논문 2011-48TC-4-8

스트림 데이터 처리를 위한 비트맵 인덱스 기반 복합 이벤트 검출 기법에 관한 연구

(A Study on The Complex Event Detection Methods Based on Bitmap Index for Stream data Processing)

박 용 민*, 오 영 환**

(Yong-Min Park and Young-Hwan Oh)

요 약

이벤트 기반 서비스 기술은 실시간으로 발생하는 이벤트를 감지하고 분석하여 이에 대한 반응으로 서비스가 연동되는 기술로, 실시간 기업 환경 구축이나 유비쿼터스 서비스 환경 구축을 위한 핵심 기반 기술이다. 실시간 기업 환경에서 요구되고 있는 기업 내 업무 프로세스에서 발생하는 다양한 정보를 실시간 모니터링, 분석하여 변화에 대한 신속한 대응을 제공하거나, 유비쿼터스 서비스 환경에서 상황에 맞게 적시에 맞춤형서비스를 제공하기 위해서는 이벤트 기반의 서비스 기술이 요구된다. 최근 이벤트 중심의 비즈니스 프로세스로 복합 이벤트 처리(CEP : Complex Event Processing) 방식이 사용된다. 복합 이벤트 처리 방식은 여러 이벤트 소스로부터 발생한 이벤트를 대상으로 이벤트들의 영향을 분석하여 대응되는 액션을 처리하는 방식으로 가장 핵심이 되는 기술은 어떻게 사용자에게 의미있는 이벤트(복합 이벤트)를 검출하는가이다. 기존의 연구에서는 복합 이벤트를 구성하는 모든 이벤트가 발생하지 않아도 부분적으로 발생하는 이벤트에 대해 계속적으로 연산을 수행하여 많은 연산과 많은 메모리를 소비하는 문제점이 있다. 이러한 문제를 해결하기 위해 본 논문에서는 대용량의 스트림 데이터에서 발생한 모든 이벤트를 처리하지 않고 응용 계층에서 등록한 복합 이벤트를 구성하는 모든 이벤트가 발생하면, 복합 이벤트를 처리하는 이벤트 검출 기법을 제안한다. 제안 하는 기법은 먼저 비트맵 인덱스를 이용하여 이벤트의 발생 유/무를 관리한다. 또한 복합 이벤트 질이의 마지막 이벤트를 트리거 이벤트로 정의하며, 이 트리거 이벤트가 발생하는 시점을 통해 이벤트의 발생을 표시한 비트맵 인덱스에 복합 이벤트를 구성하는 모든 이벤트의 발생 유/무를 검사하여 모든 이벤트가 발생하였다면, 연산을 수행할 수 있도록 제안한다. 제안하는 기법은 실험을 통해 복합 이벤트를 구성하는 이벤트의 검사를 매번 수행하지 않고 모든 이벤트가 발생하였을 때에만 연산을 수행함으로 불필요한 연산을 방지하고, 처리하는 이벤트의 수를 감소시켜 연산의 효율성을 증가 시켰다.

Abstract

Event-based service technology integrate service to detect events that occur in real time by analyzing the response. Is the core technology for real-time business and ubiquitous service environment construction. Is required event-based service technology on business processes in real-time business environment that providing rapid response to changing and custom service using a variety of information real-time monitoring and analysis in ubiquitous service environment. Last event-driven business processes can be used as a CEP(Complex Event Processing). The core of CEP technology, the event from multiple event sources analysis of events affecting and the way to handle action, is detect complex event to user. In previous studies, an event occurs that continue to perform without the need for partial operations. so many operations and spend a lot of memory is a problem. To solve these problems, event detection technique is proposed that large streams of data without processing any events, registered to configure a complex event occurs when all events in the application layer, complex event processing. The proposed method, first using a bitmap index to manage the event occurs. The complex events of the last event in response to define a trigger event. The occurrence of an event to display a bitmap index, a composite event occurrence of all event to configure the test through the point at which a trigger event occurs. Is proposed, If any event occurs to perform the operation. The proposed scheme perform operations when all event occurs without events having to perform each of the tests. As a result, avoid unnecessary operations and reducing the number of events to handle the increased efficiency of operations.

Keywords : EDA, CEP, Ubiquitous, Bitmap index, Stream data

* 정회원, 삼육보건대학 의료정보시스템과 강의전담교수(Sahmyook Health University)
** 정회원, 광운대학교 전자통신공학과 정교수(Kwangwoon University)
접수일자: 2011년3월3일, 수정완료일: 2011년4월14일

Ⅰ. 서 론

전통적인 데이터 관리 시스템의 경우 컴퓨터 저장 시스템에 저장된 데이터로부터 원하는 정보를 추출하는 것을 기반으로 하고 있지만, 최근 네트워크 및 IT 기술의 발달로 인해 중요한 데이터들은 실시간으로 생성되어 네트워크로 전달되는 경우가 매우 많다. 예를 들어, 인터넷 쇼핑몰의 경우 사용자의 구매 정보 등이 실시간으로 전달되며, 텔레매틱스와 같은 응용에서는 다수의 차량으로부터 다양한 정보가 실시간으로 수집된다. 또한 센서 네트워크 및 RFID 기술의 발전으로 인하여 물류 운송, 상품 구매 등의 정보가 실시간으로 전달되고 있다. 그 외에도 홈 네트워크, U-healthcare, 증권 등 무수히 많은 분야에서 실시간으로 많은 데이터들이 생성되고 전달된다. 이와 같이 대용량의 데이터가 실시간으로 생성되고 전달됨에 따라 현대의 데이터 관리 기술의 패러다임은 실시간으로 수집되는 데이터 중 특정 조건이나 패턴을 만족하는 데이터를 추출하는 것으로 변화하고 있다[1].

