DOI: 10.3745/KIPSTD.2010.17D.4.253

Prefix-트리를 이용한 동적 가중치 빈발 패턴 탐색 기법

정 병 수† · Ahmed Farhan††

요 약

지금까지의 빈발 패턴(Frequent Pattern) 마이닝에서는 각 항목들의 중요도(Weight)는 모든 같은 값으로 다루어 왔으나 실 환경에서는 각 항목들의 중요도가 다르게 적용되는 경우가 많이 있고 또 같은 항목이라도 시간에 따라 다른 중요도 값으로 다루어져야 할 경우가 있다. 비즈니스 데이터 분석 환경이나 웹 클릭 데이터 분석 환경과 같은 응용에서도 동적으로 변하는 중요도를 고려하여야 한다. 지금까지 항목의 중요도를 고려하는 여러 패턴 마이닝 기법들이 제안되고 있으나 동적으로 변하는 항목의 중요도를 고려하는 연구는 발표되지 않고 있다. 본 논문에서는 처음으로 동적인 항목들의 중요도(혹은 가중치)를 고려하는 빈발 패턴 마이닝 알고리즘을 제안한다. 제안하는 기법은 단 한번의 데이터베이스 스캔으로 처리되므로 스트림 데이터를 분석할 수 있다. 여러 실험을 통하여 제안하는 기법은 매우 효과적이며 확장성이 좋은 것임을 보인다.

키워드: 데이터 마이닝, 지식 탐사, 가중치 빈발 패턴, 동적 가중치

Efficient Dynamic Weighted Frequent Pattern Mining by using a Prefix-Tree

Jeong, Byeong-Soo† · Ahmed Farhan††

ABSTRACT

Traditional frequent pattern mining considers equal profit/weight value of every item. Weighted Frequent Pattern (WFP) mining becomes an important research issue in data mining and knowledge discovery by considering different weights for different items. Existing algorithms in this area are based on fixed weight. But in our real world scenarios the price/weight/importance of a pattern may vary frequently due to some unavoidable situations. Tracking these dynamic changes is very necessary in different application area such as retail market basket data analysis and web click stream management. In this paper, we propose a novel concept of dynamic weight and an algorithm DWFPM (dynamic weighted frequent pattern mining). Our algorithm can handle the situation where price/weight of a pattern may vary dynamically. It scans the database exactly once and also eligible for real time data processing. To our knowledge, this is the first research work to mine weighted frequent patterns using dynamic weights. Extensive performance analyses show that our algorithm is very efficient and scalable for WFP mining using dynamic weights.

Keywords: Data Mining, Knowledge Discovery, Weighted Frequent Pattern Mining, Dynamic Weight

1. 서 론

빈발 패턴(Frequent Pattern) 마이닝은 연관 규칙을 발견하거나 데이터간의 관련성을 파악하는 기본적인 마이닝 기법의 하나이다. 빈발 패턴을 탐색하는 초기 방법으로 *Apriori*-규칙을 이용한 *Apriori* 알고리즘[1, 2]이 제안되었으나 불필요한 후보 패턴을 많이 생성하는 단점으로 인하여 잘 활용되지 못하였다. 그 후 FP-트리를 이용한 FP-Growth[9] 알고리즘은 이러한 문제를 극복하여 많은 성능적 개선을 가져오게 되었다. 그러나 이러한 방법[9-12, 13]들은 각 항목들의 중요도(Weight)를 모든 같은 값으로 가정한 것들이어서 각 항목들의 중요도가 다르게 고려되어야 하는 실 환경에서는 활용되기 어려운 문제점을 갖게 된다.

가중치 패턴 마이닝(Weighted Pattern Mining)[3-8]은 항목들이 다른 중요도(혹은 가중치)를 가질 경우 높은 가중치 패턴을 찾아내는 마이닝 기법을 의미한다. 비즈니스 데이터 분석 환경을 예를 들면 상품에 대한 고객들의 구매 패턴은 시간에 따라 다르게 설정되어야 하며, 상품 가격 등과 같이 항목 마다 다른 가중치를 설정하여야 하는 경우가 일반적이기 때문에 실제적으로 모든 항목에 대하여 같은 가중치를 부여하는 것은 매우 현실적이지 못한 가정이다. 이외에도 웹 분석 환경에서 처리하는 각 웹 페이지는 다른 중요도를

† 종신회원 : 경희대학교 컴퓨터공학과 교수
†† 준 회 원 : 경희대학교 컴퓨터공학과 박사과정
논문접수 : 2010년 7월 2일
수 정 일 : 1차 2010년 7월 28일
심사완료 : 2010년 8월 5일

가질 수 있고, 유전자 정보들도 각각 형질 발현에 미치는 비중이 다르기 때문에 항목들에 대한 동적인 가중치를 고려하는 가중치 패턴 마이닝은 실제 응용에서 중요한 역할을 할 수 있다.

