韓國컴퓨터情報學會 論文誌
第15卷 第9號, 2010. 9.

2010-15-9-1-5

순차 데이터 스트림에서 발생 간격 제한 조건을 활용한 빈발 순차 패턴 탐색

장 중 혁*

Mining Frequent Sequential Patterns over Sequence Data Streams with a Gap-Constraint

Joong Hyuk Chang*

요 약

순차 패턴 탐색은 데이터 마이닝의 주요 기법 중의 하나로서 웹기반 시스템, 전자상거래, 생물정보학 및 USN 환경 등과 같은 여러 컴퓨터 응용 분야에서 생성되는 데이터를 효율적으로 분석하기 위하여 널리 활용되고 있다. 한편 이들 응용 분야에서 생성되는 정보들은 근래들어 한정적인 데이터 집합이 아닌 구성요소가 지속적으로 생성되는 데이터 스트림 형태로 생성되고 있다. 이러한 상황을 고려하여 데이터 스트림에서 순차패턴 탐색에 대한 연구들도 활발히 진행되고 있다. 하지만 이전의 연구들은 주로 분석 대상 데이터 스트림에서 단순 순차패턴을 구하는 과정에서 마이닝 수행 시간이나 메모리 사용량 등을 줄이는데 초점을 맞추고 있으며, 따라서 해당 데이터 스트림의 특성을 효율적으로 표현할 수 있는 보다 중요하고 의미있는 패턴들을 탐색하기 위한 연구는 거의 진행되지 못하고 있다. 본 논문에서는 데이터 스트림에서 보다 의미있는 순차패턴을 탐색하기 위한 방법으로 구성요소의 발생 간격 제한 조건을 활용한 빈발 순차패턴 탐색 방법을 제안한다. 먼저 발생 간격 정의 기준 및 발생 간격 제한 빈발 순차패턴의 개념을 제시하고, 이어서 데이터 스트림에서 발생 간격 제한 조건을 적용하여 빈발 순차패턴을 효율적으로 탐색할 수 있는 마이닝 방법을 제안한다.

Abstract

Sequential pattern mining is one of the essential data mining tasks, and it is widely used to analyze data generated in various application fields such as web-based applications, E-commerce, bioinformatics, and USN environments. Recently data generated in the application fields has been taking the form of continuous data streams rather than finite stored data sets. Considering the changes in the form of data, many researches have been actively performed to efficiently find sequential patterns over data streams. However, conventional researches focus on reducing processing time and memory usage in mining sequential patterns over a target data stream, so that a research on mining more interesting and useful sequential patterns that efficiently reflect the characteristics of the data stream has been attracting no attention. This paper proposes a mining method of sequential patterns over data streams with a gap constraint, which can help to find more interesting sequential patterns over the data streams. First, meanings of the gap for a sequential pattern and gap-constrained sequential patterns are defined, and subsequently a mining method for finding gap-constrained sequential patterns over a data stream is proposed.

• 제1저자 : 장중혁
• 투고일 : 2010. 05. 31, 심사일 : 2010. 06. 21, 게재확정일 : 2010. 07. 22.
* 대구대학교 컴퓨터IT공학부 교수
※ 이 논문은 2009년도 대구대학교 교내학술연구비 지원에 의한 논문임.

▸ Keyword : 순차패턴(Sequential patterns), 발생 간격 제한 조건(Gap-constraint), 순차 데이터 스트림 (Sequence data streams), 순차패턴 마이닝(Sequential pattern mining)

Ⅰ. 서 론

근래 들어 여러 컴퓨터 응용 분야에서는 한정적인 데이터 집합이 아니라 구성 요소가 지속적으로 발생되는 데이터 스트림 형태로 정보를 생성하고 있다. 이로 인해 데이터 처리 및 분석과 연관된 연구들도 데이터 스트림을 주요 처리 대상으로 삼고 있다 [1]. 이러한 데이터 생성 형태의 변화를 고려하여 데이터 마이닝 분야에서도 데이터 스트림에 내재된 다양한 지식들을 탐색하기 위한 연구들이 활발히 진행되고 있다. 이들 연구들은 주로 빈발 패턴 탐색[2,3,4]을 포함하는 연관규칙 탐색 및 순차패턴 탐색[5,6] 분야에서 활발히 진행되고 있다. 연관규칙 탐색 및 빈발 순차패턴 탐색 분야는 한정적인 데이터 집합을 대상으로 하는 이전의 데이터 마이닝 분야에서도 여러 응용 분야에서 유용한 것으로 이미 검증된 바 있기 때문이다. 이와 더불어 시간 흐름에 따른 가변성이 큰 데이터 스트림의 특성을 고려한 분석 방법들도 활발히 연구되어 왔다[2,7]. 하지만 데이터 스트림에 대한 대부분의 이전 연구들은 빈발 패턴이나 순차패턴 등과 같은 단순 정보를 분석하는데 머무르고 있으며, 분석 대상이 되는 데이터 스트림의 특성을 보다 잘 표현할 수 있는 새로운 관심정보를 찾는데 어려움이 있다.

순차패턴 마이닝은 순차 데이터베이스로부터 흥미로운 순차패턴을 탐색하는데 목적을 두고 있으며, 이는 주요한 데이터 마이닝 분야 중의 하나로서 웹기반 시스템, 전자상거래, 생물정보학 및 USN 환경 등 여러 컴퓨터 응용 분야에서 널리 활용되고 있다. 순차패턴 탐색은 분석 대상 데이터 집합 및 출현빈도 수 임계값이 주어졌을 때 해당 임계값 이상의 출현빈도 수를 갖는 모든 순차패턴을 찾는 작업이다. 일반적으로 마이닝 수행 결과로 얻어지는 순차패턴의 수가 매우 많으며, 이를 바로 응용 분야의 특성을 이해하기 위해서 활용하는데 어려움이 있다. 따라서 중요도나 관심도가 큰 순차패턴을 얻기 위해서는 일반적인 순차패턴 탐색으로 얻어진 결과를 다시 분석해야 한다.

한편, 하나의 순차패턴에 있어서 이를 구성하는 단위항목들의 단순 발생 순서뿐만 아니라 단위항목들 간의 발생 간격 등도 중요한 고려사항이 될 수 있다. 예를 들어 식료품점의 수집된 다음의 구매 기록을 고려해보자. 각 구매자는 매일 한 번씩 판매점에서 물건을 구입한 것으로 가정한다.

구매자_1 :
빵 → 음료수 → 커피 → **우유** → 맥주 → 녹차 → **땅콩**

구매자_2 :
빵 → **우유** → **땅콩**

두 구매자의 구매 물품 리스트에서 '빵→우유→ 땅콩'을 순차적으로 구매한 사실을 확인할 수 있다. 하지만 구매자_1의 경우는 빵을 구매한 후 우유 및 땅콩을 구매하기까지 6일이 소요된 반면, 구매자_2의 경우는 빵을 구매한 후 나머지 제품들을 구매하기까지 2일의 시간이 소요되었다. 이때 해당 식료품점의 구매 물품 리스트로부터 관심 정보를 얻기 위한 분석 과정에서 3일 이내에 이뤄진 일련의 판매 내역을 의미있는 정보로 간주하는 경우, 구매자_1의 구매 정보에 포함된 '빵→우유→땅콩' 판매실적은 무의미한 것으로 간주되는 반면 구매자_2의 구매 정보에 포함된 동일한 내용은 의미있는 정보로 분석될 수 있다. 즉, 순차패턴 탐색 과정에서 발생 간격이나 발생 시간 등을 고려하는 경우 보다 관심도가 큰 순차패턴을 마이닝 결과로 얻을 수 있다.

이러한 상황을 고려하여 순차패턴 탐색 과정에서 제한조건 등을 추가하여 보다 정제된 형태의 마이닝 결과를 얻기 위한 방법들이 활발히 연구되어 왔다. 순차패턴 탐색 과정에서 일정한 제한 조건을 제시하여 관심도가 큰 순차패턴을 마이닝 결과로 얻기 위한 방법들도 연구되었으며[8,9,10], closed 순차패턴이나 maximal 순차패턴을 탐색하기 위한 연구들도 활발히 진행되어 왔다[11,12,13,14]. 하지만 이전의 연구들은 대부분 한정적인 데이터 집합을 대상으로 하는 마이닝 방법들로서, 구성요소가 매우 빠르게 지속적으로 발생되는 데이터 스트림 환경에서 활용도나 관심도가 큰 순차패턴들을 탐색하기 위한 연구들은 거의 진행되지 못하고 있다. 앞서 기술한 바와 같이 근래 대부분의 컴퓨팅 환경에서는 데이터 스트림 형태로 정보를 발생시키고 있으며, 이를 고려할 때 데이터 스트림에 대한 마이닝 과정에서 제한 조건 등을 활용하여 관심도나 흥미도가 높은 순차패턴을 얻을 수 있는 마이닝 방법은 여러 응용 분야에서 널리 활용될 수 있을 것이다.

