지능정보연구 제16권 제3호 2010년 9월(pp.55~75)

# 발생 간격 기반 가중치 부여 기법을 활용한 데이터 스트림에서 가중치 순차패턴 탐색

장중혁
대구대학교 컴퓨터·IT공학부
(jhchang@daegu.ac.kr)

일반적인 순차패턴 마이닝에서는 분석 대상 데이터 집합에 포함되는 구성요소의 발생 순서만을 고려하며, 따라서 단순 순차패턴은 쉽게 찾을 수 있는 반면 실제 응용 분야에서 널리 활용될 수 있는 관심도가 큰 순차패턴을 탐색하는데 한계가 있다. 이러한 단점을 보완하기 위한 대표적인 연구 주제들 중의 하나가 가중치 순차패턴 탐색이다. 가중치 순차패턴 탐색에서는 관심도가 큰 순차패턴을 얻기 위해서 구성요소의 단순 발생 순서 뿐만 아니라 구성요소의 가중치를 추가로 고려한다. 본 논문에서는 발생 간격에 기반 한 순차패턴 가중치 부여 기법 및 이를 활용한 순차 데이터 스트림에 대한 가중치 순차패턴 탐색 방법을 제안한다. 발생 간격 기반 가중치는 사전에 정의된 별도의 가중치 정보를 필요로 하지 않으며 순차정보를 구성하는 구성요소들의 발생 간격으로부터 구해진다. 즉, 순차패턴의 가중치를 구하는데 있어서 구성요소의 발생순서와 더불어 이들의 발생 간격을 고려하며, 따라서 보다 관심도가 크고 유용한 순차패턴을 얻는데 도움이 된다. 한편, 근래 대부분의 컴퓨터 응용 분야에서는 한정적인 데이터 집합 형태가 아닌 데이터 스트림 형태로 정보를 발생시키고 있다. 이와 같은 데이터 생성 환경의 변화를 고려하여 본 논문에서는 순차 데이터 스트림을 마이닝 대상으로 고려하였다.

논문접수일 : 2010년 05월 13일 논문수정일 : 2010년 07월 03일 게재확정일 : 2010년 07월 21일 교신저자 : 장중혁

## 1. 서론

순차패턴 마이닝(sequential pattern mining)은 순차 데이터 집합으로부터 흥미로운 순차패턴(sequential pattern)을 탐색하는데 목적을 두고 있으며, 이는 주요한 데이터마이닝 분야 중의 하나로서 다양한 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, 따라서 이를 바로 응용 분야의 특성을 이해하기 위해서 활용하는데 어려움이 있다. 한편, 일반적으로 순차패턴 마이닝에서는 순차패턴이나 이를 구성하는 단위항목들의 중요성이 동일한 것으로 간주된다. 하지만 실제 응용 분야에서 이들 단위항목(즉, 단위 항목들이 나타내는 실제 응용 분야에서의 단위 정보)들은

* 이 논문은 2008년도 대구대학교 신임교수학술연구비 지원에 의한 논문임.

서로 다른 중요성을 가지며, 따라서 순차패턴 마이닝에서 이들의 차별화된 중요성을 고려하는 경우 보다 흥미도나 관심도가 큰 순차패턴을 얻을 수 있다. 이러한 상황을 고려하여 가중치 순차패턴 마이닝에 대한 연구들이 활발히 진행되어 왔다. 단순 지도를 기반으로 하는 일반적인 순차패턴 마이닝과는 달리 가중치 순차패턴 마이닝에서는 순차정보(sequence) 구성요소별로 차별화된 가중치를 고려하여 보다 관심도가 큰 순차패턴을 탐색한다(Lo, 2005; Yun, 2008).

가중치 순차패턴 마이닝의 필요성을 살펴보기 위하여 인터넷 쇼핑몰에서 이용자의 웹 페이지 접근 로그로부터 생성된 순차 데이터 집합을 예로 들어 보자. 이때 하나의 웹 페이지는 하나의 개별 상품에 대한 정보를 담고 있다고 가정한다. 해당 데이터 집합에서 각 순차정보는 개별 이용자의 순차적인 웹 페이지 접근 기록을 담고 있으며, 이는 해당 이용자가 관심을 갖고 정보를 검색한 물품들의 리스트로 간주할 수 있다. 단순 지지도에 기반한 일반 순차패턴 마이닝에서는 해당 데이터 집합에서 물품의 단순 검색 순서만 고려하여 순차패턴을 구한다. 하지만 해당 데이터 집합에 대한 마이닝에서 단순히 이용자들이 관심을 갖는 물품에 대한 분석 결과뿐만 아니라 물품의 가격 정보가 고려된 분석 결과를 얻을 수 있다면 해당 인터넷 쇼핑몰의 이익을 증가시킬 수 있을 것이다. 이를 고려하여 물품에 대한 검색 순서 뿐만 아니라 각 물품의 가격 정보를 고려하여 순차패턴 마이닝을 수행하는 가중치 순차패턴 마이닝을 적용하는 경우 해당 인터넷 쇼핑몰의 필요에 보다 적합한 분석 결과를 얻을 수 있다.

일반적으로 순차정보 또는 순차패턴에 있어서는 이를 구성하는 단위 항목들의 발생 순서뿐만 아니라 각각의 발생 시간이나 발생 간격 등도 중요한 정보를 제공한다. 예를 들어 두 개의 순차패턴이 서로 동일한 단위 항목들로 구성되며 이들의 발생순서가 서로 동일한 경우에도 하나의 순차패턴을 구성하는 단위 항목들의 발생 간격이 다른 하나에 비해 짧은 경우 발생 간격이 짧은 순차패턴을 보다 중요한 순차패턴으로 간주할 수 있다.

이상에서 기술한 가중치 순차패턴 마이닝의 효용성 및 순차패턴 탐색에서 구성요소의 발생 간격의 중요성 등을 바탕으로 본 논문에서는 발생 간격 기반 가중치 부여 기법을 활용한 가중치 순차패턴 마이닝 방법을 제안하고자 한다. 특히 한정적인 데이터 집합이 아니라 지속적으로 확장되는 데이터 스트림 형태로 정보를 발생시키는 근래 컴퓨터 응용 분야의 특성을 고려하여 순차 데이터 스트림을 대상으로 발생 간격 기반 가중치 순차패턴을 탐색한다. 먼저 하나의 순차패턴을 구성하는 구성요소에 대해서 발생 간격을 정의하고, 이로부터 발생 간격에 기반 한 가중치 부여 기법을 제시한다. 이어서 해당 기법을 데이터 스트림에 대한 순차패턴 마이닝에 적용하여 데이터 스트림에 대한 가중치 순차패턴 마이닝 기법을 완성한다. 논문에서 제안되는 방법은 데이터 스트림 처리의 기본적인 요구 조건들(Garofalakis, 2002)을 만족하면서 발생 간격에 기반 한 단위 항목의 가중치를 고려하여 관심도나 흥미도가 큰 순차패턴을 얻을 수 있다. 즉, 마이닝 수행 과정에서 메모리 사용량 및 수행 시간 최소화 등과 같은 데이터 스트림 처리를 위한 기본적인 요구 조건을 만족하면서 발생 간격 기반 가중치 순차패턴을 얻을 수 있다.

순차패턴 탐색을 위한 여러 데이터마이닝 방법들 중에서 발생 간격에 관심을 둔 이전의 마이닝 방법들은 발생 간격을 제한조건으로 활용하거나 하나의 개별 단위 항목 정보로 간주하여 마이닝 결과를 구하였다(Ji, 2007; Pei, 2002). 이로 인해 마이닝 수행 결과로 얻어지는 순차패턴 집합이 발생

간격의 작은 차이에도 지나치게 변화된다. 이와 달리 본 논문에서 제안하는 방법과 같이 발생 간격을 기반으로 순차정보의 가중치를 구하고 이를 활용하여 순차패턴을 탐색하는 경우 발생 간격의 차이에 유연하게 변화되는 마이닝 결과 집합을 얻을 수 있다. 한편, 가중치 순차패턴 탐색을 위한 이전의 방법들은 일반적으로 순차정보 또는 순차패턴의 가중치를 구하기 위해서 사전에 부여된 가중치 정보를 필요로 하는 반면 본 논문에서는 해당 가중치 정보를 순차정보를 구성하는 단위 항목들의 발생 간격으로부터 구한다. 즉, 본 논문에서 제안하는 발생 간격 기반 가중치 순차패턴 탐색 방법은 사전에 부여된 가중치 정보가 없이 순차정보 및 순차패턴에서 중요한 의미를 갖는 단위 항목들의 발생 간격으로부터 가중치 순차패턴을 구할 수 있으며, 이를 통해 발생 간격 변화에 유연하게 적응되는 순차패턴 집합을 마이닝 결과를 얻을 수 있다.