유비쿼터스 서비스는 상황에 맞게 적시에 찾아가는 서비스를 제공하는 것으로 다양한 서비스 모델과 이를 실현하기 위한 인터넷과 네트워크 기술들이 연구 개발되고 있다. 이와 같은 상황에 맞는 서비스 혹은 상황을 예측하여 능동적인 서비스를 제공하기 위해서는 개인의 주변에서 발생하고 있는 다양한 정보, 즉 위치 정보, 환경 정보 등과 사용자의 행동 패턴과 같은 이력 정보, 더불어 개인과 직접적인 관계는 없으나 개인에게 영향을 미칠 수 있는 이벤트 등 실세계에서 발생하는 다양한 스트림 형태의 이벤트 정보를 종합적으로 분석하여 서비스를 연동하는 기술이 요구되고 있다[2].

또한 기업이 비즈니스 환경 변화에 유연하고 민첩한 대응을 위해서는 기업 내·외부적인 프로세스의 지연 요소를 제거하고, 비즈니스 과정상에서 발생하는 이벤트와 비즈니스에 영향을 줄 수 있는 외부 이벤트의 조기 감지와 신속한 대응이 가능한 IT 구조를 갖추는 것이 필요하다. 이와 같은 실시간 기업(RTE) 구현을 위해 기업 내부 및 외부에서 발생하는 중요 이벤트에 조직이 신속히 대응하기 위해서는 이벤트 중심으로 비즈니스 프로세스가 구성되어야 한다[3].

최근 이벤트 중심의 비즈니스 프로세스로 복합 이벤트 처리(CEP : Complex Event Processing) 방식이 사용된다. 복합 이벤트 처리 방식은 여러 이벤트 소스로부터 발생한 이벤트를 대상으로 이벤트들의 영향을 분석하여 대응되는 액션을 처리하는 방식으로 가장 핵심이 되는 기술은 어떻게 사용자에게 의미있는 이벤트(복합 이벤트)를 검출하는가이다.

복합 이벤트는 단순 이벤트의 조합으로 이루어진 이벤트로, 복합 이벤트를 구성하는 모든 이벤트가 발생하여야지만 응용 계층으로 결과를 전달할 수 있다. 기존의 연구에서는 복합 이벤트를 구성하는 모든 이벤트가 발생하지 않아도 부분적으로 발생하는 이벤트에 대해 계속적으로 연산을 수행하여 많은 연산과 많은 메모리를 소비하는 문제점이 있다. 이러한 문제를 해결하기 위해 본 논문에서는 복합 이벤트를 구성하는 이벤트가 발생할 때마다 복합 이벤트를 검사하지 않고 실제 복합 이벤트를 구성하는 모든 이벤트가 발생할 때 복합 이벤트를 검사하는 검출 방법을 제안한다. 제안하는 복합 이벤트 검출 방법은 실제 복합 이벤트를 구성하는 이벤트가 발생 하였을 때 비트맵 인덱스로 관리하고, 복합 이벤트는 연산의 특성에 따라 트리를 구성한다. 실제 이벤트가 발생하면 비트맵 인덱스를 통해 복합 이벤트를 구성하는 모든 이벤트의 발생 유/무를 검사하고 복합 이벤트 추출 연산의 수행 여부를 판별한다. 따라서 이벤트의 발생량이 크게 증가해도 실제 연산양은 크게 증가하지 않는다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 관련연구로 이벤트 기반 아키텍처의 복합 이벤트 처리 기술과 기존의 이벤트 검출 기법 및 문제점에 대해 설명하며, Ⅲ장에서는 제안하는 이벤트 검출 기법에 대해 설명한다. Ⅳ장에서는 제안하는 기법에 대한 성능 평가를 기존 기법과 비교하여 설명하고 마지막으로 Ⅴ장에서는 결론에 대해 기술한다.

Ⅱ. 관련 연구

(1) 이벤트 기반 아키텍처 (Event-Driven Architecture)

이벤트 기반 아키텍처는 서비스 기반으로 통합된 정보 시스템 구조에서 다양한 이벤트 상황을 바탕으로 이벤트 교환, 이벤트 트리거, 실시간 대응을 구현하고자 한다. 이벤트 기반 아키텍처를 지원하는 다양한 기술들이 실제 응용들에서 이미 사용되고 있으며 이벤트 처리 방법에 따라 크게 다음과 같이 나눌 수 있다[4~5].