본 논문에서는 데이터 스트림 처리의 기본적인 요구 조건들[15]을 만족하면서 발생 간격에 기반한 제한 조건 설정을 통해 관심도나 흥미도가 큰 순차패턴을 얻을 수 있는 데이터 스트림에 대한 순차패턴 탐색 방법을 제안한다. 즉, 관심도나 흥미도가 큰 순차패턴을 탐색하기 위한 방법으로 순차패턴을 구성하는 항목들간의 발생 간격을 일정 수준으로 제한하는 방법을 제시하고자 한다. 한정적인 데이터 집합을 분석 대상으로 이전의 기본적인 접근 방법들과 달리 데이터 스트림에 대한 순

차패턴 탐색 과정에서 효율적으로 적용할 수 있는 발생 간격 제한 빈발 순차패턴 탐색 방법으로서, 이를 활용하여 메모리 사용량 및 수행 시간 최소화 등과 같은 데이터 스트림 처리를 위한 기본적인 요구 조건을 만족하면서 데이터 스트림에서 관심도나 흥미도가 큰 순차패턴을 효율적으로 얻을 수 있다.

본 논문의 구성은 다음과 같다. 먼저 2장에서는 논문에서 다루고자 하는 문제를 정의하고 이와 연관된 이전의 연구들을 간략히 정리하며, 3장에서는 데이터 스트림에서 순차패턴 탐색을 위한 기본적인 접근 방법을 기술한다. 4장에서는 본 논문에서 제안하는 핵심 내용인 발생 간격 제한 빈발 순차패턴의 기본 개념을 기술하고, 데이터 스트림에서 이를 효율적으로 탐색할 수 있는 마이닝 방법을 제시한다. 5장에서는 일련의 실험을 통해 제안된 방법의 효율성을 검증하고, 6장에서 논문의 결론을 맺는다.

II. 문제 정의 및 관련 연구

본 절에서는 먼저 논문에서 제안하는 순차패턴 탐색 방법이 분석 대상으로 하는 데이터 스트림을 명확히 정의하고 이를 바탕으로 논문에서 다루고자 하는 문제인 데이터 스트림 환경에서 발생 간격 제한 조건을 활용한 빈발 순차패턴 탐색을 명확히 정의한다. 더불어 해당 문제와 연관된 이전의 연구들을 간략히 정리한다.

1. 문제 정의

순차패턴 탐색을 위한 분석 대상이 되는 데이터 스트림은 시간적으로 정렬된 단위항목들의 집합인 순차정보가 지속적으로 생성되는 무한집합으로 간주할 수 있으며 다음과 같이 정의된다. **단위항목 집합** $I=\{i_1, i_2, \cdots, i_n\}$는 데이터 스트림에서 현재까지 생성된 단위항목들의 집합으로서 응용 분야에서 발생한 개별 정보를 의미한다. 하나의 **순차패턴**(sequential pattern) s는 단위항목들이 순차적으로 정렬된 리스트(ordered list)로서 $\langle e_1\ e_2\ \cdots\ e_l\rangle$와 같이 나타낸다. 여기서 ej($1\le j\le l$)는 단위항목을 나타낸다. 즉, $e_j\in I(1\le j\le l)$ 관계를 만족하며, 일반적으로 순차패턴의 구성요소라 지칭하기도 한다. 하나의 순차패턴 s에 대해서 순차패턴의 길이 $|s|$는 해당 순차패턴을 구성하는 단위항목들의 개수를 의미하며, n개의 단위항목들로 구성된 순차패턴은 **n-순차패턴**이라 지칭한다. 한편, 순차패턴 $s_1=\langle a_1\ a_2\ \cdots\ a_n\rangle$와 $s_2=\langle b_1\ b_2\ \cdots\ b_m\rangle$에 있어서 $a_1=b_{j1}$, $a_2=b_{j2}$, $\cdots$, $a_n=b_{jn}$ 관계를 만족하면서 $1\le j_1\langle j_2\langle\ \cdots\ \langle j_n\le m$ 관계를 만족하는 정수 $j_1, j_2, \cdots, j_n$이 존재할 때, s_1은 s_2의 **부분-순차패턴**(sub sequential pattern)이라 하고 s_2는 s_1의 **확대-순차패턴**(super sequential pattern)이라 한다. 즉, $s_1\subseteq s_2$ 관계를 만족하며, s_1은 s_2에 포함된다고 하고 s_2는 s_1을 포함한다고 표현한다. **순차정보**(sequence) S는 하나 이상의 단위항목들이 정렬된 집합으로서 각 순차정보는 다른 순차정보와 구별되는 **식별자**(sequence identifier) SID를 갖는다. 본 논문에서는 하나의 데이터 스트림에서 k번째 생성된 순차정보를 S_k로 나타내며, 이때 해당 순차정보의 식별자 SID는 k가 된다. 순차정보 S_k가 새롭게 생성되었을 때, **현재 데이터 스트림** D_k는 현재까지 생성된 모든 순차정보들로 구성된다. 즉, $D_k=\langle S_1, S_2, \cdots, S_k\rangle$이며, 해당 데이터 스트림에 포함된 **순차정보의 총 수**를 $|D|_k$로 나타낸다. 이와 같이 순차정보를 단위항목들의 정렬된 리스트로 정의하는 방식은 웹 사용 정보 분석이나 생물정보학 분야의 DNA 순서 분석 등 여러 분야에서 유용하게 활용될 수 있으며[9], 또한 순차정보를 항목집합(itemset)들의 정렬된 집합으로 정의하는 일반적인 표현 방식과도 상호 변환이 가능하다[15].

하나의 데이터 스트림 D_k에 대한 일반적인 순차패턴 마이닝에서는 해당 데이터 스트림에서 순차패턴 s가 발생되었을 때 s의 출현빈도 수 $C_k(s)$는 D_k에 포함되는 k개의 순차정보들 중에서 해당 순차패턴 s를 포함하는 순차정보의 수를 나타낸다. 마찬가지로 해당 시점에서 순차패턴 s의 지지도 $S_k(s)$는 D_k에 포함되는 총 순차정보의 수에 대한 출현빈도 수 $C_k(s)$의 비율로 정의된다. 따라서 데이터 스트림 D_k에 대한 일반적인 순차패턴 탐색에서 사전에 주어진 최소 지지도 S_{min}보다 크거나 같은 지지도를 갖는 순차패턴은 해당 시점에서의 빈발 순차패턴으로 정의된다. 하지만 본 논문에서 제안하는 발생 간격 제한 조건을 활용하는 순차패턴 탐색 문제에서는 순차패턴의 출현빈도 수에 대한 정의가 발생 간격 제한 조건을 적용하여 다음과 같이 변경된다. 최소지지도 S_{min}과 발생 간격 제한 조건 G_{max}가 주어졌을 때, 순차패턴 s의 발생 간격 제한 조건을 적용한 출현빈도 수는 최대 발생 간격이 G_{max}보다 작거나 같으면서 해당 순차패턴과 동일한 순차패턴을 부분패턴으로 포함하는 순차정보의 수를 나타낸다. 여기서 순차패턴의 최대 발생 간격은 순차패턴의 발생 간격을 정의하는 방식에 따라 몇 가지로 구분될 수 있으며 상세한 정의 방식은 4장에서 기술한다. 지지도 및 빈발 순차패턴에 대한 정의는 일반적인 데이터 스트림에서의 순차패턴 탐색과 동일하다. 결론적으로 본 논문에서 다루고자 하는 문제는 하나의 순차 데이터 스트림에 대해서 최소지지도 S_{min}과 발생 간격 제한 조건 G_{max}가 주어졌을 때, 해당 조건을 만족하는 모든 빈발 순차패턴을 탐색하는 것이다.

