XML 문서 관리를 위한 능동 규칙 언어

황 정 희[†] · 류 근 호[††]

요 약

XML은 웹 데이터의 표현과 정보교환을 위한 표준이다. XML의 급격한 사용증가로 인하여 XML 저장 관리 시스템 및 XML 문서의 변화에 자동으로 대응할 수 있는 규칙기반의 기술개발에 대한 연구가 활발히 진행되고 있다. 능동 규칙은 사건, 조건, 조치로 구성되며 데이터베이스의 상태 변화에 자동으로 대응할 수 있는 특성이 있으므로 이러한 요구를 충족시킨다. 따라서 이 논문에서는 XML 문서를 자동으로 관리하기 위한 XML 기반의 능동 규칙 언어를 제안하고 이 규칙언어로 정의되는 능동 규칙에 대한 종료 분석 방법을 제시한다. 아울러 XML 문서의 능동적 관리를 위한 규칙의 적용 사례를 제시하고 분석 방법의 효율성에 대해 검증한다.

Active Rule Language for XML Document Management

Jeong Hee Hwang[†] · Keun Ho Ryu[††]

ABSTRACT

XML is the standard for storing and exchanging information on the Web. As the applications of XML become more widespread, the works on rule-based technology are rapidly going on to support reactive functionality on the XML documents and the XML repositories. Active rules consist of event-condition-action, which automatically perform actions in response to status change of database. Therefore the feature of active rule satisfies the new needs in XML setting. In this paper, we propose not only a XML based active rule language to manage XML document automatically, but also an active rule analysis method to guarantee rule termination. Finally, we demonstrate some examples of active rule defined by the proposed rule language, and also verify the efficiency of our analysis method by comparing with another method.

키워드 : 능동 데이터베이스(Active Database), 능동 규칙언어(Active Rule Language), XML 문서관리(XML Document Management), 규칙 종료분석(Rule Termination Analysis)

1. 서 론

XML(External Markup Language)[1]은 웹 상에서의 데이터 교환과 저장을 위해 제안된 표준 언어이다. 웹에서의 자유로운 데이터 교환을 위해, XML은 DTD(Document Type Definition)를 통하여 문서 자체에 문서의 구조를 기술한다. 따라서 문서의 구조를 사용자가 원하는 대로 정의할 수 있으며 이러한 구조적 유동성은 모든 형태의 데이터가 XML로 기술될 수 있고, 웹에서 운용되는 모든 데이터가 동일한 형태로 통합, 저장, 처리될 수 있는 기반을 제공한다[2]. 또한 하나의 소스로부터 다양한 디스플레이 메커니즘을 만들 수 있는 정보의 재사용(repurpose), 재정의, 디스플레이가 가능하다. 즉, 서버의 데이터베이스에 데이터를 저장하고 데이터를 추출하여 데이터의 구조를 표현하는 XML 마크업 문서로 만든 후 이를 다시 웹 브라우저, PDA(Personal Digital Assistant), 무선폰 등 다양한 종류의 디바이스에서 보여 줄 수 있다[3]. 이러한 XML의 장점으로 인하여 웹에서의 다양한 전자상거래 및 여러 응용분야에서 XML를 데이터 모델로 하여 활용, 개발되고 있다.

이러한 XML은 데이터 웨어하우스(Data Warehouse), 전자상거래(e-commerce)등과 같은 분야에서의 더욱 활발한 사용 증가로 인하여 XML 저장소(repository)의 데이터 변화에 자동으로 대응할 수 있는 기능을 지원하기 위한 규칙 기반의 기술개발 필요성이 빠르게 대두되고 있고, 능동 규칙(Active Rule)은 이러한 요구사항을 충족시킬 수 있는 가장 주요한 요소로서 주목받고 있으며 이를 이용하여 XML 데이터를 다루고자 하는 많은 연구[4-8]가 최근에 진행되고 있다.

능동 규칙은 사건(event), 조건(condition), 그리고 조치(action)로 정의되며 능동 데이터베이스를 포함하는 여러 환경에서 상태 변화에 자동으로 대응할 수 있는 능동적 특성을 지니고 있기 때문에 네트워크 관리, 데이터 무결성 관리, 워크플로우 관리 등의 다양한 분야에 적용되고 있다. 또한 최근에는 데이터 웨어하우스에서 형성 뷰의 점진적 관리,

※ 이 논문은 2002년도 한국학술진흥재단의 지원에 의하여 연구되었음(KRF 2002-002-D00152).
† 준 회 원 : 충북대학교 대학원 전자계산학과
†† 종신회원 : 충북대학교 전기전자 컴퓨터공학부 교수
논문접수 : 2002년 7월 4일, 심사완료 : 2002년 9월 17일

XML 문서에 대한 유효성 검사, 실시간으로 변화되는 정보를 먼 거리에 있는 사용자에게 자동으로 제공하는 푸시(push) 기법 등에 적용하기 위한 다양한 응용[6, 8, 9] 방안이 제시되고 있다.

그러나 기존의 대부분의 연구에서는 XML 저장소의 변화에 대해 반응할 수 있는 능동 규칙의 적용 방법에만 초점을 두고 있을 뿐 다양한 XML 데이터의 형태에 적용될 수 있는 XML 기반의 능동 규칙언어의 구문을 정형화하여 명시하지 않았다. 반면에 [6]에서는 XML 데이터를 관리하기 위해 기존의 능동 규칙구문을 변형한 구문형태를 명시하고, 규칙의 종료여부를 분석하기 위하여 추상적 해석기법을 적용할 수 있음을 제시하였으나 규칙 집합에 포함되는 모든 규칙에 대하여 변화되는 데이터베이스의 상태와 함께 규칙의 실행여부를 예측하는 것은 잠재적 트리거 관계를 정확하게 분석하기 어렵다.

따라서 이 논문에서는 능동 규칙의 다양한 응용 기반이 될 수 있는 XML 문서의 능동적 관리를 위한 새로운 능동 규칙 언어를 제안한다. 그리고 제안하는 능동 규칙언어를 이용하여 XML 문서를 관리하는 적용 예를 보이며 능동 규칙 설계의 중요한 관건인 종료분석방법에 대한 알고리즘을 기술한다. 아울러 기존 연구의 종료 분석방법과 비교하여 우리의 분석방법의 효율성을 입증한다. 이 논문에서 제안하는 XML 기반의 능동 규칙언어를 이용하여 능동 규칙을 정의하면 XML 문서의 생성과 삭제 수정시에 자동으로 대응할 수 있으며 사용자나 응용프로그램에서 요구하는 형태로 문서를 관리할 수 있는 문서의 자동 생성과 통합이 가능하다는 특성이 있다.

이 논문의 구성은 다음과 같다. 2장에서는 기존에 제시된 XML를 위한 ECA(Event Condition Action) 규칙 언어에 대해 알아보고 이 논문과의 차이점을 기술하며 3장에서는 이 논문에서 제안하는 XML를 위한 능동 규칙언어를 정의하고 4장에서는 XML 기반의 능동 규칙에 대한 종료 분석방법을 설명한다. 그리고 5장에서는 3장에서 정의된 능동 규칙언어를 이용하여 XML 문서를 관리하는 방법의 적용 사례를 기술하고 6장에서는 기존에 제안된 XML 기반의 능동 규칙 분석방법과의 비교를 통해 우리가 제안하는 분석방법의 효율성을 검증하고 7장에서는 결론을 맺는다.

2. 관련 연구

최근 인터넷의 발전이 진전되면서 이 기종간의 시스템에서 상호적인 데이터 사용과 교환을 목적으로 하는 XML의 이용이 확산되고 있다. 즉, 사용자나 웹의 응용분야에서 XML 문서를 만들고 조작하기 위한 것이 일반화되어 가고 있다. 이것은 XML 문서를 저장하는 XML 저장소(repository)의 관리와 더불어 XML 문서의 자동적 관리에 대한 필요성을 암시하고 있다. 이러한 응용의 관심과 발전은 앞으로 더 많은 응용 분야에 대한 잠재적 가능성을 가져올 수 있으며 현재 분리되어 제공되는 정보의 제공을 통합하여 제공할 수 있는 무한한 응용의 발전 가능성에 대한 중요성을 나타내는 것이다[4, 7].