- 단순 이벤트 처리 : 발생한 이벤트들은 모두 의미 있는 이벤트로 간주하고 각각의 이벤트 내용에 따라 액션을 수행한다. Publish/subscribe 방식이나 중재 방식에 의해 이벤트 처리를 제공한다.
- 스트림 이벤트 처리 : 의미있는 이벤트와 무의미한 이벤트가 같이 발생하는 대량의 이벤트 스트림을 대상으로 하며, 필터링 등을 수행하여 의미있는 이벤트 정보만 뽑아서 응용에 전달 혹은 서비스와 연동한다. RFID, USN 미들웨어 등이 이에 해당한다.
- 복합 이벤트 처리 : 여러 이벤트 소스로부터 발생한 이벤트를 대상으로 이벤트들이 영향을 분석하여 대응되는 액션을 수행한다. 단순 이벤트 처리가 하나의 이벤트를 대상으로 한 반면, 복합 이벤트 처리는 여러 이벤트간의 다양한 관계를 분석한다. 복합 이벤트 처리는 BAM (Business Activity Monitoring) 패키지에 내장되고 제공되거나 별도의 복합 이벤트 처리 시스템으로 제공되기도 한다.

이 중 복합 이벤트 처리는 기업에서 발생하는 복잡한 이벤트들을 탐지하고 관리하기 위해 필요한 기술이다. 복합 이벤트 처리의 목적 중 하나는 복잡한 비즈니스 환경에서 수 없이 발생하는 이벤트들에 대한 이해를 돕는 것이다. 복합 이벤트 처리를 통해서 특정 이벤트가 무엇에 의해서 발생했는지 알 수 있고, 그것을 바탕으로 대응하는 룰을 만들어 실행에 옮길 수 있다. 결국, 복합 이벤트 처리는 어떠한 액션을 취하는 데에 기본적인 바탕을 제공해 준다[2].

일반적인 복합 이벤트 처리에서 이벤트는 단순 이벤트(simple event)와 복합 이벤트(complex event)로 구성된다. 단순 이벤트는 하나의 이벤트로 표현되며, 복합 이벤트는 단순 이벤트의 조합으로 표현된다. 예를 들어, RFID의 단순 이벤트는 (ID, L, T)로 정의 할 수 있다. 여기서, ID는 RFID 태그 ID를 나타내며, L은 위치, T는 발생한 이벤트의 시간을 나타낸다.

복합 이벤트는 단순 이벤트의 연관성을 기초로 만들어지며, 의미있는 데이터로 변환하여 응용 계층으로 전송한다. 복합 이벤트 언어는 DBMS에서 사용하는 SQL과 비슷하며 그림 1과 같이 나타낸다[6].

SELECT는 입력된 단순 이벤트를 고려한 조합된 복합 이벤트를 나타내며 복합 이벤트 처리의 최종 결과로 상위 응용 계층으로 전송된다. FROM은 입력된 단순 이벤트 스트림을 나타내며, WHERE은 의미있는 복합 이벤트를 나타내기 위한 조건이며, WITHIN은 이벤트 발생 시간을 나타낸다.

Complex Evnet 구문
SELECT [result] *FROM [input event streams]* *WHERE [condition]* *WITHIN [time]*

연산자(기호)	의미
AND(∧)	E1∧E2 : E1, E2 이벤트가 동시에 발생
OR(∨)	E1∨E2 : E1 또는 E2 이벤트가 발생
NOT(!)	!E1 : E1 이벤트가 발생하지 않음
SEQ(→)	E1→E2 : E1이벤트가 발생하고 그 후 E2 이벤트가 발생

그림 1. 복합 이벤트 구문 및 연산자
Fig. 1. Complex event syntax and operators.

(2) 기존 복합 이벤트 검출 기법

[7]에서는 연속적으로 발생하는 대용량의 RFID 스트림 데이터에서 이벤트를 처리하기 위한 SASE를 제안하였다. SASE는 단순 이벤트의 패턴과 조건 그리고 시간 간격으로 구성을 통해 복합 이벤트를 정의하기 위한 언어를 제안하였다. SASE는 5가지의 처리 과정을 수행하여 복합 이벤트의 적합성을 검사한다. 먼저 등록된 복합 이벤트는 오토마타 형태로 구성하고 이벤트가 발생함에 따라 이벤트의 상태 변화를 스택에 저장한다. 이때, 변화가 발생하기 전의 상태는 포인터로 연결된다. 이벤트 발생에 따라 상태의 변화가 오토마타의 최종 단계에 도착하게 되면 포인터를 따라 발생한 이벤트의 조합을 구성한다. 이 처리 과정을 SSC(Sequence Scan and Con- struction)이라 한다. SSC를 통하여 구성된 이벤트 조합은 실제 복합 이벤트의 후보 집합으로 SELECTION 연산을 수행하여 복합 이벤트의 조건을 만족하는지 검사한다. 또한, WINDOW 연산을 통해 시간적인 조건을 만족하는지 검사한다. 마지막으로 NG(negation) 연산을 통하여 발생하지 않는 데이터를 처리하게 된다. 모든 조건을 만족하는 이벤트 조합은 복합 이벤트를 등록한 응용 계층에게 전달한다.

