2001년 한국정보처리학회 춘계 학술발표논문집 제8권 제1호

시간 데이터베이스에서 능동 규칙 시스템의 설계 및 구현

김홍균*, 남광우*, 류근호*, 이종훈**
*충북대학교 컴퓨터과학과
**한국전자통신연구원 컴퓨터소프트웨어연구소
e-mail:kwnam@dblab.chungbuk.ac.kr

Design and Implementation of an Active Rule System on Temporal Databases

Hong Gyun Kim*, Kwang Woo Nam*, Keun Ho Ryu*, Jong Hun Lee**
*Dept of Computer Science, Chungbuk National University
**Computer and Software Technology Laboratory, ETRI

요 약

이 논문은 시간 데이터베이스에 능동 규칙 시스템을 결합한 능동적 시간 데이터베이스의 능동 규칙 및 실행 모델을 제안하며 그 모델을 설계 구현한다. 기존의 시간 데이터베이스 능동 규칙 연구들이 시간 릴레이션에 대한 조건-조치 형태의 규칙 지원에 머물렀던데 비하여 제안하는 능동 규칙 모델은 유효 시간 및 트랜잭션 시간을 지원하는 이원 시간 릴레이션에 대해 SQL3 트리거의 기능 모두를 시간 데이터베이스 상에서 지원한다. 즉, 전이 릴레이션 및 전이 변수, 조건-조치의 튜플 및 문 단위 수행을 포함한다. 또한, 능동 규칙의 유효 시간 정의 및 유효 시간 사건 등의 새로운 개념을 제시하고 시스템 구조 및 설계 구현에 대하여 기술한다.

1. 서 론[1]

능동적 시간 데이터베이스 시스템(active temporal database system)은 시간 데이터베이스 시스템[1]에 능동 규칙 시스템[2]을 결합한 진보된 데이터베이스 시스템으로 과거 데이터를 바탕으로 능동적으로 데이터베이스의 상황을 판단하여 적절한 조치를 취할 수 있다[3, 4]. 이 시스템은 추세 분석, 상태 관찰을 통한 경보, 고장에 대한 예측, 금융 시스템 및 공정 제어 등과 같은 고도의 응용에 효과적으로 대응할 수 있다. 실제 플랜트 컴퓨터 시스템(plant computer system), 주식 관리 시스템(stock management system), 네트워크 데이터 관리 시스템(network data management system)과 같은 응용들은 능동적 시간 데이터베이스 기능을 필요로 하는 미래 데이터베이스 응용들로 지적되고 있다[5].

능동 데이터베이스 시스템에 시간 개념 지원을 확장하려는 연구들은 그 동안 계속적으로 진행되어 왔다. 비교적 근접한 형태의 연구는 Chomicki[6]와 Sistla[7, 8]의 연구로 시간 데이터베이스가 아닌 일반 능동 데이터베이스 상에서 보조 릴레이션과 능동 규칙을 사용하여 PTL(past temporal logic) 등을 바탕으로 한 소급 갱신 및 선행 갱신 처리를 지원하고 있다. 또한 완전한 형태의 E-C-A (event-condition-action) 규칙이 아닌 C-A 형태를 지원한다. 시간 데이터베이스를 기반으로 능동 규칙 시스템을 결합하려는 연구는 Etzion 등이 시간 속성을 갖는 데이터 모델 상에서 능동 규칙을 지원하기 위한 PARDES[3]와 TALE[9] 등이 있다. 그러나, 이들에 의해 제시된 능동 시간 규칙은 C-A 형태를 가지며 매우 제한된 형태에 머물고 있다. 이외에 많은 연구들이 시간 데이터 모델을 고려하지 않은 일반 데이터베이스 상에서의 능동 규칙에 시간적 사건이나 시간 관련 표현을 포함하는 조건부의 표현을 가능하게 해주는 형태이다.