[5]에서는 XML 기반의 저장소에서 새로운 사건에 반응하여 사용자에게 관련된 정보를 보내줄 수 있는 푸시기법(Push Technology))에 적용 가능한 Active XML Rule 개념을 소개하였고 이것은 XML을 위한 SOAP프로토콜[10]을 이용하여 먼 거리에 있는 사용자에게 XML 저장소에서 발생하는 사건에 대해 능동 규칙을 이용하여 새롭게 변경되는 사용자의 관심사항을 제공한다. 아래의 규칙 정의의 예는 <cd> 엘리먼트가 삽입되면 규칙이 트리거되고 <price>가 20보다 크고 <author>가 "Milli Vanilli"를 포함하면 SOAP 메소드 "Notify"를 통해 서버 131.175.16.105에 통지한다.

```
<event> insert (//cd) </event>
<condition> for $a in // cd
            where $a= $new and
                  $a/price < 20 and
                  contains($a/author, "Milli Vanilli")
</condition>
<action>
 <SOAP-ENV : Envelope>
  ...
 <SOAP-ENV : Body>
  <m : Notify xmlns : m = "http:// 131.175.16.105/methods">
</action>
```

[6]에서는 XML 저장소의 변화에 대해 반응할 수 있는 XML 데이터의 관리를 위한 ECA 규칙을 소개하고 규칙의 구문과 실행의미에 대하여 기술하였다. 그리고 XML 기반의 능동 규칙에 대한 종료 분석방법과 함께 기존의 능동 데이터베이스에서의 능동 규칙과는 달리 XML의 특성을 포함하고 있는 능동 규칙의 분석방법이 간단하지 않음을 언급하였다. 이 논문에서 제안하고 있는 XML를 위한 능동 규칙의 예는 다음과 같다.

두 개의 XML 문서 s.xml, p.xml에서 s.xml는 매일 각 store에서 각 상품에 대한 판매량을 나타내는 문서이고 p.xml는 각 상품의 판매량을 포함하는 상품에 대한 정보를 나타내는 문서이다. 다음의 ECA rule은 하나이상의 새로운 상품이 s.xml 문서의 store 아래에 추가될 때 p.xml 문서를 수정하는 규칙이다.

```
on INSERT documents('s.xml')/sales/day/store/product
if TRUE
do DELETE store[@id = @delta /../@id])
       BELOW document('p.xml')/products/
             product[@id = $delta/@id] ;
   INSERT <store id = '{$delta/../@id}'>
       BELOW document('p.xml')/products/
             product[@id = $delta/@id] AFTER TRUE
```

이 규칙에서의 첫 번째 조치는 s.xml에 삽입된 상품의 부

모요소인 store 엘리먼트가 이미 p.xml의 products의 자식 노드로 존재한다면 p.xml 로부터 이를 제거하는 것이고 두 번째 조치는 p.xml에 있는 새로운 상품의 자식노드로 store를 추가한다.

또한 이 논문에서는 XML 데이터 관리를 위해 정의된 규칙의 종료 분석에 대해 그들이 기존의 능동 규칙에 대해 제안하였던 방법인 규칙 동작에 대한 추상적 해석(Abstract Interpretation)[11]을 기반으로 하는 분석 방법을 소개하였다. 이 분석 방법은 이 논문에서 제안하고자 하는 분석방법과 많은 차이가 있으며 이 논문의 6장에서 그 차이점을 비교 분석할 것이다.

[7]에서는 e-service에서 최신의 정보를 빠르게 제공하기 위해 XML 정보를 관리하는 새로운 개발 흐름의 능동 XML 규칙을 언급하고 XSLT(Extensible Stylesheet Language Transformations)기반의 접근방법으로써 템플릿(Template) 규칙의 집합으로 구성되는 XSLT을 제시하였다. XSLT의 템플릿 규칙은 XML 문서의 노드를 처리하는 규칙처리기(Rule Processor)로 간주하며 명세된 규칙 명세에 의해 적용 변환된다. 그리고 하나의 노드가 여러 템플릿과 일치할 때 즉, 규칙 적용에 따른 충돌 발생의 경우에 대비하여 템플릿 규칙에 대한 우선순위를 명시하고 우선순위가 같은 경우에는 가장 나중에 정의된 템플릿을 적용하도록 하였다. 다음의 예는 웹 사용자를 연령별로 구분하여 각각 다른 형태로 정보를 제공하고자 하는 웹 페이지의 개별화(Personalization) 수행을 위한 규칙 정의이다.

```
CREATE RULE Personalize
PRIORITY : 0
EVENT : <xsl : variable name = "U" select = "User">
        <rule : login select = "$U"/>
COND_ACT : <xsl : template match = "$U"/>
            <xsl : variable name = "age_user" select = "$U/age"/>
            <xsl : choose>
                <xsl : when test = "$age_user & lt : = 27">
                    <xsl : include href = "junior.xsl"/>
                </xsl : when>
                <xsl : when test = "$age_user
                    27 and age_user & lt ; 55">
                <xsl : include href = "standard.xsl"/>
                </xsl : when>
                <xsl : otherwise>
                    <xsl : include href = "senior.xsl"/>
                </xsl : otherwise>
            </xsl : choose>
    </xsl : template>
```

이 규칙은 로깅된 웹 사용자의 연령이 27살 미만이면 junior.xsl, 55살 미만이면 standard.xsl, 그리고 그외의 연령에게는 senior.xsl을 적용하여 서로 다른 형태의 웹 내용을 제공한다.

이와 같이 기존에 제시되었던 연구들에서, [5]는 능동 규칙을 이용하여 XML 저장소의 변화에 반응하는 규칙의 예가 제시되었으나 능동 규칙언어의 정형화된 구문의 구조를 명시하지 않았다. 그리고 [6]은 XML 기반의 능동 규칙을 종료 분석하기 위하여 추상적 해석기법을 이용하는 분석방법을 제안하였으나 규칙 집합에 포함되는 모든 규칙에 대하여 실행 동작을 예측한다는 것은 다양한 상황에서 발생할 수 있는 규칙의 잠재적 트리거 관계를 정확하게 반영하기 어렵고 XML 기반의 능동 규칙은 기존의 능동 규칙보다 더 복잡한 구조로 형성되기 때문에 정확한 분석 결과를 기대할 수 없다. 또한 [7]은 XML 데이터를 관리하기 위한 기존의 연구와는 다르게, XML 데이터를 보여주기 위한 XSLT에서의 능동 규칙을 이용하여 사용자에 따라 다른 형태로 정보를 제공하는 것에 중점을 두고 있다.

따라서 이 논문에서는 기존 연구의 문제점을 개선하고 XML 문서의 능동적 관리를 목적으로 하는 능동 규칙언어의 구문을 정의한다. 그리고 규칙 동작의 종료여부를 결정하기 위하여 XML의 특성인 엘리먼트 경로를 포함하는 트리거 관계를 정의하고 비활성화 관계[15]에 의해 연속적 실행 관계여부를 결정하는 종료 분석 알고리즘을 제안한다. 아울러 실제적으로 능동 규칙이 어떻게 적용되어 XML 문서가 관리되는지의 규칙 적용 예와 [6]에서 제시된 XML 기반의 능동 규칙의 종료 분석 방법과 비교하여 우리가 제안하는 분석 방법의 효율성을 검증한다.

3. XML를 위한 능동 규칙

이 논문에서는 XML 문서의 능동적 관리를 위한 새로운 능동 규칙 언어를 정의한다. 이것은 기존 데이터베이스에서의 능동 규칙의 사용과 마찬가지로 XML 문서 변화에 대해 자동 대응할 수 있는 규칙 제공의 기반이 된다.