[8]에서는 그래프를 이용하여 RFID 스트림 데이터를 위한 RCEDA의 복합 이벤트를 처리기법을 제안하였다. RCEDA에서는 복합 이벤트를 효율적으로 처리하기 위하여 복합 이벤트를 정의하는 규칙을 제안하였고 복합 이벤트를 구성하기 위해 단순 이벤트를 조합하기 위한 연산자를 정의하였다. 이러한 연산자는 AND, OR,

NOT과 같은 기본 연산자와 순차적인 이벤트를 묶는 SEQ, 시간 조건이 포함된 TSEQ 연산, 동일한 이벤트의 연속적인 처리를 위한 SEQ+, TSEQ+ 연산 등이 있다. 실제 복합 이벤트를 처리하기 위해 어플리케이션에서 등록한 복합 이벤트는 계층적인 그래프로 저장한다. 그래프의 단말노드는 단순 이벤트로 구성되어 있으며 비 단말노드는 이벤트의 연산자로 구성되어 있다. 단순 이벤트가 발생하면 그래프의 단말 노드에서부터 이벤트의 조건을 검사하여 상위 노드로 전달된다. 이때, 발생하지 않는 이벤트의 연산자가 포함된 노드를 만난 경우는 상위노드에서 Pseudo 이벤트를 발생 시킨다. 발생하지 않는 연산은 하위 노드에서 상위노드로 이벤트를 전달하는 시점을 선정 할 수 없기 때문에 발생된 Pseudo 이벤트는 정해진 시간에 하위 노드로 질의를 전달하고 조건을 만족하는 경우 다시 상위 노드로 전달된다. 이벤트가 최종 루트노드에서 조건을 만족하게 되면 복합 이벤트를 등록한 응용 계층에게 최종 이벤트가 전달된다.

(3) 기존 복합 이벤트 검출 기법의 문제점

복합 이벤트를 효과적으로 처리하기 위하여 기존에 많은 연구가 수행되어 왔다. 하지만 대용량 데이터 스트림에서 이벤트를 처리하는데 연산이 크게 증가하고 이로 인해 주어진 시간 내에 이벤트를 검출하지 못하여 시간 지연이 발생하는 문제점이 있다.

SASE는 복합 이벤트의 패턴이 SEQUENCE 연산으로 한정되어 수행된다. SASE에서는 복합 이벤트를 오토마타의 형태로 정의한다. 그리고 단순 이벤트가 발생함에 따라 복합 이벤트의 상태를 단계별로 나누어 변화시킨다. 이벤트의 상태는 발생하는 단순 이벤트의 순서에 따라 결정됨으로 순서를 고려하는 SEQUENCE연산만이 오토마타로 구성될 수 있다. SEQUENCE 이외의 연산은 오토마타의 형태로 구성과 복합 이벤트의 상태를 단계별로 정의하는 명확한 기준을 제공하지 못한다. 따라서 SASE는 다양한 형태의 복합 이벤트를 처리할 수 없는 문제점이 있다.

SASE는 등록된 여러 복합 이벤트들 사이에 정보를 공유 할 수 없는 단점을 지니게 된다. SASE에서는 각각의 복합 이벤트는 인스턴스 스택에 단순 이벤트를 저장하고 발생된 시간에 따라 포인터를 연결하는 구조를 가지고 있다. 따라서 같은 단순 이벤트로 구성되어 있는 복합 이벤트도 각각의 단순 이벤트를 저장하고 있어야 한다. 이는 데이터의 중복 저장을 의미한다. 그림 2는 이러한 문제점을 나타낸 것이다.

그림 3은 RCEDA 방식의 문제점을 나타낸 것이다. E_1, E_2, E_3, E_4 의 이벤트로 구성된 복합 이벤트가 있고 단순 이벤트는 그림 3과 같이 E_1, E_3, E_4 는 빈번히 발생하지만 E_2 이벤트는 발생하지 않는다. 이러한 경우에서 중간단계의 후보 집합인 E_3, E_4 의 조합 결과는 계속 생성되지만 최종 복합 이벤트는 발생하지 않는다.

RCEDA에서 단순 이벤트가 발생하면 하위 노드에서 상위 노드로 전달된다. 이벤트를 전달 받은 상위노드는 하위 노드에서 발생한 이벤트를 저장하고 하위 노드들로부터 전달받은 이벤트를 이용하여 중간 단계의 후보 집합을 생성하여 상위 노드로 전달해야 한다. 따라서 상위 노드는 모든 중간 단계의 후보 집합 이벤트를 저장하기 위한 추가적인 구조가 필요하다. 실제 복합 이벤트의 발생에 영향을 주지 않는 불필요한 중간 단계 후보 집합도 상위 노드로 전달되어 저장하기 때문에 불

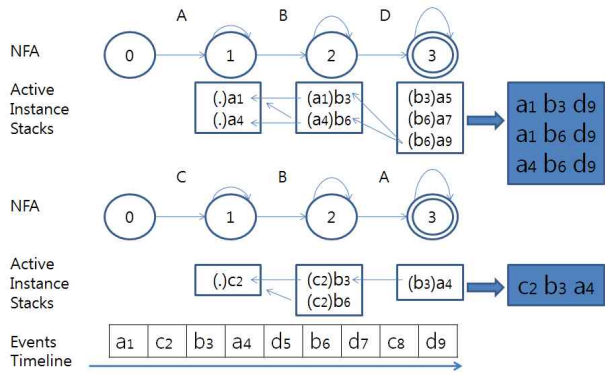


그림 2. 동일한 이벤트로 생성된 복합 이벤트
Fig. 2. Complex event generated by the same event.

