능동 규칙의 실행의미를 반영한 종료 분석기

신 예 호[†] · 황 정 희[††] · 류 근 호[†††]

요 약

능동 데이터베이스는 미리 정의된 규칙(rule)에 의해 규칙에 대응하는 사건이 발생하면 자동으로 트리거 되어 조건을 평가하고 조치를 수행한다. 이러한 능동 규칙은 연속적으로 서로 다른 규칙을 트리거 할 수 있고 그로 인해 종료하지 못하고 무한한 순환적 동작을 반복할 가능성이 있다. 따라서 이 논문에서는 규칙집합의 종료를 보장하는 종료 분석방법을 제안한다. 여기서 제안한 방법은 정확한 종료 분석을 위해 복합사건의 규칙과 규칙 실행시간을 고려한다. 아울러 규칙간에 형성되는 비활성화 관계를 이용하여 그래프로 표현한 비활성화 그래프를 기존의 트리거 그래프와 결합시킴으로서 규칙 종료의 복잡성 개선과 분석 결과의 정확성을 향상시킨다.

A Termination Analyzer Including Execution Semantics of Active Rules

Ye Ho Shin[†] · Jeong Hee Hwang[††] · Keun Ho Ryu[†††]

ABSTRACT

Active databases execute an action of active rule defined in advance which is triggered automatically, whenever an event with the matching event specifications occurs, its condition is evaluated. Because these rules may in turn trigger other rules, the set of rules may be triggered each other indefinitely. Therefore, we propose a termination analysis method to guarantee termination. This proposed method considers composite event as well as rule execution time. Above all, the method not only uses deactivation graph combined to trigger graph for exact analysis, but also improves the complexity of analysis. Also the proposed method enhances accuracy of analysis result.

키워드 : 능동 데이터베이스(Active Database), 능동규칙(Active Rule), 규칙 종료 분석(Rule Termination Analysis), 규칙 실행 의미(Rule Execution Semantics)

1. 서 론

능동 데이터베이스는 사건(event), 조건(condition), 그리고 조치(action)로 정의된 규칙에 의해 데이터베이스 상태 변화에 자동으로 대응할 수 있는 능동적 특성을 지니고 있다. 규칙은 정의되어 있는 사건범주에 일치하는 데이터베이스의 상태변화가 감지되면 트리거 된다. 그러나 규칙이 트리거 된다고 해서 트리거 된 모든 규칙이 조치까지의 완전한 수행이 이루어지는 것은 아니다. 왜냐하면 규칙의 조건절에서 명시된 평가 결과가 참일 때에만 조치를 수행할 수 있기 때문이다. 이 규칙의 능동적 특성은 사용자 개입 없이 데이터베이스 스스로 적절한 조치를 수행할 수 있도록 하기 때문에 데이터베이스 보안, 데이터 무결성 관리, 워크플로우 관리 등 다양한 응용에 적용 가능하다[1].

이러한 능동 데이터베이스는 여러 응용 분야에서 유용하게 쓰일 수 있는 많은 장점이 알려져 있음에도 불구하고 실제적으로 널리 사용되고 있지 않는 실정이다[2]. 이것은 능동 데이터베이스에서의 능동 규칙의 실행 동작과 규칙간의 상호 연관성 등에 대한 예측이 어렵기 때문이다. 그 중에서도 가장 중요한 문제는 규칙집합의 실행 종료를 얼마나 보장할 수 있느냐 하는 점이다. 그러므로 규칙간에 서로를 무한하게 트리거하는 관계를 미리 예측하여 조정할 수 있는 종료 분석 연구가 필요하다[2].

규칙 종료 분석방법으로는 정적분석(static analysis)방법과 동적 분석(dynamic analysis) 방법이 있다. 동적 분석방법[3, 4]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.

정적 분석방법[5]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 이

※ 이 연구는 과학재단 특정기초연구(1999-2-30300-006-3)의 연구비 지원으로 수행되었음.
† 준 회 원 : 충북대학교 대학원 전자계산학과
†† 준 회 원 : 충북대학교 대학원 전자계산학과
††† 종신회원 : 충북대학교 전기전자및컴퓨터공학부 교수
논문접수 : 2001년 5월 28일, 심사완료 : 2001년 7월 3일

벤트 형식과 규칙의 실행의미(rule execution semantics)를 반영하는 것이 어렵다.

규칙의 사건절에 명세되는 사건의 형태에는 기본사건(Primitive Event)과 기본사건의 조합으로 이루어지는 복합사건(Composite Event)이 있고, 규칙의 실행시점(Rule execution time)을 나타내는 before, after를 명세할 수 있다[6, 7]. before 규칙은 규칙을 트리거하는 사건이 발생하였을 때 사건을 발생시킨 데이터베이스 연산을 처리하기 전에 규칙을 먼저 수행하는 반면, after 규칙은 규칙을 트리거하는 사건이 되는 데이터베이스 연산을 처리한 후에 규칙을 수행하는 특성이 있다.

복합 사건과 규칙 실행 시점을 규칙의 사건절에 명세하면 규칙의 동작흐름이 달라질 수 있음에도 불구하고 규칙 종료에 대한 기존의 연구에서는 기본사건의 규칙만을 분석 대상에 반영했을 뿐, 복합사건이나 규칙의 실행시점을 나타내는 before, after 규칙을 고려하여 종료 분석방법을 제시하고 있지 않다. 따라서 이 논문에서는 정확한 종료 분석결과를 위해 기존의 연구에서 제시되지 않았던 복합사건의 규칙과 규칙 실행 시간을 포함시킨다. 아울러 정적 분석기법에서의 문제점인 규칙 종료 결정의 복잡성을 개선하기 위하여 비활성화 그래프(Deactivation Graph : DG)와 트리거 그래프(Trigger Graph : TG)를 결합시키는 능동 규칙의 종료 분석 방법을 제안한다. 이 논문에서 제시하는 규칙 종료 분석 방법에서 비활성화 그래프는 규칙의 조건 값을 거짓(false)으로 변경시키는 비활성화 관계[8]를 나타내고 규칙간의 상호 트리거 분석이 가능한 장점이 있다.