본 논문의 구성은 다음과 같다. 먼저 제 2장에서는 본 논문에서 다뤄지는 문제와 연관된 이전의 연구들을 간략히 정리하며, 제 3장에서는 순차 데이터 스트림에 대한 정의를 포함하여 본 논문의 전개과정에서 필요한 기본 개념을 정리한다. 제 4장에서는 본 논문에서 제안하는 핵심 내용인 발생 간격 기반 가중치 부여 기법에 대해서 상세히 기술하고, 데이터 스트림에 대한 순차패턴 마이닝 과정에서 해당 기법을 적용하여 발생 간격 기반 가중치 순차패턴을 탐색하는 데이터 스트림 마이닝 방법을 제 5장에서 기술한다. 제 6장에서는 일련의 실험을 통해 제안된 방법의 효용성을 검증하고, 끝으로 제 7장에서 논문의 결론을 맺는다.

## 2. 관련 연구

순차패턴 마이닝은 데이터마이닝의 주요 기법 중의 하나로서 웹 기반 시스템, 생물정보학, 전자상거래 등과 같은 컴퓨터 응용 분야에서 생성되는 데이터를 효율적으로 분석하기 위하여 널리 활용되고 있으며, 다양한 연구들이 지속적으로 진행되어 왔다. 순차패턴 마이닝은 탐색하고자 하는 결과 패턴의 특성이나 탐색 조건 등에 따라 일반적인 순차패턴 마이닝(Lin, 2008; Pei, 2004), closed 순차패턴 및 maximal 순차패턴 마이닝(Luo, 2005; Wang, 2007), 제한 조건을 활용한 순차패턴 마이닝(Ji, 2007; Pei, 2002), 근사 순차패턴 마이닝(Kum, 2003; Kum, 2006), 가중치 순차패턴 마이닝(Lo, 2005; Yun, 2008) 등의 분야로 세분화 된다.

가중치 순차패턴 마이닝은 단순 지지도 뿐만 아니라 가중치 정보를 추가로 활용하여 관심도가 큰 순차패턴을 마이닝 결과로 얻고자 하는 것으로, 이를 통해 실제 응용 분야에서 순차패턴 마이닝의 활용도를 높일 수 있다. 가중치 순차패턴 마이닝을 위한 대부분의 이전 연구들은 일반적으로 사전에 정의된(혹은 부가적으로 명시된) 별도의 가중치 정보를 필요로 한다. 예를 들어 도소매 업체의 물품 판매 데이터 집합에서는 판매 물품의 단가나 각 판매에 따른 판매량 등의 정보를 가중치 정보로 활용한다. 하지만 사전에 정의된 가중치를 활용하는 이러한 접근 방법은 부가적인 정보를 발생시키지 않고 단순히 구성요소의 순차적인 발생 여부만을 판단할 수 있는 응용 분야에서는 효율적으로 활용되는데 한계가 있다. 한편, 상대적으로 관심도가 큰 순차패턴을 탐색하기 위한 다른 연구들도 활발히 진행되어 왔다. Ji(2007) 및 Pei(2002)에서는 제한 조건을 고려한 순차패턴 탐색 방법을 제안하였다. 이들 방법에서는 단위 항목들의 발생 시간이나 발생 간격을 제한 조건으로 활용하고 있다. Chen(2005) 및 Chen(2003)에서는 발생 시간 정보를 갖는 순차 데이터 스트림에서 하나의 순차정보

에 존재하는 인접한 두 단위 항목들 사이의 발생 시간 차이 정보를 활용하여 관심도가 높은 순차패턴을 찾기 위한 마이닝 방법이 제안되었다. 하지만 이들 연구들은 주로 한정적인 데이터 집합을 대상으로 연구되어 왔으며, 따라서 마이닝 수행 과정에서의 분석 대상 데이터 집합에 대한 탐색 횟수, 메모리 사용량 및 처리 시간 등의 문제로 데이터 스트림 환경에서는 효과적으로 적용되는데 어려움이 있다.

데이터 스트림 처리 및 마이닝에 대한 기존 연구들은 주로 마이닝 수행과정에서 메모리 사용량 및 수행시간을 최소화하는데 관심을 두고 진행되어 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 시간에 효율적으로 탐색하는 것이 매우 어렵다. 따라서 데이터 스트림에서 순차패턴 탐색을 위한 다수의 이전 연구들(Chang, 2005; Huang, 2009)은 가중치 정보 등과 같은 부가적인 정보를 활용하지 못하고 순차정보를 구성하는 단위항목들의 단순 출현빈도 수만을 고려하여 마이닝 결과를 구한다.

## 3. 기본 정리

본 절에서는 발생 간격 기반 가중치 순차패턴 마이닝을 위한 기본적인 내용으로서 분석 대상이 되는 순차 데이터 스트림을 정의하고 가중치 순차패턴 마이닝의 기본 개념을 정리한다.

### 3.1 순차 데이터 스트림

순차패턴 마이닝의 분석 대상이 되는 데이터 스트림은 시간적으로 정렬된 단위항목들의 집합인 순차정보가 지속적으로 생성되는 무한집합으로 간주할 수 있으며 다음과 같이 정의된다.

i) 단위 항목 집합 $\boldsymbol{I} = \{i_1, i_2, \cdots, i_n\}$는 데이터 스트림에서 현재까지 생성된 단위 항목들의 집합으로서 응용 분야에서 발생한 개별 정보를 의미한다.

ii) 하나의 순차패턴(sequential pattern) $\boldsymbol{s}$는 단위 항목들이 순차적으로 정렬된 리스트(ordered list)로서 $<e_1\ e_2\ \cdots\ e_l>$와 같이 나타낸다. 여기서 $e_j(1 \le j \le l)$는 단위항목을 나타낸다. 즉, $e_j \in I(1 \le j \le l)$ 관계를 만족하며, 일반적으로 순차패턴의 구성요소라 지칭하기도 한다. 하나의 순차패턴 $s$에 대해서 **순차패턴의 길이** $|s|$는 해당 순차패턴을 구성하는 단위항목들의 개수를 의미하며, $n$개의 단위항목들로 구성된 순차패턴은 ***n*-순차패턴**이라 지칭한다. 한편, 순차패턴 $s_1 = <a_1\ a_2\ \cdots\ a_n>$과 $s_2 = <b_1\ b_2\ \cdots\ b_m>$에 있어서 $a_1 = b_{j1}$, $a_2 = b_{j2}$, $\cdots$, $a_n = b_{jn}$ 관계를 만족하면서 $1 \le j_1 < j_2 < \cdots < j_n \le m$ 관계를 만족하는 정수 $j_1$, $j_2$, $\cdots$, $j_n$이 존재할 때, $s_1$은 $s_2$의 **부분-순차패턴**(sub sequential pattern)이라 하고 $s_2$는 $s_1$의 **확대-순차패턴**(super sequential pattern)이라 한다. 즉, $s_1 \subseteq s_2$ 관계를 만족하며, $s_1$는 $s_2$에 포함된다고 하고 $s_2$는 $s_1$을 포함한다고 표현한다.

iii) 순차정보(sequence) $S$는 하나 이상의 단위항목들이 정렬된 집합으로서 각 순차정보는 다른 순차정보와 구별되는 **식별자**(sequence identifier) SID를 갖는다. 본 논문에서는 하나의 데이터 스트림에서 $k$번째 생성된 순차정보를 $S_k$로 나타내며, 이때 해당 순차정보의 식

별자 SID는 $k$가 된다.

iv) 순차정보 $S_k$가 새롭게 생성되었을 때, **현재 데이터 스트림** $D_k$는 현재까지 생성된 모든 순차정보들로 구성된다. 즉, $D_k = <S_1, S_2, \cdots, S_k>$이며, 해당 데이터 스트림에 포함된 **순차정보의 총 수**를 $|D|_k$로 나타낸다.

일반적으로 순차정보는 항목집합(itemset)들의 정렬된 집합으로 표현되는 반면 본 논문에서는 순차정보를 단위 항목들의 정렬된 리스트로 정의하였다. 하지만, 본 논문의 정의 방식으로도 기존의 순차정보인 항목집합들의 정렬된 집합을 효과적으로 표현할 수 있다. 즉, 항목집합들의 정렬된 집합인 기존의 순차정보에서 각 단위 항목들을 발생 시간 순서에 따라 정렬하여 단위 항목들의 정렬된 집합으로 변경하는 경우 본 논문의 표현방식과 동일한 형태로 변경할 수 있다. 이때 발생 시간이 동일한 단위 항목들은 단위항목을 나타내는 기호의 알파벳 순서로 정렬된다. 유사한 방법으로 본 논문에서 사용한 방식으로 표현된 순차정보도 기존의 표현 방식으로 변경할 수 있다. 즉, 동시에 발생한 단위 항목들을 항목집합으로 결합하고 이들을 발생 시간 순서에 따라 정렬함으로써 기존의 표현 방식으로 변경할 수 있다(Garofalakis, 2002). 한편, 본 논문의 정의 방식 그 자체로도 웹 사용 정보 분석이나 생물정보학 분야의 DNA 순서 분석 등 여러 분야에서 유용하게 활용될 수 있다(Ji, 2007).