3.1 XML 기반의 능동 규칙 언어

규칙은 규칙 기술 형식을 위한 문법적 구조와 그 구조에 대한 의미를 명확하게 하는 정의된 규칙 언어를 통해 명시적으로 표현할 수 있고 이를 바탕으로 능동 규칙 시스템은 사용자의 요구를 반영하는 규칙에 의해 정확하게 처리할 수 있다[8, 9]. 능동 규칙 표현을 위한 언어들은 기존의 여러 능동 데이터베이스 시스템 상에서 다양한 형태로 제시되어 왔고 최근에는 XML 데이터를 다루기 위한 능동 규칙에 대한 정의가 제시되고 있다[6, 7].

XML 문서들로 이루어져 있는 XML 데이터베이스를 위한 능동 규칙에서도 기존의 형식과 같은 ECA 형태로 구성되며 (그림 1)은 이 논문에서 제안하는 XML의 특성을 반영하는 규칙 언어의 구문 정의이다.

(그림 1)의 규칙 정의에서는 기존의 규칙정의에 시간 사건을 추가하였다. 이것은 XML 문서들에 대해 일정시간 혹은 주기적으로 문서를 통합하거나 로그 파일과 같이 일정

한 간격으로 자료를 기록해야 할 필요가 있는 문서에 대한 규칙 실행을 적용하기 위한 것이다.

```
CREATE TRIGGER <rule_name>
{<temporal event>}
<event action time> <event_action> <XML_path> ON
<xml_name>
[REFERENCING {OLD, NEW} AS <identifier>]
[FOR {EACH-VALUE | SET-VALUE ]}
WHEN <condition containing predicate>
DO <action>{<action_position>}<element name> UNDER
    <XML_path> ON <xml_name>

<temporal event> ::= <absolute time event> |
                     <periodic time event>
<absolute time event> ::= AT <time point>
<periodic time event> ::= EVERY {<time amount> |
                                 <time point>}
                          [DURING <time interval>]
<event action time> ::= BEFORE | AFTER
<event_action> ::= INSERT | DELETE | UPDATE
<condition> ::= XQuery | BOOLEAN
<action> ::= INSERT | DELETE | UPDATE
<action_position> ::= FRONT | END
```

(그림 1) XML 기반의 능동 규칙 언어

데이터 접근 사건은 예약어 ON 다음에 XML 문서 이름을 명시하여 규칙의 트리거 적용 대상 문서를 지정할 수 있으며 데이터 접근사건 절의 가장 앞에는 BEFORE나 AFTER를 사용하여 사건영역에 포함되는 문서의 실행을 사건 발생 시점 전으로 할 것인지, 사건 발생 시점 후로 할 것인지를 명시할 수 있다.

<trigger event>는 사건의 중심이 되는 데이터베이스 연산 즉, 삽입(INSERT), 삭제(DELETE), 갱신(UPDATE) 연산 중의 하나에 의해 데이터베이스에 접근할 수 있는 사건의 범주를 한정한다.

일반적으로 규칙은 과거 트랜잭션에 의해 변경된 데이터들을 참조하기 위해 REFERENCING 절을 갖는다[8, 16]. REFERENCING 절은 데이터 접근사건과 함께 기술되어 OLD AS에 의한 데이터의 과거 값이나, NEW AS에 의한 새로운 값의 참조를 가능하게 한다. 참조되는 데이터의 단위는 정의되는 규칙의 규칙실행 단위에 따라 문서 혹은 엘리먼트가 된다. 그리고 규칙의 실행 단위를 나타내는 EACH-VALUE와 SET-VALUE는 정의된 규칙의 사건에 해당하는 문서의 노드 각각에 대해 실행할 것인지, 동시에 실행할 것인지를 명시하며 기본적으로는 개별적 수행을 적용한다.

WHEN 절은 규칙의 조건에 대한 표현으로 규칙의 조치를 수행할 것인가를 결정하는 부분이고, 조건 평가를 위한 질의(query)에서는 XML 문서를 위한 질의어, 즉 XQuery를 사용하고 특정 엘리먼트의 내용 값을 비교하는 술어 연산을 포함하여 조건 평가 결과를 얻을 수 있다.

DO 절은 조치를 기술해 주는 부분으로 XML 문서에 대한 삽입, 삭제, 수정이 가능하다. <action_position>에서는 ON 다음의 XML 문서에 대한 규칙의 조치 실행 적용위치를 지정한다. 즉 문서에 대한 자식 노드를 추가할 경우 해당되는 형제노드의 앞에 추가할 것인지(FRONT), 뒤에 추가할 것인지(END)를 명시하고 기본적으로는 END가 적용된다. 그리고 조치에서도 ON 다음에 XML 문서의 이름을 명시할 수 있는데 만약 명세가 생략되면 사건에 명세된 문서와 같은 문서에 대한 조치의 적용을 의미한다.

3.2 규칙언어의 확장

3.2.1 사 건

사건은 어떤 규칙이 트리거되어야 하는지를 명세 하는데 이용된다[9, 18]. 사건을 발생시키는 데이터베이스 연산은 삽입, 삭제, 수정 등과 같은 데이터베이스 연산과 시간 사건에 의해 발생할 수 있으며 이렇게 물리적으로 발생하는 사건들을 기본사건이라 하며 이 기본사건들의 논리적인 결합을 통해 사건을 명세할 수 있는데 이러한 사건들은 복합사건이라 한다[8, 19]. 따라서 XML를 위한 능동 규칙의 사건에는 단순 사건과 더불어 다음과 같은 복합사건을 명세할 수 있다.

① 같은 문서에서의 사건 발생 연산자에 대한 OR, AND 결합의 복합사건
(예) BEFORE INSERT or UPDATE bookstore/book ON book.xml

② 같은 문서에서 경로가 다른 사건에 대한 복합사건
(예) BEFORE INSERT bookstore/book OR bookstore/booktitle/title ON book.xml

서로 연관된 데이터를 갖는 다른 문서간의 무결성 검사 즉, 개인 주소 변경시 전체 직원 주소록에서의 변경이나, 학생의 개인 성적 변경시 전체 평균 변경 등과 같은 참조 데이터에 대한 일관성 유지를 위해 각 문서의 변경을 감시할 수 있는 규칙들을 생성하여 변경된 문서를 연관된 문서에도 반영할 수 있도록 하는 규칙의 정의에 필요하다.

3.2.2 조 건

조건은 사건의 발생에 의해 기동된 규칙의 조치 여부를 결정하는 부분으로써 XPath(XML Path Language)[3, 20]를 이용하여 검색 연산을 구성하는 데이터베이스 술어와 XML의 질의어인 XQuery(XML Query Language)[21]가 명세될 수 있으며, 조건 평가결과가 참이 될 때에 조치가 실행된다.

XPath는 XML 문서의 노드를 트리 형태로 관리하여 문서의 부분 요소들에 대한 식별기능을 갖는 특징이 있다. 그러므로 규칙 명세에서는 간결한 경로 표현을 위해 XPath의 특성을 이용한다. 즉, 사건에서 명시된 엘리먼트 경로를 현재노드로 간주하고 조건에서는 상대위치를 표현한다. 이것은 사건에서 명시된 엘리먼트 경로를 재작성하는 번거로움

을 방지하기 위한 것이다.

또한 XPath는 XML 문서의 특정 부분의 위치를 찾을 수 있는 표현식 이외에도 값을 계산하거나 패턴매칭시 사용될 수 있다는 장점이 있다. 그리고 커다란 문서의 특정 부분을 식별할 때에는 XPath의 완전한 표현의 길이가 길어진다는 단점을 간단한 문서 경로로 표현할 수 있도록 축약형 표현식을 허용한다. 그러므로 조건에서는 이러한 XPath에서의 간단한 경로 접근 문법의 표현을 사용하여 문서의 위치 경로에 접근한다. XPath의 축약형 문법의 예는 다음과 같다.