이 논문의 구성은 다음과 같다. 2장에서는 종료 분석 방법에 대한 기존의 연구 내용과 문제점에 대해 알아보고 3장에서는 이 논문에서 제안하는 규칙의 종료 분석을 하기 위해 규칙관계 정의와 규칙의 실행 관계를 나타내는 TG와 DG 결합그래프를 설명하고 4장에서는 3장에서의 규칙 관계정의와 규칙 실행 관계 그래프를 이용하여 사이클을 분석하는 방법과 절차를 기술한다. 그리고 5장에서는 제안된 종료 분석방법의 유용성을 증명하기 위한 적용사례로써 규칙 집합의 예를 이용하여 규칙이 종료 분석되는 과정을 설명하고, 기존연구와의 비교 및 평가를 한다. 마지막으로 6장에서는 결론을 맺는다.

2. 관련 연구

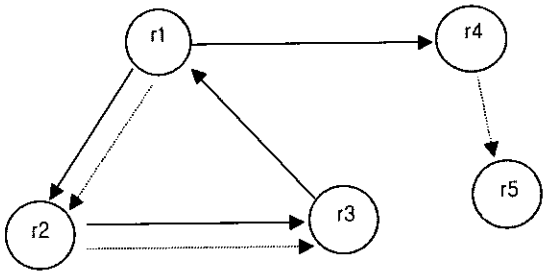
규칙종료는 능동 규칙을 구성하는 데 있어 기본적으로 분석, 검증되어야 할 특성으로 규칙들의 상호작용을 정확하게 이해해야 한다. 규칙종료 분석의 전형적인 방법은 트리거 그래프를 형성하여 규칙간의 상호관계를 파악하는 것이다[3, 4].

이 장에서는 기존의 연구들에서 이 논문과 가장 밀접한 관련이 있는 다음의 몇 가지 분석 방법을 소개한다.

2.1 활성화 그래프를 이용한 종료 분석 방법

능동 규칙 분석의 가장 기본적이고 중요한 문제라고 할 수 있는 규칙종료를 해결하기 위해서 최근까지 많은 연구들이 진행되고 있으며 이들 대부분은 트리거 그래프를 기반으로 한다[4]. 트리거 그래프[4, 9]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)[4, 10]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 1)은 [4, 11]에서 제시된 TG와 AG를 결합하여 규칙의 관계를 표현한 예이다.



(그림 1) 트리거그래프(TG : 실선)와 활성화그래프(AG : 점선)

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

또한 [12]에서는 규칙 관계를 나타내는 각각의 트리거 그래프와 활성화, 비활성화 그래프를 모두 고려하여 규칙의 실행 상태를 예측하고, 이를 전개 그래프(Evolution Graph)로 표현하여 같은 규칙 실행 상태가 존재하지 않으면 종료할 수 있다는 종료 분석 방법을 제시하였다. 이 방법에서는 각각의 규칙 실행 상태를 다음과 같이 네 가지로 나누어 기술하였다. 규칙 ri가 트리거되고 활성화 되는 상태를 s_i로, 트리거되고 비활성화되는 상태를 s_i^t로, 활성화되고, 트리거되지 않는 상태는 s_i^a로, 그리고 비활성화되고, 트리거되지 않는 상태는 s_i^n으로 표기한다.

이러한 네 가지 상태로써 트리거 그래프에서 사이클을 형성하는 각 규칙의 실행으로 인해 발생 가능한 다른 규칙과의 변화 상태를 함께 기술하여 같은 실행 상태가 반복되면 규칙 실행이 종료하지 않을 가능성이 있으므로 그 규칙 집합은 종료하지 않는다는 결정을 한다.

2.2 노드 제거에 의한 규칙종료 결정

규칙종료문제는 규칙간의 관계를 나타내는 그래프에서 사이클을 분석하는 것이다. 즉, 규칙관계를 표현한 트리거

그래프에서 사이클이 형성되지 않으면 규칙집합이 종료한다는 것을 결정을 할 수 있다[13, 14].

[4]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.

2.3 기존 연구의 문제점

규칙 집합에 대한 종료 여부를 예측하여 결정하기 위한 기존의 연구 방법에 대하여 위에서 설명하였다. 그러나 기존의 연구에서는 규칙 실행의미를 반영할 수 없다는 문제점과 그래프 표현상의 어려움 등이 있다.

- 규칙 실행 의미 미반영 문제

기존의 연구 방법에서는 단지 규칙을 구성하는 사건, 조건, 조치의 기본적인 역할만을 고려하여 종료 분석하는 방법을 제시하였을 뿐 사건절에 명세되어 규칙의 실행 의미를 포함하고 있는 복합사건의 형태와 규칙의 실행 시점을 나타내는 before, after 규칙의 실행 특성을 반영하고 있지 않다. 그러므로 복합사건의 경우에는 규칙의 실행 여부를 정확하게 반영하지 못하고, before, after 규칙의 경우에는 규칙의 실행 순서를 정확하게 반영하지 못하기 때문에 종료 분석 결과가 달라질 수 있다.

- 그래프 표현 문제

종료 분석은 규칙간의 관계를 표현한 그래프를 이용한다. 즉 규칙간의 트리거 관계 그래프(TG), 비활성화 관계 그래프(DG) 그리고 활성화 관계 그래프(AG)를 방향성 간선으로 표현하여 사이클을 분석하고 종료 여부를 결정한다.

그런데 기존의 연구에서는 TG, AG, DG 각각의 그래프[12], 또는 TG와 AG의 결합 그래프[4,11]를 통해 규칙 종료 분석을 수행하였다. 그러나 각각의 그래프를 상호 참조하여 종료 분석하는 방법은 대상 규칙의 관계를 각각의 그래프에서 참조하여 정보를 결합하여야 하므로 과정이 복잡하다. 또한 TG와 AG 결합 그래프는 연속적인 실행이 가능한 능동 규칙의 특성상, 실행 가능성만을 나타내는 것은 오히려 규칙 관계에 대한 표현이 더 복잡해 질 수 있기 때문에 종료 결정도 간단하지 않다.

3. 규칙 실행의미 반영 종료 분석

이 장에서는 기존에 제시되었던 종료 분석 방법의 문제점을 보완한 새로운 분석 방법, 즉 복합사건과 규칙 실행 시간을 고려하는 분석 방법을 제시하기 위하여 기본적인 규칙 관계를 정의하고 이를 표현한 규칙 실행 관계 그래프를 제시한다. 먼저, 트리거 수행의 종료를 결정하는 규칙 종료는 정의 1과 같다.

[정의 1] 규칙 종료 : 규칙은 규칙과 규칙사이에 서로 다른 규칙을 트리거하는 복잡한 처리들이 무한히 반복되지 않고 처리를 마친 마지막 상태에 도달한다.