### 3.2 데이터 스트림에서 가중치 순차패턴 마이닝

하나의 데이터 스트림 $D_k$에 대한 가중치 순차패턴 마이닝에서 해당 데이터 스트림에 출현한 순차패턴 $s$의 출현빈도 수 및 지지도는 다음과 같이 정의된다. 먼저 해당 시점에서 $s$의 가중치를 고려한 출현빈도 수 $WC_k(s)$는 $D_k$에 포함되는 $k$개의 순차정보들에 대해서 각 순차정보에서 s의 가중치를 구하여 총합을 구한 값이다. 이때 해당 순차패턴 $s$를 포함하지 않는 순차정보에서 $s$의 가중치는 0으로 간주한다. 마찬가지로 해당 시점에서 순차패턴 $s$의 가중치를 고려한 지지도 $WS_k(s)$는 $D_k$에 포함되는 총 순차정보의 수 $k$에 대한 가중치를 고려한 출현빈도 수 $WC_k(s)$의 비율로 정의된다. 따라서 하나의 데이터 스트림 $D_k$에 대한 가중치 순차패턴 마이닝에서 해당 데이터 스트림에 출현한 순차패턴 $s$의 가중치를 고려한 지지도 $WS_k(s)$가 사전에 주어진 최소 지지도 $S_{min}$보다 크거나 같으면 순차패턴 $s$는 해당 시점에서 빈발 가중치 순차패턴으로 정의된다. 결론적으로 데이터 스트림에서 가중치 순차패턴 마이닝이란 분석 대상이 되는 데이터 스트림에 대해서 마이닝 결과를 얻고자 하는 시점에서의 모든 빈발 가중치 순차패턴을 구하는 작업이다. 이러한 정의는 데이터 스트림에 대한 가중치 순차패턴 마이닝에 대한 일반적인 정의를 기술한 것으로서 발생 간격 기반 가중치를 적용한 가중치 순차패턴 마이닝에 대해서는 제 4장에서 상세히 정의한다.

## 4. 발생 간격 기반 가중치

### 4.1 발생 간격 기반 가중치 부여 기법

순차패턴 마이닝에서 각 순차패턴에 발생 간격 기반 가중치를 정의하기 위해서는 먼저 순차패턴의 발생 간격을 명확히 정의할 필요가 있다. 일반적으로 하나의 순차패턴에서 발생 간격이라 함은 해당 순차패턴을 구성하는 구성요소들의 순차정보 내에서의 발생 순서 차이를 의미하며, 본 논문

에서는 인접한 구성요소들의 발생 순서 차이를 해당 순차패턴의 발생 간격으로 정의한다. 즉, 순차 데이터 스트림을 구성하는 하나의 순차정보 $S = < a_1 a_2 \cdots a_m >$와 이에 출현한 하나의 순차패턴 $s = < b_1 b_2 \cdots b_n >$ 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 $j_1, j_2, \cdots, j_n$이 존재한다. 이때, 해당 순차패턴 $s$에서 인접한 두 개의 단위항목 $b_p$ 및 $b_q$ $(1 \le p \le n-1, q = p+1)$ 사이의 **발생 간격** $G_{pq}$는 다음과 같이 정의되며 $n-1$개의 발생 간격이 정의된다.

$$G_{Pq} = j_q - j_p$$

이와 같이 정의되는 발생 간격을 활용하여 순차패턴의 가중치를 정의하기 위해서는 크게 두 가지 과정을 필요로 한다. 하나는 서로 다른 크기로 구해지는 발생 간격에 대한 정규화 과정이며, 다른 하나는 하나의 순차패턴에 다수가 정의되는 발생 간격(또는 이의 정규화 된 값)을 통합하여 해당 순차패턴을 대표할 수 있는 가중치를 정의하는 것이다. 먼저 0이상의 다양한 정수 값으로 구해진 발생 간격들 간의 공평한 비교를 위한 정규화의 과정으로 순차패턴에 존재하는 하나의 발생 간격에 대한 가중치를 [정의 1]에서와 같이 정의한다.

[정의 1 : 발생 간격의 가중치] 발생 간격 기반 가중치 부여 기법에서 가중치를 설정하는 기준이 되는 단위 발생 간격을 u(u > 0)라 하고, 단위 발생 간격 u마다 감소되는 가중치의 양을 결정하는 감쇠기본값을 b(0 < b < 1)라 하며, 또한 단위항목들 사이에 발생 간격이 존재하더라도 가중치가 감소되지 않는(즉, 발생 간격이 없는 것으로 간주되는) 최대 발생 간격을 의미하는 허용 발생 간격을 Ga라 하자. 이때, 하나의 순차패턴 $s = < b_1 b_2 \cdots b_n >$의 인접한 두 항목 $b_p$ 및 $b_p$ $(1 \le p \le n-1, q = p+1)$ 사이의 발생 간격 $G_{pq}$에 대한 가중치 $w(G_{pq})$는 다음과 같이 정의된다.

$$w(G_{pq}) = b^{(G_{pq} - G_a)/u}$$ ■

이어서 하나의 순차패턴에 존재하는 다수의 인접한 단위항목들의 조합으로부터 구해진 발생 간격 기반 가중치를 통합하여 해당 순차패턴에 대한 발생 간격 기반 가중치를 정의하며 [정의 2]에서와 같이 정의된다.

[정의 2 :. 순차패턴의 발생 간격 기반 가중치] 하나의 순차정보 $Sk$에 출현한 순차패턴 $s = < b_1 b_2 \cdots b_n >$에 대해서 해당 순차패턴에 존재하는 다수의 인접한 항목들간의 발생 간격 기반 가중치들 중에서 최소값을 해당 순차정보에서 순차패턴 $s$의 발생 간격 기반 가중치 $Wk(s)$라 정의하며, 다음과 같이 구해진다. 이때, $n = 0$인 경우는 해당 순차정보에서 $s$가 출현하지 않은 경우를 의미한다.

$$W_k(s) = \begin{cases} \min_{1 \le p \le n-1, q=p+1}(w(G_{pq})) & (n \ge 2) \\ 1 & (n = 1) \\ 0 & (n = 0) \end{cases}$$ ■

순차패턴에 대한 발생 간격 기반 가중치가 정의되면 이를 바탕으로 순차 데이터 스트림에서 발생한 순차패턴의 발생 간격 기반 가중치 출현빈도 수 및 지지도를 정의할 수 있다. 단순 지지도 기반의 순차패턴 출현빈도 수 및 지지도에 대한 정의와 유사하나 해당 순차패턴의 출현 빈도 수 계산 시각 순차정보에서 해당 순차패턴의 발생 간격 기반

가중치가 고려되어 [정의 3]에서와 같이 정의된다.

[정의 3 : 발생 간격 기반 가중치 출현빈도 수 및 지지도] 순차 데이터 스트림 Dk에서 발생한 하나의 순차패턴 s에 대해서 발생 간격 기반 가중치 출현빈도 수 gwCk(s)는 Dk에 포함되는 k개의 순차정보에서 해당 순차패턴의 발생간격 기반 가중치를 구하고 이를 전부 더한 값으로서 다음과 같이 정의된다.

$$gwC_k(s) = \sum_{S_i:(s\subseteq S_i)\wedge(S_i\in D_k)} W_i(s)$$

따라서 $D_k$에서 해당 순차패턴 $s$의 발생 간격 기반 가중치 지지도 $gwS_k(s)$는 다음과 같이 정의된다.

$$gwS_k(s)=\frac{gwC_k(s)}{|D|_k}=\frac{\sum_{S_i:(s\subseteq S_i)\wedge(S_i\in D_k)} W_i(s)}{|D|_k}$$ ■

순차패턴의 발생 간격 기반 가중치 지지도가 정의되면, 이를 바탕으로 순차 데이터 스트림에서 가중치 순차패턴 마이닝을 위한 빈발 발생 간격 기반 가중치 순차패턴이 정의된다. 즉, 순차 데이터 스트림 $D_k$에 대해서 최소 지지도 $S_{min}$이 주어졌을 때, 해당 데이터 스트림에서 발생한 하나의 순차패턴 $s$의 발생 간격 기반 가중치 지지도 $gwS_k(s)$가 $S_{min}$보다 크거나 같은 값을 가질 때 해당 순차패턴을 빈발 발생 간격 기반 가중치 순차패턴이라 정의한다.