각 위치 경로 단계의 첫 번째 부분을 나타내는 액시스(axis)를 나타낼 때 일반적으로 child는 생략하며, attribute::은 @으로, self::node()는 .으로, 문서의 세 번째 요소를 선택하기 위한 [position() = 3]은 [3]으로 대체할 수 있다.

```
<bookinventory>
  <nation>
   <America>
     <booktitle> Oh, Pioneers! </booktitle>
   </America>
   <English>
     <booktitle> Great Expectations </booktitle>
   </English>
  </nation>
  <book>
    <title> Great Expectations </title>
    <author>
      <name> Charles Dickens </name>
      <born> 1812 </born>
      <died> 1879 </died>
      <nationality> English </nationality>
    </author>
    <count> 10 </count>
  </book>
  <book>
    <title> Oh, Pioneers! </title>
     <author>
      <name> Willa Cather </name>
      <born> 1873 </born>
      <died> 1947 </died>
      <nationality> American </nationality>
     </author>
   <count> 5 </count>
  </book>
</bookinventory>
```

위 예는 책 재고를 관리하기 위한 문서(bookinventory.xml)로서 각각의 책에 대한 제목, 저자, 저자와 관련된 사항을 기록하고 있으며 저자의 국적 별로 책 관리를 하는 <nation> 엘리먼트를 두고 있다. 다음 규칙은 위 문서에 대한 책 재고수량이 수정될 때 트리거 되는 규칙 정의의 예로써 count 값이 0이면 책 정보를 담고 있는 하위노드를 포함하는 부모 노드 <book>를 삭제하라는 규칙이다.

```
CREATE TRIGGER bookinventory_update
AFTER UPDATE bookinventory/book/count ON bookinventory.xml
WHEN $<count> = 0 and
DO DELETE bookinventory/book
```

이 규칙의 실행은 다음의 규칙을 트리거하여 국적별로 관리하는 책제목 리스트에서도 책의 제목을 제거한다. 그런데 이 규칙 정의에서는 책제목이 일치하는 엘리먼트를 찾는 조건질의가 필요하다. 이러한 경우에 XSL(Extensible Stylesheet Language)에서 사용되는 XML 문서에 대한 간단한 질의 패턴을 이용하여 아래의 보기와 같이 정의된 규칙을 생성할 수 있다.

```
CREATE TRIGGER book_delete
BEFORE DELETE bookinventory/book ON bookinventory.xml
WHEN [$any$ booktitle = $<title>]
DO DELETE <booktitle>
```

위 규칙의 조건에서 any 키워드는 동일한 이름의 자식 엘리먼트가 하나이상인 경우에 적용되어 엘리먼트의 값을 비교할 때 사용한다. 즉, [any booktitle = $<title>]는 각 <booktitle> 엘리먼트의 값을 조사하여 <title>의 값과 같은 <booktitle> 엘리먼트가 존재하면 조치에서 그 <booktitle> 엘리먼트를 삭제하라는 규칙 명세이다. 이와 같이 조건에서 패턴 질의를 이용하여 얻어진 결과 엘리먼트가 존재하면 조치에서 UNDER 키워드 이하에 문서 경로를 명시할 필요가 없다.

3.2.3 조 치

조치는 궁극적으로 규칙이 처리하고자 하는 동작들을 명세한다. 즉, 규칙에 명세된 조치의 평가가 참일 때 조치를 수행한다.

XML 문서에서 조건을 만족하는 엘리먼트의 값에 대해 그에 해당하는 엘리먼트의 하위요소에 새로운 엘리먼트를 생성하고자 할 때 문서의 삽입 경로가 조건에서 명시된 값에 따라 달라질 수 있다. 이러한 경우에 대한 규칙 명세 방법을 설명하기 위한 규칙의 예는 다음과 같다.

(규칙 1)

```
CREATE TRIGGER bookinventory_insert
BEFORE INSERT bookinventory/book ON bookinventory.xml
WHEN $<nationality> = "American"
DO INSERT <booktitle> UNDER bookinventory/nation/American
```

(규칙 2)

```
CREATE TRIGGER bookinventory_insert
BEFORE INSERT bookinventory/book ON bookinventory.xml
WHEN $<nationality> = "English"
DO INSERT <booktitle> UNDER bookinventory/nation/English
```

위 (규칙 1)과 (규칙 2)는 조건의 값에 따라 생성되어야 하는 하위 엘리먼트의 위치가 달라지는 경우의 규칙일 때에는 각 규칙을 각각 작성하지 않고 일반적인 프로그램의 case 문처럼 각 엘리먼트에 따라 적용되는 대상 엘리먼트를 다르게 할 수 있도록 유연한 규칙 명세가 필요하다. 따라서

<nationality> 엘리먼트의 값이 "American" 이면 <American> 요소의 하위요소에 엘리먼트를 생성하고 "English" 이면 <English> 엘리먼트의 하위요소에 엘리먼트를 생성하는 규칙은 다음과 같이 작성할 수 있다.

```
CREATE TRIGGER bookinventory_insert
BEFORE INSERT bookinventory/book ON bookinventory.xml
WHEN $<nationality> = "English | America"
DO INSERT <booktitle> UNDER
            bookinventory/nation/$<nationality>
```

이처럼 문서 경로가 조건에 따라 변경되므로 정확한 문서경로를 명시할 수 없는 경우에는 엘리먼트의 내용 값을 직접 명시하여 문서 경로를 결정한다.

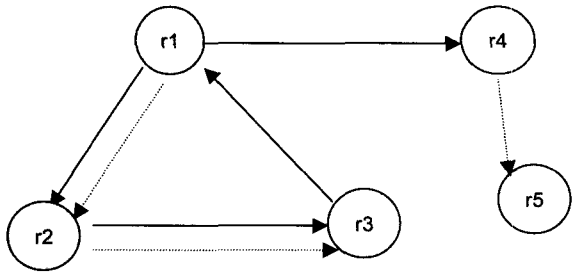
XML 데이터에 대하여 규칙을 정의하는 것은 관계형 데이터베이스에 대한 규칙의 명세보다 더 복잡하다. 다양한 문서의 크기와 엘리먼트의 순서 그리고 계층적인 구조로 인하여 규칙의 명세가 복잡해지고 같은 문서 내에서도 같은 엘리먼트가 반복적으로 나타날 수 있으므로 XML 문서가 갖는 독특한 특성을 모두 고려하여 가능한 명료하고 유연하게 규칙을 명세할 수 있는 규칙 언어의 정의와 그에 따른 구현 방법이 요구된다.

4. XML 기반의 규칙 종료 분석

능동 데이터베이스에서 능동 규칙의 종료 분석은 중요한 연구 주제이며 이에 대한 많은 연구가 진행되어 왔다. 특히 지금까지는 관계형 데이터에 대한 연구가 주된 대상이었다. 규칙 분석의 주요 관심은 초기 데이터베이스 상태와 규칙을 트리거하는 초기 사건에 대하여 대응하는 규칙의 실행 종료를 보장할 수 있느냐 하는 것이다[14, 22]. 따라서 XML 문서 관리를 위한 능동 규칙에서도 규칙간에 서로를 무한하게 트리거하는 관계를 미리 예측하여 조정할 수 있는 종료 분석 연구가 반드시 필요하다.

규칙 종료 분석방법으로는 정적분석(static analysis) 방법과 동적 분석(dynamic analysis) 방법이 있다. 동적 분석방법[12, 13]은 규칙이 실행되는 실행시간(run-time)에 규칙의 동작을 조사하는 방법으로, 단점은 능동 데이터베이스를 실제적으로 사용하기 전에는 종료를 보장할 수 없으며 실행시간 분석에 따른 성능 저하의 문제를 수반한다. 반면에 정적 분석방법[22]은 컴파일 시간(compile time)에 정의된 규칙을 조사하여 규칙간의 트리거 관계를 예측하고 규칙집합의 종료 보장을 결정하는 방법이다. 이 방법의 단점은 컴파일시간에 분석이 이루어지기 때문에 정의된 모든 규칙의 분석결과를 컴파일과정을 통해 완료해야 하므로 다양한 이벤트 형식과 규칙의 실행의미(rule execution semantics)를 반영하는 것이 어렵다.