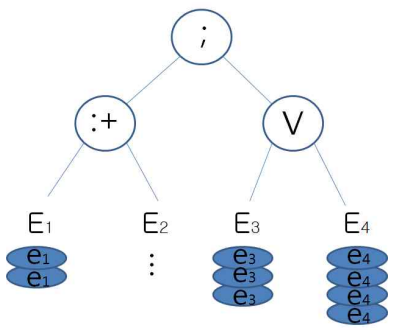


그림 3. RCEDA에서 이벤트 처리과정의 문제점
Fig. 3. Process in the event of a problem RCEDA.

필요한 저장비용을 소모한다.

RCEDA는 여러 복합 이벤트들의 그래프 공유에 대한 문제점이 발생한다. 복합 이벤트는 서로 다른 복합 이벤트와 그래프의 노드를 공유할 수 있다. 하지만 연산을 구성하는 연산자와 하위노드 연산시간이 모두 일치하지 않으면 공유할 수 없다. 따라서 그래프의 단말 노드들은 공유가 용이하지만 비 단말 노드는 쉽게 공유되지 않는다.

Ⅲ. 제안하는 복합 이벤트 검출 기법

(1) 복합 이벤트 처리를 위한 제안 모델

2장에서 언급한 문제점을 해결하기 위해 제안하는 복합 이벤트 검출 기법은 복합 이벤트를 구성하는 단순 이벤트들이 모두 발생할 때 복합 이벤트를 검출하도록 검출 시점을 조절한다. 제안하는 복합 이벤트 처리를 위한 모델은 그림 4와 같이 복합 이벤트를 구성하는 모든 단순 이벤트들이 발생되었는지를 판별하기 위해 비트맵 인덱스를 구성하며, 복합 이벤트를 위한 질의는 복합 이벤트를 구성하는 단순 이벤트 중 마지막 이벤트를 트리거 이벤트로 정의하고, 트리거 이벤트를 기준으로 복합 이벤트 질의를 등록한다. 또한 복합 이벤트 질의는 연산자의 특성에 의해 트리형태로 구성된다. 실제 단순 이벤트가 발생하면 비트맵 인덱스를 통해 발생 시간을 기록하며, 만약 복합 이벤트를 구성하는 모든 단순 이벤트가 발생하면 실제 복합 이벤트를 검출하기 위해 복합 이벤트 트리를 검사한다.

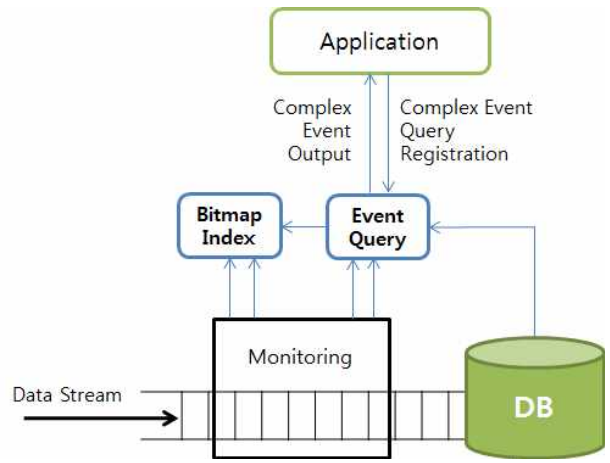


그림 4. 복합 이벤트 처리를 위한 제안 모델
Fig. 4. Proposed model for complex event processing.

(2) 비트맵 인덱스(Bitmap Index)

본 논문에서는 비트맵 인덱스를 이용하여 단순 이벤트의 발생 이력을 저장한다. 그림 5와 같이 비트맵 인덱스는 논리적인 그리드 영역에 대응하는 비트 열을 가진 긴 복합 이벤트 연산 시간동안 유지하는 2차원 배열 형태를 가진다.

단순 이벤트가 발생하게 되면 단순 이벤트는 발생한 시간대에 일치하는 비트열에 해당 그리드 영역에 대응하는 비트를 갱신한다. 비트열은 단순히 어느 시간대에 어떤 이벤트가 발생했는지의 정보만을 유지하고 실제 발생한 단순 이벤트는 데이터베이스에 저장하게 된다. 즉, 비트맵 인덱스는 단순 이벤트의 발생 유/무와 데이터베이스에 저장된 이벤트를 검색하기 위한 인덱스로 사용된다. 그림 5는 비트맵 인덱스 구조를 나타낸다. 예를 들어 T_t 시간에 e 이벤트가 발생하였다면 비트열을 '1'의 값으로 기록된다.