이러한 배경으로 본 논문에서는 항목들의 동적인 중요도를 고려할 수 있는 가중치 패턴 마이닝을 위한 새로운 알고리즘 DWFPM(Dynamic Weighted Frequent Pattern Mining)을 제안한다. 제안하는 기법은 동적으로 변하는 중요도를 고려할 수 있는 가중치 패턴 마이닝 기법으로 단 한 번의 데이터베이스 스캔을 통하여 처리되므로 스트림 데이터 환경[13-16]과 같은 실시간 처리가 필요한 환경에서도 잘 적용될 수 있다. 또한 우리가 아는 한에서는 제안하는 기법은 동적인 가중치를 고려하는 첫 번째 알고리즘이 된다.

본 논문의 구성은 2장에서 배경이 되는 기본 지식을 서술하고 3장에서는 제안하는 (Dynamic Weighted Frequent Pattern Mining)에 대하여 설명한다. 4장에서는 실험 데이터를 분석한 내용을 기술하고 5장에서 결론을 맺는다.

2. 연구 배경

2.1 빈발 패턴 마이닝(Frequent Pattern Mining)

집합 I (= $\{i_1, i_2, \cdots i_m\}$)를 항목들의 집합, 집합D (= $\{T_1, T_2, \cdots T_n\}$)를 트랜잭션으로 구성된 데이터베이스라 하고, 트랜잭션 $T_i \in D$는 I의 부분 집합으로 구성된다고 하자. 이때 패턴의 빈도수는 데이터베이스에서 패턴 X (= $\{x_1, x_2, \cdots x_n\}$)를 포함하고 있는 트랜잭션의 개수를 말한다. 빈발 패턴 마이닝은 트랜잭션 데이터베이스에 나타난 여러 패턴들 중에서 빈도수가 주어진 임계값보다 크거나 같은 패턴을 의미한다. 빈발 패턴 마이닝이란 이 조건을 만족하는 모든 패턴을 찾는 문제이다. 만일 어떤 패턴 p가 빈발하지 않은 패턴이면 p의 모든 초월집합(Super Set)은 빈발하지 않은 패턴이 된다. 이를 Anti-monotone 성질[1, 2]이라고 한다. 빈발 패턴 마이닝 알고리즘으로 초기에 *Apriori* [1, 2] 알고리즘이 제안되었으나 불필요한 후보 패턴을 많이 생성하는 단점으로 인하여 잘 활용되지 못하였다. 그 후 FP-트리를 이용한 FP-Growth[9] 알고리즘은 이러한 문제를 극복하여 많은 성능적 개선을 가져오게 되었다. 최근에는 스트림 데이터를 처리하기 위하여 데이터베이스를 한번만 스캔하여 빈발 패턴을 찾아내는 알고리즘[14, 18]도 제안되고 있다. 그러나 이러한 알고리즘들은 모두 항목들간의 중요도(가중치)를 모두 같은 값으로 가정한 것들이다.

2.2 가중치 빈발 패턴 마이닝(Weighted Frequent Pattern Mining)

항목에 대한 가중치(Weight)는 트랜잭션 데이터베이스에서 항목의 중요성을 나타내는 지표로 항목 집합 I = $\{i_1, i_2, \cdots i_n\}$에 대하여 패턴 P$\{x_1, x_2, \cdots x_m\}$의 가중치 Weight(P)는 식(1)과 같이 정의한다:

$$\text{Weight(P)} = \frac{\sum_{i=1}^{length(P)} Weight(x_i)}{length(P)} \quad (1)$$

패턴 P의 가중치 지지도(Weighted Support)는 식(2)와 같이 패턴 P의 가중치 값과 지지도를 곱한 값이다:

$$Wsupport(P) = Weight(P) * Support(P) \quad (2)$$

Wsupport(P) 값이 정해진 최소 임계값(Minimum Threshhold)보다 크거나 같을 때 패턴 P를 가중치 빈발 패턴이라 한다.