2. 관련 연구

데이터 스트림에 내재된 다양한 형태의 정보나 지식을 탐색

하기 위해서 다수의 알고리즘들이 활발히 제안되어 왔다. 이들 알고리즘들은 주로 마이닝 수행과정에서 메모리 사용량 및 수행시간을 최소화하는데 관심을 두고 연구되어 왔다. 특히 순차패턴 탐색은 탐색과정에서 고려되는 후보 패턴의 수가 매우 많고 마이닝 결과를 얻기 위한 수행시간이 상대적으로 긴 편이다. 따라서 순차정보가 지속적으로 발생되는 데이터 스트림 환경에서 각 시점의 빈발 순차패턴 집합을 한정적인 시간 안에 효율적으로 탐색하는 것이 매우 어렵다. 이러한 상황을 고려하여 데이터 스트림에서 순차패턴 탐색을 위한 다수의 이전 연구들은 마이닝 수행 과정에서의 메모리 사용량 최소화 및 수행시간 단축에 초점을 맞추어 진행되어 왔으며, 일반적으로 순차정보를 구성하는 단위항목 정보들의 단순 출현빈도 수만을 고려하여 결과를 구한다. 대표적으로 eISeq 방법[5]에서는 새로 발생된 순차패턴을 지연추가(delayed-insertion) 하거나 발생빈도가 현격히 감소된 일부 순차패턴을 메모리에서 더 이상 관리하지 않고 전지하는(pruning) 작업을 통해 마이닝 수행 과정에서 메모리 사용량 및 수행 시간을 감소시킨다. 하지만 이들 방법들은 순차정보를 구성하는 단위항목들의 발생 간격이나 발생 시간 등과 같은 추가적인 정보를 고려하여 보다 관심도가 큰 순차패턴을 탐색하지 못한다.

관심도나 흥미도가 보다 큰 순차패턴을 얻기 위한 이전 연구는 대부분 한정적인 데이터 집합을 대상으로 하는 순차패턴 탐색 분야에서 연구되었다. 즉, 한정적인 데이터 집합을 대상으로 순차정보에 내재된 발생시간 정보 및 발생 간격 정보 등을 활용하여 보다 관심도가 큰 순차패턴을 얻고자 하는 연구들이 진행되어 왔다. [9] 및 [10]에서는 제한 조건을 고려한 순차패턴 탐색 방법을 제안하였다. 이들 방법에서는 단위항목들의 발생 시간이나 발생 간격을 제한 조건으로 활용하고 있다. [16] 및 [17]에서는 발생 시간 정보를 갖는 순차 데이터 스트림에서 하나의 순차정보에서 인접한 두 단위항목들 간의 발생 시간 차이 정보를 활용하여 관심도가 높은 순차패턴을 찾기 위한 마이닝 방법이 제안되었다. 하지만 이들 연구들은 마이닝 수행 과정에서의 분석 대상 데이터 집합에 대한 탐색 횟수, 메모리 사용량 및 처리 시간 등의 문제로 데이터 스트림 환경에서는 효과적으로 적용되는데 어려움이 있다.

III. 데이터 스트림에서 순차 패턴 탐색을 위한 기본 연구

본 논문에서 제안하는 발생 간격 제한 조건을 활용한 순차패턴 탐색 방법은 데이터 스트림에서 순차패턴 탐색을 위한 기본적인 방법으로 선행 연구된 eISeq 방법을 기반으로 순차패턴을 구성하는 항목들의 발생 간격을 계산하여 제한 조건에 적합한 순차패턴 패턴을 마이닝 결과 집합으로 구한다. 본 절에서는 eISeq 방법에 대해서 간단히 기술함으로써 본 논문에서 제안하는 발생 간격 제한 조건을 활용한 빈발 순차패턴 탐색 방법의 기본적인 틀에 대한 이해도를 높이고자 한다.

하나의 데이터 스트림에 대해서 최소 지지도 S_{min}이 주어졌을 때, eISeq 방법은 해당 데이터 스트림에서 발생한 순차패턴들 중에서 S_{min} 이상의 지지도를 갖는 모든 순차패턴을 빈발 순차패턴으로 구해준다. eISeq 방법은 데이터 스트림을 구성하는 각 순차정보를 하나씩 차례로 탐색하며 별도의 후보패턴(candidate pattern) 생성 과정을 수행하지 않고 마이닝 결과를 얻는다. 해당 방법에서는 각 순차정보에서 발생한 순차패턴들 중에서 상세 출현빈도 수 정보를 관리해야 할 정도로 중요성이 큰 순차패턴 정보를 *모니터링 트리*(monitoring tree)라 불리는 전위트리(prefix tree) 구조를 이용하여 메모리에서 관리한다. 이때 상세 출현빈도 수 관리 대상이 되는 순차패턴을 구분하기 위해서 추가적인 임계값을 활용한다. 해당 임계값을 중요 지지도(significant support) $S_{sig}(0<S_{sig}<S_{min})$라 정의하고, 각 시점에서 S_{sig}보다 큰 지지도를 갖는 순차패턴은 중요 순차패턴(significant sequential pattern)으로 정의하고 이들의 출현빈도 수를 메모리상의 모니터링 트리에서 관리한다.

모니터링 트리의 각 노드는 하나의 단위항목을 가지며, 이는 트리의 루트 노드로부터 해당 노드까지 이르는 경로에 존재하는 단위항목들로 구성되는 하나의 순차패턴을 나타낸다. 각 노드는 해당 노드가 나타내는 순차패턴과 연관되어 (*cnt, cnt_r, sid, sid_r*) 정보를 관리한다. *cnt*는 현재 데이터 스트림 D_k에서 해당 노드가 나타내는 순차패턴의 *출현빈도* 수를 의미한다. *cnt_r*은 해당 순차패턴의 *후위-출현빈도* 수(remaining count)를 의미하며, 여기서 후위 출현빈도 수라 함은 현재 데이터 스트림 D_k에 포함된 순차정보들 중에서 *후위-순차정보*(remaining- sequence)가 해당 노드가 나타내는 순차패턴을 포함하는 순차정보의 수를 나타낸다. 하나의 순차정보 S_k에 대해서 해당 순차정보의 후위-순차정보(remaining-sequence) $R(S_k)$및 *전위-단위항목*(prefix-item) $P(S_k)$는 다음과 같이 정의된다[5]. *전위-단위항목*(prefix-item) $P(S_k)$는 순차패턴 S_k의 첫 번째 단위항목을 지칭하며, 후위-순차정보 $R(S_k)$는 $P(S_k)$를 제외한 모든 단위항목들을 포함하는 S_k의 부분-순차패턴을 지칭한다. 모니터링 트리의 각 노드에서 관리되는 (*cnt, cnt_r, sid, sid_r*) 정보에서 *sid*는 해당 노드가 나타내는 순차패턴을 포함하는 가장 최근 순차정보의 식별자를 의미하며, *sid_r*은 후위-순차정보가 해당 노드가 나타내는 순차패턴을 포함하는 가장 최근 순차정보의 식별자를 의미한다. 이 밖의 트리 구조의 유지 및 탐색을 위한 기본적인 구조는 일반적인 전위트리 구조에서와 동일하다.

이어서 eISeq 방법의 개략적인 수행과정을 살펴보면 다음과 같다. 데이터 스트림 D_k에 순차패턴 S_k가 생성되었을 때 다음에 기술하는 일련의 과정들이 순차적으로 진행된다. 이때, 빈발 순차패턴 탐색 과정은 각 시점에서 최신의 마이닝 결과를 얻고자 하는 경우에만 수행되며, 강제 전지 과정은 메모리 사용량을 줄이기 위한 작업으로서 일반적으로 주기적으로 수행되거나 또는 수행 중 메모리 사용량이 사전에 설정된 일정 기준에 근접했을 때 메모리 사용량을 감소시키기 위해 수행된다.