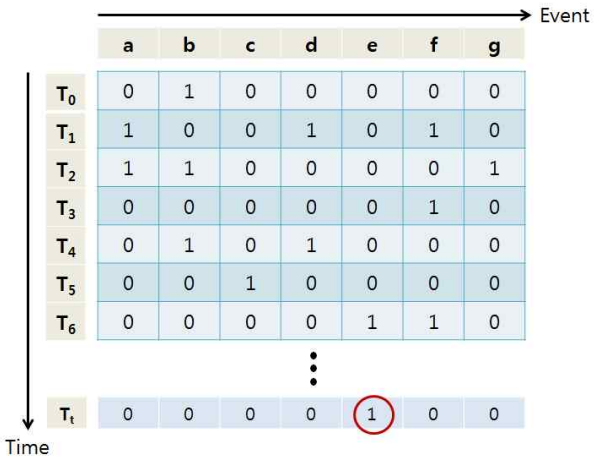


그림 5. 비트맵 인덱스 구조
Fig. 5. Structure of bitmap index.

(3) 이벤트 질의(Event Query)

이벤트 질의는 응용 계층에서 등록한 복합 이벤트를 효율적으로 처리하기 위하여 복합 이벤트 정보를 저장하는 구조이다. 이벤트 질의는 응용 계층에서 등록된 복합 이벤트를 연산자의 특성에 의해 트리 형태로 구성한다. 또한 등록된 복합 이벤트를 구성하는 모든 이벤트 중 마지막 이벤트를 트리거 이벤트로 정의하며, 이 트리거 이벤트를 기준으로 복합 이벤트 질의를 등록하게 된다. 트리거 이벤트는 정의 1과 같다.

[정의 1] 트리거 이벤트(Trigger Event)

트리거 이벤트는 복합 이벤트를 추출할 수 있는 기준 시점을 제공하는 이벤트로 연산자에 따라 정의 되며 SEQUENCE 연산의 경우 마지막에 위치한 이벤트를 트리거 이벤트로 정의하며, 다른 연산자에 의한 연결된 이벤트들은 마지막 항의 연결된 모든 이벤트를 트리거 이벤트로 정의한다.

예를 들어, 그림 6과 같이 복합 이벤트를 위한 질의 Q1, Q2, Q3가 응용 계층에서 등록되었다면, 이벤트 질의는 트리형태로 구성되며, 트리거 이벤트를 기준으로 복합 이벤트 질의를 등록하게 된다. 여기서 트리거 이벤트는 d, c, h가 된다. 또한 그림 6에서와 같이 하나의 트리거 이벤트(d)는 다른 복합 이벤트들 사이에서 공유될 수 있다.

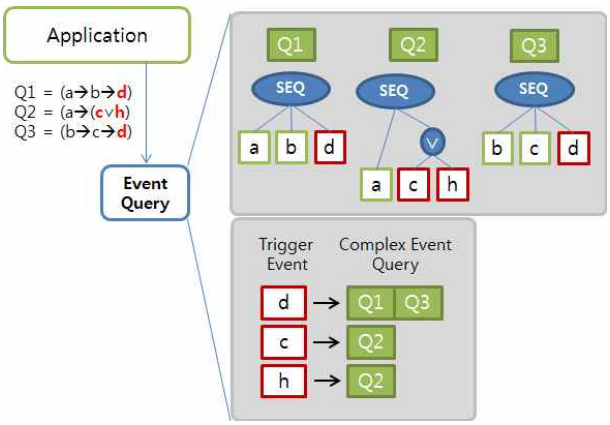


그림 6. 이벤트 질의
Fig. 6. Event query.

(4) 제안하는 복합 이벤트 검출 과정

그림 7은 복합 이벤트 검출을 위한 처리 과정을 나타낸다. 먼저 복합 이벤트 질의는 트리 형태와 트리거 이벤트를 기준으로 복합 이벤트 질의를 등록한다. 수신된 스트림 데이터에서 트리거 이벤트인 d의 단순 이벤트가 발생하였다면, 비트맵 인덱스에 발생한 시간을 기록하며, 이벤트 질의를 통해 자신을 포함한 복합 이벤트 질의인 Q1과 Q3에 접근한다. Q1과 Q3 각각의 질의는 d 이벤트를 제외한 다른 이벤트의 발생 유/무를 식별하기 위해 비트맵 인덱스를 통해 검색한다. 비트맵 인덱스의 검색은 복합 이벤트 질의가 최초 등록된 시점에서 트리거 이벤트가 발생한 시점까지를 시간 범위로 하여 그 범위 내에서 각 질의를 구성하는 이벤트의 발생 유/무를 검색한다. 만약 모든 이벤트가 발생하였다면 각각의 질의는 연산을 수행하며 그렇지 못하다면 연산을 수행하지 않는다.

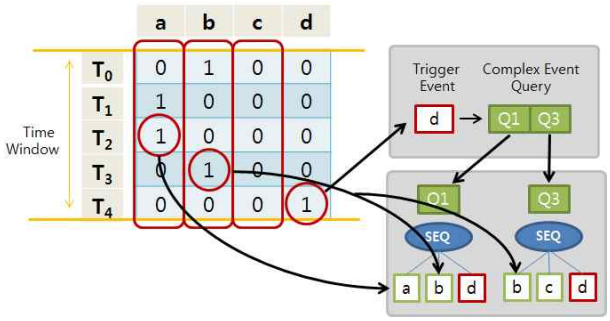


그림 7. 복합 이벤트 검출을 위한 처리 과정
Fig. 7. For the detection of complex event processing.