한편, 현재 데이터베이스 질의어 표준으로 제정된 SQL3는 능동 규칙을 지원하기 위하여 트리거를 포함함고 있다. 이와 함께 시간 데이터베이스 모델 및 질의어의 표준도 SQL3와 상호 변환 가능한 SQL/Temporal 표준 제안이 진행 중이다[10]. 이 논문에서 제안하고 있는 능동 규칙 모델은 SQL/Temporal 데이터 모델을 기반으로 SQL3 트리거의 모든 기능을 시간 데이터베이스 상에서 지원하도록 확장한다. 즉, 유효 시간 릴레이션, 트랜잭션 시간 릴레이션, 이원 시간 릴레이션에 대하여 전이 릴레이션 및 전이 변수, 조건-조치의 튜플 및 문 단위 수행을

이 연구는 한국전자통신연구원의 "4D 시공간 데이터 제공자 컴포넌트 개발" 위탁과제의 연구비 지원에 의해 수행되었음.

지원 할 수 있다. 또한, 능동 규칙을 이원시간 릴레이션에 저장함으로써 능동 규칙의 활성화 시간을 유효 시간으로 정의 할 수 있으며 트랜잭션 시간을 통해 능동 규칙의 변화 이력을 추적할 수 있는 기능을 제공한다. 뿐만 아니라, 시간 데이터베이스 상의 특정 유효 시간 영역에 대한 삽입, 삭제, 갱신에 대한 능동 규칙을 정의할 수 있도록 유효 시간 사건 등의 새로운 개념을 제시한다.

논문의 제 2장에서는 능동 규칙 언어의 의미, 그리고 능동 규칙에서 사용되는 전이 릴레이션과 전이 변수, 사건의 의미를 설명한다. 제 3장에서는 2장에서 제안된 모델을 위한 시스템의 구조에 대하여 설명하고, 제 4장에서 구현 알고리즘에 대하여 기술한다. 제 5장은 기존의 연구와 비교를 수행하며, 제 6장에서 결론을 맺는다.

2. 시간 능동 규칙 모델

제 2장에서는 능동 규칙의 기반이 되는 시간 데이터베이스에 대하여 정의하고, 이원 시간 릴레이션에서의 삽입, 삭제, 갱신 연산의 실행 의미에 대하여 기술하였다. 이 장에서는 시간 데이터베이스에서의 능동 규칙 베이스의 의미에 대하여 기술한다.

시간 능동 규칙 언어는 시간 릴레이션에 대한 트리거를 지원하는 SQL3 표준 트리거 언어의 확장이다. 시간 데이터베이스 능동 규칙 언어의 구문은 다음에서 보이는 것과 같이 트리거의 유효시간과 시간 데이터베이스 사건, 조건, 조치, 전이 변수 선언, 조건-조치 처리 단위 부분으로 이루어진다.

VALIDTIME *vt*
CREATE TRIGGER *rulename*
E_t
$[TV_t]$
$[G_t]$
[WHEN C_t]
A_t

위의 시간 데이터베이스 능동 규칙 선언에서 vt는 트리거가 유효하게 활성화되어 있는 시간을 정의하며, Et는 활성화 시간 동안 능동 규칙 모니터 해야 하는 시간 데이터베이스에서의 사건의 정의이다. Ct는 Et에 정의된 사건이 발생 했을 때 데이터베이스의 현재 상태와 TVt에 정의된 전이 변수에 의한 조건 판단에 대한 정의이며, At는 Ct의 조건을 만족할 때 수행해야 할 일련의 데이터 조작 연산으로 이루어진 조치들에 대한 정의이다. Gt는 Et가 발생했을 때 Ct와 At를 수행하는 단위에 대한 정의이다. 위의 시간 능동 규칙 언어에 대한 실행 모델이 다음에서 정의된다.