이것은 규칙의 조치 실행으로 발생되는 사건에 의해 트리거되는 모든 규칙의 실행이 반복적으로 지속되지 않고 트리거된 모든 규칙의 실행이 완료되는 상태를 말한다. 그러므로 이러한 규칙집합은 종료한다는 보장을 할 수 있다.

규칙 집합 R = {r1, r2, r3, …, rn}이라고 하면 각 규칙의 사건절은 r.e, 조건절은 r.c, 조치절은 r.a로 표현한다. 규칙 관계의 기본 정의를 위해 규칙 ri의 실행은 Out(ri)로 표기하며, 규칙 rj을 트리거하는 규칙은 TR-by(rj)로 표기한다.

[정의 2] 트리거 관계 TR은 규칙 ri의 조치실행으로 인해 발생하는 사건이 rj를 트리거한다.

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

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

규칙의 사건절에는 기본사건외에 and, or의 논리연산자를 포함하는 복합사건과 규칙의 실행시점을 나타내는 before, after를 사용할 수 있다. 그러므로 이러한 규칙의 다양한 사건형태와 규칙의 실행 시점 의미를 포함하여 종료분석이 이루어진다면 더 정확한 분석 결과를 얻을 수 있다. 따라서 다음절에서는 이러한 특성들을 그래프에 포함하여 종료 분석하기 위한 규칙 실행 관계 그래프를 제시한다.

3.1 규칙 실행 관계 그래프

기존의 트리거 그래프에서 실선은 기본 사건의 트리거 관계를 나타낸다. 그러나 복합사건은 기본사건의 조합으로 형성되기 때문에 복합사건 규칙의 트리거는 기본사건과 구별하기 위해 점선으로 나타내어 트리거관계를 표시한다[6]. 이것은 복합사건을 트리거하는 잠정적인 트리거 관계를 표현하는 것이다. 그러므로 복합사건의 규칙을 트리거하는 규칙 관계에서 복합사건의 조합이 and 연산자로 이루어진 경우에는 트리거 관계의 모든 잠정적 기본사건이 발생해야 규칙이 트리거 될 수 있고, or인 경우에는 적어도 하나의 잠정적 기본사건이 발생해야 복합사건의 규칙이 트리거되므로 이와 같은 트리거 조건의 만족 여부가 종료분석에서 고려되어야 한다.

그리고 복합사건의 규칙에 대한 트리거관계를 나타낼 때는 조치절의 실행으로 트리거되는 규칙의 사건을 간선에 표시하여, 어떠한 사건으로 규칙이 트리거 되는지 원인이 되는 사건을 구분하고, 사건의 발생순서를 중요시하는 복합사건일 경우에는 사건의 발생순서를 참조한다. 이것은 복합사건의 트리거 여부를 정확하게 결정할 수 있도록 한다. 이

러한 복합사건의 트리거 관계를 정의하면 다음과 같다.

[정의 3] 복합사건의 규칙 r_j는 규칙 r_i, r_k에 의해 트리거되는 and, or 연산자에 의한 규칙이라고 할 때 다음의 조건에 의해 트리거 된다.

$$TR(r_i, r_k \rightarrow r_{j(and)}) = \{r_i, r_j, r_k \in R | E(Out(r_i) \cap TR\text{-}by(r_j) \neq \phi) \cup E(Out(r_k) \cap TR\text{-}by(r_j) \neq \phi) = r_j.E\}$$

$$TR(r_i, r_k \rightarrow r_{j(or)}) = \{r_i, r_j, r_k \in R | E(Out(r_i) \cap TR\text{-}by(r_j) \neq \phi) \cup E(Out(r_k) \cap TR\text{-}by(r_j) \neq \phi) \subseteq r_j.E\}$$

여기서 $E(Out(r_i) \cap TR\text{-}by(r_j) \neq \phi)$은 $TR(r_i, r_j)$관계에서 발생하는 사건을 의미한다. and 연산자에 의한 복합사건의 규칙 r_j은 잠정적으로 트리거하는 규칙관계 $TR(r_i, r_j)$, $TR(r_k, r_j)$ 에서 발생하는 사건이 규칙 r_j의 사건절에 명세되어 있는 모든 사건과 일치해야 규칙 r_j가 트리거 될 수 있다는 것을 의미하고, or 연산자에 의한 복합사건일 때는 규칙 r_j에 명세되어 있는 사건 중 어느 하나만 $TR(r_i, r_j)$, $TR(r_k, r_j)$ 관계에서 발생하면 트리거 될 수 있다는 것을 말한다.

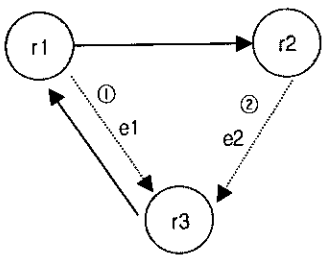
(그림 2)는 그래프에서 복합사건의 규칙 r3를 트리거하는 관계를 표현한 예이고, (1)과 (2)는 정의 3에 의한 복합사건의 트리거 조건을 and(conjunction)와 or(disjunction)표현한 것이다.

(1) 규칙 r1, r2의 and 결합으로 발생하는 복합사건에 의해 트리거되는 규칙 r3

$$r1(e1) \text{ and } r2(e2) = r3(e1 \wedge e2)$$

(2) 규칙 r1, r2의 or 결합으로 발생하는 복합사건에 의해 트리거되는 규칙 r3

$$r1(e1) \text{ or } r2(e2) = r3(e1 \vee e2)$$



(그림 2) 잠정적 사건 e1과 e2에 의한 복합사건의 트리거 표현

규칙의 실행 시점을 의미하는 before, after를 고려한 종료 분석을 위해서는 기존의 트리거 그래프에서 노드에 규칙 이름만을 나타내었던 것과는 다르게 규칙 이름에 before(b), after(a)를 구분하여 표시한다. 이것은 규칙의 실행시점을 명시한 before, after에 의해서 규칙의 실행 순서가 달라지므로 종료 분석 과정에서 고려되어야 하기 때문이다.

before, after 규칙의 실행의미를 반영할 경우 실제로 규칙이 실행되는 순서를 알아보기 위한 예로써, 규칙{r1-b, r2-a, r3-b}의 우선순위는 규칙의 순서에 의하고, 규칙이 트리거되어 실행된다고 할 때 실제적으로 규칙이 실행되어 완료되는 순서는 규칙 실행 시점을 나타내는 before와 after의 사용으로 인해 예측과는 다른 순서로 실행될 수 있다. 즉, r1의 실행에 이어 r2가 트리거 되고 조치가 실행될 때 before 규칙의 r3가 트리거되어 실행된 후 r2의 조치실행이 완료된다.