그림 7에서 Q1과 Q3는 SEQUENCE 연산자로 발생 순서가 중요하다. 따라서 Q1에서 d 이벤트가 T_4에서 발생하였다면, 그 다음 순서인 b 이벤트는 T_0~T_4 시간 범위에서 발생 유/무를 검색한다. b 이벤트는 T_3 시점에서 발생하였으므로, a 이벤트를 b 이벤트의 발생 시점을 기준으로 T_0~T_3 시간범위에서 다시 발생 유/무를 검색한다. 그 범위의 T_2에서 발생하였기 때문에 Q1 질의는 모든 이벤트가 발생하였으므로 연산을 수행한다. Q3의 경우 마찬가지로 d 이벤트가 T_4에서 발생하였다면 c 이벤트는 T_0~T_4 시간범위에서 발생 유/무를 검색한다. c 이벤트는 그 시간범위에서 발생하지 않았으므로 연산을 수행하지 않는다.

Ⅳ. 성능 평가 및 결과

본 장에서는 Ⅲ장에서 제안한 복합 이벤트 검출 기법을 토대로 기존 복합 이벤트 검출 기법과의 성능 평가를 수행하여 우수성을 입증한다. 성능 평가에 사용된 시스템은 Intel Dual Core 2.4GHz 프로세스와 2Gbyte의 메모리를 가지고 있으며 운영체제는 Fedora Linux 9를 사용한다. 복합 이벤트 처리를 위한 모듈은 NS2 시뮬레이터를 이용하여 구현하였다. SASE 기법은 SEQUENCE 연산 이외에 다양한 연산을 포함하는 복합 이벤트를 처리하지 못하는 단점이 있다. 따라서 제안하는 복합 이벤트 처리 기법의 성능 평가는 RCEDA 기법과의 비교 평가를 수행한다. 성능 평가는 복합 이벤트의 수와 초당 발생하는 단순 이벤트의 수에 따라 기존 연구에서 복합 이벤트 검출 연산 수와 제안하는 기법에서의 검출 연산 수를 비교한다. 또한 복합 이벤트 수와 초당 발생하는 단순 이벤트 수를 변경하여 기존 제안 기법과의 평균 처리 시간을 비교 분석한다.

(1) 복합 이벤트 검출 연산

본 절에서는 초당 10,000개의 단순 이벤트가 발생하는 상황에서 복합 이벤트 수를 50~500개로 변경시켜 등록하고 복합 이벤트 검출 연산 수를 측정하였다. 또한 복합 이벤트 수를 500개로 고정한 후 단순 이벤트를 10,000~50,000개로 변경 시키며 실험을 반복 수행한다. 그림 8은 복합 이벤트 수의 변화에 따라 실험 결과를 나타낸 그림이다. 초당 10,000 개의 단순 이벤트가 발생할 때 제안하는 기법은 기존연구에 비하여 연산 수가 감소한 것을 볼 수 있다. 즉, 복합 이벤트를 구성하는 단순 이벤트의 수가 많고 연산이 복잡하면 본 논문에서 제안하는 기법이 처리하는 복합 이벤트 검출 연산의 수가 현저히 감소함을 볼 수 있다.

그림 9는 초당 발생하는 단순 이벤트 수를 변화 시켜가며 제안하는 기법과 기존 기법의 결과 그림이다. 초당 발생하는 단순 이벤트 수가 증가함에 따라 기존 기법과 제안하는 기법 모두 초당 발생하는 이벤트에 따라 증가하는 모습을 볼 수 있다. 하지만 제안하는 기법이 기존 기법보다 증가량이 적음을 알 수 있다. 이것은 기존 기법은 증가된 단순 이벤트에 따라 대응하는 복합 이벤트의 수가 증가하기 때문이며, 제안하는 기법은 단순 이벤트 수가 증가하여도 복합 이벤트의 발생 조건을 만족하는 복합 이벤트 수는 크게 증가하지 않는다.

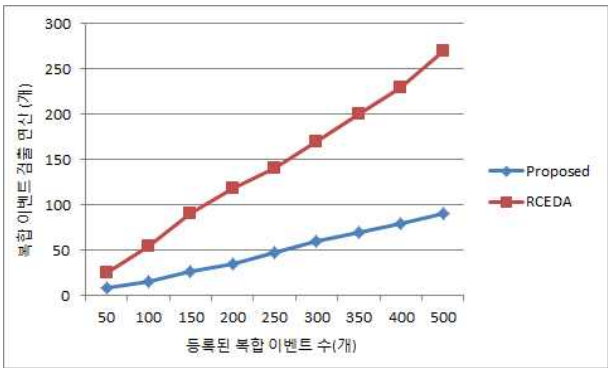


그림 8. 복합 이벤트 수에 따른 연산 수
Fig. 8. The operand of the number of complex events.