능동 규칙의 종료 분석은 대부분 트리거 그래프(Trigger Graph : TG)를 기반으로 한다[13]. 트리거 그래프[12, 13]는 규칙을 노드로 표현하고 규칙의 실행 결과가 임의의 규칙을 트리거할 수 있는 사건의 범주에 포함되면 실행된 규칙과 트리거가 가능한 규칙을 연결하는 방향성 간선(edge)을 사용하여 표현한다. 이와 더불어 트리거 그래프에 대해 상호 보완적인 활성화 그래프(AG : activation graph)[13, 14]는 임의의 두 규칙 ri, rj (i ≠ j)에 대해 ri의 조치가 rj의 조건을 참으로 변경시킬 때 형성되므로 규칙종료 분석에서 더 정확한 결과를 얻을 수 있다. (그림 2)는 [8, 13]에서 제시된 TG와 AG를 결합하여 규칙의 관계를 표현하였다.



(그림 2) 트리거그래프(TG : 실선)와 활성화그래프(AG : 점선)

이 그래프에서는 적어도 하나의 도착간선(incoming edge)을 소유하는 규칙은 실행될 가능성이 있다는 것을 의미한다.

규칙종료문제는 규칙간의 관계를 나타내는 그래프에서 사이클을 분석하는 것이다. 즉, 규칙관계를 표현한 트리거 그래프에서 사이클이 형성되지 않으면 규칙집합이 종료한다는 결정을 할 수 있다[22].

[13]에서는 규칙 집합으로부터 TG 또는 AG의 도착간선을 갖지 않는 규칙을 제거하는 축소(reduction) 알고리즘을 제시하였고, 이에 의해 그래프에 있는 모든 사이클이 제거되면 규칙집합의 종료를 보장할 수 있다는 분석 방법을 소개하였다.

이러한 기존의 연구에서는 규칙 실행의미의 미반영과 그래프 표현상의 어려움 등이 있었다. 그래서 우리는 [23]에서 규칙의 실행 의미를 반영하는 종료 분석 방법을 제안하였다. 이것은 사건에 명세되는 복합사건의 형태와 규칙의 실행 시점을 나타내는 "BEFORE", "AFTER" 규칙의 실행의미를 반영하고, 규칙간의 관계를 그래프로 표현할 때 발생되는 그래프의 복잡성에 대한 해결방법으로 비활성화 그래프(Deactivation Graph : DG)와 트리거 그래프를 결합시키는 능동 규칙의 종료 분석 방법이다.

XML 문서 관리를 위한 능동 규칙의 종료 분석에서도 [23]에서 제시되었던 기본적인 종료 분석 방법에 근거하여 XML 문서의 특성을 고려하는 분석 방법을 제시한다. 또한 우리는 능동 규칙의 트리거를 정의하기 위하여 아래식. [23]의 임의 규칙 ri, rj에 대한 트리거 관계를 사용한다.

$TR(ri, rj) = \{ri, rj \in R \mid Out(ri) \cap TR\text{-}by(rj) \neq \phi\}$

즉, 규칙 ri의 조치실행으로 발생하는 사건은 규칙 rj의 사건에 명세된 사건과 공통된 것이 적어도 하나 존재한다는 것을 의미한다.

이것을 XML를 위한 능동 규칙의 트리거 관계로 확장하여 정의하면 다음과 같다.

규칙 종료 분석을 위해 위에서 정의된 예제 규칙의 명세를 사건, 조건, 조치로 다음과 같이 분리하면,

E : [BEFORE|AFTER] INSERT **XML_path(E)**
ON XML_name
C : WHEN trigger_condition(**XML_path(C)**)
A : DO INSERT <element> UNDER **XML_path(A)**

임의의 규칙에 대해

P_E = 사건 명세에 포함된 문서경로 XML_path(E)의 하위 엘리먼트를 포함하는 집합
P_C = 조건 명세에 포함된 문서경로 XML_path(C)의 하위 엘리먼트를 포함하는 집합
$Re(P_A)$ = 조치 명세에 포함된 문서경로 XML_path(A)에 대하여 조치 실행으로 생성, 삭제, 변경된 결과 노드의 하위 엘리먼트를 포함하는 집합이라고 할 때

조치의 구문 실행 결과로 생성, 삭제, 변경된 하위 엘리먼트를 포함하는 $Re(P_A)$가 **XML_path(E)**에 포함되지 않는다면 트리거 관계가 형성되지 않는다. 이것을 식으로 표현하면 규칙 집합(R)에 포함되는 규칙 ri, rj의 트리거 관계 정의는 다음과 같다.

[정의 1] 트리거 관계 TR은 규칙 ri의 조치실행으로 인해 발생하는 사건이 rj를 트리거한다.

$$TR(ri, rj) = \{ri, rj \in R \mid Re(P_A(ri)) \cap P_E(rj) \neq \phi\}$$

이와 같은 트리거 관계를 기반으로 규칙간의 활성화 관계와 비활성화 관계도 정의 될 수 있다. 활성화 관계는 규칙의 조치 실행으로 임의 규칙의 조건 값을 거짓에서 참으로 변화시키는 것을 말하며 비활성화 관계는 조건 값을 참에서 거짓으로 변화시키는 것을 의미한다. 그러므로 규칙의 조치 실행으로 새롭게 생성된 문서경로와 임의 규칙의 조건에 명세된 경로와는 공통된 경로를 포함하고 있고 그것의 조건값을 거짓으로 변화시키면 비활성화 관계가 성립하고 참으로 변화시키면 활성화 관계가 성립한다.

[정의 2] ri의 조치 실행으로 rj의 조건 값을 참으로 변화시키면 규칙 ri, rj는 활성화 관계(AR)이다. 이것을 식으로 표현하면 다음과 같다.

$$AR(ri, rj) = \{ri, rj \in R \mid (Re(P_A(ri)) \cap P_C(rj) \neq \phi) \cap Trans\text{-}into(r_j.Ce = True) \neq \phi\}$$

규칙 r의 조건 평가의 값(Condition Evaluation)은 r.Ce으로 표기하며 값의 범위는{True, False}이다. 그리고 Trans-into는 규칙의 조건 값을 변화시키는 규칙을 말한다.

[정의 3] ri의 조치 실행으로 rj의 조건 값을 거짓으로 변화시키면 규칙 ri, rj 는 비활성화 관계(DR)이다. 이것을 식으로 표현하면 다음과 같다.

$$DR(ri, rj) = \{ri, rj \in R \mid (Re(P_A(ri)) \cap P_C(rj) \neq \phi) \cap Trans\text{-}into(r_j.Ce = False) \neq \phi\}$$

이러한 규칙간의 비활성화 관계는 임의의 규칙 조치에서 자신을 포함하는 임의의 다른 규칙의 P_C에 포함되는 경로를 delete하는 경우를 포함하여 형성한다. 다음은 비활성화 관계를 나타내는 규칙의 예이다.

(예)
규칙 r1 : 책 재고 증가를 관리하는 규칙

```
CREATE TRIGGER book_insert
BEFORE INSERT bookstore/book ON bookinventory.xml
WHEN [$any$bookinventory/book/title = $<title>]
DO UPDATE NEW.count = OLD.count + NEW.count
```

규칙 r2 : 책 재고 감소를 관리하는 규칙

```
CREATE TRIGGER count_update
AFTER UPDATE bookstore/book/count ON bookinventory.xml
WHEN count = 0
DO DELETE bookinventory/book
```

규칙 r1과 규칙 r2는 [정의 3]에 의해 규칙 r1의 조치와 규칙 r2의 조건에 공통된 경로집합 bookstore/book/count이 존재하며 규칙 r1의 조치에서는 count의 값을 증가시키고 규칙 r2의 조건에서는 count = 0을 검사하므로 규칙 r1는 규칙 r2조건 값을 거짓으로 변화시키는 비활성화 관계를 이룬다.