시간 능동 규칙의 실행 모델은 다음과 같이 정의된다.

$$[E_t \overset{\Delta_t,\delta_t}{\rightarrow} (G)(C_t \overset{\Delta_t,\delta_t}{\rightarrow} A_t)](vt)$$

시간 능동 규칙 TR은 유효한 활성화 시간을 정의하는 vt를 만족할 때 능동 규칙이 모니터 해야 하는 사건들의 집합 정의 Et, 사건 Et에 의해 트리거 되었을 때 처리해야 하는 데이터베이스의 현재 상태와 규칙의 전이 릴레이션 △t와 전이 변수 에 의한 조건 판단 정의 Ct, 조건 Ct를 만족할 때 수행해야할 일련의 데이터 조작 연산으로 이루어진 조치에 대한 정의 At의 네 개 구성요소에 의해 이루어진다. 이 논문에서는 각 규칙의 실행 의미를 표현하기 위해 다음과 같은 표현을 사용한다.

위의 표현에서 보는 것과 같이 시간 능동 규칙의 선언은 트리거 유효성 검사부, 사건 검출부, 조건 평가부, 조치 실행부로 이루어진다.

[예 2.1] EMP 릴레이션에 대해 2000년 12월 25일부터 2001년 1월 2일까지 휴가기간 동안 데이터를 삭제하려는 시도에 대해 audit 릴레이션에 접근 기록을 저장하라는 능동 규칙은 다음과 같다.

```
VALIDTIME PERIOD '[2000/12/25-2001/01/02)'
CREATE TRIGGER vocation_modify_audit
BEFORE DELETE ON emp
FOR EACH STATEMENT
INSERT INTO audit VALUES( $USER, 'emp', CT);
```

위의 능동 규칙에서 WHEN절은 생략될 수 있다. 이때 WHEN절은 항상 true를 의미한다. 또한, FOR EACH STATEMENT문에 의하여 사용자가 EMP릴레이션의 데이터를 수정하려는 접근에 대한 예 4.1의 능동 규칙은 각 수정문에 대하여 단 한번씩만 수행된다.

3. 시스템 구조

시간 데이터베이스 규칙 시스템은 규칙 카탈로그, 규칙 관리자, 사건 검출기, 조건 평가기, 조치 수행기 등 다섯 가지 구성요소로 이루어진다. 이들 각 구성요소의 구체적 기능은 다음과 같다.

- 규칙 카탈로그(TRB_{bt}) : 시간 능동 규칙을 저장하기 위한 이원 시간 릴레이션이다. 정의 3.3에서 설명된 것과 같이 규칙의 식별자 rid 속성, 규칙의 이름 name 속성, E-CA 수행 단위 granularity 속성, 규칙의 사건 정의 E_t를 저장하기 위한 event 속성, 조건부 C_t를 위한 condition 속성, 조치부 A_t를 위한 action 속성으로 으로 구성된다. 각 규칙 튜플의 유효시간은 규칙이 활성화 되어 있어야 되는 시간을 의미한다.

- 규칙 관리자(rule manager) : 규칙 카탈로그 TRB_{bt}의 규칙들의 유효 시간을 체크하여 활성화 시간에 도달한 규칙들을 메인 메모리 상의 유효 규칙 캐쉬로 적재하며, 유효시간을 지난 규칙들을 캐쉬로부터 제거한다.

○ 사건 검출기(event detector) : 정의 3.6에 정의된 규칙들의 사건부 E_t에 대해 사건 트리(event tree)의 형태로 관리하며, 질의 수행기로부터 발생한 사건에 대한 메시지를 받고 사건 트리로부터 사건을 검출한다.

○ 조건 평가기(condition evaluator) : 데이터베이스의 현재 상태와 정의 3.4와 정의 3.5에 기술된 전이 릴레이션 및 전이 변수에 대한 질의를 통하여 조건을 평가한다. 조건 평가는 규칙 처리 중 가장 많은 시간을 소모하는 부분이다. 그러므로 점진적 조건 평가 알고리즘을 사용한다. 조건 평가를 위한 전이 릴레이션 및 전이 변수는 질의 수행기로부터 전달 받는다.