한편, 순차 데이터 스트림 $D_k$에서 발생한 두 개의 순차패턴 $s_1$과 $s_2$에서 $s_2$는 $s_1$의 확대-순차패턴($s_1s_2$)이라고 가정할 때 하나의 순차정보 $S_i$에서 이들이 동시에 출현한 경우[정의 2]에 의해서 $W_i(s_1) \geq W_i(s_2)$ 관계를 만족한다. 또한 $s_1s_2$ 관계를 만족하므로 $s_2$를 포함하는 모든 순차정보는 $s_1$도 포함한다. 따라서 해당 데이터 스트림에서 $s_1$과 $s_2$의 발생 간격 기반 가중치 출현빈도 수 사이에 다음과 같은 관계가 성립한다.

$$gwC_k(s_1)=\sum_{S_i:(s_1\subseteq S_i)\wedge(S_i\in D_k)} W_i(s)$$
$$\geq \sum_{S_j:(s_2\subseteq S_j)\wedge(S_j\in D_k)} W_j(s) = gwC_k(s_2)$$

마찬가지로 $s_1$과 $s_2$의 발생 간격 기반 가중치 지지도 사이에는 다음과 같은 관계가 성립한다.

$$gwS_k(s_1)=\frac{\sum_{S_i:(s_1\subseteq S_i)\wedge(S_i\in D_k)} W_i(s)}{|D|_k} \geq$$
$$\frac{\sum_{S_j:(s_2\subseteq S_j)\wedge(S_j\in D_k)} W_j(s)}{|D|_k} = gwS_k(s_2)$$

포함 관계에 있는 두 순차패턴의 발생 간격 기반 가중치 지지도에 대한 이러한 관계를 발생 간격 기반 가중치 지지도의 anti-monotone 속성이라 하며, 이를 활용하여 발생 간격 기반 가중치 순차패턴 탐색 과정에서 데이터 집합에 대한 탐색 횟수나 탐색 범위를 크게 감소시킬 수 있다. 즉, 만약 하나의 순차패턴 $s$의 발생 간격 기반 가중치 지지도 $gwS_k(s)$가 최소 지지도 $S_{min}$보다 작아서 빈발 순차패턴이 되지 못한다면 발생 간격 기반 가중치 지지도의 anti-monotone 속성에 의해 $s$의 확대-순차패턴들도 빈발 순차패턴이 될 수 없음을 판단할 수 있다.

### 4.2 발생 간격 기반 가중치 적용 예제

본 절에서는 예제 데이터 집합을 이용하여 앞서 기술한 발생 간격 기반 가중치 적용에 대해서 설명한다. 먼저 예제 데이터 집합은 <그림 1>에서 보는 바와 같이 4개의 순차정보로 구성되며, $u = 1$, $b = 0.9$ 및 $G_a = 1$을 갖는 발생 간격 기반 가중치를

| SID | 순차정보 |
|---|---|
| 1 | <a, b, a, d, e> |
| 2 | <c, d, f> |
| 3 | <a, b, c, d, f> |
| 4 | <c, d> |

<그림 1> 예제 데이터 집합

적용한다. 여기서 두 개의 순차패턴 $s_1$ = <a b>와 $s_2$ = <a d >에 대해서 발생 간격 기반 가중치 적용에 따른 지지도 변화를 살펴보자. 먼저, 각 순차정보에서 두 순차패턴의 발생 간격 기반 가중치를 구하면 다음과 같다.

$$W_1(s_1) = 0.9^{(1-1)/1} = 1.0,$$

$$W_2(s_1) = 0,\ W_3(s_1) = 0.9^{(1-1)/1} = 1.0,\ W_4(s_1) = 0$$

$$W_1(s_2) = 0.9^{(3-1)/1} = 0.81,\ W_2(s_2) = 0,$$

$$W_3(s_2) = 0.9^{(3-1)/1} = 0.81,\ W_4(s_2) = 0$$

따라서 두 순차패턴의 발생 간격 기반 지지도 $gwS_k(s_1)$ 및 $gwS_k(s_2)$는 다음과 같이 구해진다.

$$gwS_k(s_1) = (1 + 0 + 1 + 0)/4 = 0.5$$
$$gwS_k(s_2) = (0.81 + 0 + 0.81 + 0)/4 = 0.405$$

두 순차패턴의 단순 지지도는 0.5로 서로 동일한 값을 갖는 반면 발생 간격 기반 가중치 지지도에서는 0.5와 0.405로 차별화된 값을 갖는다. 만약 해당 데이터 집합에 대한 빈발 순차패턴 탐색에서 최소 지지도가 0.5로 설정되었다면, 단순 지지도를 적용하는 경우 모두가 빈발 순차패턴으로 탐색되나 발생 간격 기반 가중치를 적용하는 경우 $s_1$은 빈발 순차패턴이 되지만 $s_2$는 빈발 순차패턴이 되지 못한다. 그 이유는 <그림 1>에서 보는 바와 같이 $s_1$ = <a b>를 구성하는 두 단위 항목들 사이의 발생 간격은 작은 반면에 $s_2$ = <a d>를 구성하는 두 단위항목들 사이의 발생 간격은 상대적으로 크기 때문이다. 즉, 순차패턴을 구성하는 단위항목들의 발생 간격에 따라 해당 순차패턴의 중요성이 차별화되어 서로 다른 지지도 값을 갖는다.

## 5. 데이터 스트림에서 발생 간격 기반 가중치 순차패턴 탐색

데이터 스트림에서 발생 간격 기반 가중치 순차패턴 마이닝은 분석 대상이 되는 순차 데이터 스트림에 대해서 해당 데이터 스트림에서 발생한 모든 빈발 발생 간격 기반 가중치 순차패턴들을 찾는 작업으로 정의할 수 있다. 본 절에서는 데이터 스트림 처리를 위한 기본적인 요구사항(Garofalakis, 2002)을 만족하면서 발생 간격 기반 가중치 순차패턴을 효율적으로 탐색하는 데이터 스트림 마이닝 기법을 제안한다. 먼저 제 5.1절에서는 데이터 스트림에서 순차패턴 마이닝을 위한 선행 연구를 간략히 기술하고, 이에 더하여 앞서 기술한 발생 간격 기반 가중치 부여 기법을 적용한 발생 간격 기반 가중치 순차패턴 탐색 방법인 ***WSGap***(**W**eighted **S**equential Pattern Mining via a **Gap**-based Weighting Approach) 방법을 제 5.2절에서 기술한다.

### 5.1 데이터 스트림에서 순차패턴 마이닝을 위한 선행 연구

본 논문에서 제안하는 발생 간격 기반 가중치 순차패턴 탐색 방법은 데이터 스트림에서 순차패턴 탐색을 위한 기본적인 방법으로 선행 연구된 eISeq(Chang, 2005) 방법을 기반으로 발생 간격 기반 가중치 부여 기법을 적용하여 마이닝 결과 집

합을 구한다. 본 절에서는 해당 방법에 대한 간략히 기술함으로써 본 논문에서 제안하는 *WSGap* 방법의 기본적인 틀에 대한 이해를 높이고자 한다.

eISeq 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. 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 $D_k$에서 해당 노드가 나타내는 순차패턴의 *출현빈도 수*를 의미한다. *cnt_r*은 해당 순차패턴의 *후위-출현빈도 수*(remaining count)를 의미하며, 여기서 후위 출현빈도 수라 함은 현재 데이터 스트림 $D_k$에 포함된 순차정보들 중에서 *후위-순차정보*(remaining-sequence)가 해당 노드가 나타내는 순차패턴을 포함하는 순차정보의 수를 나타낸다. 하나의 순차정보 $S_k$에 대해서 해당 순차정보의 후위-순차정보(remaining-sequence) $R(S_k)$및 전위-단위 항목(prefix-item) $P(S_k)$는 다음과 같이 정의된다(Chang, 2005). *전위-단위항목*(prefix- item)$P(S_k)$는 순차패턴 $S_k$의 첫 번째 단위항목을 지칭하며, 후위-순차정보(remaining-sequence)$R$ $(S_k)$는 $P(S_k)$를 제외한 모든 단위항목들을 포함하는 $S_k$의 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. 한편, 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 크거나 같은 지지도를 갖는 순차패턴은 중요 순차패턴(significant sequential pattern)으로 정의하고 이들의 출현빈도 수를 모니터링 트리에서 관리한다.

eISeq 방법의 개략적인 수행과정을 살펴보면 다음과 같다. 데이터 스트림 $D_k$에 순차패턴 $S_k$가 생성되었을 때 해당 순차정보를 처리하기 위하여 다음에 기술하는 일련의 과정들이 순차적으로 진행된다.

- 매개변수 갱신 : 현재 데이터 스트림을 구성하는 순차정보의 총 개수 등의 정보가 갱신된다.
- 출현빈도 수 갱신 : 순차정보 $S_k$에서 출현한 각 순차패턴 $s$에 대해서 이와 연관된 노드가 모니터링 트리에 존재하고 해당 순차정보 $S_k$에 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 $s$가 후위-순차정보 $R(S_k)$에 포함되고 후위-출현빈도 수가 아직 갱신되지 않았다면(즉, $sid\_r < k$), *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 순차패턴은 비중요 순차패턴으로 간주되어 상세 출현빈도 수를 더 이상 관리할 필요가 없으며 해당 노드는 모니터링 트리로부터 제거된다.

- 순차패턴 추가 : 모니터링 트리의 각 노드 중에서 순차정보 $S_k$에서 발생된 순차패턴과 연관된 노드에 대한 출현빈도 수 갱신 작업이 종료된 후 $S_k$에 출현하였으나 모니터링 트리에 관리되고 있지 않는 중요 순차패턴을 모니터링 트리에 추가하기 위한 작업을 수행한다. 이때, 하나의 순차패턴이 비중요 단위 항목을 하나 이상 포함하는 경우 해당 순차패턴은 중요 순차패턴이 될 수 없으므로 $S_k$를 구성하는 각 단위 항목의 출현빈도 수를 모니터링 트리를 탐색하여 확인하고 비중요 단위항목을 제외하여 *정제된 순차정보* $\overline{S_k}$를 구한 후 이를 활용하여 새로운 중요 순차패턴 발생 여부를 판단한다. 정제된 순차정보 $\overline{S_k}$를 구한 후 모니터링 트리를 한 번 더 탐색하면서 새롭게 발생한 중요 순차패턴을 찾고 이를 나타내는 연관 노드를 모니터링 트리에 추가한다. 추가된 노드의(*cnt*, *cnt_r*, *sid*, *sid_r*) 정보는 (Chang, 2005)에서 기술된 바와 같은 방법으로 값이 계산되어 초기화된다.