그러므로 규칙 실행 순서에 따라 생성되는 사건과 생성 순서가 달라질 수 있고 그로 인해 트리거 되는 규칙과 규칙의 실행여부 결정에 많은 영향을 준다. 즉, 규칙 실행 시점을 나타내는 before, after는 규칙의 우선 순위와 더불어 종료분석 결정에 중요한 요소로 작용한다.

종료 분석은 트리거 그래프를 이용한다. 이는 규칙의 실행 가능성을 표현한 것이다. 그러나 임의의 규칙 실행으로 인해 트리거된 규칙은 조건을 만족하지 않으면 조치까지의 완전한 실행은 불가능하다. 그러므로 규칙간에 형성되는 비활성화 관계는 규칙 종료 분석에 정확성을 제공하는 요소가 된다. 따라서 이 논문에서는 기존의 트리거 그래프에 비활성화 관계를 나타내는 비활성화 그래프를 결합하여 종료 분석한다. 비활성화 관계의 규칙 정의는 다음과 같고 이러한 조건을 만족하는 규칙 관계를 비활성화 그래프에 포함한다.

[정의 4] ri의 조치 실행으로 rj의 조건 값을 거짓으로 변화시키면 규칙 ri, rj는 비활성화 관계(DR)이다. 이것을 식으로 표현하면 다음과 같다.

$$DR(ri, rj) = \{ri, rj\ (i \neq j) \in R | Out(ri) \cap Trans\text{-}into(r_j.Ce = False) \neq \phi\}$$

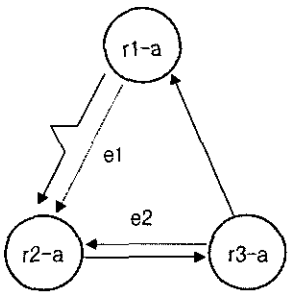
규칙 r의 조건 평가의 값(Condition Evaluation)은 r.Ce으로 표기하며 값의 범위는{True, False}이다. 그리고 Trans-into는 규칙의 조건 값을 변화시키는 규칙을 말한다.

따라서 비활성화 관계의 규칙 ri, rj는 비활성화 그래프에 포함된다.

(그림 3)은 복합사건의 규칙과 before, after 규칙을 포함하여 TG와 DG 결합그래프를 나타낸 예이다.

규칙의 조건절에 기준을 두는 비활성화 관계는 트리거되는 시점에 따라 실행여부가 달라질 수 있다. 즉, 규칙이 비활성화되었다고 해서 무조건 규칙이 실행되지 않는 것은 아니며 트리거 되는 시점과의 간격사이에 또 다른 임의의 규칙에 의해 활성화상태로 변경될 수 있다는 것을 고려해

야 정확한 종료 분석이 가능하다. 그러므로 사이클의 실행 여부는 규칙 베이스에 저장되어 있는 활성화 관계를 참조하여 결정한다.



(그림 3) 복합사건 및 before, after 규칙을 포함한 트리거 그래프(TG : 실선 및 점선)와 비활성화 그래프(DG : 각진 실선)

4. 사이클 분석 및 분석 절차

규칙 종료분석은 규칙관계를 나타내는 그래프에서 규칙이 종료하지 못하는 원인이 되는 사이클을 분석하는 것이다. 그러므로 그래프에 존재하는 사이클 제거를 위해 거짓(false) 사이클을 정의하면 아래와 같다.

[정의 5] 트리거 관계의 사이클을 형성한다 할지라도, 사이클 형성의 규칙집합에서 실행이 불가능한 규칙을 포함하고 있는 사이클은 거짓 사이클(False Cycle)이고 다음 조건을 만족한다.

$$\text{False Cycle(CR)} = \{r_i, r_j, r_k \in CR \mid (r_i.Ce = \text{False}) \text{ or } (r_j.Ce = \text{False}) \text{ or } (r_k.Ce = \text{False})\}$$

규칙 집합에 속하는 규칙 중 어느 하나라도 조건 값이 거짓이 되는 규칙을 포함하는 사이클, 즉 규칙의 실행에 있어 조건 값이 거짓이 되는 규칙과 비활성화 관계에 의해 조건 값이 거짓으로 변화되는 규칙(일정한 횟수의 반복 수행후 비활성화 되는 규칙 포함)을 포함하는 사이클은 순환적 연속실행이 불가능하므로 참(True) 사이클이 아니다.

그러므로 거짓 사이클 정의에 의해 (전제 1)에서는 사이클을 제거하기 위한 조건을 제시한다.

- 규칙 ri와 rj가 트리거 관계일 때는 TR(ri, rj)로 표기하고, 비활성화 관계일 때는 DR(ri, rj)로 표기한다.

(전제 1) 트리거 그래프에서의 사이클 제거 조건

TR(ri, rj)에서 rj가 and, or 연산자에 의한 복합사건의 규칙인 경우에 before, after 규 칙과 규칙의 우선순위를 고려하여 실제로 실행 불가능한 규칙을 포함하고 있는 사이클은 제거한다. 즉, or 연산자의 조합에 의한 규칙은 적어도 하나의 사건 발생이 존재해야 하고 and 연산자에 의한 경우에는 모든 기본사건이 발생해야 한다.

(예 1) TR(r1, rj), TR(r2, rj)의 트리거 관계에서 규칙 ri의 조치로 인해 발생하는 사건을 E(ri)라 하면 or의 경우에는 $E(r1 \vee r2) \subseteq E(rj)$, and의 경우에는 $E(r1 \wedge r2) = E(rj)$를 만족해야 한다.

1차 분석에서 정의 5에 의해 거짓 사이클을 제거하고, 존재하는 사이클에 대해 2차 분석 과정에서 제거되는 사이클 제거 조건은 정의 6과 같고 (전제 2)에서는 이 조건을 설명한다.