○ 조치 수행기(action executor) : 규칙의 조치부 C_t에 기술된 데이터베이스에 대한 검색 질의 및 데이터의 삽입, 삭제, 갱신 연산문 및 사용자 정의 함수를 호출한다. 조치의 수행 결과는 시간 데이터베이스의 데이터를 변경시킬 수 있으며 이러한 변경은 또 다른 규칙을 트리거 시킬 수 있다. 이 때 규칙을 트리거 시킨 사건과 조치에 의해 발생하는 변경은 같은 트랜잭션으로 간주된다.

시간 데이터베이스의 능동 규칙 시스템에서 사건 검출기, 조건 평가기, 조치 수행기는 규칙 관리자의 유효 규칙 적재 및 제거 메시지에 의해 동적으로 처리해야할 데이터를 추가하거나 삭제한다. 즉, 규칙 관리자가 유효 시간 시작에 도달한 규칙을 규칙 카탈로그 TRB_{bt}로부터 유효 규칙 캐쉬에 적재 할 때 사건 검출기에 적재되는 규칙의 사건부 E_t를 적재 메시지를 통해 보내며, 조건부 C_t는 조건 평가기에, 조치부 A_t는 조치 수행기에 보내진다. 또한, 유효 시간이 종료된 규칙을 캐쉬에서 제거 할 때 제거되는 규칙에 대한 메시지가 사건 검출기, 조건 평가기, 조치 수행기에 보내지며 메시지를 전달받은 각 시스템은 그 규칙에 대한 정보를 제거한다.

4. 구현 알고리즘

시간 데이터베이스의 데이터 수정문을 수행하는 질의 수행기는 알고리즘 4.1의 수행 알고리즘과 같이 각 삽입, 삭제, 갱신 연산문에 대한 실제 변경을 수행하며 각 수행 단계에서 능동 규칙 시스템의 사건 검출기, 조건 평가기, 조치 수행기를 호출한다. 다음과 같이 네 개의 주요 함수로 구성된다.

○ compute_delta(query^{statement},type,target$_{bt}$,attr,vt,Δ_t)

compute_delta 함수는 수정문을 해석하고 전이 릴레이션 Δ_t를 계산하는 함수이다. 삭제, 갱신 수정문을 입력으로 받아서 질의 수정문의 타입 type, 대상이 되는 릴레이션 target$_{bt}$, 대상이 되는 속성 attr, 수정문이 대상으로 하는 유효시간 영역 vt, 영향을 받는 튜플 들의 전이 릴레이션 Δ_t를 반환한다. attr은 update에서 속성이 명시된 경우에 값을 가지며 그렇지 않을 경우 NULL값을 갖는다.

```
execute_bt(q_bt^statement)
{
  compute_delta(q_bt^statement, type, target_bt, attr, vt, Δ_t);
  TriggeredRules ← event_dectect(type, target_bt, attr, vt, BEFORE);
  for each tr ∈ TriggeredRules
    if tr[granularity] = STATEMENT
      if (condition_evaluation(tr, Δ_t, NULL) = true)
          action_activation(tr, Δ_t, NULL);
    else /* if tr[granularity] = ROW */
        for each δ_t ∈ Δ
          if (condition_evaluation(tr, Δ_t, δ_t) = true)
              action_activation(tr, Δ_t, δ_t);
  apply_delta(target_bt, Δ_t);
  // temporal constraint here
  TriggeredRules ← event_dectect(type, target_bt, attr, vt, AFTER);
  for each tr ∈ TriggeredRules
    if tr[granularity] = STATEMENT
      if (condition_evaluation(tr, Δ_t, NULL) = true)
          action_activation(tr, Δ_t, NULL);
    else /* if tr[granularity] = ROW */
      for each δ_t ∈ Δ
        if (condition_evaluation(tr, Δ_t, δ_t) = true)
            action_activation(tr, Δ_t, δ_t);
}
```