<표 1>은 전자 매장에서의 판매 데이터로 지지도와 가중치의 한 예를 보여준다. 가중치 값으로 상품의 실제 가격을 사용하는 것보다 정규화를 통하여 (0~1)의 값을 갖는 정규화된 가중치 값을 사용하는 것이 편리할 수 있다. 가중치 빈발 패턴 마이닝이란 대량의 판매 데이터에서 정해진 최소 매출(즉 임계값) 이상의 판매를 보인 모든 상품들의 집합(즉 패턴, 항목 집합)을 찾아내는 작업으로 생각할 수 있다. 가중치 빈발 패턴 마이닝 알고리즘으로 초기에 발표된 방법은 *Apriori* 알고리즘을 기반으로하는 MINWAL [6], WARM [7], WAR [8] 등이 있으나 여러 번의 데이터베이스 스캔을 필요로 하여 속도가 느려지는 단점을 가지고 있었다.

이러한 성능적 문제를 해결한 방법으로 FP-트리를 기반으로 하는 WFIM[3]이 제안되었다. WFIM에서는 항목들의 최소 가중치와 가중치 범위를 정하고 FP-트리를 가중치 오름 차순으로 구성하여 FP-트리가 하향 닫힘(Downward Closure) 성질을 만족하도록 하고 있다. 같은 저자에 의하여 제안된 WIP[6] 알고리즘에서는 가중치 패턴의 성질을 Weight Affinity 개념을 이용하여 정의하고 Weight Affinity 값이 큰 패턴을 유용한 패턴으로 하여 탐색하고 있다. WFIM과 WIP는 FP-Growth 방식을 기반으로 하는 알고리즘들로 FP-Growth 알고리즘과 같이 두 번의 데이터베이스 스캔을 요구하게 되므로 한번의 처리 기회만을 갖는 스트림 데이터의 처리에는 사용할 수 없는 제약이 있다. Wcloset[17]는 같은 방식으로 닫힌 가중치 빈발 패턴(Closed Weighted Frequent Patterns)을 찾는 알고리즘이다.

WFIM과 WIP는 가중치 빈발 패턴을 탐색하는 과정에서는 빈발 패턴 마이닝에서 적용하던 Anti-monotone 성질을 가중치 빈도수에는 적용할 수 없는 것이 중요한 문제점으로

〈표 1〉 전자 매장에서의 판매 데이터 예

Bar Code	Item	Price	Support (frequency)	Normalized Weight
1	Personal computer	800$	500	0.8
2	Laser printer	450$	320	0.45
3	Bubble jet printer	250$	450	0.25
4	Digital Camera	600$	700	0.6
5	Memory stick	200$	825	0.2
6	Hard disk	130$	350	0.13
7	DVD drive	100$	450	0.1
8	CD drive	50$	250	0.05

지적하고 있다. 예를 들면 항목 “a”의 가중치가 0.6 빈도수가 4이고 항목 “b”의 가중치와 빈도수가 각각 0.2, 5일 때 만약 항목집합 “ab”의 빈도수가 3이라면 공식(1)과 공식(2)에 따르면 “ab”의 가중치는 (0.6 + 0.2)/2 = 0.4가 되고 가중치 빈도수는 0.4 × 3 = 1.2가 된다. 그리고 항목 “a”와 “b”의 가중치 빈도수는 각각 0.6 × 4 = 2.4 과 0.2 × 5 = 1.0이 된다. 가중치 빈발 패턴에서 가중치 빈도수의 임계값이 1.2라면 항목 ‘b”는 가중치 빈발 패턴이 아닌 반면 항목집합 “ab”는 가중치 빈발 패턴이 된다. 이는 Anti-monotone 성질에 위배되는 것 이다. WFIM와 WIP에서는 전역적 최대 가중치(Global Maximum Weight)를 설정하여 트리 탐색 과정에서 Anti-monotone 성질이 가능하도록 하고 있다. 즉 위의 예에서 항목 “a”의 가중치 0.6을 최대 가중치로 설정하고 이 값을 이용하여 모든 항목집합에 대한 가중치 빈도수를 구하면 “b”의 가중치 빈도수는 3.0이 되고 따라서 패턴 “ab”가 초기에 제거되는 경우가 발생하지 않게 된다. 그리고 과대 계상된 “b”의 가중치 빈도수는 “b”의 실제 가중치를 이용하여 나중에 제거하게 된다.