- 매개변수 갱신 : 현재 데이터 스트림을 구성하는 순차정보의 총 개수 등의 정보가 갱신된다.
- 출현빈도 수 갱신 : 순차정보 S_k에서 출현한 각 순차패턴 s에 대해서 이와 연관된 노드가 모니터링 트리에 존재하고 해당 순차정보 S_k에 의해서 탐색되지 않았을 경우 해당 노드의 정보 (*cnt, cnt_r, sid, sid_r*)는 다음과 같이 갱신된다. 먼저 *cnt* 값이 1 증가되고 *sid* 값은 현재 순차정보의 *SID* 값으로 갱신된다. 다음으로 순차패턴 s가 후위-순차정보 $R(S_k)$에 포함되고 후위-출현빈도 수가 아직 갱신되지 않았다면(즉, $sid\_r\langle k$), *cnt_r* 값이 1 증가되고 *sid_r* 값은 현재 순차정보의 *SID* 값으로 갱신된다. 이때 해당 노드의 갱신된 출현빈도 수로부터 구해진 지지도가 중요 지지도 S_{sig}보다 작다면, 해당 노드가 나타내는 순차패턴은비중요 순차패턴으로 간주되어 상세 출현빈도 수를 더 이상 관리할 필요가 없으며 해당 노드는 모니터링 트리로부터 제거된다.
- 순차패턴 추가 : 모니터링 트리의 각 노드 중에서 순차정보 S_k에서 발생된 순차패턴과 연관된 노드에 대한 출현빈도 수 갱신 작업이 종료된 후 S_k에 출현하였으나 모니터링 트리에 관리되고 있지 않는 중요 순차패턴을 모니터링 트리에 추가하기 위한 작업을 수행한다. 이를 위해서 S_k에서 비중요 단위항목을 제외하고 정제된 순차정보 $\overline{S}_k$ 를 구한다. 하나의 순차패턴이 비중요 단위항목을 하나 이상 포함하는 경우 해당 순차패턴은 중요 순차패턴이 될 수 없기 때문에 비중요 단위항목들은 미리 제거하고 정제된 순차정보를 이용하여 새로운 중요 순차패턴 발생 여부를 판단한다. 정제된 순차정보 $\overline{S}_k$ 를 얻기 위해서 순차정보 S_k를 구성하는 각 단위항목의 출현빈도 수를 모니터링 트리를 탐색하여 확인하고 중요 지지도 S_{sig}보다 작은 지지도를 갖는 단위항목을 제거한다. 이때 모니터링 트리에 관리되고 있지 않는 새로운 단위항목은 이를 나타내는 노드를 모니터링 트리에 추가한다. 정제된 순차정보 $\overline{S}_k$ 가 구해지면 이를 이용하여 모니터링 트리를 한 번 더 탐색하면서 S_k에 새롭게 발생한 중요 순차패턴을 찾고 이를 나타내는 연관 노드를 모니터링 트리에 추가한다. 추가된 노드의 (*cnt, cnt_r, sid, sid_r*) 정보는 [5]에서 기술된 바와 같은 방법으로 값이 계산되어 초기화된다.
- 빈발 순차패턴 탐색 : 각 시점에서 최신의 빈발 순차패턴 집합을 얻기 위해서 전위트리를 기반으로 하는 일반적인 마이닝 방법에서와 동일한 과정으로 모니터링 트리를 탐색한다. 모니터링 트리 각 노드에서 지지도가 최소 지지도 S_{min}보다 크거나 같은 경우 해당 노드가 나타내는 순차패턴은 빈발 순차패턴으로 구해진다.
- 강제 전지 : 모니터링 트리의 각 노드를 순차적으로 탐색하면서 현재 관리되고 있는 순차패턴(즉, 해당 순차패턴을 나타내는 노드)들 중에서 중요 지지도 S_{sig}보다 작은 지지도를 갖는 비중요 순차패턴들을 해당 트리로부터 제거한다.

Ⅳ. 데이터 스트림에서 발생 간격 제한 빈발 순차패턴 탐색

본 절에서는 발생 간격 제한 빈발 순차패턴의 기본 개념을 정의하고 이를 탐색하는 과정에서 활용될 수 있는 기본 속성을 살펴본다. 이어서 해당 기본 개념 및 속성들을 바탕으로 순차 데이터 스트림에서 발생 간격 제한 빈발 순차패턴 탐색 방법인 **GConDS**(**G**ap-**Con**strained Sequential Pattern Mining over **D**ata **S**treams) 방법을 제시한다. GConDS 방법은 데이터 스트림 처리를 위한 기본적인 요구사항을 만족하면서 발생 간격 제한 조건을 고려하여 보다 관심도나 흥미도가 큰 순차패턴을 탐색할 수 있도록 지원한다.

1. 발생 간격 제한 빈발 순차패턴

발생 간격 제한 빈발 순차 패턴을 정의하기 위해서는 먼저 하나의 순차패턴에 대해서 해당 순차패턴의 발생 간격을 정의할 필요가 있다. 일반적으로 하나의 순차패턴에서 발생 간격이라 함은 해당 순차패턴을 구성하는 항목들의 순차정보 내에서의 발생 순서 차이를 의미한다. 예를 들어 하나의 순차정보 $S=\langle a\ b\ c\ d\ e\ f\rangle$에서 출현한 두 개의 서로 다른 순차패턴 $s_1=\langle a\ b\rangle$ 및 $s_2=\langle a\ f\rangle$를 고려해 보자. 전자는 발생 간격이 1인 반면에 후자의 경우는 발생 간격이 5가 된다. 이를 바탕으로 순차패턴의 발생 간격을 [정의 1]에서와 같이 정의할 수 있다. 이때 순차패턴의 발생 간격은 분석 대상 데이터 스트림을 생성하는 응용 분야의 특성이나 사용자의 관점에 따라 차별화하여 정의될 수 있으며, 본 논문에서는 [10] 등의 접근법을 고려하여 대표적인 두 가지 기준으로 발생 간격을 정의한다.

〔정의 1. 순차패턴의 발생 간격〕 순차 데이터 스트림을 구성하는 하나의 순차정보 $S=\langle a_1\ a_2\ \cdots\ a_m\rangle$와 이에 출현한 하나의 순차패턴 $s=\langle b_1\ b_2\ \cdots\ b_n\rangle$ 사이에는 $b_1=a_{j1}$, $b_2=a_{j2}$, $\cdots$, $b_n=a_{jn}$ 관계를 만족하면서 $1\le j_1<j_2<\cdots<j_n\le m$ 관계를 만족하는 정수 j_1, j_2, $\cdots$, j_n이 존재한다. 이때, 해당 순차패턴 s의 최대 발생 간격 $G(s)$를 다음과 같이 정의한다.

- 통합 발생 간격 기준(Type 1) : 해당 순차패턴 s를 구성하는 단위항목들 중에서 발생 순서가 가장 빠른 단위항목과 가장 늦은 단위항목의 순차정보 S에서의 발생 순서 차이를 순차패턴 s의 최대 발생 간격으로 정의한다. 즉, $G(s)$는 다음과 같이 정의된다.

  $G(s) = j_n - j_1$

- 인접 항목간 간격 기준(Type 2) : 해당 순차패턴 s를 구성하는 단위항목들 중에서 서로 인접한 두 단위항목의 순차정보 S에서의 발생 순서 차이를 계산하고 이들 값 중에서 가장 큰 값을 순차패턴 s의 최대 발생 간격으로 정의한다. 즉, $G(s)$는 다음과 같이 정의된다.

  $G(s) = max(j_n-j_{n-1},\ j_{n-1}-j_{n-2},\ \cdots,\ j_2-j_1)$

순차패턴에 대한 발생 간격이 정의되면 이를 바탕으로 순차 데이터 스트림에서 발생한 순차패턴의 발생 간격 제한 출현빈도 수 및 지지도를 정의할 수 있다. 순차패턴의 일반적인 출현빈도 수 및 지지도에 대한 정의와 유사하나 해당 순차패턴의 출현 여부를 판단하는데 있어서 발생 간격 제한 조건을 고려하여 〔정의 2〕에서와 같이 정의된다.

〔정의 2. 순차패턴의 발생 간격 제한 출현빈도 수 및 지지도〕 순차 데이터 스트림 D_k에서 발생한 하나의 순차패턴 s에 대해서 발생 간격 제한 조건 G_{max}가 주어졌을 때 해당 순차 패턴의 발생 간격 제한 출현빈도 수 $GC_k(s)$는 다음과 같이 정의된다.

$$GC_k(s) = \left|\{S \mid (s \subseteq S) \wedge (G(s) \le G_{\max}) \wedge (S \in D_k)\}\right|$$

따라서 D_k에서 해당 순차패턴 s의 발생 간격 제한 지지도 $GS_k(s)$는 다음과 같이 정의된다.

$$GS_k(s) = \frac{\left|\{S \mid (s \subseteq S) \wedge (G(s) \le G_{\max}) \wedge (S \in D_k)\}\right|}{|D|_k}$$

순차패턴의 발생 간격 제한 지지도가 정의되면, 이를 바탕으로 순차 데이터 스트림에서 최소 지지도 및 발생 간격 제한 조건이 주어졌을 때 발생 간격 제한 빈발 순차 패턴을 〔정의 3〕에서와 같이 정의할 수 있다.