규칙 관계의 트리거 그래프에서 사이클을 형성하는 규칙이 순차적으로 실행되지 않으면 연속적인 반복 실행이 불가능하다. 따라서 규칙의 연속적 반복 실행의 고리를 제거할 수 있는 비활성화 관계를 이용하면 규칙의 종료여부를 좀 더 간결하고 정확하게 결정할 수 있다. 그러므로 XML 기반의 능동 규칙에 대한 종료 분석에도 이 방법을 적용한다. 즉, 트리거 관계를 기초로 1차 분석을 수행하여 한정된 횟수의 실행 사이클을 제거하고 2차 분석을 통해 비활성화 관계의 거짓 사이클(False Cycle)을 제거하여 종료 분석한다. (그림 3)은 이러한 종료 분석 수행 과정을 나타내는 알고리즘이며, 이 알고리즘에 의해 사이클이 형성되는 규칙관

계가 존재하지 않는 규칙 집합은 종료한다고 결정한다.

```
• Triggering Step
  fetch rules related in rulebase
  search TR(ri[ai] = rj[ej])
  generation trigger graph
  if non-execution rule ∈ cycle CR
    delete CR
  if not exist cycle then
    stop
  else execute deactivation step.

• Deactivation Step
  search DR(ri[ai] = rj[cj])
  generation deactivation graph
  if deactivation relation ⊂ cycle CR
    delete false cycle CR
  if not exist cycle then
    stop
  else request rule redesign to user
```

(그림 3) 규칙 종료 분석 수행 알고리즘

5. 적용 사례

이 장에서는 앞에서 제시된 XML 문서 관리를 위한 능동 규칙 언어를 이용하여 XML 문서를 관리하는 규칙의 적용 사례를 설명한다.

5.1 문서의 자동관리

XML 기반으로 정의된 능동 규칙은 XML 문서의 변화에 대해 자동으로 실행된다. 이것은 정의된 규칙의 사건범주에 해당하는 XML 정보의 삽입, 삭제, 수정이 발생하면 규칙이 자동으로 트리거되는 것을 보장하므로 XML 문서를 자동으로 관리할 수 있다.

<bookstore.xml> : 분야별 책제목 리스트의 내용을 포함하는 문서이다.

```
<bookstore>
  <fiction_book>
    <title> Seven </title>
      <num> 20 </num>
      <input_date> 2001-10-1 </input_date>
    <title> Rainbow </title>
      <num> 15 </num>
      <input_date> 2001-10-7 </input_date>
    <title> Rose </title>
    ...
  </fiction_book>
  <nonfiction_book>
    ...
  </nonfiction_book>
</bookstore>
```

<fiction_book.xml> : 소설책에 관한 정보를 포함하고 있는 문서이다.

```
<fiction_book>
   <title> Seven </title>
     <author>
       <first_name> Ju </first_name>
       <last_name> Suk </last_name>
       <birthdate> 1950-4-3 </birthdate>
       <nationality> America </nationality>
     </author>
       <issue_year> 2001 </issue_year>
       <serial> 1-111-1 </serial>
       <count> 20 </count>
   </fiction_book>
```

위 예에서 서점에서 새로운 책을 구매하여 책 재고가 증가했을때 책 재고 리스트(bookstore.xml)에 따른 책의 내용(fiction_book.xml)을 추가하는 능동 규칙은 다음과 같다.

```
CREATE TRIGGER book_insert
AFTER INSERT bookstore/fiction_book/title ON bookstore.xml
WHEN $<num> > 0
DO INSERT fiction_book/title ON fiction_book.xml
```

사건에서 bookstore/fiction_book/title의 삽입은 하위 엘리먼트를 포함하는 title 엘리먼트에 대한 삽입이 이루어진 것을 의미하며, 조치에서는 추가된 책에 대한 내용을 fiction_book.xml 문서에 추가한다.

다음 예 <book_order.xml>는 책 주문 리스트와 주문 내용을 담고 있는 문서이다.

```
<book_order>
  <book_order_list>
    <book_order_id> A10001 </book_order_id>
      ...
    <book_order_id> B22110 </book_order_id>
  </book_order_list>
  <book_order_contents>
    <book_order_id> A10001 </book_order_id>
      <book_title> Seven </book_title>
      <num> 2 </num>
      <date> 2002-5-5 </date>
    <name> Hwang Jeong Hee </name>
    <address> Cheong Ju Chungbuk University </address>
    <phone>
      <area_code> 043 </area_code>
      <phone_number> 267-2252 </phone_number>
    </phone>
  </book_order_contents>
  ...
<book_order_contents>
  <book_order_id> B22110 </book_order_id>
    <book_title> Science </book_title>
    <num> 1 </num>
      <date> 2002-5-10 </date>
    <name> Hwang Jeong Hee </name>
    <address> Cheong Ju Chungbuk University </address>
    <phone>
       <area_code> 043 </area_code>
       <phone_number> 267-2252 </phone_number>
    </phone>
  </book_order_contents>
</book_order>
```

다음 규칙은 고객이 책을 주문했을 때 책 주문 번호에 따

른 주문 내용을 문서에 추가하는 규칙이다.

```
CREATE TRIGGER book_order_insert
AFTER INSERT book_order/book_order_list/book_order_id ON
                book_order.xml
WHEN TRUE
DO INSERT book_order_contents UNDER book_order
```

다음 규칙의 예는 책 주문번호는 다르지만 주문자의 성명이 같은 경우에 주문 내용을 하나로 통합하는 규칙이다. 규칙의 조건에서는 주문자의 이름과 전화번호가 같은 경우에만 내용을 통합한다. 이것은 주문자의 이름이 같아도 동명이인이 있을 경우에 대비하여 전화번호까지 같을 때 같은 고객으로 간주하고 주문 내용을 통합한다.

```
CREATE TRIGGER book_order
BEFORE INSERT book_order/book_order_contents ON
                book_order.xml
WHEN [$any$ name = $<name>] and
     [$any$ address = $<address>]
DO INSERT FRONT book_order_id, book_title, num, date
```

조건에서 $<name>, $<address>는 새로 삽입되는 <book_order_contents> 엘리먼트의 하위 엘리먼트 값을 의미하며 이미 존재하는 <name>, <address> 엘리먼트와의 비교를 통해 동일 고객에 대한 주문 정보가 이미 존재하면 주문 정보를 통합한다. 따라서 조치에서는 조건의 패턴질의에서 발견된 동일 고객의 <book_order_contents> 경로 정보에 최근의 주문 내용, 즉 주문번호, 책이름, 수량, 날짜를 주문 정보의 가장 앞에 추가하기 위하여 <FRONT>를 사용하였으며, 이 규칙에 의해 book_order.xml에서 동일한 고객의 책 주문 내용 통합에 따라 생성되는 엘리먼트 <book_order_contents>의 결과는 다음과 같다.

```
<book_order_contents>
  <book_order_id> B22110 </book_order_id>
    <book_title> Science </book_title>
    <num> 1 </num>
    <date> 2002-5-10 </date>
  <book_order_id> A10001 </book_order_id>
    <book_title> Seven </book_title>
    <num> 2 </num>
    <date> 2002-5-5 </date>
  <name> Hwang Jeong Hee </name>
  <address> Cheong Ju Chungbuk University </address>
  <phone>
      <area_code> 043 </area_code>
      <phone_number> 267-2252 </phone_number>
  </phone>
</book_order_contents>
```

5.2 유효성 검사를 위한 제약 강화

XML 문서의 구조를 나타내는 DTD는 문서의 구성과 관계에 대하여 엘리먼트와 속성을 이용하여 지원할 뿐 DTD에 대한 조작 즉, 구조에 대한 검색이나 다른 문서로의 변형이 불가능하고 엘리먼트의 내용에 대한 데이터 타입도 텍스트 데이터(PCDATA)만을 표시하므로제한적이다. 이러한 DTD의 제한적인 문서 구조표현을 개선하고자 XML 스키마의 사용이 점차로 대두되고 있으며 이러한 XML 스키마는 DTD에 비해 XML의 기본 문법 형태로 정의할 수 있고 엘리먼트의 삽입과 삭제가 가능하다는 장점이 있다. 그러나 아직까지 많은 XML 문서가 DTD를 중심으로 하여 사용되고 있는 것을 고려하여 앞에서 제시한 능동 규칙을 사용하여 DTD에서 지원되지 않는 제한적인 구조 표현을 다음과 같이 개선할 수 있다.