알고리즘 4.1 능동 규칙의 처리 알고리즘

○ event_detect(type, target$_{bt}$, attr, vt, mode)

event_detect 함수는 사건 검출을 위해 규칙 시스템의 사건 검출기에 사건 발생 메시지를 보내는 함수이다. 메시지는 수정문의 타입 type, 대상이 되는 릴레이션 target$_{bt}$, 대상 속성 attr, 사건의 대상이 되는 유효시간 영역 vt, BEFORE와 AFTER 사건 처리를 위해 발생한 사건의 전인지 후인지를 표현하는 mode로 구성된다. 사건 검출기는 처리 결과로 위의 사건을 만족하는 규칙들의 리스트를 결과 값으로 반환한다.

○ condition_evaluation(tr, Δ_t, δ_t)

condition_evaluation 함수는 규칙 시스템의 조건 평가기에 조건 발생 메시지를 보내는 함수이다. 메시지는 조건 평가에 평가해야하는 규칙 tr, 사건에 의해 생성된 전이 릴레이션 Δ_t, 현재 처리중인 변수 δ_t로 구성된다. 만약 규칙이 STATEMENT 단위의 실행으로 설정되었다면 δ_t는 NULL이다. 조건 평가의 반환 값은 조건이 만족할 때 true, 만족하지 않을 때 false를 반환한다.

○ action_activation(tr, Δ_t, δ_t)

action_activation 함수는 규칙 시스템의 조치 수행기에 조치 수행 메시지를 보내는 함수이다. 메시지는 조치 수행에 사용되는 규칙 tr, 사건에 의해 생성된 전이 릴레이션 Δ_t, 현재 처리중인 전이 변수 δ_t로 구성된다. 조치문에서 SET new.attr=value 형태의 구문을 사용하여 δ_t의 값을 변경할 수 있다. 조치의 수행은 다른 규칙들을 트리거 할 수 있다. 단 BEFORE문에 의한 조치문은 데이터의 회귀적 변경을 초래할 수 있으므로 조치문에 수정문이 허용되지 않는다.

알고리즘 4.1의 질의 수행기 알고리즘에서 만약 트리거 된 규칙이 E-CA가 명령문 단위로 수행되는 STATEMENT 모드라면 규칙은 트리거 된 사건에 대해 단 한번 수행된다. 그리고 튜플 단위의 수행 모드인 ROW라면 각 전이 릴레이션에 있는 각 튜플당 한번씩 조건을 평가하고 조건을 만족할 경우 조치를 수행한다. 또한, event_detect 함수에 의해 트리거 되는 규칙이 하나 이상일 경우 규칙들은 규칙들이 TRB_{bt}에 삽입된 트랜잭션 시간 순으로 수행된다. 이 것은 먼저 정의 된 규칙의 수행 결과에 대해 항상 후에 정의된 규칙의 조치가 수행되는 것을 보장한다.

5. 기존 연구와의 비교

Sistla의 연구들[7,8]은 능동 시간 데이터베이스와의 시간 능동 데이터베이스의 중간점에 존재한다. 그러나, 이 연구들은 시간 데이터베이스를 기반으로 하지 않고 관계형 데이터베이스 상에서 보조 릴레이션을 통해 이미 선언된 PTL과 FTL을 통해 시간 개념을 지원하므로 능동 규칙 선언 이후의 시간 데이터에 변경에 대해서만 조건 평가를 수행할 수 있다는 점에서 이 논문의 능동 시간 데이터베이스와 차이점을 갖는다.