〔정의 3. 발생 간격 제한 빈발 순차패턴〕 순차 데이터 스트림 D_k에 대해서 최소지지도 S_{min}과 발생 간격 제한 조건 G_{max}가 주어졌을 때, 해당 데이터 스트림에서 발생한 하나의 순차패턴 s의 발생 간격 제한 지지도 $GS_k(s)$가 S_{min}보다 크거나 같은 값을 가질 때 해당 순차패턴을 발생 간격 제한 빈발 순차패턴이라 정의한다.

2. 속성 분석

단순 지지도를 기반으로 하는 순차패턴 탐색 과정에서 데이터 집합에 대한 탐색 횟수나 탐색 범위를 크게 감소시켜 마이닝 수행의 효율성을 높이는 대표적인 속성은 단순 지지도의 anti-monotone 속성이다. anti-monotone 속성은 동일한 데이터 집합에서 발생한 두 개의 순차패턴 s_1과 s_2에 대해서 s_2가 s_1의 확대-순차패턴($s_1 \subseteq s_2$)일 때 순차패턴 s_2의 지지도 $S(s_2)$는 순차패턴 s_1의 지지도 $S(s_1)$보다 항상 작거나 같음을 의미한다. 〔속성 1〕에서 보듯이 발생 간격 제한 지지도도 단순 지지도와 마찬가지로 anti-monotone 속성을 만족하며, 이는 발생 간격 제한 순차패턴 탐색 과정에서 마이닝 수행 속도 및 메모리 사용량 등을 감소시키는 중요한 역할을 한다.

〔속성 1. 발생 간격 제한 지지도의 anti-monotone 속성〕 순차 데이터 스트림 D_k에서 발생한 두 개의 순차패턴 s_1과 s_2에서 s_2는 s_1의 확대-순차패턴($s_1 \subseteq s_2$)이라고 가정하자. 이때 순차패턴 s_1의 발생 간격 제한 지지도는 〔정의 2〕에 의해 다음과 같이 구해진다.

$$GSk(s1) = \frac{\left|\{S \mid (s_1 \subseteq S) \wedge (G(s_1) \le G_{\max}) \wedge (S \in D_k)\}\right|}{|D|_k} \quad \cdots\cdots (1)$$

한편, $s_1 \subseteq s_2$인 관계를 만족하기 때문에 다음의 관계를 만족한다.

$$\left|\{S \mid (s_1 \subseteq S) \wedge (G(s_1) \le G_{\max}) \wedge (S \in D_k)\}\right| \ge \left|\{S \mid (s_2 \subseteq S) \wedge (G(s_2) \le G_{\max}) \wedge (S \in D_k)\}\right| \quad \cdots\cdots (2)$$

식 (1)과 (2)에 의해 다음의 관계가 만족함을 알 수 있다.

$$\frac{\left|\{S \mid (s_1 \subseteq S) \wedge (G(s_1) \le G_{\max}) \wedge (S \in D_k)\}\right|}{|D|_k} \ge \frac{\left|\{S \mid (s_2 \subseteq S) \wedge (G(s_2) \le G_{\max}) \wedge (S \in D_k)\}\right|}{|D|_k}$$

즉, $GS_k(s_1) \ge GS_k(s_2)$ 관계를 만족한다. 따라서 순차패턴

s_2의 발생 간격 제한 지지도 $GS_k(s_2)$는 항상 순차패턴 s_1의 발생 간격 제한 지지도 $GS_k(s_1)$보다 작거나 같다.

데이터 스트림에 대한 발생 간격 제한 순차패턴 탐색 과정에서, 만약 하나의 순차패턴 s의 발생 간격 제한 지지도 $GS_k(s)$가 최소 지지도 S_{min}보다 작다면, 〔속성 1〕에 따라 s의 확대-순차패턴들은 빈발 순차패턴이 될 수 없음을 판단할 수 있다. 이와 더불어 발생 간격 제한 조건도 anti- monotone 속성을 만족한다. 즉, 순차 데이터 스트림 D_k에서 발생한 두 개의 순차패턴 s_1과 s_2에서 s_2는 s_1의 확대-순차패턴($s_1 \subseteq s_2$)이라고 가정하자. 이때 순차패턴 s_2의 발생 간격 $G(s_2)$는 〔정의 1〕에 의해 항상 순차패턴 s_1의 발생 간격 $G(s_1)$보다 크거나 같다. 따라서 순차패턴 s_1이 발생 간격 제한 조건 G_{max}보다 커서 발생 간격 제한 순차패턴이 될 수 없는 경우 이의 확대-순차패턴인 s_2도 발생 간격 제한 순차패턴이 될 수 없다.

3. GConDS 방법

발생 간격 제한 조건을 활용하여 데이터 스트림에서 관심도가 높은 순차패턴을 탐색하기 위한 방법으로 본 논문에서 제안하는 *GConDS* 방법은 먼저 데이터 스트림 처리를 위한 기본적인 요구사항들을 만족해야 한다. 이를 위해서 *GConDS* 방법은 3장에서 기술된 데이터 스트림에서 순차패턴 탐색을 위한 기본적인 접근 방법을 기반으로 구현되었다. 이에 더불어 발생 간격 제한 조건을 적용하기 위한 과정을 추가하여 데이터 스트림 환경에서 발생 간격 제한 빈발 순차패턴을 효율적으로 탐색할 수 있도록 지원한다.

Input: A sequence data stream D_k,
A support threshold S_{min}, A significant support threshold S_{sig}, A gap-constraint G_{max}
Output: A complete set of gap-constrained frequent sequential patterns $GConSP_k$

ML : A monitoring tree that maintains a set of significant sequential patterns

```
ML = ∅;
for each new sequence S_k in D_k {

  // 매개변수 갱신
  Update the total number of sequences |D|_k in the current sequence data stream;

  // 출현빈도 수 갱신 및 순차패턴 추가
  for each sequential pattern s⊆S_k {
    Compute G(s) of s, i.e., the maximum gap of s;
     // '인접 항목간 발생 간격 기준' 또는 '통합 발생 간격 기준'을 적용

    if G(s)≤G_max {
      If its corresponding node with an entry (cnt, cnt_r, sid, sid_r) is in ML and sid<k {
        The cnt and sid of the entry are updated subsequently;
        if (cnt / |D|_k) < S_sig
          The corresponding entry is pruned from ML;  // s는 비중요 순차패턴
        if (s⊆R(S_k)) && (sid_r < k)   // R(S_k) is a remaining-sequence of S_k
          The cnt_r and sid_r of the entry are updated subsequently;
      }
      If its corresponding node with an entry (cnt, cnt_r, sid, sid_r) is not in ML {
        If s is a significant sequential pattern, its corresponding node with an entry (cnt, cnt_r, sid, sid_r) is inserted into ML,
        and the values of cnt, cnt_r, sid and sid_r in the entry are initialized as described in [5];
      }
    } // if
  } // for

  // 빈발 순차패턴 탐색
  GConSP_k = ∅;
  for all sequential pattern s whose corresponding node is in ML
    if GS_k(s) ≥ S_min
      GConSP_k = GConSP_k ∪ {s};  // s는 최대 발생 간격이 G_max 이하인 빈발 순차패턴
}
```

그림 1. *GConDS* 방법
Fig. 1. *GConDS* method

분석 대상이 되는 순차 데이터 스트림에서 하나의 순차정보가 새롭게 생성되었을 때, 해당 순차정보를 처리하기 위한 매개변수 갱신 과정, 출현빈도 수 갱신 및 순차패턴 추가 과정 등 일련의 작업들이 차례로 수행된다. 일정 시점에서의 마이

닝 결과를 얻고자 하는 경우 빈발 순차패턴 탐색 작업을 수행하고, 마이닝 수행 과정에서 메모리 사용량을 줄이기 위한 방법으로 강제 전지 작업을 수행하기도 한다. 순차패턴의 중요 순차패턴 여부를 즉시 검사하고 중요 순차패턴으로 확인되는 경우 이를 모니터링 트리에 추가한다. 이때 발생 간격 제한 조건을 적용하기 위해서는 출현빈도 수 갱신 및 순차패턴 추가 과정에서 처리 대상이 되는 순차패턴의 발생 간격 정보를 고려하여 사전에 정의된 발생 간격 제한 조건 G_{max} 보다 큰 발생 간격을 갖는 순차패턴들은 처리 과정에서 더 이상 고려되지 않는다. 이를 통해서 G_{max}보다 작은 발생 간격을 갖는 관심도가 큰 순차패턴들만 마이닝 수행 과정에서 다뤄지도록 한다.