각 시점에서 최신의 빈발 순차패턴 집합을 얻기 위해서 전위트리를 기반으로 하는 일반적인 마이닝 방법에서와 동일한 과정으로 모니터링 트리의 각 노드를 탐색하여 해당 노드의 지지도가 최소 지지도 $S_{min}$보다 크거나 같은 경우 해당 노드가 나타내는 순차패턴은 빈발 순차패턴으로 구해지며, 이를 빈발 순차패턴 탐색 과정이라 한다. 또한, 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 제거하는 강제 전지 과정을 수행할 수도 있다. 강제 전지 과정은 마이닝 수행 과정에서 메모리 사용량을 줄이기 위한 작업으로서 일반적으로 주기적으로 수행되거나 또는 수행 중 메모리 사용량이 사전에 설정된 일정 기준에 근접했을 때 메모리 사용량을 감소시키기 위해 수행된다.

### 5.2 *WSGap* 방법

데이터 스트림에서 발생 간격 기반 가중치 순차패턴 탐색을 위한 *WSGap* 방법에서는 분석 대상이 되는 순차 데이터 스트림에서 하나의 순차정보가 새롭게 생성되었을 때, 제 5.1절에서 기술한 순차정보를 처리하기 위한 일련의 과정들이 차례로 수행되며, 일정 시점에서 마이닝 결과를 구하기 위한 빈발 순차패턴 탐색 과정 및 마이닝 수행 과정에서 메모리 사용량을 줄이기 위한 방법으로 강제 전지 작업을 수행하기도 한다.

*WSGap* 방법에서는 단순 지지도가 아닌 각 순차패턴의 발생 간격을 고려한 가중치를 구하고 이를 활용하여 출현빈도 수 갱신 과정 및 순차패턴 추가 과정을 수행한다. 즉, 출현빈도 수 갱신 과정에서 하나의 순차정보에 출현한 각 순차패턴에 대해서 출현빈도 수를 갱신하는데 있어서 단순 지지도를 기반으로 하는 기존의 방법에서는 1만큼 출현빈도 수를 증가시켰으나 *WSGap* 방법에서는 해당 순차패턴을 구성하는 단위항목들의 발생 간격으로부터 가중치를 구한 후 해당 가중치만큼 출현빈도 수를 증가시킨다. 순차패턴 추가 과정에서도 새로 발생된 순차패턴의 출현빈도 수를(Chang, 2005)에서 제시된 방법으로 계산하는데 있어서 해당 순차패턴의 발생 간격 기반 가중치를 활용한다. *WSGap* 방법의 수행 과정은 <그림 2>에서와 같다.

순차패턴 탐색 과정에서 발생 간격을 고려하는 이전의 대다수 방법들은 한정적인 데이터 집합을 대상으로 하며 마이닝 수행 과정에서 분석 대상

```
ML = ∅; //중요 순차패턴을 관리하는 모니터링 트리의 초기화
for each new sequence S_k in D_k {  // D_k : 분석 대상이 되는 순차 데이터 스트림

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

// 출현빈도 수 갱신
for each sequential pattern s = <a_1 a_2 ⋯ a_n>  S_k {
    // 순차패턴 s의 발생 간격 기반 가중치를 구함
    // b : 감쇠기본값, u : 단위 발생 간격, G_a : 허용 발생 간격
    if (|s|==1)  W_k(s) = 1;
    else  W_k(s) = min_{1≤p≤n−1,q=p+1} (b^{(G_pq − G_a)/u}); // G_pq : a_p와 a_q 사이의 발생 간격

    if (its corresponding node with an entry (cnt, cnt_r, sid, sid_r) is in ML) && (sid<k) {
        The cnt and sid of the entry are updated subsequently considering W_k(s);
        if (cnt / |D|_k) < S_sig  // S_sig : 중요 지지도
            The corresponding entry is pruned from ML; // s는 비중요 순차패턴
        if (s⊑R(S_k)) && (sid_r < k)
            The cnt_r and sid_r of the entry are updated subsequently considering W(S_k);
    }
}

// 순차패턴 추가
S̄_k = Get_FilteredSequence(S_k); // 새로운 출현한 단위항목 추가 및 정제된 순차패턴 생성
for each new sequential pattern s' = <b_1 b_2 ⋯ b_m > (m2) induced by S̄_k {
    W_k( s' ) = min_{1≤p≤m−1,q=p+1} (b^{(G_pq − G_a)/u});
    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 (Chang, 2005)
    considering W_k( s' );
}

// 빈발 발생 간격 기반 가중치 순차패턴 탐색
GWSP_k = ∅; // GWSP_k : D_k에 대한 빈발 발생 간격 기반 가중치 순차패턴 집합
for all sequential pattern s whose corresponding node is in ML
    if gwS_k(s) ≥ S_min    // S_min : 최소 지지도
        GWSP_k = GWSP_k ∪ {s}  // s는 빈발 발생 간격 기반 가중치 순차패턴
}
```

<그림 2> WSGap 방법

데이터 집합을 반복적으로 탐색한다.

따라서 이들 방법을 지속적으로 확장되고 갱신되는 데이터 스트림에 적용하는 경우 최신의 마이닝 결과를 얻고자 하는 각 시점마다 전체 데이터 스트림을 반복적으로 탐색하게 되므로 마이닝 수행 시간이 크게 증가된다. 반면 *WSGap* 방법은 분석 대상이 되는 데이터 스트림이 지속적으로 갱신되는 경우에도 새로 추가된 하나의 순차정보에 대한 처리를 통해 최신의 마이닝 결과를 얻을 수 있다. 즉, 분석 대상 데이터 스트림을 구성하는 전체 순차정보에 대한 재탐색이나 반복적인 탐색없이 추가된 순차정보에 대한 처리만으로 최신의 마이닝 결과를 얻을 수 있다. 따라서 매우 짧은 시간에 해당 마이닝 결과를 얻을 수 있다.

## 6. 실험 결과 고찰

본 절에서는 발생 간격 기반 가중치 부여 기법 및 *WSGap* 방법의 유용성 및 효율성을 검증하기 위한 일련의 실험 결과를 제시하고 이에 대한 분석 결과를 기술한다. 실험에 사용된 데이터 집합은 두 가지이며, 하나는 *SD_IBM* 데이터 집합으로서 순차패턴 마이닝 방법의 효율성 검증을 위한 실험용 데이터 집합 생성에 널리 활용되는 IBM 데이터 생성기(IBM data generator)(Agrawal, 1995)를 이용하여 생성되었으며, 다른 하나는 *SD_Web* 데이터 집합으로서 실제 응용 분야에서 *WSGap* 방법의 효율성을 검증하기 위한 것으로 웹 사이트의 사용자 접근 로그로부터 생성되었다. *SD_IBM* 데이터 집합은 1,000개의 단위 항목으로부터 생성된 100,000개의 순차정보로 구성되며, *SD_IBM* 데이터 집합은 545개의 단위항목(즉, 개별 웹 페이지)으로부터 생성된 1,000,000개의 순차정보로 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 처리하였다. *WSGap* 방법 수행을 위한 설정값 중에서 중요 지지도 $S_{sig}$ 값은 발생 간격 기반 가중치 순차패턴 탐색에는 직접적인 영향을 미치지 않는 것으로서 각 실험에서 동일하게 $S_{min}$의 30%로 설정하였다. 한편, *WSGap* 방법 수행에 있어서 데이터 스트림 처리를 위한 기본적인 요구조건인 메모리 사용량 및 마이닝 수행 시간 단축 등과 관련된 성능 분석 결과는 이전의 선행 연구에서 충분히 검증되었으며 본 논문에서는 이를 생략한다.

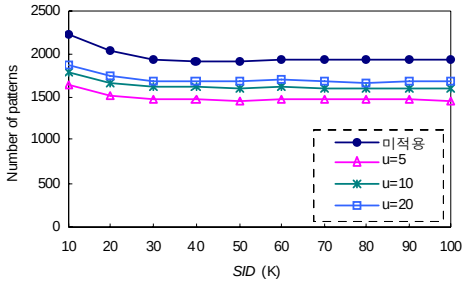
먼저 발생 간격 기반 가중치 부여 기법의 효용성을 검증하기 위해서 *SD_IBM* 데이터 집합에 대해서 발생 간격 기반 가중치 적용시 마이닝 결과로 얻어지는 빈발 순차패턴 집합을 분석하였다. 해당 데이터 집합에 대한 다음의 실험들에서 $S_{min}$ 값은 0.005로 설정되었다.