지금까지의 모든 가중치 빈발 패턴 마이닝 알고리즘은 항목들의 가중치가 고정된 값을 가지는 경우만을 다루었고 항목들의 가중치 값이 시간에 따라 변하는 환경은 고려하지 않았다. 본 논문에서는 동적인 가중치 값을 갖는 환경에서 가중치 빈발 패턴을 찾는 마이닝 알고리즘을 제안한다. 제안하는 기법은 단 한번의 데이터베이스 스캔을 통하여 처리되므로 스트림 데이터 환경[13-16]과 같이 실시간 처리가 필요한 환경에서도 잘 적용될 수 있다. 또한 제안하는 기법은 우리가 아는 바로는 동적인 가중치를 고려하는 첫 번째 알고리즘이 된다.

3. 동적 가중치 빈발 패턴 마이닝(DWFPM)

3.1 정의

[정의 1] 패턴 P에 대한 동적 가중치 지지도(Dynamic Weighted Support)는 다음과 같이 정의된다.

$$DWsupport(P) = \sum_{i=1}^{N} Weight_i(P) \times Support_i(P) \quad (3)$$ ■

여기서 N은 배치(Batch)의 개수를 말하고 Weight(P) 와 Support(P)는 공식 (1)과 (2)에서 얻어진다. 예를 들면 <표 2>에서 첫 번째 배치의 패턴 “bd”의 DWsupport(bd)는 ((0.9 +0.3) / 2) × 1 = 0.6 이고 두 번째, 세 번째 배치의 DWsupport(bd)는각각 ((0.7 +0.5) / 2) × 1 = 0.6 와 ((0.3 +0.4) / 2) × 0 = 0가 된다. 따라서 전체 DWsupport(bd) = 0.6 + 0.6 + 0 = 1.2가 된다.

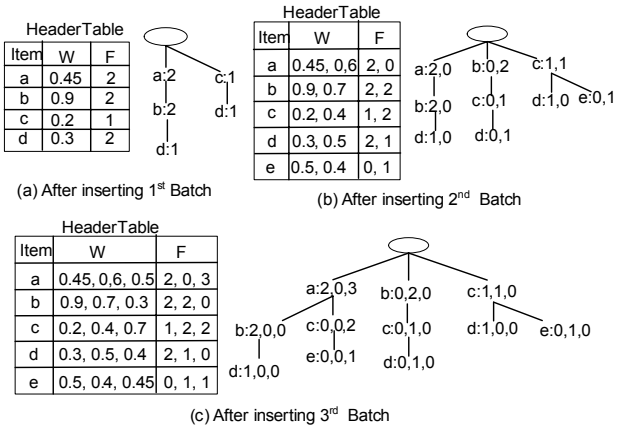
[정의 2] 패턴 p의 DWsupport(p) 값이 주어진 최소 임계값보다 크거나 같을 때 패턴 p를 동적 가중치 빈발 패턴이라 한다. 만일 최소 임계값이 1.2라면 표 2.에서 패턴 “bd”는 동적 가중치 빈발 패턴이다. ■

〈표 2〉 동적 가중치를 갖는 트랜잭션 데이터베이스의 예

Batch	TID	Trans.	Weight				
			a	b	c	d	e
1st	T_1 T_2 T_3	a, b, d c, d a, b	0.45	0.9	0.2	0.3	0.5
2nd	T_4 T_5 T_6	b b, c, d c, e	0.6	0.7	0.4	0.5	0.4
3rd	T_7 T_8 T_9	a, c, e a a, c	0.5	0.3	0.7	0.4	0.45

3.2 트리의 생성

이 절에서는 동적 가중치를 갖는 항목들로 이루어진 트랜잭션들의 내용을 저장하는 Prefix-트리의 구조 및 생성 방법을 설명한다. Prefix-트리는 FP-트리[9]에서와 같은 헤더 테이블을 가지며 테이블 내에는 항목 id, 빈도수(Frequency)와 더불어 항목들의 가중치 값도 함께 저장한다. 이러한 값들은 각 배치(Batch)별로 구분하여 헤더 테이블의 필드 및 트리의 노드에 저장된다. 트리의 구성 방법은 FP-트리의 구성 방법과 같고 (그림 1)은 <표 2>의 데이터베이스를 이용하여 구성한 트리의 모습을 보여준다.