한편, *GConDS* 방법에서는 마이닝 수행 과정에서 처리 속도 개선을 위해서 출현빈도 수 갱신 과정 및 순차패턴 추가 과정을 통합하여 처리하도록 수정하였다. 즉, 출현빈도 수 갱신 과정에서 검색 대상이 되는 순차패턴이 모니터링 트리에 존재하지 않는 경우(즉, 출현빈도 수 갱신 작업이 실패한 경우), 해당 이를 통해 기존 처리 방식의 순차패턴 추가 과정에서 정제된 순차정보 생성 과정 및 이를 활용한 모니터링 트리 재탐색 과정을 생략함으로써 수행시간을 일부 단축할 수 있다. *GConDS* 방법의 수행 과정은 그림 1에서와 같다.

V. 실험 결과 분석

1. 실험 데이터 집합

논문에서 제안된 방법의 효용성을 검증하기 위해서 *SD_IBM* 및 *SD_Web* 이라는 두 가지 데이터 집합을 사용하여 실험을 수행하였다. *SD_IBM*은 데이터 마이닝 분야에서 실험용 데이터 집합 생성에 널리 사용되는 IBM 데이터 생성기(IBM data generator)[18]를 이용하여 생성되었으며, *SD_Web*은 실제 응용 분야에서 생성된 데이터 집합이다. *SD_IBM* 데이터 집합은 1,000 개의 단위항목으로부터 생성된 100,000 개의 순차정보로 구성되며 각 순차정보는 평균적으로 20개의단위항목으로 구성된다. *SD_Web* 데이터 집합은 웹 사이트의 사용자 접근 로그로부터 생성된 데이터 집합이다. 사용자 접근 로그에서 각 웹페이지를 단위항목으로 간주하고 연속적으로 접근된 웹 페이지들을 하나의 순차정보로 구성하였다. 이때 일정 시간 동안 사용자 입력이 없거나 사용자가 접속을 종료한 경우 하나의 순차정보가 생성된 것으로 간주하였다. 해당 데이터 집합은 545 개의 단위항목을 가지며 1,000,000 개의 순차정보로 구성된다. 또한 가장 긴 순차정보는 30 개의 단위항목으로 구성된다.

본 논문에서 제시되는 모든 실험에서는 구성요소가 지속적으로 발생되고 이를 순차적으로 처리해야 하는 데이터 스트림 환경을 구현하기 위해서 각 데이터 집합을 구성하는 순차정보를 하나씩 차례로 탐색하여 처리한다. 논문에서 제시되는 각 실험에서 *GConDS* 방법 수행을 위한 매개변수들 중 S_{sig}는 S_{min}의 30%로 설정되었으며 메모리 사용량을 줄이기 위한 강제 전지 작업은 1,000 개의 순차정보마다 반복하여 수행되도록 설정하였다. 이들 두 가지 매개변수는 발생 간격 제한 조건을 활용한 빈발 순차패턴 탐색에는 직접적인 영향을 미치지 않는 것으로서 각 실험에서 동일하게 설정하였다.

2. 기본 성능 평가 실험

제안된 방법의 기본적인 특성을 분석하기 위해서 SD_IBM 데이터 집합에 대해서 마이닝 결과로 얻어지는 빈발 순차패턴 집합을 분석하고 이를 얻기 위한 마이닝 수행 시간을 비교하였다. 해당 실험에서 최소 지지도는 0.005로 설정되었으며, 두 가지 발생 간격 정의 기준(즉, Type1 및 Type2)에 대해 발생 간격 제한 조건 Gmax를 변화시키면서 실험하였다.

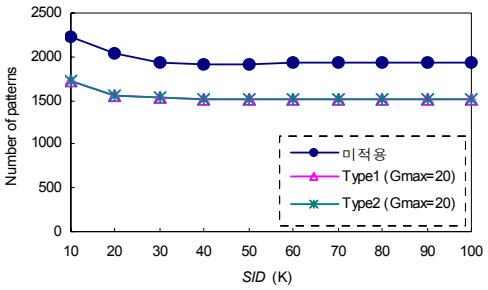


그림 2 . 결과 패턴 수 변화
Fig. 2. Change of the number of resulting patterns

표 1. 결과 패턴에 대한 상세 분석
Table 1. Detailed analysis of resulting patterns
(a) 인접 항목간 발생 간격 기준 (Type1)

G_{max}	L_1	L_2	L_3	L_4	L_5	L_6 이상	계
미적용	735	1177	20	6	1	0	1939
5	735	145	20	6	1	0	907
10	735	290	20	6	1	0	1052
20	735	741	20	6	1	0	1503
30	735	998	20	6	1	0	1760
50	735	1152	20	6	1	0	1914

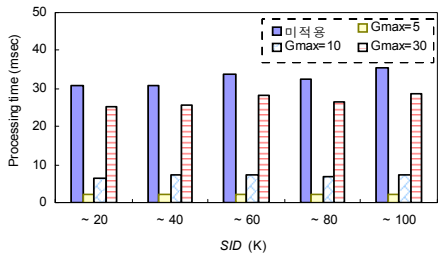
(b) 통합 발생 간격 기준 (Type2)

G_{max}	L_1	L_2	L_3	L_4	L_5	L_6 이상	계
미적용	735	1177	20	6	1	0	1939
5	735	145	17	3	0	0	900
10	735	290	20	6	1	0	1052
20	735	741	20	6	1	0	1503
30	735	998	20	6	1	0	1760
50	735	1152	20	6	1	0	1914

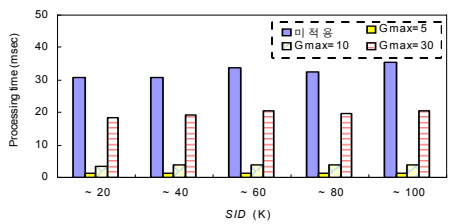
먼저 그림 2는 발생 간격 제한 조건 설정에 따른 결과 패턴 수의 변화를 분석하기 위한 것으로 두 가지 발생 간격 정의 기준에 의해 구해진 결과 집합을 발생 간격 제한 조건을 적용하지 않은 경우와 비교하였다. *SD_IBM*에 속하는 순차정보가 지속적으로 발생되는 상황에서 순차정보가 매번 10,000 개씩 증가되는 시점에서의 마이닝 결과를 구하여 비교하였다. 그림에서 알 수 있듯이 발생 간격 제한 조건 미적용시 결과로 구해지는 빈발 순차패턴들 중에서 발생 간격이 큰 일부 순차패턴들은 발생 간격 제한 조건을 적용하는 경우 결과 집합에서 제외된다. 따라서 마이닝 결과로 구해지는 순차패턴의 수가 감소됨을 알 수 있다. 한편, 발생 간격 정의 기준 두 가지는 본 실험에서 사용된 데이터 집합에서는 큰 영향을 미치지 않음을 알 수 있다.

표 1은 G_{max} 변화에 따른 결과 집합의 상세 변화 내용을 제시하고 있으며, 해당 데이터 집합을 구성하는 100,000 개의 순차정보가 모두 처리된 후의 결과를 분석하였다. 표에서 $L_k(k=1,2,\cdots)$는 k개의 단위항목으로 구성된 빈발 순차패턴을 의미한다. 하나의 단위항목으로 구성되는 L_1의 경우는 순차패턴 내에 발생 간격이 존재하지 않으므로 모든 경우에 얻어지는 결과가 동일하다. L_3 이상의 경우에도 G_{max} 값이 매우 작게 설정된 경우(Type2에 대해서 G_{max}=5인 경우)를 제외하고는 발생 간격 제한 조건을 설정하더라도 거의 유사한 결과를 얻음을 알 수 있다. 반면 L_2의 경우는 발생 간격 제한 조건의 영향을 크게 받는다. *SD_IBM* 데이터 집합은 실제 응용 분야 데이터 집합이 아니라 실험을 위해 인위적으로 생성된 데이터 집합으로서 각 순차패턴의 지지도 및 발생 간격 등이 적절히 분포되어 있으므로 큰 발생 간격을 갖는 순차패턴이 자주 발생되어 빈발 순차패턴이 되는 경우는 거의 존재하지 않는다. 따라서 길이가 긴 순차패턴에 있어서는 발생 간격 제한 조건 적용에 무관하게 유사한 결과를 보이며, 상대적으로 짧은 길이의 순차 패턴인 L_2의 경우에만 큰 차이를 보인다.