<그림 3>은 발생 간격 기반 가중치 적용에 따른 빈발 순차패턴 개수의 변화를 분석하기 위한 것으로서 발생 간격 기반 가중치 설정에 필요한 세가지 매개변수 $u$, $b$ 및 $G_a$ 값 변화에 따른 빈발 순차패턴 수 변화를 보여준다. 각각의 결과는 *SD_IBM*에 속하는 순차정보가 지속적으로 발생되는 상황에서 순차정보가 매번 10,000개씩 처리된 시점에서의 마이닝 결과를 구하여 비교하였다. <그림 3(a)>는 단위 발생 간격 $u$의 변화에 따른 빈발 순차패턴 수의 변화를 보여준다.

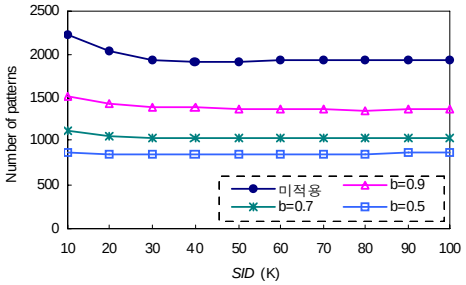
그림에서 보는 바와 같이 단위 발생 간격이 작을수록 마이닝 결과로 얻어지는 순차패턴의 수가 감소된다. 그 이유는 동일한 발생 간격을 갖는 순차패턴이라 하더라도 단위 발생 간격이 작을수록 해당 순차패턴의 가중치는 작아지므로 일정 시점에

서 해당 순차패턴의 출현빈도 수 및 지지도가 작아지며, 이로 인해 빈발 순차패턴으로 탐색되는 패턴의 수가 줄어들기 때문이다.

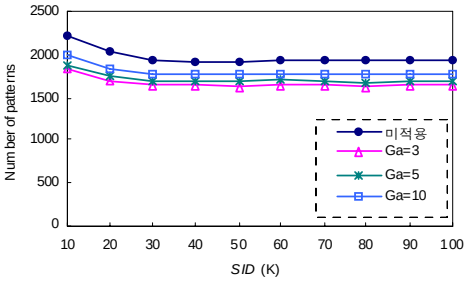
<그림 3(b)>는 감쇠기본값 $b$의 변화에 따른 빈발 순차패턴 수의 변화를 보여준다. 단위 발생 간격 $u$와 마찬가지로 감쇠기본값 $b$ 값이 작을수록




(a) 단위 발생 간격 $u$ 변화[$Ga = 5$, $b = 0.9$]




(b) 감쇠기본값 $b$ 변화[$u = 10$, $Ga = 0$]




(c) 허용 발생 간격 $Ga$ 변화[$u = 20$, $b = 0.9$]

<그림 3> 발생 간격 기반 가중치 적용에 따른 순차패턴 수 변화

동일한 발생 간격을 갖는 순차패턴의 가중치가 작게 부여된다. 따라서 그림에서 보듯이 감쇠기본값이 작을수록 마이닝 결과로 얻어지는 순차패턴의 수가 감소된다. 특히 감쇠기본값은 [정의 1]의 발생 간격의 가중치 정의에서 보듯이 지수형태를 띄는 가중치 함수에서 밑수가 되는 값으로서 감쇠기본값 감소에 따른 빈발 순차패턴 수의 감소폭이 크다. <그림 3(c)>는 허용 발생 간격 $G_a$ 변화에 따른 빈발 순차패턴 수의 변화를 보여준다. 허용 발생 간격은 발생 간격에 대한 가중치를 부여하는데 있어서 가중치가 감소되지 않고 허용될 수 있는 최대 발생 간격을 의미하므로, 허용 발생 간격이 클수록 동일한 발생 간격을 갖는 순차패턴에 있어서 가중치가 증가된다. 따라서 허용 발생 간격이 클수록 분석 대상 데이터 스트림에 발생한 각 순차패턴이 큰 가중치를 가지며 마이닝 결과로 얻어지는 빈발 순차패턴의 수가 많아진다.

<표 1> 마이닝 결과의 길이별 순차패턴 개수 [Ga = 5, b = 0.9]

| u | $L_1$ | $L_2$ | $L_3$ | $L_4$ | $L_5$ | $L_6$ 이상 | 계 |
|---|---|---|---|---|---|---|---|
| 미적용 | 735 | 1177 | 20 | 6 | 1 | 0 | 1939 |
| 5 | 735 | 695 | 20 | 6 | 1 | 0 | 1457 |
| 10 | 735 | 843 | 20 | 6 | 1 | 0 | 1605 |
| 20 | 735 | 916 | 20 | 6 | 1 | 0 | 1678 |

<표 1>은 발생 간격 기반 가중치 적용에 따른 가중치 순차패턴 마이닝 결과 집합에서 길이별 순차패턴의 개수 변화를 제시하고 있으며, *SD_IBM* 데이터 집합을 구성하는 100,000개의 순차정보가 모두 처리된 후의 결과를 분석하였다. 본 실험에서 감쇠기본값 $b$ 및 허용 발생 간격 $G_a$는 각각 0.9 및 5로 설정되었으며, 단위 발생 간격 $u$의 변화에 따른 순차패턴 수의 변화를 분석하였다. 표에서 $L_k$

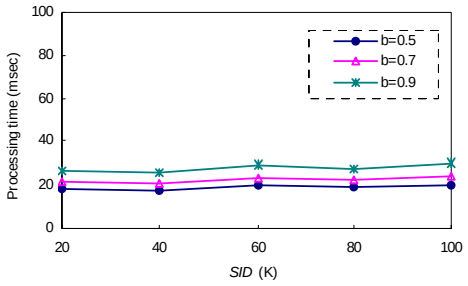
<표 2> 결과 패턴의 지지도 변화
[$u$ = 20, $Ga$ = 10, $b$ = 0.9]

| 패턴 | 단순 지지도 | 가중치 적용 지지도 | 지지도 차이 |
|---|---|---|---|
| <186,308> | 0.00526 | 0.00482 | 0.00044 |
| <223,991> | 0.00530 | 0.00489 | 0.00041 |
| <970,862> | 0.00518 | 0.00481 | 0.00037 |
| <378,991> | 0.00508 | 0.00472 | 0.00036 |
| <446,228> | 0.00511 | 0.00476 | 0.00035 |
| <223,542> | 0.00514 | 0.00479 | 0.00035 |
| <403,780> | 0.00523 | 0.00488 | 0.00035 |
| <72,42> | 0.00530 | 0.00497 | 0.00033 |
| <479,348> | 0.00501 | 0.00468 | 0.00033 |
| <657,533> | 0.00527 | 0.00494 | 0.00033 |

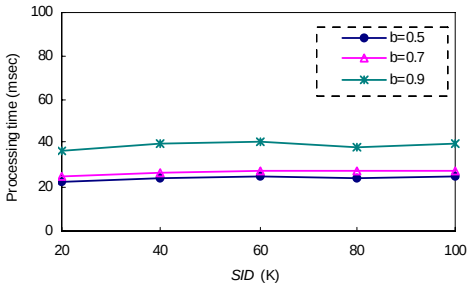
($k$ = 1, 2, …)는 $k$개의 단위 항목으로 구성된 빈발 순차 패턴을 의미한다. 하나의 단위항목으로 구성되는 $L_1$의 경우는 순차패턴 내에 발생 간격이 존재하지 않으므로 발생 간격 기반 가중치 적용 여부에 무관하게 얻어지는 결과가 동일하다. $L_3$이상의 경우에도 발생 간격 기반 가중치를 적용하는 경우에도 미 적용시와 동일한 결과를 보인다. 반면 $L_2$의 경우는 발생 간격 기반 가중치에 크게 영향을 받음을 알 수 있다. 실험에 사용된 *SD_IBM* 데이터 집합은 실제 응용 분야 데이터 집합이 아니라 인위적으로 생성된 데이터 집합으로서 각 순차패턴의 지지도 및 발생 간격 등이 적절히 분포되도록 생성된다. 또한 일반적으로 큰 발생 간격을 갖는 순차패턴이 자주 발생되어 빈발 순차패턴이 되는 경우는 거의 존재하지 않는다. 따라서 길이가 긴 순차패턴에 있어서는 발생 간격 기반 가중치 적용 여부에 무관하게 서로 동일한 결과를 보며, 상대적으로 짧은 길이의 순차 패턴인 $L_2$의 경우에만 큰 차이를 나타낸다.

발생 간격 기반 가중치 적용에 따른 순차패턴의 지지도 변화를 보다 명확히 파악하기 위하여 <표 2>에서와 같이 동일한 순차패턴에 대해서 단순 지지도와 발생 간격 기반 가중치 지지도를 비교하였다. 표에서 순차패턴을 구성하는 숫자들은 개별 단위 항목을 나타낸다. 본 실험에서 단위 발생 간격 $u$, 감쇠기본값 $b$ 및 허용 발생 간격 $G_a$는 각각 20, 0.9 및 10으로 설정되었다. <표 2>에서 제시된 순차패턴은 상대적으로 발생 간격이 큰 순차패턴으로서 발생 간격 기반 가중치 적용 시 지지도가 감쇠되는 것들이다. 표에서 제시된 것보다 많은 수의 순차패턴들이 발생 간격 기반 가중치 지지도가 감소되었으며, 표에서는 지지도 감소폭이 큰 10개를 제시하였다. 표에서 알 수 있듯이 각 순차패턴들은 단순 지지도 기반으로 순차패턴 마이닝을 수행하는 경우 $S_{min}$ = 0.005보다 큰 지지도를 가지므로 전부 빈발 순차패턴으로 탐색된다. 하지만 발생 간격 기반 가중치 순차패턴 탐색에서는 해당 최소 지지도보다 작은 지지도를 갖게 되므로 빈발 순차패턴이 될 수 없다.