[정의 6] 사이클에서, 일정한 횟수의 실행 또는 조건을 거짓으로 변경시키는 비활성화 관계에 의해 $DR(r_i, r_j)$가 존재하는 사이클은 제거한다(단, 활성화 관계 $AR(r_k, r_j)$의 존재여부로 결정한다). 이것은 다음과 같이 표현한다.

$$\exists\{DE(r_i, r_j)\{\neg AR(r_k, r_j)\}$$

다음의 조건을 만족하는 사이클(CR)은 비활성화 관계를 이용하는 2차 분석에서 제거된다.

$$\text{Rid(CR)} = \{r_i, r_j, r_k \in CR \mid (\text{Out}(r_i) \cap \text{Trans-into}(r_j.Ce = \text{False}) \neq \phi) \text{ and } (\text{Out}(r_k) \cap \text{Trans-into}(r_j.Ce = \text{True}) = \phi)\}$$

(전제 2) 비활성화 그래프에서의 사이클 제거 조건

1) 일정 횟수의 반복 수행을 하면 더 이상 실행되지 않는 규칙 관계[15]를 포함하는 사이클, 즉 DR(ri, rj)에서
   ① ri의 조치절에서 속성값을 감소(예, -)시키고, rj의 조건절에서 [속성 > 정수] 값을 비교하는 경우와
   ② ①과 반대로, ri의 조치절에서 속성값을 증가(예, +)시키고, rj의 조건절에서 [속성 < 정수] 값을 비교하는 경우이다.

   즉, 규칙 관계가 사이클을 형성한다 할지라도 일정횟수만 반복 수행되는 경우에는 사이클을 제거하여 종료 분석한다.
2) 우선순위가 높은 규칙 또는 before 규칙의 수행으로 비활성화 되는, 실제적으로 실행 되지 못하는 규칙을 포함하고 있는 사이클을 제거한다.
3) ri의 조치로 인해 rj의 조건을 만족하지 않는 일반적인 비활성화 관계의 규칙을 포함 하는 사이클을 제거한다.

위에 제시된 조건에 의해 (그림 3)의 그래프를 이용하여 종료분석을 하기 위한 가정으로 규칙 r2-b는 or 연산자에 의한 복합사건이고 규칙의 우선순위가 r1-a, r2-b, r3-a의 순서로 실행된다고 할 때, 규칙의 실행은 우선순위에 의해

r1-a가 먼저 실행되고 그의 조치 실행전에 r2-b가 실행되고 r1-a의 조치실행 완료와 r3-a가 실행되는 과정을 통해 사이클이 형성된다. 그러나 사이클 형성의 규칙간에 비활성화 관계의 간선(r1-a, r2-b)이 존재하며, 이것은 r2-b의 조건값이 거짓이 되어 더 이상 실행이 불가능한 규칙관계를 의미하므로 이러한 규칙 관계를 포함하고 있는 사이클은 제거할 수 있다.

규칙 종료 분석방법에서의 효율성이란 종료를 결정하는 과정의 간결성과 분석결과의 정확성을 의미한다. 이 논문에서 제시하는 트리거 그래프와 비활성화 그래프를 이용한 종료 분석 절차는 다음과 같다.

단계 1 : 1차 분석

트리거 그래프를 구성하는 기초 분석과정으로써 동일한 목표 테이블에 대해 정의된 규칙 들의 트리거 관계를 그래프로 표현하여 사이클의 존재여부와 실행여부를 점검하여 종료 여부를 결정한다. 즉, 1차 분석은 규칙의 조치 수행으로 자신을 포함한 임의의 규칙을 트리 거하는 규칙 관계에 대해 분석하는 것이다.

분석 결과로써 사이클의 제거조건에 의해 제거되는 사이클을 제외한, 사이클 관계의 규칙 들에 대해서는 비활성화 관계를 이용하는 2차 분석을 통해 규칙 집합의 종료 여부를 결정 한다.

단계 2 : 2차 분석

비활성화 그래프는 규칙의 조치실행으로 자신을 제외한 임의의 규칙의 조건 값을 참에서 거짓으로 변화시킬 때 생성되는 규칙 관계의 간선을 그래프에 포함시켜 형성한다.

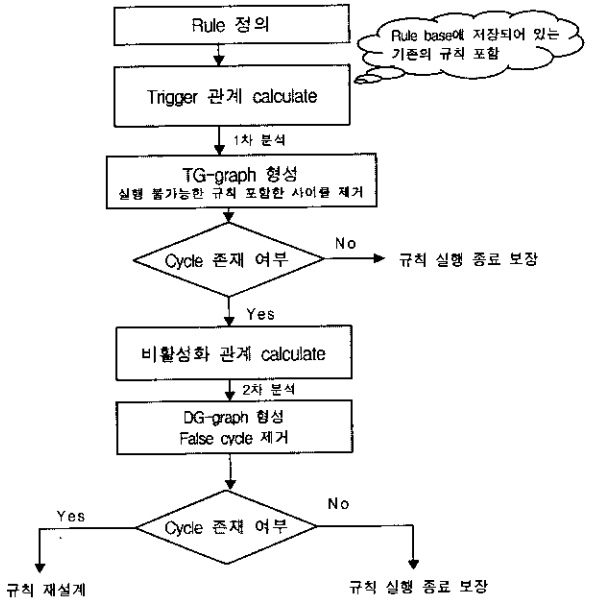
1차 분석의 트리거 그래프에서 사이클을 형성하는 규칙들 중에서 비활성화 관계에 의해 제거될 수 있는 사이클 즉, 거짓(False) 사이클의 규칙들은 제거된다. 단, 비활성화 관계의 규칙은 다시 활성화되지 않는 관계이어야 한다.

1차 분석은 규칙간의 조치절과 사건절의 관계를 분석하는 것이며 2차 분석은 1차 분석 결과를 가지고 조치절과 조건절을 상세하게 분석하는 것이다.

1, 2차의 분석과정을 마친 결과로써 규칙들의 관계를 표현한 그래프에 사이클이 존재하지 않는다면 규칙집합은 종료를 보장한다고 결정할 수 있다. (그림 4)는 규칙 종료 분석 절차를 나타낸 것이다.

지금까지 복합사건과 before, after 규칙을 고려하는 종료 분석을 위해, 기존의 트리거 그래프를 변형하여 규칙 실행 관계를 표현하는 트리거 그래프와 비활성화 그래프를 결합하여 종료 분석하는 방법을 설명하였다. 이 분석 방법에서 가장 최종적인 분석 결정은 비활성화 그래프를 이용한 2차 분석에서 이루어진다.