(그림 1) Prefix-트리의 구성 예

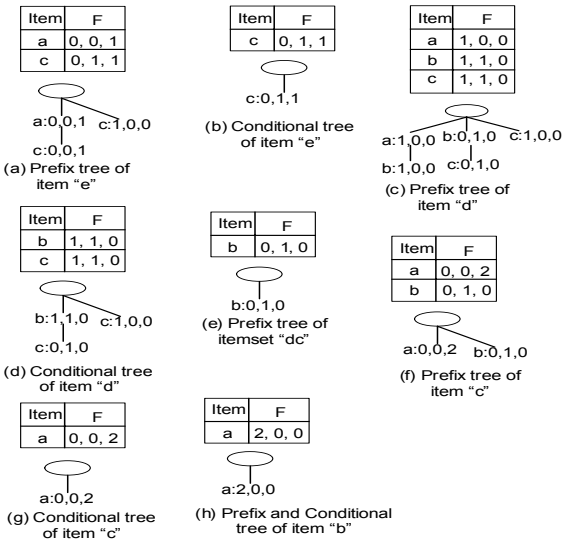
(그림 1)에서 알 수 있듯이 Prefix-트리는 트랜잭션들이 삽입 삭제되는 과정에서 항목들의 순서는 변하지 않으며, 상위 노드의 빈도수 값은 항상 모든 자식 노드들의 빈도수 값을 더한 값보다 크거나 같게 된다. 또 위 Prefix-트리는 단 한번의 데이터베이스 스캔을 통하여 만들어 질 수 있다.

3.3 마이닝 과정

제안하는 DWFPM 알고리즘은 FP-Growth 알고리즘처럼 트리의 상향 탐색을 통하여 수행된다. 2.2절에서 언급하였듯이 항목집합의 가중치 빈도수는 Anti-monotone 성질을 갖지 않기 때문에 이 성질을 유지하기 위하여 전역적 최대 가중치인 GMAXW 값을 활용한다. GMAXW는 데이터베이스의 모든 항목들 중 최대 가중치 값을 갖는 항목의 가중치 값으로 설정된다. <표 2>에서 보면 항목 “b”의 가중치 0.9

가 GMAX가 된다. <표 2>의 데이터를 살펴 보면 항목 “e”는 항목 “a”와 “c” 이외는 다른 항목들과 같이 트랜잭션에 나타나지 않고 있다. 이 경우 항목 “e”에 대한 마이닝 작업을 할 때 GMAXW 값 대신 국지적인 최대값인 “c”의 가중치(세 번째 배치에서) 0.7을 사용하여도 Anti-monotone 성질을 만족할 수 있다. 이와 같이 불필요한 항목 노드들을 초기에 가지치기(Prune) 하기 위하여 국지적 최대 가중치 LMAXW를 활용한다. 즉 GMAXW보다 항상 같거나 작은 값을 갖는 LMAXW를 사용하면 더 많은 불필요한 항목들이 초기에 제거될 수 있게 된다.

<표 2>의 데이터베이스와 이를 이용하여 구성한 (그림 1)의 트리를 가지고 최소 임계 값이 1.2일 경우 마이닝하는 과정을 살펴 보면 우선 GMAXW는 0.9가 되고 이를 이용하여 계산한 가중치 빈도수는 <a:4.5, b:3.6, c:4.5, d:2.7, e:1.8>가 된다. 즉 모든 단일 항목들은 가중치 빈발 패턴이 될 수 있는 가능성이 있고, 각 항목에 대하여 트리의 상향 탐색을 거쳐 동적 가중치 빈발 패턴을 찾게 된다. 먼저 헤더 테이블의 가장 밑에 있는 항목 “e”에 대한 Prefix-트리를 구성하기 위해서는 (그림 1)의 Prefix-트리에서 항목 “e”를 포함하는 가지들을 빈도수를 고려하여 추출하면 된다(그림 2)(a). 그리고 불필요한 항목을 제거하기 위하여 “e”의 LMAXW = 0.7을 이용하여 동적 가중치 빈도수를 구하면 <a:0.7, c:1.4>되는 데 여기서 임계 값 1.2보다 작은 항목 “a”(0.7)는 Conditional Prefix-트리에서 제거된다(그림 2)(b). 이 과정에서 “e”와 “ce”가 후보 패턴으로 추출된다. 같은 방법으로 항목 “d”에 대한 Prefix-트리와 Conditional Prefix-트리를 구성하면 각각 (그림 2)(c)와 (그림 2)(d)와 같고 Pattern-Growth 방식에 따라 “bd”, “cd”, “d”가 후보 패턴으로 인식된다. 계속해서 “dc”의 Prefix-트리는 (그림 2)(e)와 같이 구성되고 “b”의 동적 가중치 빈도수가 0.9이므로 Conditional 트리가 더 이상 생성되지 않는다. 그림 2(f)는 항목 “c”의 Prefix-트리이고 동적 가중치 빈도수는 <a:1.8, b:0.9>가 되며 이때



(그림 2) 마이닝 과정

GMAXW 대신 LMAXW = 0.9가 사용된다. 이와 같은 방식으로 모든 후보 패턴들을 찾으면 <표 3>의 맨 왼쪽 행과 같고 실제의 가중치를 이용하여 실제 가중치 빈도수를 계산하여 임계 값과 비교하면 <표 3>의 맨 오른쪽 행과 같은 결과가 나오게 된다.