<그림 4>는 *WSGap* 방법을 이용한 발생 간격 기반 가중치 순차패턴 탐색의 효율성을 검증하기 위한 실험 결과로서 마이닝 수행시간을 제시하고 있다. 본 실험에서는 분석 대상이 되는 데이터 스트림을 2만 개의 순차정보로 구성되는 다섯 개의 동일 크기 구간으로 구분하여 각 구간에 포함되는 순차정보를 처리하는데 소요된 평균 시간을 보여주고 있다. <그림 4(a)>는 분석 대상 데이터 스트림을 구성하는 각 순차정보를 처리하는데 필요한 시간을 나타내고 있으며, 그림에서 보듯이 30 msec 이하의 시간에 하나의 순차정보를 처리하고 있다. <그림 4(b)>는 최신의 발생 간격 기반 가중치 순차패턴을 얻기 위한 순차패턴 탐색 과정에 소요된 시간을 보여주고 있으며, 각 경우에 있어서 40msec 이하의 시간에 최신의 마이닝 결과 집합을 얻을 수 있다. 이러한 결과를 통해 매우 짧은 시간에 최신의 분석 결과를 얻게 됨을 알 수 있다.




(a) 순차정보 처리 시간



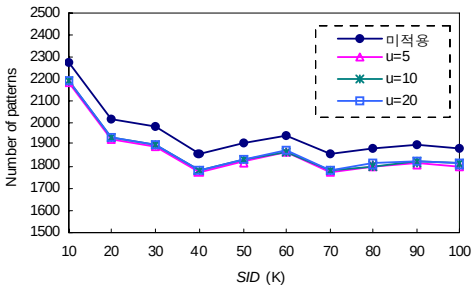

(b) 순차패턴 탐색 시간

<그림 4> 마이닝 수행 시간[$u$ = 10, $Ga$ = 0]

한편, 마이닝 수행 과정에서 메모리 사용량은 (Chang, 2005)에서 제시된 바와 같이 순차정보가 지속적으로 발생되는 경우에도 크게 증가되지 않고 일정 범위로 유지되며, 따라서 지속적으로 확장되는 데이터 스트림 환경에서도 효율적으로 마이닝 결과를 얻을 수 있다.

발생 간격 기반 가중치 부여 기법 및 *WSGap* 방법의 실제 응용 분야에서 발생한 데이터에 대한 유용성을 검증하기 위해서 *SD_Web* 데이터 집합에 대한 실험을 수행하였다. 해당 데이터 집합에 대한 다음의 실험들에서 $S_{min}$ 값은 0.005로 설정되었다.

<그림 5>는 단위 발생 간격 $u$의 변화에 따른 빈발 순차패턴 수의 변화를 보여준다. 실험 데이터 집합에 속하는 순차정보가 지속적으로 발생되는 상황에서 순차정보가 매번 100,000개씩 처리된 시점에서의 마이닝 결과를 구하여 비교하였다. 동일




<그림 5> *SD_Web* 데이터 집합에서 결과 패턴 수 변화[$Ga$ = 5, $b$ = 0.9]

한 발생 간격을 갖는 순차패턴이라 하더라도 단위 발생 간격이 작을수록 해당 순차패턴의 가중치가 감소되어 일정 시점에서 해당 순차패턴의 출현빈도 수 및 지지도가 작아지므로 *SD_IBM* 데이터 집합에 대한 실험 결과와 마찬가지로 단위 발생 간격이 작을수록 마이닝 결과로 얻어지는 순차패턴의 수가 감소된다. 한편, *SD_Web* 데이터 집합에 대한 감쇠기본값 $b$ 및 허용 발생 간격 $G_a$에 변화에 따른 순차패턴 수 변화 실험에서도 *SD_IBM* 데이터 집합에 대한 실험 결과와 유사한 결과를 확인할 수 있었다.

<표 3>은 *SD_Web* 데이터 집합에 대한 실험에서 해당 데이터 집합을 구성하는 1,000,000개의 순차정보가 모두 처리된 후에 구해진 마이닝 결과 집합에서 길이별 순차패턴의 개수 변화를 보여준다. 본 실험에서 감쇠기본값 $b$ 및 허용 발생 간격 $G_a$는 각각 0.9 및 5로 설정되었으며, 단위 발생 간

<표 3> SD_Web 데이터 집합에서 마이닝 결과의 길이별 순차패턴 개수[$Ga$ = 5, $b$ = 0.9]

| U | $L_1$ | $L_2$ | $L_3$ | $L_4$ | $L_5$ | $L_6$ | $L_7$ | $L_6$ 이상 | 계 |
|---|---|---|---|---|---|---|---|---|---|
| 미적용 | 90 | 387 | 605 | 492 | 249 | 56 | 1 | 0 | 1880 |
| 5 | 90 | 376 | 587 | 466 | 234 | 50 | 1 | 0 | 1804 |
| 10 | 90 | 378 | 590 | 467 | 235 | 50 | 1 | 0 | 1811 |
| 20 | 90 | 378 | 590 | 467 | 236 | 50 | 1 | 0 | 1812 |

격 $u$의 변화에 따른 순차패턴 수의 변화를 분석하였다. *SD_IBM* 데이터 집합에서와는 달리 $L_3$, $L_4$, $L_5$ 및 $L_6$ 등 비교적 길이가 긴 순차패턴에서도 발생 간격 기반 가중치를 적용하는 경우 결과 순차패턴의 수가 감소됨을 알 수 있다. 실제 응용 분야에서 생성된 *SD_Web* 데이터 집합은 단순 지지도 기반의 순차패턴 마이닝 수행 시 길이가 길고 큰 발생 간격을 갖는 순차패턴들이 마이닝 결과에 포함되기 때문이다. 하지만 발생 간격이 큰 이러한 순차패턴들은 상대적으로 낮은 중요성이나 관심도를 갖는 순차패턴으로서 굳이 마이닝 결과로 구해질 필요가 없는 것들이다. 따라서 발생 간격 기반 가중치를 적용하여 이들을 마이닝 결과집합에서 제외하는 가중치 순차패턴 마이닝을 통해 보다 효율적인 순차패턴 마이닝을 수행할 수 있다. <표 4>는 *SD_Web* 데이터 집합에 대한 실험에서 동일한 순차패턴에 대한 단순 지지도와 발생 간격 기반 가중치 지지도를 비교하였다. 표에서 순차패턴을 구성하는 숫자들은 개별 단위항목을 나타낸다. 본 실험에서 단위 발생 간격 $u$, 감쇠기본값 $b$ 및 허용 발생 간격 $G_a$는 각각 20, 0.9 및 5로 설정되었다. *SD_IBM* 데이터 집합에 대한 실험 결과에서와 마찬가지로 큰 발생 간격을 갖는 일부 순차패턴들은 발생 간격 기반 가중치를 적용하는 경우 지지도가 감소됨을 알 수 있다.

<표 4> SD_Web 데이터 집합에서 결과 패턴의 지지도 변화[$u$ = 20, $Ga$ = 5, $b$ = 0.9]

| 패턴 | 단순 지지도 | 가중치 적용 지지도 | 지지도 차이 |
|---|---|---|---|
| <1, 3, 40, 57> | 0.00541 | 0.00495 | 0.00046 |
| <1, 4, 40, 57> | 0.00518 | 0.00477 | 0.00040 |
| <1, 3, 4, 6, 57> | 0.00525 | 0.00485 | 0.00040 |
| <1, 4, 6, 48> | 0.00538 | 0.00500 | 0.00038 |
| <3, 4, 40, 48> | 0.00517 | 0.00482 | 0.00036 |
| <1, 3, 23, 57> | 0.00505 | 0.00471 | 0.00034 |
| <1, 115> | 0.00513 | 0.00481 | 0.00032 |
| <4, 5, 38> | 0.00501 | 0.00469 | 0.00031 |
| <3, 4, 5, 6, 40> | 0.00512 | 0.00482 | 0.00030 |
| <1, 3, 8, 57> | 0.00502 | 0.00473 | 0.00030 |

실제 응용 분야에서 생성된 *SD_Web* 데이터 집합에 대한 이상의 실험 결과에서 볼 때 발생 간격 기반 가중치 부여 기법 및 *WSGap* 방법은 실제 응용 분야에서 발생되는 데이터에도 유용하게 적용되어 상대적으로 작은 발생 간격을 갖는 관심도가 큰 가중치 순차패턴을 탐색할 수 있음을 알 수 있다.