임의의 규칙 실행으로 인해 비활성화된 규칙은 트리거가 되더라도 조건을 만족하지 않으면 조치까지의 완전한 규칙 실행은 불가능하다. 그러므로 규칙의 조건에 기준을 두는 비활성화 관계는 사건에 기준을 두는 트리거 관계보다 더 상세한 규칙 정보를 제공한다. 단, 비활성화 된 규칙이 실행되기 전에 다시 활성화되지 않는다는 규칙 의미를 포함해야 한다. 따라서 규칙 베이스에 규칙 관계를 저장할 때 정확한 종료분석을 위해 활성화 관계(AR)를 저장한다.



(그림 4) 규칙 종료 분석 절차

5. 적용 사례

이 장에서는 적용사례를 통해 앞에서 제시된 종료 분석 방법에 의한 분석 과정을 설명한다.

(그림 5)는 종료 분석과정에 사용할 스키마의 예이다. 그리고 <표 1>은 규칙의 종료 분석에 적용하기 위한 규칙 집합이다. 규칙의 우선 순위는 아래와 같이 가정한다.

우선순위 : P(r1) > P(r2) > P(r3) > P(r4) > P(r5) > P(r6)

```
Account(no, bal, rate)     ; 고객번호, 잔액, 이율
Client(num, count, level)  ; 고객번호, 고객 보유 계좌수(1-10),
                             고객 등급(1-7)
Loan(no, zan, flag)        ; 고객번호, 대출잔액, check(Y,N)
```

(그림 5) 규칙 예의 스키마

〈표 1〉 규칙 집합의 규칙 정의

```
규칙 r1 : create trigger after-r1
          after update of bal on Account
          when bal > 1000
          then update rate = rate + 0.1

규칙 r2 : create trigger after-r2
          after update of rate on Account
          when rate > 7
          then update level = level + 1
```

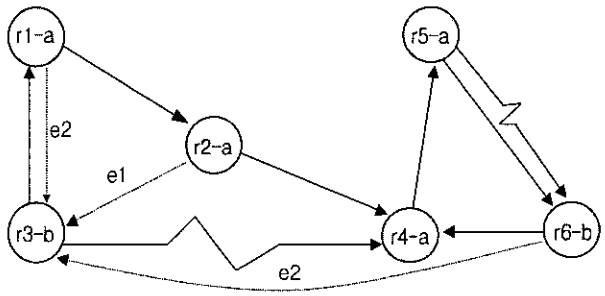
```
규칙 r3 : create trigger before-r3
          before update of rate on Account and level on Client
          when zan > 0
          then update bal = bal - zan
          set update flag = "Y"

규칙 r4 : create trigger after-r4
          after update of level on Client
          when level > 3 and flag = "N"
          then update zan = zan - bal

규칙 r5 : create trigger after-r5
          after update of zan on Account
          when zan = 0
          then update count = count - 1

규칙 r6 : create trigger before-r6
          before update of count on Client
          when count > 5
          then update level = level + 1
```

(그림 6)은 <표 1>에 정의된 규칙들의 관계를 TG와 DG 결합 그래프로 나타낸 것으로 규칙들과 사건 e1, e2로 이루어진 복합사건 관계를 표시하였다.



(그림 6) <표 1>의 규칙 관계를 나타낸 TG와 DG 결합 그래프

<표 2>는 규칙 정의에서 탐색된 각 규칙간의 TR, DR, AR의 관계이다. AR 관계는 규칙간에 형성되는 비활성화 관계를 결정할 때 참고 자료가 된다.

<표 2> 규칙간에 형성되는 TR, DR, AR 관계

rule ID	TR		DR	AR
r1-a	r2	r3		
r2-a	r3	r4		
r3-b	r1		r4	
r4-a	r5			
r5-a	r6		r6	
r6-b	r3	r4		

TR(ri, rj) - 규칙 ri의 조치절과 규칙 rj의 사건절에 같은 연산과 속성을 사용하는 트리거 관계

DR(ri, rj) - 규칙 ri의 조치절의 실행결과가 규칙 rj의 조건절의 값을 false가 되도록 하는 비활성화 관계

AR(ri, rj) - 규칙 ri의 조치절의 실행결과가 규칙 rj의 조건절의 값을 true가 되도록 하는 활성화 관계

위의 규칙 관계 데이터를 기초로 다음과 같은 식으로 규칙 관계를 표현하고 각 규칙의 트리거 관계를 전개하여 규칙간에 형성되는 사이클을 탐색할 수 있다.

① Out(r1) = TR[r2, r3], ② Out(r2) = TR[r3, r4]
③ Out(r3) = TR[r1] + DR[r4], ④ Out(r4) = TR[r5]
⑤ Out(r5) = TR[r6] + DR[r6], ⑥ Out(r6) = TR[r3, r4]

- Out(ri)는 ri의 규칙이 임의의 규칙을 트리거, 활성화 그리고 비활성화 시키는 규칙의 관계를 표현한다.
- 사이클이 발견된 규칙(CR)과 중복 수행하는 규칙(d)은 더 이상 경로를 전개하지 않는다.

① Out(r1) = TR[r2, r3]
⇒ {(r1, r2), (r1, r3)}
⇒ {(r1, r2, r3), (r1, r2, r4)} + **(r1, r3, r1)**^{CR1}
⇒ **(r1, r2, r3, r1)**^{CR2}, (r1, r2, r4, r5)
⇒ (r1, r2, r4, r5, r6)
⇒ {(r1, r2, r4, r5, r6, r3),
(r1, r2, r4, r5, r6, r4)^{d}}
⇒ **(r1, r2, r4, r5, r6, r3, r1)**^{CR3}

② Out(r2) = TR[r3, r4]
⇒ {(r2, r3), (r2, r4)}
⇒ (r2, r3, r1) + (r2, r4, r5)
⇒ {**(r2, r3, r1, r2)**^{CR2}, (r2, r3, r1, r3)}
+ (r2, r4, r5, r6)
⇒ {(r2, r4, r5, r6, r3), *(r2, r4, r5, r6, r4)*^{d}}
⇒ (r2, r4, r5, r6, r3, r1)
⇒ {**(r2, r4, r5, r6, r3, r1, r2)**^{CR3},
(r2, r4, r5, r6, r3, r1, r3)^{d}}

..