〈표 3〉 동적 가중치를 이용한 가중치 빈발 패턴

Candidate Patterns	*DWsupport calculation*	Result
(1) c e: 0,1,1	(((0.4+0.4)/2)×1) + (((0.7+0.45)/2)×1) = 0.4+0.575 = 0.975	Pruned
(2) e: 0,1,1	0.4×1+0.45×1 = 0.85	Pruned
(3) c d: 1,1,0	(((0.2+0.3)/2)×1) + (((0.4+0.5)/2)×1) = 0.25+0.45 = 0.7	Pruned
(4) b d: 1,1,0	(((0.9+0.3)/2)×1) + (((0.7+0.5)/2)×1) = 0.6+0.6 = 1.2	Pass
(5) d: 2,1,0	0.3×2+0.5×1 = 1.1	Pruned
(6) a c: 0,0,2	((0.5+0.7)/2))×2 = 1.2	Pass
(7) c: 1,2,2	0.2×1+0.4×2+0.7×2 = 2.4	Pass
(8) a b: 2,0,0	((0.9+0.45)/2))×2 = 1.35	Pass
(9) b: 2,2,0	0.9×2+0.7×2 = 3.2	Pass
(10) a: 2,0,3	0.45×2+0.5×3 = 2.4	Pass

위에서 설명한 마이닝 과정을 알고리즘으로 기술하면 아래와 같다.

Input: DB contains batches of transactions with dynamic weights, minimum threshold (δ)
Output: Dynamic weighted frequent patterns

```
Begin
  For each transaction Ti in DB
  Sort the items inside Ti according to lexicographical order
      Insert Ti into Tree
      Update the header table H
  End For
  GMAXW is the global maximum weight
  For each item ai from the bottom of H
    If (total_frequency(ai)×GMAX W) > δ then
        Create Prefix tree PTi for item ai
      Calculate LMAXW
      Call Mining ( PTi, ai, LMAXW)
    End If
  End For
End
Procedure Mining (T, a, LMAXW)
Begin
  Create conditional tree CDT of a by deleting each item di
  from T having total_frequency(di)×LMAX W < δ
  For each item βi in the header table of CDT
    Call Test_Candidate(aβi)
    Create Prefix tree Tβ for itemset aβi
    Calculate LMAXW
    Call Mining ( Tβ, aβi, LMAXW)
  End For
End
Procedure Test_Candidate (X )
Begin
  Let Dynamic weighted support of X is DWX
  Set DWX = 0
  For each batch Bi
    DWX = DWX + (frequency(XBi)×Weight(XBi))
  End For
  If DWX ≥ δ then
    Add X in the Dynamic weighted frequent pattern list
  End If
End
```

4. 실험 결과 및 분석

4.1 실험 환경 및 데이터 집합

제안된 트리 구조 및 알고리즘의 성능적 특징을 분석하기

위하여 <표 4>와 같은 데이터 집합을 사용하였다. <표 4>의 데이터 집합, mushroom, chess, kosarak들은 빈발 패턴 탐색 알고리즘의 성능을 실험하기 위하여 사용되는 실 환경의 데이터 집합이다. T10I4D100k평균 트랜잭션의 길이가 10, 항목의 개수가 1,000, 트랜잭션 수를 100,000로 하여 실험을 목적으로 생성한 데이터 집합이다. 그러나 해당 데이터에는 가중치 값이 나와 있지 않기 때문에 확률 변수를 이용하여 0.1~0.9까지의 값을 생성 추가하였다. 실험은 Window XP(Pentium Dual Core 2.13 GHz CPU, 1GB 메모리)하에서 Microsoft Visual C++ 6.0으로 구현하여 수행되었다.