(1) 엘리먼트의 내용과 속성 값에 대한 데이터 타입에 제한을 두어야 하는 경우, 즉 문자 혹은 숫자만 가능한 경우, 또는 숫자 값이 0이면 안 되는 경우 등과 같이 데이터 타입과 속성 값의 도메인 범위를 능동 규칙의 조건에서 비교하여 문서 변화에 따른 데이터의 유효성 검사가 자동으로 이루어 질 수 있다.

(2) DTD 선언에서 엘리먼트의 반복 횟수를 나타내는 *, +는 일정한 횟수의 반복만을 허용 하는 경우에 DTD에서는 일정한 횟수에 대한 제한을 둘 수 없다, 따라서 능동 규칙을 이용하여 일정한 횟수의 엘리먼트 생성만을 허용하도록 하는 반복 횟수 제어가 가능하다.

(3) 숫자를 취급하는 엘리먼트의 내용에 대한 산술 계산이 필요한 경우에 DTD에서는 숫자를 취급하는 데이터 타입에 대한 별도 선언을 지원하지 않으므로 어느 특정 엘리먼트들의 내용 값들을 계산하여 새로운 엘리먼트를 생성하고자 할 때 많은 어려움이 있다. 이것은 능동 규칙을 이용한 산술 계산을 통해 엘리먼트의 생성이 가능하다.

(4) 문서내에서 동일 엘리먼트에 대한 내용 값에 따라서 정렬이 필요한 경우에 능동 규칙을 이용하면 동일 엘리먼트내에서도 정렬이 가능하다. DTD에서는 엘리먼트간의 순서 만 지정할 수 있을뿐 동일 엘리먼트가 반복적으로 생성되는 경우에 엘리먼트에 대한 정렬이 불가능하다. 따라서 능동 규칙을 사용하면 엘리먼트들의 내용에 따른 정렬도 가능하다.

이렇게 DTD에 의한 XML 문서 생성시 제한적인 문서 구조 표현을 능동 규칙을 이용함으로써 각 문서가 갖는 구조의 특성을 살릴 수 있고 자칫 지나치게 길어질 수 있는 문서의 크기와 정렬이 필요한 경우에 문서의 통합과 정렬이 가능하므로 문서의 구조와 내용을 함축적으로 표현할 수 있어 문서의 자동 관리에 유용하다.

6. 기존 연구와의 비교 분석

이 장에서는 [6]에서 제시된 XML 기반의 능동 규칙에 대

한 종료 분석 방법에 대하여 이 논문에서 제안하는 분석 방법과의 비교를 통해 우리가 제안하는 분석 방법의 효율성을 검증한다.

[6]에서는 그들이 기존의 능동 규칙에 대해 제안하였던 방법인 규칙 동작에 대한 추상적 해석[11] 기반의 종료 분석 방법을 제시하였다. 다음 규칙은 [6]에서 종료 분석 방법을 설명하기 위해 사용된 규칙의 예이다. 우리도 이 규칙을 이용하여 [6]의 분석 방법과 비교한다.

```
rule 1 : on INSERT/a/*          rule 2 : on DELETE /a/*
         if /a/b and a/c                  if TRUE
         do DELETE b                      do INSERT <c/>
            BELOW /a                         BELOW /a
```

[6]의 분석방법은 정의된 각 규칙에 대하여, 추상적 초기 데이터베이스 상태에서 규칙의 조건 값이 알려져 있지 않은(Unknown) 모든 동작 가능한 규칙의 초기상태에 대한 추상적 실행에 따라 규칙 관계에서 발생하는 각 규칙의 조건 값의 변화를 예측하고 조건이 거짓으로 변하여 더 이상 실행이 불가능하다는 추상적 결과가 나오면 그 규칙의 집합은 종료할 수 있다는 결정을 한다. 즉, 위 예에서 규칙 ri의 조건과 조치를 ci, ai라 할 때 초기의 규칙 실행이 r1으로 시작된다고 가정하면 r1, r2의 반복적 실행으로 인한 조건의 결과는 (그림 4)와 같이 예측하여 r1은 3번 반복 실행 전에 종료할 것이고, r2가 초기에 실행된다고 가정하면 (그림 5)에서와 같이 c1의 조건값이 더 이상 실행이 불가능한 거짓이 될 것이라고 예측하여 4번 반복 실행 이내에 종료하게 될 것이라는 결정을 한다. 그러므로 이러한 조건 값의 추상적 예측 결과에 의해 r1과 r2를 포함하는 규칙 집합은 종료한다고 결정한다.

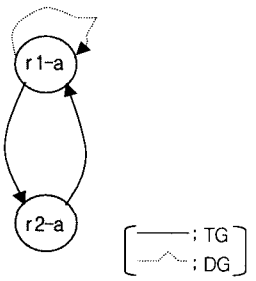
Iteration	c 1	c 2	Schedule
1	Unknown	True	[a 1]
2	False	True	[a 2]
3	False	True	[ ]

(그림 4) r1의 초기 실행 예측

Iteration	c 1	c 2	Schedule
1	Unknown	True	[a 2]
2	Unknown	True	[a 2]
3	False	True	[a 2]
4	False	True	[ ]

(그림 5) r2의 초기 실행 예측

그러나 우리가 제안하는 비활성화 관계를 이용하는 종료 분석 방법은 [6]에서 제안한 분석 방법과는 달리 트리거 그래프와 비활성화 그래프의 결합그래프[23]를 이용하여 간단하게 종료 분석할 수 있다. 위 규칙 r1과 규칙 r2의 규칙 관계는 앞에서 제시한 [정의 1]과 [정의 3]에 의하여 (그림 6)과 같은 그래프를 형성한다.



(그림 6) 트리거 그래프(TG)와 비활성화 그래프(DG)

(그림 6)에서 규칙 이름을 나타내는 r1-a의 a는 "AFTER" 규칙을 의미하고, 규칙 r1과 규칙 r2는 서로를 트리거 하는 관계이다. 그러므로 규칙 r1 또는 규칙 r2의 규칙 실행은 서로 다른 규칙을 연속적으로 트리거할 수 있다. 그러나 규칙 r1은 앞에서 정의한 [정의 3]에 의해 자신의 조건 값을 거짓으로 변화시키는 비활성화 관계이다. 따라서 (그림 3)의 종료 분석 수행 알고리즘에 의하여 비활성화 관계를 포함하는 규칙 r1과 규칙 r2의 실행 사이클은 그래프에서 제거되고 이 규칙 집합은 종료한다는 결정을 한다.

이와 같은 분석 방법의 차이에서 나타나는 것과 같이, [6]에서 제시한 분석 방법은 발생 가능한 규칙의 모든 실행 관계를 개별적으로 예측하여 조건 값의 변화에 기반한 분석 과정을 거치므로 분석 결과를 얻기까지의 분석 과정이 복잡하다. 그리고 데이터베이스의 상태 변화에 자동으로 대응하는 규칙 실행의 모든 상태를 정확하게 예측하여 반영하기 어려우므로 정확한 분석 결과를 기대할 수 없다. 반면에 규칙실행 그래프를 기반으로 하는 우리의 분석 방법은 사이클 내에 존재하는 비활성화 관계를 이용하여 트리거되더라도 조건을 만족하지 않아 조치까지의 완전한 규칙 실행이 불가능한 연속적 실행의 차단 요소를 도출해내는 분석 방법이므로 규칙의 종료 여부를 간결하게 결정할 수 있는 효과를 얻을 수 있다.