④ Out(r4) = TR[r5]
⇒ (r5, r6)
⇒ {(r5, r6, r3), (r5, r6, r4)}
⇒ (r5, r6, r3, r1) + **(r5, r6, r4, r5)**^{CR4}
⇒ {(r5, r6, r3, r1, r2), *(r5, r6, r3, r1, r3)*^{d}}
⇒ {*(r5, r6, r3, r1, r2, r3)*^{d},
{(r5, r6, r3, r1, r2, r4)}
⇒ **{(r2, r4, r5, r6, r3, r1, r2)**^{CR3},
(r2, r4, r5, r6, r3, r1, r3)^{d}}

규칙 집합의 각 규칙에 대해 이와 같은 방식으로 트리거 경로를 전개하면 존재하는 모든 사이클을 탐색할 수 있다.

(그림 6)에서 제시한 규칙집합에 존재하는 사이클은 CR1 <r1, r3>, CR2<r1, r2, r3>, CR3<r1, r2, r4, r5, r6, r3>, CR4 <r4, r5, r6> 4개를 찾을 수 있다. 이들 사이클은 CR1 ⊂ CR2 ⊂ CR3, CR4 ⊂ CR3 관계를 갖는다. 실행이 불가능

한 규칙을 포함하는 사이클이 제거되듯이 실행이 불가능한 사이클을 포함하는 사이클도 제거된다. 그러므로 사이클 CR1과 CR4가 제거되면 CR2와 CR3도 제거할 수 있다.

1차 분석에서 사이클을 제거할 수 있는 요소는 복합연산의 실행 여부이다. 그러므로 복합사건 규칙 r3의 실행 여부를 먼저 조사한다. and 연산자의 결합에 의한 복합사건의 규칙 r3-before는 먼저 실행되는 r1에 의한 잠정적 사건 e2와 r2의 e1에 의해 실행되어야 하나 규칙 r2는 규칙 r1의 실행 완료 후에 트리거 되어 실행되므로 r1의 조치 이전에 실행되는 r3의 before, and 규칙은 트리거되도록 하는 규칙 2의 사건 e1을 제공받지 못해 실행이 불가능하다(정의 5). 그러므로 이러한 규칙을 포함하고 있는 사이클 CR1<r1, r3>, CR2<r1, r2, r3>은 제거될 수 있고 이러한 사이클을 포함하는 CR3<r1, r2, r4, r5, r6, r3>도 제거된다. 실제적으로 실행이 불가능한 규칙 r3은 다시 수정되어 정의되어야 실행 가능한 규칙이 될 수 있다.

이로써 1차 분석 결과에서 존재하는 모든 사이클이 제거되고 남아있는 사이클은 CR4<r4,r5,r6>뿐이고, 이 사이클은 비활성화 관계를 이용하는 2차 분석을 통해 사이클의 실행여부를 결정하여 최종의 종료 분석 결과를 얻는다.

사이클 CR4에는 DR(r5, r6)관계가 존재한다. 이것은 r6가 r5에 의해 비활성화 되는 규칙 관계를 의미하므로 정의 6에 의해 사이클 CR4를 제거할 수 있다. 비활성화 관계에 대해 좀 더 자세히 살펴보기 위해 규칙 r5와 r6의 조건절과 조치절을 대수식으로 표현하면 다음과 같다.

$$\text{rule r5}: R_{C5} = \Pi\text{count}(\sigma_{\text{zan}=0}(\text{Loan}) \propto_{\text{no}=\text{num}} (\text{Client}))$$
$$R_{A5} = \text{Upd}\ [\text{count} = \text{count}-1]\ \sigma_{RC5}$$
$$\text{rule r6}: R_{C6} = \Pi_{\text{num,level}}(\sigma_{\text{count}} > 5\ (\text{Client})))$$
$$R_{A6} = \text{Upd}\ [\text{level} = \text{level}+1\ ]\ \sigma_{RC6}$$

여기서, 트리거 관계의 rule r5의 조치절, R_{A5}= Upd [count = count-1] σ_{RC5}과 r6의 조건절 $R_{C6} = \Pi_{\text{num,level}}(\sigma_{\text{count}} > 5$ (Client)))은 한정된 범위의 값(1-10)을 갖는 속성 count에 대해 일정한 횟수의 실행만이 가능한 비활성화 관계라는 것을 알 수 있다.

그러므로 정의 6에 의해 활성화 관계가 존재하지 않고, 비활성화 관계 DR(r5, r6)만 존재하는 사이클 CR3을 제거할 수 있다. 따라서 <표 1>의 규칙집합에 존재하는 모든 사이클이 제거되므로 종료를 보장할 수 있다는 분석 결과를 얻는다.

6. 기존 연구와의 비교 및 평가

이 논문의 종료 분석 방법은 복합사건의 규칙과 before, after 규칙의 실행의미를 반영한다. 그러므로 기존의 연구에서는 복합사건의 규칙과 before, after 규칙을 종료 분석 대상에서 고려하지 않았기 때문에 규칙 언어 사용이 제한적이고, 분석 결과에 정확성을 기할 수 없었다.

기존 연구 [11]에서 비활성화 시키는 규칙보다 우선 순위가 더 높은 규칙에 의해 트리거 되는 경우의 규칙 실행여부 결정에 고려되어야 하는 우선 순위의 언급과 마찬가지로, 트리거를 유발시킨 규칙보다 더 먼저 실행되는 before 규칙의 특성을 반영해야 정확한 종료 결과를 얻을 수 있다 그러므로, 이 논문에서는 규칙의 실행 시점이 명시적으로 표현되는 before, after 규칙의 실행의미를 반영했다는 것이 기존 연구와의 차이점이 있다.

[5]에서는 복합사건을 고려하는 종료 분석 방법을 제안하고 있다. 이 연구에서는 사이클을 이루는 규칙집합의 내부 규칙에 의해 트리거되는 규칙은 지속적으로 실행 가능하다는 결정을 하며, 비종료를 방지하기 위해서는 규칙에 의해 발생하는 사건이 아닌 외부 사건(Outside Event)을 복합사건에 포함시켜 규칙을 정의해야 한다고 기술하고 있다. 그러나 이것은 규칙의 조치실행으로 비활성화되는 규칙의 조건값과의 관계를 충분히 고려하지 않았기 때문이다. 즉, 외부 사건을 포함하지 않은 복합사건이 사이클 내부 규칙에 의해 트리거되어도 실행 불가능한 사이클이 될 수 있는다는 것을 고려하지 않았다. 또한 그래프의 표현에서도 복합사건이 어떠한 사건으로 일부 트리거되는지를 나타내지 않아 규칙의 실행여부를 정확하게 판단하기 어렵다.

그러나 이 논문에서 제시하는 복합사건의 그래프 표현은 규칙의 실행 가능성을 빠르고 정확하게 식별할 수 있는 장점이 있다.