〈표 4〉 데이터 집합의 특징

Datasets	Size (MB)	No. of Trans.	Distinct items	Avg. Trans. Len.
mushroom	0.56	8124	119	23
chess	0.34	3196	75	37
T10I4D100K	3.83	100000	942	10.2
kosarak	30.5	990002	41270	8.1

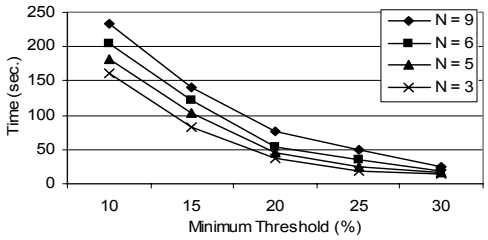
4.2 성능 분석

데이터 집합 Mushroom은 중간 정도의 조밀도(Dense)를 갖는 데이터 집합이다. 전체 항목들 중에서 20%정도의 항목들이 모든 트랜잭션에 나타나는 분포 특성을 지니고 있다. 그리고 데이터 집합 Chess는 매우 조밀한 분포 특성을 갖는 데이터 집합으로 전체 항목들 중에서 50%정도의 항목들이 모든 트랜잭션에 포함 된다. 따라서 가중치 빈발 패턴의 길이도 매우 길게 되고 임계 값에 따라 추출되는 가중치 빈발 패턴의 개수도 많아지게 된다. 실험은 데이터 집합을 여러 개의 배치(Batch)(N)로 나누어 알고리즘을 수행하고 임계 값에 따른 수행 속도를 배치의 개수(N)에 따라 구분하여 측정하였다.

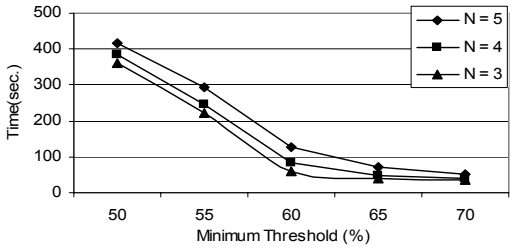
데이터 집합 T10I4D100K와 Kosarak은 항목의 수가 많고 트랜잭션의 길이가 상대적으로 짧아 조밀하지 않은 특성을 보이는 데이터 집합이다. 따라서 임계 값에 따라 추출되는 가중치 빈발 패턴의 개수가 그리 많지 않게 된다. 각각의 데이터 집합에 대한 마이닝 속도는 그림 3~그림 6에 나타난다. 실험에 사용한 임계 값은 데이터베이스 전체 가중치 빈도수와의 비율(%)로 하고 마이닝의 속도 변화를 잘 나타내는 구간을 실험 대상으로 적용하였다. 데이터의 특성에 따라 조밀한 데이터 집합은 임계 값이 상대적으로 큰 구간에서 속도 변화를 보이고 있으며, 조밀하지 않은 데이터 집합은 임계 값의 작은 변화에도 많은 속도 차이를 보이고 있다. 그 이유는 임계 값에 따른 후보 패턴의 수와 밀접한 관련이 있을 것으로 보인다.

다른 데이터 집합과는 달리 아주 조밀하지 않은 Kosarak 데이터 집합에서는 (그림 6)과 같이 배치의 개수(N)에 따라서도 속도 차이를 보이고 있는 데, 그 이유는 각 배치에 따라 다른 가중치 값이 항목에 부여되어 전체 데이터베이스의 가중치 빈도수 합을 낮추는 효과를 나타내고 이에 따라 후보 패턴의 수가 변화하기 때문인 것으로 보인다. 본 실험에서는 제안하는 기법이 동적 가중치를 다루는 첫 번째 연구이므로 다른 알고리즘과의 비교보다 데이터 집합의 특성에 따른 성능적 특징을 주로 분석하였다. 제안하는 기법은 한 번의 데이터베이스 스캔으로 마이닝 작업을 하므로 두 번의 스캔을 필요로 하는 FP-Growth를 기반으로 하는 알고리즘보다 자연히 좋은 성능을 보일 수 있으며(이에 대한 분석은 [18]에 잘 나타나 있다.), 또한 한 번의 데이터 처리만을 요구하는 스트림 데이터 처리 환경에서도 잘 적용될 수 있을 것이다.