7. 결 론

인터넷, 전자상거래 등과 같은 웹의 발달과 더불어 웹의 표준으로 제시되고 있는 XML의 급속한 사용 증가로 XML 저장관리 시스템 및 XML 데이터의 변화에 자동으로 대응할 수 있는 기능을 지원하기 위한 규칙기반의 기술개발에 대한 연구가 최근에 빠르게 진행되고 있다. 따라서 이 논문에서는 XML 문서를 자동으로 관리하기 위한 능동 규칙 언어를 제안하였고 이 규칙언어를 이용하여 정의된 능동 규칙의 비종료를 방지하기 위한 종료 분석 방법에 대하여 제시하였다. 그리고 XML 문서의 능동적 관리를 위한 규칙의 적용 사례를 설명하였다.

XML 기반의 능동 규칙언어는 XML 문서들로 이루어져 있는 XML 데이터베이스에 대하여 능동 규칙을 표현할 수 있는 문법적 구조를 정의하는 것으로 기존의 형식과 같은 ECA 형태로 구성되며 XML의 특성을 반영하도록 하였다. 따라서 이 규칙언어에 의해 정의된 능동 규칙은 XML 문서의 생성과 삭제 수정시에 자동으로 대응할 수 있으며 사용자나 응용프로그램에서 요구하는 형태로 문서를 관리할 수 있고 문서의 자동 생성과 통합도 가능하다.

또한 XML 기반의 능동 규칙집합에 대하여 종료 여부를 결정하는 분석 방법으로는 XML 문서의 특성을 고려하고 규칙의 실행관계 그래프를 이용하는 분석 방법을 제시하였다. 즉, 규칙의 사건, 조건, 조치에서 명세되는 XML 문서의 경로를 기반으로 하는 규칙간의 트리거 관계와 규칙의 연속적 반복 실행의 고리를 제거할 수 있는 비활성화 관계를 이용하는 분석 방법을 제시하므로써 규칙의 종료 여부를 쉽게 식별 가능하도록 하였다.

마지막으로 제시된 분석 방법의 효율성을 검증하기 위하여 기존에 제시된 분석 방법과의 비교를 통하여 우리가 제안하는 분석 방법과의 차이점을 기술하였다.

현재 이 논문에서 제안한 XML 기반의 능동 규칙 언어는 XML의 특성에 적합한 능동 규칙의 실행의미를 반영하는 연구가 계속되고 있고 이것은 XML 기반의 웹 데이터에 대한 능동 마이닝 기법에 적용할 계획이다.

참 고 문 헌

[1] W3C, Extensible Markup Language(XML) 1.1. http://www.w3.org/TR/xml11, W3C Working Draft. April, 2002.

[2] 박상원, 정재목, 정태선, 김형주, "XML과 데이터베이스", 정보과학회지, 특집호, 2001.

[3] Natanya Pitts, editor, "XML Black Book 2nd Edition," Young-Jin, 2001.

[4] A. Bonifati, "Active Behaviors within XML Document Management," EDBT, 2000.

[5] A. Bonifati, S. Ceri, and S. Paraboschi, "Pushing Reactive Services to XML Repositories using Active Rules," In Proc. 10th World-Wide-Web Conference, 2001.

[6] J. Bailey, A. Poulovassilis, P. Wood, "Analysis and Optimization of ECA rules on XML," Computer Networks Journal, pp.1-21, 2001.

[7] A. Bonifati, S. Ceri and S. Paraboschi, "Active rules for XML : A New Paradigm for E-services," VLDB Journal, Vol.10, No.1, pp.39-47, 2001.

[8] N. Paton, editor, Active Rules in Database Systems, Springer-Verlag, 1999.

[9] J. Widom, S. Ceri, editor, Active Database Systems, Morgan-Kaufmann, San Mateo, California, 1995.

[10] "Simple Object Access Protocol (SOAP)1.1 (W3C Note)," http://www.w3c.org/TR/2000/note-soap-20000508/, 2000.

[11] J. Bailey, A. Poulovssilis, "An Abstract Interpretation Framework for Termination Analysis of Active Rules," In Proc. 7th Int. Workshop on Database Programming Languages, LCNS. Scotland, pp.249-266, 1999.

[12] E. Baralis, S. Ceri, S. Paraboschi, "Run-Time Detection of Non-Terminating Active Rule System," Proc. of the 4th Intl. Conf. on Deductive and Object-Oriented Databases, DOOD'95, Singapore, 1995.

[13] E. Baralis, S. Ceri, S. Paraboschi, "Improved Rule Analysis by Means of Triggering and Activation Graphs," Proc.of 2nd intl. Workshop on Rules in Database Systems, RIDS '95, Athens, Greece, 1995.

[14] S. Ceri and J. Widom, "Deriving Production Rules for Constraint Maintenance," In Dennis McLeod, Ron Sacks David, and Hans Schek, editors, Proc. Sixteenth Int'l Conf. on Very Large Data Bases, Brisbane, Australia, pp.566-577, 1990.

[15] J. Bailey, L. Crnogorac, K. Ramamohanarao, H. Sondergaard, "Abstract Interpretation of Active Rules and Its Use in Termination Analysis," ICDT'97, Lecture Notes in Computer Science, 99, pp.199-202, 1997.

[16] Mattos, Nelson M., An Overview of the SQL3 Standard, Database Technology Institute IBM-Santa Teresa Lab., 1996.

[17] S. Abiteboul, B. Amann, S. Cluet, A. Eyal, L. Mignet and T. Milo. Active views for electronic commerce. In Int. Conf. on Very Large DataBases (VLDB), Edinburgh, Scotland, 1999.

[18] C. Zaniold, S. Ceri, C. Faloutsos, R. T. Snodgrass, V. S. Subrahmanian, R. Zicari, "Design Principles for Active Rules," Chapter 4, Advanced Database Systems, Morgan Kaufman Pub, 1997.

[19] A. Vaduva, S. Gatziu, Klaus R. Dittrich, "Investigating Termination in Active Database Systems with Expressive Rule Languages," RIDS, pp.149-164, 1997.

[20] World Wide Web Consortium, XML Path Language (XPath), Version 1.0, http://www.w3.org/TR/xpath, W3C Recommendation, 1999.

[21] World Wide Web Consortium. XQuery1.0 : An XML Query Language, http://www.w3.org/TR/xquery, W3C Working Draft, 2002.

[22] A. Aiken, J. M. Hellerstein, J. Widom, "Static Analysis Techniques for Predicting the Behavior of Active Database Rules," ACM Transaction on Database System, Vol. 20, No.1, pp.3-41, 1995.

[23] Ye Ho Shin, Jeong Hee Hwang, Keun Ho Ryu, "Termination Analyzer including Rule Execution Semantics," KIPSD, Vol.8-D, No.5, pp.513-522, 2001.

[24] Bing Ji Jiang, Jeong Hee Hwang, Ye Ho Shin, Keun Ho Ryu, "A Study on Termination Analysis for Rule Compiler," KIPSD, Vol.8-D, No.6, pp.823-834, 2001.

황 정 희

e-mail : jhhwang@dblab.chungbuk.ac.kr

1991년 충북대학교 전산통계학과 졸업 (이학사)

2001년 충북대학교 대학원 전자계산학과 (이학석사)

2001년~현재 충북대학교 대학원 전자계산학과 박사과정

관심분야 : 능동 데이터베이스, 시공간 데이터베이스, XML, 데이터 마이닝

류 근 호

e-mail : khryu@dblab.chungbuk.ac.kr

1976년 숭실대학교 전산학과(이학사)

1980년 연세대학교 공학대학원 전산전공 (공학석사)

1988년 연세대학교 대학원 전산전공 (공학박사)

1976~1986년 육군군수 지원사 전산실(ROTC 장교), 한국전자통신연구원(연구원), 한국방송통신대 전산학과(조교수) 근무

1989년~1991년 Univ. of Arizona Research Staff(TempIS 연구원, Temporal DB)

1986년~현재 충북대학교 전기전자 컴퓨터공학부 교수

관심분야 : 시간 데이터베이스, 시공간 데이터베이스, Temporal GIS, 객체 및 지식기반 시스템, 지식기반 정보검색 시스템, 데이터 마이닝 및 데이터베이스 보안, 바이오인포메틱스