구성요소가 지속적으로 확장되는 특징으로 데이터 스트림에 대한 처리 및 마이닝 방법들에서는 수행 시간 측면의 유용성을 검증하기 위해서 일반적으로 순차정보 등과 같은 하나의 단위 정보를 처리하는데 소요되는 시간을 측정하여 비교한다. 그림의 3의 결과는 SD_IBM 데이터 집합에 대한 마이닝 수행 과정에서 수행시간을 비교하여 보여주는 것으로서, 해당 데이터 집합에 포함되는 전체 순차정보를 발생 순서에 따라 20,000 개씩 5개의 구간으로 나누어 각 구간별 평균 수행 시간을 제시하고 있다. 그림에서 볼 수 있듯이 기본적으로 하나의 순차정보를 처리하기 위한 수행시간이 30 msec 미만으로 매우 작다. 또한, 발생 간격 제한 조건을 적용하는 경우 미적용시보다 마이닝 수행시간이 감소됨을 알 수 있다. 이러한 결과를 고려할 때, 발생 간격 제한 조건을 적용하는 경우 큰 발생 간격을 갖는 관심도가 낮은 순차패턴을 제외하여 결과를 구할 수 있을 뿐만 아니라 마이닝 수행 시간도 단축할 수 있음을 알 수 있다.



(a) 인접 항목간 발생 간격 기준 (Type1)



(b) 통합 발생 간격 기준 (Type2)

그림 3. 마이닝 수행 시간
Fig. 3. Mining processing time

3. 실제 응용 분야 데이터 실험

실제 응용 분야에서 발생한 데이터에 대한 제안된 방법의 유용성을 검증하기 위해서 *SD_Web* 데이터 집합에 대한 실험을 수행하였다. 본 실험에서 최소 지지도는 0.005로 설정되었으며, 논문에서 제안한 두 가지의 발생 간격 정의 기준 중에서 '통합 발생 간격 기준 (Type1)'을 적용하여 실험하였다. 앞서 *SD_IBM*에 대한 실험에서도 확인할 수 있듯이 '인접 항목간 발생 간격 기준 (Type2)'를 적용하는 경우에도 유사한 결과를 나타내므로 *SD_Web* 데이터 집합에서는 이에 대한 실험은 생략하였다.

그림 4는 *SD_Web* 데이터 집합에 대한 실험에서 마이닝 결과로 얻어지는 순차패턴의 개수 변화를 보여주며, *SD_Web*에 속하는 순차정보가 지속적으로 발생되는 상황에서 순차정보가 매번 100,000 개씩 처리된 시점에서의 마이닝 결과를 구하여 비교하였다. 그림에서 알 수 있듯이 G_{max} 값이 매우 작게 설정된 경우에는 결과로 구해지는 순차패턴의 수가 상당히 감소되었으나 G_{max} 값이 증가함에 따라 순차패턴 개수의 감소폭이 크게 작아짐을 알 수 있다. 발생 간격 제한 조건을 적용했을 때 마이닝 결과로 구해지는 순차패턴의 길이별 변화를 상세히 분석하기 위한 실험으로 다양한 G_{max} 값에 대한 길이별 순차패턴의 개수를 분석한 결과를 표 2에서 보여주고 있다. *SD_IBM* 데이터 집합에서와는 달리 L_3, L_4 및 L_5 등 비교적 길이가 긴 순차패턴에서도 발생 간격 제한 조건을 적용하는 경우 결과 순차패턴의 수가 감소됨을 알 수 있다. 이러한 상황은 실제 응용 분야에서 생성된 *SD_Web* 데이터 집합에서는 발생 간격 제한 조건을 적용하지 않는 경우 길이가 길고 발생 간격이 큰 순차패턴이 마이닝 결과집합에 포함되어 있기 때문이다. 하지만 발생 간격이 큰 이러한 순차패턴들은 앞서 설명한 바와 같이 상대적으로 낮은 중요성이나 관심도를 갖는 순차패턴이며, 굳이 마이닝 결과로 구해질 필요가 없는 것들이다. 따라서 본 논문에서 제안한 바와 같이 발생 간격 제한 조건을 적용하여 마이닝 수행 단계에서 발생 간격이 큰 순차패턴들을 결과집합에서 제외함으로써 보다 효율적인 순차패턴 마이닝을 수행할 수 있다.

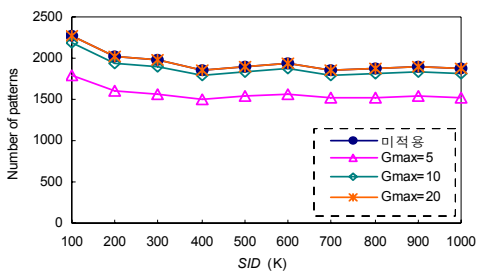


그림 4. *SD_Web* 데이터 집합에 대한 결과 패턴 수 변화
Fig. 4. Change of the number of resulting patterns on the data set *SD_Web*

표 2. SD_Web 데이터 집합에 대한 결과 패턴 상세 분석
Table 2. Detailed analysis of resulting patterns on the data set SD_Web

G_{max}	L_1	L_2	L_3	L_4	L_5	L_6	L_7	L_6 이상	계
미적용	90	387	605	492	249	56	1	0	1880
5	90	337	477	382	194	43	1	0	1524
10	90	375	590	473	238	54	1	0	1821
20	90	386	605	492	249	56	1	0	1879

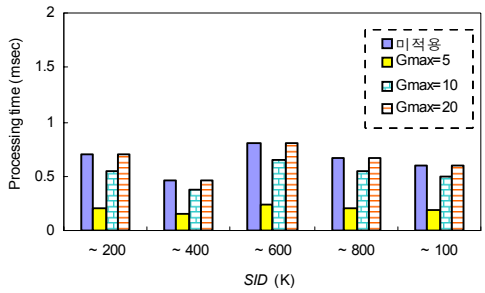


그림 5. *SD_Web* 데이터 집합에 대한 마이닝 수행 시간
Fig. 5. Mining processing time on the data set *SD_Web*

그림 5는 *SD_Web* 데이터 집합에 대한 *GConDS* 방법 수행 과정에서 마이닝 수행 시간 변화를 보여주는 것으로서, 해당 데이터 집합에 포함되는 순차정보를 발생 순서에 따라 200,000 개씩 5개의 구간으로 나누어 각 구간별 평균 수행 시간을 제시하고 있다. 그림에서 볼 수 있듯이 기본적으로 하나의 순차정보를 처리하기 위한 수행시간이 1 msec 미만으로 매우 작으며, 발생 간격 제한 조건을 적용하는 경우 이를 적용하지 않는 경우보다 수행시간이 감소됨을 알 수 있다. *SD_Web*에 대한 이상의 실험 결과를 고려할 때 논문에서 제안된 방법은 실제 응용 분야에서 발생되는 데이터에도 유용하게 적용되어 매우 짧은 수행시간에 발생 간격 제한 빈발 순차패턴을 탐색할 수 있음을 알 수 있다.

Ⅵ. 결 론

여러 컴퓨터 응용 분야에서 발생되는 순차정보 형태에 대한 마이닝 과정에서는 순차정보 구성요소의 단순 발생 순서뿐만 아니라 발생 시간이나 발생 간격 등도 매우 유용한 정보를 제공할 수 있다. 이러한 부가 정보를 활용하는 경우 해당 응용 분야의 특성을 보다 잘 표현하거나 사용자의 관심도가 큰 순차패턴을 마이닝 결과로 얻을 수 있으며, 이를 통해 기존의 단순 순차패턴 탐색 과정의 비효율적인 부분들도 보완할 수 있다. 즉, 기존의 단순 순차패턴 탐색 과정에서 지나치게 많은 수의 순차패턴들이 마이닝 결과로 구해짐에 따라 이를 다시 분석하여 보다 중요하거나 관심도가 큰 순차패턴을 얻는 과정을 피할 수 있다.