## 7. 결 론

분석 대상이 되는 데이터 스트림이나 데이터 집합에 내재된 지식이나 정보를 찾는데 있어서 구성요소의 발생 순서까지 고려하는 순차패턴 마이닝은 이전의 한정적인 데이터 집합 뿐만 아니라 근래의 변화된 컴퓨터 응용 환경에서 생성되는 데이터 스트림의 형태의 정보를 분석하는데도 효율적으로 적용되고 있다. 하지만 기존의 단순 지지도 기반 순차패턴 마이닝은 관심도나 중요도가 작은 순차패턴들까지 포함된 마이닝 결과 집합을 구한다. 이를 보완하기 위한 순차패턴 마이닝 관련 연구들 중에서 주목받는 연구 분야 중의 하나인 가중치 순차패턴 마이닝은 구성요소의 발생 순서 뿐만 아니라 구성요소나 순차정보의 가중치를 산정할 수 있는 부가적인 정보를 활용하여 보다 관심도가 큰 순차패턴을 마이닝 결과로 제공함으로써 탐색된 순차패턴의 활용도를 높일 수 있다.

이러한 연구 흐름을 고려하여 본 논문에서는 발생 간격 기반 가중치 부여 기법 및 이를 활용한 데이터 스트림에서의가중치 순차패턴 탐색 방법인 *WSGap* 방법을 제안하였다. 하나의 순차패턴에 있어서 발생 간격은 분석 대상이 되는 데이터 스트림의 특성이나 사용자의 관심도에 따라 다양한

기준으로 정의될 수 있다. 본 논문에서 제안한 발생 간격 기반 가중치 부여 기법에서는 하나의 순차패턴을 구성하는 여러 단위 항목들에 대해서 인접한 두 단위항목의 순차정보에서의 발생순서 차이를 발생 간격으로 정의하고, 이를 활용하여 발생 간격 기반 가중치를 부여한다. 먼저 가중치 함수를 이용하여 하나의 순차패턴에 존재하는 각 발생 간격에 대한 가중치를 구하고 이로부터 해당 순차패턴의 가중치를 구한다. 이어서 분석 대상이 되는 하나의 데이터 스트림에서 각 순차패턴의 가중치를 총합하여 해당 순차패턴의 지지도를 구하고, 이로부터 해당 순차패턴이 빈발 순차패턴이지를 판단한다. 한편, 일련의 실험 결과로부터 논문에서 제안한 방법이 데이터 스트림에 대한 가중치 순차패턴 탐색에 효과적으로 적용될 수 있으며 발생 간격을 고려하여 보다 정제된 형태의 순차패턴들을 마이닝 결과로 구해줌을 확인하였다. 특히, 실제 응용 분야에서 발생된 데이터 집합에 대해서도 효과적으로 적용될 수 있음을 확인하였다.

웹 기반 시스템, 전자상거래, 생물정보학 및 USN 환경 등 근래의 다양한 컴퓨터 응용 환경에서는 순차 데이터 스트림 형태로 정보를 발생시키고 있다. 따라서 본 논문에서 제안한 발생 간격 기반 가중치 부여 기법 및 이를 활용한 데이터 스트림에 대한 가중치 순차패턴 탐색 방법은 이들 응용 분야에서 유용하게 활용될 수 있으며, 특히 관심도 큰 순차패턴을 얻는데 도움이 될 것이다.

## 참고문헌


Agrawal, R. and R. Srikant., "Mining Sequential Patterns", *Proc. of the 1995 Int'l Conf. on Data Engineering*, (1995), 3~14.

Chang, J. H. and W. S. Lee., "Efficient Mining Method for Retrieving Sequential Patterns over Online Data Streams", *Journal of Information Science*, Vol.31, No.5(2005), 420~432.

Chen, Y. L. and T. C.-H. Huang., "Discovering Time-Interval Sequential Patterns in Sequence Databases", *Expert Systems with Applications*, Vol.25, No.1(2003), 343~354.

Chen, Y. L., 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*, Vol.35, No.5(2005), 959~972.

Garofalakis, M., 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. onVery Large Data Bases, (2002).

Huang, Q. and W. Ouyang., "Mining Sequential Patterns in Data Streams", *Proc. of the 6th Int'l Symposium on Neural Networks*, (2009), 865~874.

Ji, X., J. Bailey. and G. Dong., "Mining Minimal Distinguishing Subsequence Patterns with Gap Constraints", *Knowledge and Information Systems*, Vol.11, No.3(2007), 259~296.

Kum, H. C., J. Pei., W. Wang. and D. Duncan., "ApproxMAP : Approximate Mining of Consensus Sequential Patterns", *Proc. of the 2003 SIAM Int'l Conf. on Data Mining(SDM '03)*, 311~315, (2003).

Kum, H. C., J. H. Chang. and W. Wang., "Sequential Pattern Mining in Multi- Databases via Multiple Alignment", *Data Mining and Knowledge Discovery*, Vol.12, No.2(2006),

151~180.

Lin, M. Y., S. C. Hsueh. and C. W. Chang., "Fast Discovery of Sequential Patterns in Large Databases using Effective Time-Indexing", *Information Sciences*, Vol.178, No.22(2008), 4228~4245.

Lo, S., "Binary Prediction based on Weighted Sequential mining method", *Proc. of the (2005) Int'l Conf. on Web Intelligence*, pp. 755~761, 2005.

Luo, C. 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)*, 64~72, (2005).

Pei, J., 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)*, (2002), 18~25.

Pei, J., J. Han., B. Mortazavi-Asl., J. Wang., H. Pinto., Q. Chen., U. Dayal. and M. C. Hsu., "Mining Sequential Patterns by Pattern-Growth: The PrefixSpan Approach", *IEEE Transactions on Knowledge and Data Engineering*, Vol.16, No.11(2004), 1424~1440.

Wang, J., J. Han. and C. Li., "Frequent Closed Sequence Mining without Candidate Maintenance", *IEEE Transactions on Knowledge and Data Engineering*, Vol.19, No. 8(2007), 1042~1056.

Yun, U., "A New Framework for Detecting Weighted Sequential Patterns in Large Sequence Databases", *Knowledge-Based Systems*, Vol.21, No.2(2008), 110~122.

Abstract

# Finding Weighted Sequential Patterns over Data Streams via a Gap-based Weighting Approach

Joong Hyuk Chang*

Sequential pattern mining aims to discover interesting sequential patterns in a sequence database, and it is one of the essential data mining tasks widely used in various application fields such as Web access pattern analysis, customer purchase pattern analysis, and DNA sequence analysis. In general sequential pattern mining, only the generation order of data element in a sequence is considered, so that it can easily find simple sequential patterns, but has a limit to find more interesting sequential patterns being widely used in real world applications. One of the essential research topics to compensate the limit is a topic of weighted sequential pattern mining. In weighted sequential pattern mining, not only the generation order of data element but also its weight is considered to get more interesting sequential patterns.

In recent, data has been increasingly taking the form of continuous data streams rather than finite stored data sets in various application fields, the database research community has begun focusing its attention on processing over data streams. The data stream is a massive unbounded sequence of data elements continuously generated at a rapid rate. In data stream processing, each data element should be examined at most once to analyze the data stream, and the memory usage for data stream analysis should be restricted finitely although new data elements are continuously generated in a data stream. Moreover, newly generated data elements should be processed as fast as possible to produce the up-to-date analysis result of a data stream, so that it can be instantly utilized upon request. To satisfy these requirements, data stream processing sacrifices the correctness of its analysis result by allowing some error.

Considering the changes in the form of data generated in real world application fields, many researches have been actively performed to find various kinds of knowledgeembedded in data streams. They mainly focus on efficient mining of frequent itemsets and sequential patterns over data streams, which have been proven to be useful in conventional data mining for a finite data set. In addition, mining algorithms have also been proposed to efficiently reflect the changes of data streams over time

* Department of Computer and Information Technology, Daegu University

into their mining results. However, they have been targeting on finding naively interesting patterns such as frequent patterns and simple sequential patterns, which are found intuitively, taking no interest in mining novel interesting patterns that express the characteristics of target data streams better. Therefore, it can be a valuable research topic in the field of mining data streams to define novel interesting patterns and develop a mining method finding the novel patterns, which will be effectively used to analyze recent data streams.

This paper proposes a gap-based weighting approach for a sequential pattern and amining method of weighted sequential patterns over sequence data streams via the weighting approach. A gap-based weight of a sequential pattern can be computed from the gaps of data elements in the sequential pattern without any pre-defined weight information. That is, in the approach, the gaps of data elements in each sequential pattern as well as their generation orders are used to get the weight of the sequential pattern, therefore it can help to get more interesting and useful sequential patterns. Recently most of computer application fields generate data as a form of data streams rather than a finite data set. Considering the change of data, the proposed method is mainly focus on sequence data streams.

**Key Words** : Weighted Sequential Pattern, Gap-based Weight, Sequential Pattern, Sequence Data Stream, Data Mining

# 저 자 소 개

**장중혁**

연세대학교에서 컴퓨터과학 전공으로 이학사(1996), 공학석사(1998) 및 공학박사(2005)를 취득하였다. UIUC 박사 후 연구원을 역임하였으며, 2008년 9월부터 대구대학교 컴퓨터IT공학부 교수로 재직하고 있다. 주요 연구관심 분야로는 데이터 스트림, 데이터마이닝, 데이터베이스, 지능형 웹 서비스, USN 환경의 데이터 처리 및 생물정보학 등이 있다.