Gal, Etzion 및 Segev는 [3]에서 시간 함수들, 특히 선행 갱신과 소급 갱신의 처리와 관련된 PARDES의 확장에서 능동적 시간 데이터베이스(active temporal databases) 모델을 제안하고 있다. 그러나, 이 논문의 모델에서는 전이 릴레이션과 전이 변수, 규칙의 수행 단위, 조치에서의 전이 변수 수정과 같은 기능에 중점을 둔 것이 큰 차이이다. 아울러 의미 모델과 실행 알고리즘을 구체적으로 제시하였다.

6. 결 론

이 논문은 SQL/Temporal 데이터 모델을 기반으로 유효 시간 릴레이션, 트랜잭션 시간 릴레이션, 이원 시간 릴레이션에 대하여 전이 릴레이션 및 전이 변수, 조건-조치의 튜플 및 문 단위 수행을 지원하도록 하였고, 능동 규칙의 활성화 유효 시간과 유효 시간 사건 등의 새로운 개념을 갖는 능동 규칙 모델과 실행 모델을 제안하였다. 아울러 제안된 모델을 증명하기 위하여 예를 제시하고 구현 알고리즘을 제시하였다.

한편 이 논문에서는 시간 데이터베이스 상에서의 규칙의 충돌에 대한 내용과 시간 무결성 제한자의 지원에 대한 문제점에 대한 해결책은 차후 연구로 남겨두었다.

참고문헌

[1] R. Snodgrass and I. Ahn, "Temporal Databases," *IEEE Computer*, vol.19, no.9, pp.35-42, September 1986.

[2] J. Widom and S. Ceri, ed. *Active Database System, Triggers and Rules For Advanced Database Processing*, Morgan Kaufmann Publishers Inc., p.2, 1996.

[3] O. Etzion, A. Gal, and A. Segev, "Data Driven and temporal Rules in PARDES," *in Proc. of Conf. on Rules in Database Systems*, pp.92-108, Edinburgh, Scotland September 1993.

[4] 박정석, 신예호, 남광우, 류근호, "시간지원 능동 규칙의 점진적 조건 평가," 정보과학회 논문지(B), 제 26권 제 4호, 462-472쪽, 1999년 4월.

[5] R. Chandra and A. Segev, "Managing Temporal Financial Data in an Extensible Database," *in Proc. of VLDB Conf.*, pp.302-313, Dublin, Ireland, August 1993.

[6] J. Chomicki and D. Toman, "Implementing Temporal Integrity Constraints using an Active DBMS," *IEEE Trans. on Knowledge and Data Engineering*, vol.7, no.4, pp.566-581, April 1995.

[7] Sistla, P. and O. Wolfson, "Temporal Conditions and Integrity Constraints in Active Database Systems," *in Proc. of ACM SIGMOD Conf.*, pp.269-280, San Jose, USA, May 1995.

[8] Sistla, P. and O. Wolfson, "Temporal Triggers in Active Databases," *IEEE Trans. on Knowledge and Data Engineering*, vol.7, no.3, pp. 471-486, June 1995,

[9] Gal, Avigdor, Opher Etzion, and Arie Segev, "TALE: A Temporal Active Language and Execution Model," *Lecture Note in Computer Science*, vol.1080, pp.61-81, May 1996.

[10] R. T. Snodgrass, M. Bohlen, C. Jensen, and A. Steiner, "Transitioning Temporal Support in TSQL2 to SQL3," *Lecture Note in Computer Science*, vol.1399, pp.150-194, June 1997.

[11] D. Klaus and S. Gatziu, "Events in an Active Object-Oriented Database System," *in Proc. of the Int. Workshop on Rules in Database Systems*, pp.23-39, Edinburgh, Scotland, September 1993.

[12] N. Gehani, H. Jagadish, and I. Mumick, "Event Specification in an Active Object-Oriented Database." *in Proc. of ACM SIGMOD Conf.*, pp.81-90, San Diego, USA, June 1992.

[13] E. N. Hanson and L. Noronha, "Timer-Driven Database Triggers and Alerters: Semantics and a Challenge," *SIGMOD Record*, vol.28, no.4, pp.11-16, December 1999.