본 논문에서는 데이터 스트림에 대한 순차패턴 탐색 과정에서 순차패턴을 구성하는 단위항목들의 발생 간격을 고려하는 마이닝 방법을 제안하였다. 즉, 데이터 스트림에서 발생 간격 제한 빈발 순차패턴 탐색 방법을 제안하였다. 하나의 순

차패턴에 있어서 발생 간격은 분석 대상이 되는 데이터 스트림의 특성이나 사용자의 관심도에 따라 다양한 기준으로 정의될 수 있으며, 본 논문에서는 하나의 순차패턴을 구성하는 처음 단위항목과 마지막 단위항목의 발생 순서 차이에 기반하는 '통합 발생 간격 기준' 및 인접한 두 단위항목의 순차정보 내에서의 발생 순서 차이 중에서 가장 큰 발생 간격에 기반하는 '인접 항목간 발생 간격 기준'에 따라 발생 간격을 구하고, 이를 활용하여 발생 간격 제한 빈발 순차패턴을 탐색하였다. 실험을 통해 논문에서 제안한 방법이 데이터 스트림에 대한 순차패턴 탐색 과정에 효과적으로 적용될 수 있으며, 발생 간격 제한 조건 값에 따라 보다 정제된 형태의 순차패턴 집합을 마이닝 결과로 구해줌을 확인하였다. 또한 실제 응용 분야에서 발생된 데이터 집합에 대해서도 효율적인 마이닝 수행을 통해 유용한 결과를 얻을 수 있음을 보여준다.

근래에는 다양한 컴퓨터 응용 분야에서 데이터 스트림 형태로 정보를 발생 시키고 있다. 더욱이 유통 관련 회사의 구매 정보, 전자상거래 관련 데이터 및 웹 접근 로그 등 다수의 응용 분야에서 발생되는 데이터 스트림은 순차정보 형태를 갖는다. 따라서 데이터 스트림에 대한 순차패턴 탐색 방법은 이들 응용 분야에서 유용하게 활용될 수 있으며, 본 논문에서 제안한 발생 간격 제한 빈발 순차패턴 탐색 방법을 활용하여 보다 정제된 형태의 순차패턴 집합을 얻을 수 있을 것이다.

일반적으로 데이터 스트림에 대한 마이닝에서는 각 시점의 분석 결과를 메모리상에서 유지함으로써 지속적으로 확장되는 데이터 스트림 환경에서 최신의 분석 결과를 효율적으로 얻을 수 있음은 물론이고, 시간 흐름에 따른 마이닝 결과의 변화를 파악할 수 있다. 앞서 기술된 실험 결과에서와 같이 데이터 스트림에서 순차패턴 탐색 과정에서 발생 간격 제한 조건을 활용하여 보다 관심도가 높은 순차패턴들로 구성되는 마이닝 결과를 얻음으로써 데이터 스트림에 대한 순차패턴 탐색의 효율성을 높일 수 있다.

한편 순차정보를 발생시키는 여러 응용 분야들 중에는 특별히 큰 발생 간격을 갖는 특이상황이나 예외상황에 해당되는 순차패턴이 중요한 의미를 갖는 경우가 있다. 즉, 순차패턴을 구성하는 단위항목들의 발생 간격이 일정 범위 이내인 경우는 관심을 두지 않고 특정 기준 이상의 발생 간격을 갖는 순차패턴들이 보다 중요하게 간주될 수 있다. 순차패턴 탐색에 있어서 특이패턴(outlier)으로 간주되는 이러한 순차패턴은 본 논문에서 제안한 발생 간격 제한 순차패턴 마이닝 방법으로는 탐색하는데 어려움이 있다. 이와 같은 데이터 스트림에서의 특이 순차패턴 탐색을 위한 연구들도 흥미로운 향후 연구주제가 될 수 있을 것이다.

참고문헌

[1] J. Kang, J.F. Naughton, and S.D. Viglas, "Evaluating Window Joins over Unbounded Streams," in Proc. of the 19th Int'l Conf. on Data Engineering, pp. 341-352, 2003.

[2] J.H. Chang and W.S. Lee, "A Sliding Window Method for Finding Recently Frequent Itemsets over Online Data Streams," Journal of Information Science and Engineering, Vol. 20, pp. 753-762, 2004.

[3] G. Mao, X. Wu, X. Zhu, G. Chen, and C. Liu, "Mining Maximal Frequent Itemsets from Data Streams," Journal of Information Science, Vol. 33, pp. 251-262, 2007

[4] J.X. Yu, Z. Chong, H. Lu, Z. Zhang, and A. Zhou, "A False Negative Approach to Mining Frequent Itemsets from High Speed Transactional Data Streams," Information Sciences, Vol. 176, pp. 1986-2015, 2006

[5] J.H. Chang and W.S. Lee, "Efficient Mining Method for Retrieving Sequential Patterns over Online Data Streams," Journal of Information Science, Vo. 31, pp. 420-432, 2005.

[6] Q. Huang and W. Ouyang, "Mining Sequential Patterns in Data Streams," in Proc. of the 6th Int'l Symposium on Neural Networks, pp. 865-874, 2009.

[7] C.-H. Lin, D.-Y. Chiu, Y.-H. Wu, and A.L.P. Chen, "Mining Frequent Itemsets from Data Streams with a Time-Sensitive Sliding Window," in Proc. of th 5th SIAM Int'l Conf. on Data Mining, pp. 68-79, 2005.

[8] E. Chen, H. Cao, Q. Li, and T. Qian, "Efficient Strategies for Tough Aggregate Constraint-based Sequential Pattern Mining," Information Sciences, 178(6), pp. 1498-1518, 2008.

[9] X. Ji, J. Bailey, and G. Dong, "Mining Minimal Distinguishing Subsequence Patterns with Gap Constraints," Knowledge and Information Systems, 11(3), pp. 259-296, 2007.

〔10〕 J. Pei, J. Han, and W. Wang, "Mining Sequential Patterns with Constraints in Large Databases," Proc. of the 2002 ACM Int'l Conf. on Information and Knowledge Management (CIKM '02), pp. 18-25, 2002.
〔11〕 C. Luo and S.M. Chung, "Efficient Mining of Maximal Sequential Patterns Using Multiple Samples," Proc. of the 2005 SIAM Int'l Conf. on Data Mining (SDM '05), pp. 64-72, 2005.
〔12〕 P. Tzvetkov, X. Yan, and J. Han, "TSP: Mining Top-K Closed Sequential Patterns," Knowledge and Information Systems, 7(4), pp. 438-457, 2005.
〔13〕 J. Wang and J. Han, and C. Li, "Frequent Closed Sequence Mining without Candidate Maintenance," IEEE Transactions on Knowledge and Data Engineering, 19(8), pp. 1042-1056, 2007.
〔14〕 X. Yan, J. Han, and R. Afshar, "CloSpan: Mining Closed Sequential Patterns in Large Datasets," Proc. of the 2003 SIAM Int'l Conf. on Data Mining (SDM '03), pp. 166-177, 2003.
〔15〕 M. Garofalakis, J. Gehrke, and R. Rastogi, "Querying and Mining Data Streams: You Only Get One Look," in The tutorial notes of the 28th Int'l Conf. on Very Large Data Bases, 2002.
〔16〕 Y.-L. Chen, M.-C. Chiang, and M.-T. Ko, "Discovering Fuzzy Time-Interval Sequential Patterns in Sequence Databases," IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics, 35(5), pp. 959-972, 2005.
〔17〕 Y.-L. Chen and T. C.-H. Huang, "Discovering Time-Interval Sequential Patterns in Sequence Databases," Expert Systems with Applications, 25(1), pp. 343-354, 2003.
〔18〕 R. Agrawal and R. Srikant, "Mining Sequential Patterns," in Proc. of the 1995 Int'l Conf. on Data Engineering, pp. 3-14, 1995.

저 자 소 개

장 중 혁
1996년 : 연세대학교 컴퓨터과학과 졸업 (이학사)
1998년 : 연세대학교 대학원 컴퓨터과학과 (공학석사)
2005년 : 연세대학교 대학원 컴퓨터과학과 (공학박사)
2006년1월~2008년7월 : UIUC, Wright State Univ. 박사 후 연구원
2008년 9월~현재 : 대구대학교 컴퓨터IT공학부 교수