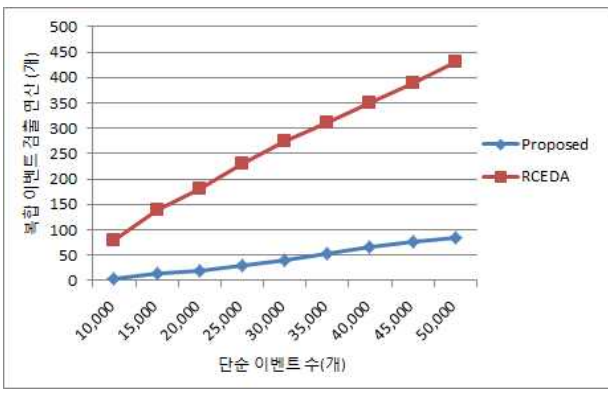


그림 9. 단순 이벤트 수에 따른 연산 수
Fig. 9. The operand of the number of simple events.

(2) 평균 처리 시간

이 절에서는 등록된 복합 이벤트 수에 따라 초당 이벤트를 처리하는 평균 시간을 측정한다. 그림 10은 등록된 복합 이벤트 수에 따른 평균 처리 시간을 나타낸다. 기존 기법은 등록된 복합 이벤트 수에 따라 처리 시간이 비교적 크게 증가하는 것을 볼 수 있다. 반면 제안하는 기법은 등록된 복합 이벤트 수에 따라 평균 처리 시간이 크게 증가하지 않는다.

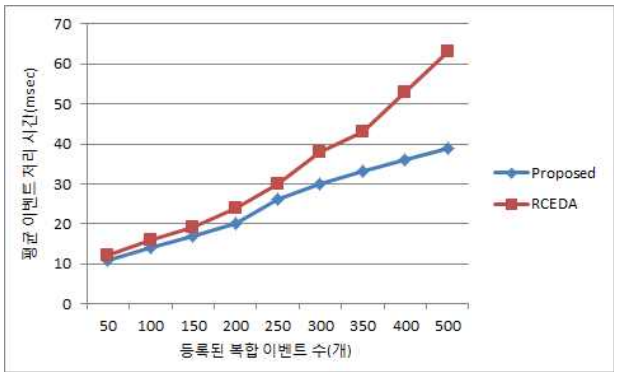


그림 10. 복합 이벤트 수에 따른 평균 처리 시간
Fig. 10. The average time of complex event processing number.

V. 결 론

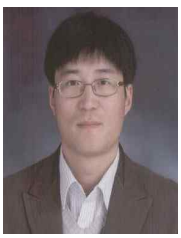
본 논문에서는 복합 이벤트 검출을 효율적으로 처리하기 위해 복합 이벤트를 구성하는 모든 단순 이벤트가 발생하는 시점에서 복합 이벤트를 검출하는 방법을 제안하였다. 제안하는 복합 이벤트 검출 기법은 복합 이벤트를 구성하는 단순 이벤트의 발생 조건을 빠르게 판별하기 위해 비트맵 인덱스를 구성하고, 복합 이벤트를 구성하는 단순 이벤트를 연산자에 따라 트리 형태로 구성된다. 비트맵 인덱스를 통해 복합 이벤트를 구성하는 모든 이벤트가 발생하면, 트리로 구성된 복합 이벤트 질의를 수행하게 된다. 제안하는 복합 이벤트 검출기법은 모든 이벤트가 발생하였을 때 복합 이벤트 검출 연산을 수행하기 때문에 연산의 수와 처리시간을 감소시킬 수 있다. 실험을 통해 제안하는 기법이 기존의 기법

에 비해 이벤트 처리 수와 평균 처리 시간에서 우수한 성능을 나타낸 것을 확인하였다.

참 고 문 헌

[1] 김종익, “동적 데이터 흐름 처리를 위한 이벤트 스트림 관리 기술 개발”, 교육과학기술부 기초연구사업, 2009.
[2] 이미영, 김명준, “이벤트 기반 서비스 기술 동향”, 전자통신동향분석 제21권 제5호, 2006.
[3] 백현진, “RTE 구현을 위한 전략”, SDS Consulting Review, 2004.
[4] Roy W. Schulte, “Event-Driven Applications : De- finition and Taxonomy”, Gartner, July 2003.
[5] Brenda M. Michelson, “Event-Driven Architecture Overview”, Patricia Seybold Group, Feb. 2006.
[6] M. Mohania, D. Swamini, S. K. Gupta, S. Bhowmick, and T. Dillon, “Event composition and Detection in Data Stream Management Systems”, Proc. International Conference on Database and Expert Systems Applications, 2005.
[7] D. Gyllstrom, E. Wu, H. J. Chae, Y. Diao, P. Stahlberg, and G. Anderson SASE: Complex Event Processing over Streams. In Proceedings of the Third Biennial Conference on Innovative Data Systems Research (CIDR 2007)
[8] F. Wang, S. Liu, and Y. Bai, “Bridging Physical and Virtual Worlds: Complex Event Processing for RFID Data stream”, Proc International Conference on Extending Database Technology, 2006.

저 자 소 개



박 용 민(정회원)
2005년 광운대학교 전자통신 공학과 공학석사 졸업
2007년 광운대학교 전자통신 공학과 박사수료
2011년~현재 삼육보건대학 의료정보시스템 강의전담교수
<주관심분야 : 의료정보시스템, USN, Complex Event Processing, Stream Mining>

오 영 환(정회원)
대한전자공학회 논문지 제37권 TC편 제2호 참조