[15]에서는 한정된 횟수로 실행되는 사이클은 종료를 보장할 수 있다고 제안하고 이러한 사이클의 그래프를 변형하여 사이클이 형성되지 않도록 한다. 그러나 그래프 변형 과정의 복잡성과 규칙간의 관계 표현 범위가 더 확대되어 본래의 규칙 관계를 식별하기 어렵다. 그러므로 이 논문에서는 이러한 그래프 표현의 복잡성을 개선하기 위하여 일정횟수의 수행가능 사이클을 비활성화 관계에 포함시켜 표현함으로써 사이클이 한정된 횟수의 실행 후 종료한다는 것을 쉽게 예측할 수 있도록 한다.

7. 결 론

능동 데이터베이스 시스템은 능동 규칙의 사용으로 데이터베이스 상태변화에 자동으로 대응하는 유용성을 제공한다. 그러나 능동 규칙은 자신을 포함한 다른 규칙을 트리거할 수 있다는 특성으로 인해 규칙이 종료하지 못하고 무한하게 반복 실행되어 시스템의 성능과 데이터베이스 상태에 바람직하지 않은 결과를 초래할 가능성이 있다. 그러므로 최근까지 이러한 능동 규칙의 실행을 미리 예측하여 무한

한 규칙의 반복 실행을 방지하기 위한 연구들이 제시되고 있다.

이 논문에서는 사용자나 응용 프로그램에 의해 트리거되는 사건이 발생했을 때 실제적으로 규칙들에 대해 상호 적용되는 방식을 의미하는 규칙의 실행의미를 반영한 종료 분석 방법을 제시하였다. 즉, 정확한 종료분석 결과를 위해 능동 규칙의 사건절에 명세 가능한 복합사건의 규칙과 규칙의 실행시점을 나타내는 before, after 규칙이 갖는 특성을 분석 과정에서 고려하여 일반적으로 사용되고 있는 능동 규칙에 대한 종료 분석 범위를 확장하였다.

그리고 트리거 그래프를 기반으로 하는 기존의 종료 분석 방법에서 복합사건의 규칙과 before, after 규칙을 고려하기 위해 트리거 그래프를 변형하였고, 또한 비활성화 관계의 그래프를 결합시킴으로써 규칙의 반복적 사이클 실행 여부를 간결하게 결정할 수 있는 효과를 얻을 수 있었다. 또한 분석 방법의 검증을 위한 적용사례로써 규칙 집합의 예를 이용하여 규칙이 종료 분석되는 과정을 설명하였고, 기존연구와의 비교를 통해 이 논문의 종료 분석 방법에 대한 효율성을 평가하였다.

사이클을 형성하는 규칙이 순차적으로 실행되지 않으면 연속적인 반복 실행이 불가능하다. 따라서 규칙의 연속적 반복 실행의 고리를 제거하는 조건을 찾아냄으로써 규칙의 종료 결정을 좀 더 명확히 할 수 있으며, 이는 규칙의 실행의미를 고려하고 비활성화 관계를 이용하기 때문에 쉽게 식별이 가능하고 정확성을 증가시킨다.

현재 이 논문에서 제안한 분석기는 능동적 데이터 마이닝 기능 지원 데이터 관리 시스템에서 능동규칙의 컴파일을 수행하는 규칙 컴파일러의 일부로서 구현 중에 있다. 향후 연구방향으로는 이 논문에서 고려하지 않은 규칙의 실행의미인 규칙 실행 단위(rule execution granularity)를 포함하는 분석 방법에 대한 연구가 필요하며 아울러 복합사건을 포함하는 규칙 종료 분석 방법에 있어서도 현재 이진 복합 사건만을 적용할 수 있도록 하고 있으나 이를 두 개 이상의 사건이 결합된 다중 복합 사건에 대해서도 적용할 수 있도록 하는 연구가 필요하다.

참 고 문 헌

[1] Jeong-Seok Park, Ye Ho Shin, Kwang Woo Nam, Keun Ho Ryu, "Incremental Condition Evaluation for Active Temporal Rules," Journal of KISS(B), Vol.26, No.4, pp.462-472, April 1999.

[2] 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.

[3] 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, December 95.

[4] 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, September 95.

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

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

[7] ANSI/ISO/IEC International Standard (IS), Database Language SQL-Part 2 : Foundation (SQL/Foundation), ISO/IEC 9075-2 : 1999 (E), September, 1999.

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

[9] S. Ceri, J. Widom. "Application of Active Databases," Active Database Systems-Triggers and Rules for Advanced Databases Processing, 1996.

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

[11] E. Baralis. "Rule Analysis," Chapter 3, Active Rules in Database Systems, Springer-Verlag Pub, 1999.

[12] D. Montesi, M, Bagnato,C. Dallera. "Termination Analysis in Active Database," Database Engineering and Applications, 1999.IDEAS '99 International Symposium Proceedings, pp.288-297 1999.

[13] A. Aiken, J. M. Hellerstein, "Behavior of database production rules : Termination, Confluence, and Observable determinism," In Proceeding of the ACM SIGMOD conf. 59-68, San Diego, California, June, 1992.

[14] 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, March, 1995.

[15] S. Yeung, T. Wang LING "Unrolling Cycle to Decide Trigger Termination," Proc 25th VLDB Conf. pp.483-493, Edinburgh, 1999.

[16] E. Baralis and J. Widom, "An Algebric approach to rule analysis in expert Database Systems," Proc 20th VLDB Con, Santiago, Chile, September, 1994.

신 예 호

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

1996년 군산대학교 컴퓨터과학과 졸업(학사)

1998년 충북대학교 대학원 전자계산학과 졸업(석사)

1998년 충북대학교 대학원 전자계산학과 입학(박사과정)

2000년~현재 충북대학교 대학원 전자계산학과 수료

관심분야 : 능동 데이터베이스, 시간 데이터베이스, 공간 데이터베이스, 데이터 마이닝

황 정 희

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

1991년 충북대학교 전산통계학과 졸업(학사)

1999년~현재 충북대학교 대학원 전자계산학과 재학중(석사)

관심분야 : 능동 데이터베이스, 시공간 데이터베이스, 데이터 마이닝

류 근 호

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

1976년 숭실대학교 전산학과(이학사)

1980년 연세대학교 산업대학원 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, 지식기반 정보검색 시스템, 데이터 마이닝 및 데이터베이스 보안, Bioinformatics