상한다. 그리고 그것들을 JAXB를 사용하여 XML 문서로 변환한다. 이때 데이터베이스가 XML 스키마 중심의 구조이기 때문에 JAXB를 통하여 관계형 데이터베이스로부터XML 문서를 생성하는 것은 매우 효율적이다. 그러나 이 XML 문서는 완전하지 않다. 왜냐하면 XML 스키마 중심의 데이터베이스 설계 전에 XML 스키마는 이미 단순화되었기 때문이다. 그래서 스키마로부터 완전한 XML 문서를 재구조화하기 위해 XSLT(XML Stylesheet Language Transformations)[14]를 사용 하였다. 그림 6에서 볼 수 있는 것처럼 각각의 최하위 자식 엘리먼트들은 객체형태(object_type)뿐만 아니라 부모의 ID(parent_id)도 가지고 있다. 따라서 XSLT 처리기는 쉽게 원본 XML 스키마에 명시되어 있는 순서를 알 수 있다. 그림 8은 이러한 문서 재구조화의 흐름을 나타낸다.

결론

본 논문에서는 생물학적 정보(biological information: 많은 엘리먼트들과 속성들이 정의된)를 위한 XML 스키마로부터 구조적 유사성을 가지는 엘리먼트들을 추출하기 위한 방법을 제안한다. 실험의 결과들은 이 개선된 스키마가 테이블 사이의 결합을 줄여서 XML 데이터의 저장과 로딩(loading)의 성능을 현격히 개선하는 것을 보여준다.

이후 연구에서, 유사 엘리먼트들의 분류를 위한 규칙을 확장할 계획이다. 본 논문에서는 XML스키마의 계층의 전환을 최소화 하기위해 분류의 한계를 *LCG*에서 *LPG*까지로 한정하였다. 만약 *LCG*에서 *ULPG*로 범위를 확장시킨다면, 데이터베이스의 테이블들 사이의 결합은 줄어들 것이다. 그리고 그것은 XML 데이터의 저장과 로딩의 성능을 현재 현재보다 향상시킬 수 있을 것이다.

참고문헌

[1] H. Schoning, Tamino - A DBMS designed for XML, In Proceedings of the 17th International Conference on Data E ngineering 2-6 April 2001, Heidelberg Germany, pp.149-154, 2001.

[2] I. Tatarinov and S. Viglas, Storing and Querying Ordered XML Using a Relational Database System, In Proceedings of the 2002 ACM SIGMOD international conference on Management of data, 03-06 june 2002, Madison, Wisconsin, U.S.A., pp.204-215, 2001.

[3] K. Runapongsa and J. M. Patel, Storing and Querying XML Data in Object-Relational DBMS, In EDBT 2002 Workshop on XML-Based Data Management and Multimedia Engineering, LNCS 2490,

pp.266-285, 2002.

[4] J. Shanmugasundaram, K. Tufte, G. He, C. Zhang, D. Detwitz and J. Naughton, Relational databases for querying xml documents: Limitations and opportunities, In Proc. Intl. Conf. on 25th VLDB, 1999.

[5] A. Schmidt, M. L. Kersten, M. Windhouwer, and F. Waas, Efficient relationalstorage and retrieval of XML documents, In proceedings of WebDB, 2000.

[6] A. Schdmidt, M. F. Fernandez, and D. Suciu, Storing Semistructured Data with STORED, Inproceedings of the ACM SIGMOD International Conference on Management of Data, pp.431-442, 1999.

[7] I. H. Witten, E. Frank, Data Mining Practical Machine Learning Tools and Techniques with Java Implementations, Morgan Kaufmann Publishers, 2005.

[8] U. Sarkans, H. Parkinson, G. G. Lara, A. Oezcimen, A. Sharma, N. Abeygunawardena, S. Contrino,E. Holloway, P. Rocca-Serra, G. Mukherjee, M. Shojatalab, M. Kapushesky, S. A. Sansone, A. Farne, T. Rayner, A. Brazma, The ArrayExpress gene expression database: a software engineering and implementation perspective, Bioinformatics. 21(8), pp.495-501, 2005.

[9] A. Catherine Ball, A. B. Ihab. Awad, Janos Demeter, Jeremy Gollub, Joan M. Hebert, Tina Hernandez-Boussard, Heng Jin, C. Matese John, Michael Nitzberg, Farrell Wymore, K. Zachariah, O. Patrick Brown and Gavin Sherlock, The Stanford Microarray Database accommodates additional microarray platforms and data formats, Nucleic Acids Research, pp.33, 2005.

[10] S. Ambler, D. A. Chapam, Agile Database Techniques: Effective Strategies for the Agile Software Developer, WILEY, 2003.

[11] W. Martin, R.M. Horton, Magebuilder, A schema translation tool for generating MAGE-ML from tabular microarray data, Bioinformatics Conference, CSB 2003, pp.431 - 432, 2003

[12] P. T. Spellman, Design and implementation of microarray gene expression markup language (MAGE-ML), Genome Biol-23-3 (9)RESEARCH0046

[13] JAXB (Java Architecture for XML Binding), http://java.sun.com/xml/downloads/jaxb.html

[14] XSLT (XML Stylesheet Language Transformations), http://www.w3.org/Style/XSL/