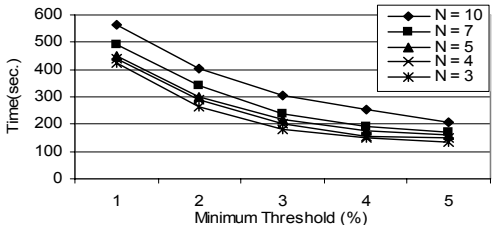
또한 데이터 집합의 크기가 매우 크고 항목들의 수가 많아짐에 따라 메모리에 저장되는 Prefix-트리의 크기가 염려될 수 있으나, 위의 데이터 집합을 사용했을 경우 Mushroom은 0.615MB, Chess는 0.581MB, T10I4D100K는 14.37MB, 그리고 Kosarak은 201.14MB 정도로 GB급의 현재 메모리 용량에 비추어 그리 문제가 되지 않을 것으로 보인다.



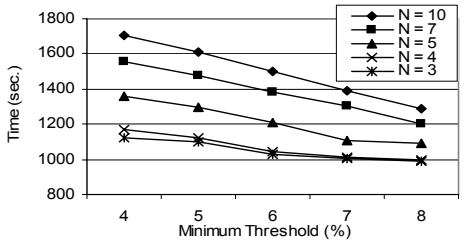
(그림 3) Mushroom의 마이닝 속도



(그림 4) Chess의 마이닝 속도



(그림 5) T10I4D100K의 마이닝 속도



(그림 6) Kosarak의 마이닝 속도

5. 결 론

본 논문의 주요 공헌은 가중치 빈발 패턴 마이닝 환경에서 동적 가중치의 개념을 처음으로 도입하여 이를 적용하는 마이닝 알고리즘을 개발한 것이다. 제안한 DWFPM 알고리즘은 배치 별로 항목들의 가중치와 빈도수를 저장하여 정확한 동적 가중치 빈발 패턴을 찾아 낼 수 있었으며 데이터 집합의 특성에 따라 조밀한 데이터 집합에 대해서는 배치의 수에 관계없이 일정한 수행 속도를 보인 반면 아주 조밀하지 않은 데이터 집합에 대해서는 배치의 수에 따라 비교적 큰 속도 차이를 보이고 있었다. 이는 조밀하지 않은 데이터 집합의 경우는 항목들의 가중치 변화가 전체 데이터베이스의 가중치 합에 많은 영향을 주고 있기 때문인 것으로 보인다. 제안하는 방법은 또한 단 한번의 데이터베이스 스캔만을 필요로 하므로 스트림 데이터의 마이닝에도 적합하다. 여러 실험을 통하여 제안하는 기법이 매우 효과적이며 다양한 데이터 집합에서 좋은 성능을 보임을 입증하였다.

참 고 문 헌

[1] R. Agrawal, T. Imieliński and A. Swami, "Mining association rules between sets of items in large databases," Proc. of the 12th ACM SIGMOD Int. Conf. on Management of Data, May 1993, pp.207-216.

[2] R. Agrawal, R. Srikant, "Fast Algorithms for Mining Association Rules in Large Databases," Proc. of the 20th Int. Conf. on Very Large Data Bases, Sep., 1994, pp.487-499.

[3] U. Yun, J.J. Leggett, "WFIM: weighted frequent itemset mining with a weight range and a minimum weight," Proc. of the Fourth SIAM Int. Conf. on Data Mining, USA, 2005, pp.636-640.

[4] U. Yun, "Efficient Mining of weighted interesting patterns with a strong weight and/or support affinity," Information Sciences, vol. 177, 2007, pp.3477-3499.

[5] C.H. Cai, A.W. Fu, C.H. Cheng, W.W. Kwong, "Mining association rules with weighted items," Proc. of Int. Database Engineering and Applications Symposium, IDEAS 98, Cardiff, Wales, UK, 1998, pp. 68-77.

[6] C.F. Ahmed, S.K. Tanbeer, B.-S. Jeong and Y.-K. Lee, "Mining Weighted Frequent Patterns in Incremental Databases", Proc. of the 10th Pacific Rim Int. Conf. on Artificial Intelligence, Dec. 2008, pp.933-938.

[7] F. Tao, "Weighted association rule mining using weighted support and significant framework," Proc. of the 9th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, USA, 2003, pp. 661-666.

[8] W. Wang, J. Yang, P.S. Yu, "WAR: weighted association rules for item intensities," Knowledge Information and Systems, Vol.6, 2004, pp.203-229.

[9] J. Han, J. Pei, Y. Yin, R. Mao, "Mining frequent patterns without candidate generation: a frequent-pattern tree approach," Data Mining and Knowledge Discovery, Vol.8, 2004, pp. 53-87.

[10] G. Grahne, and J. Zhu, "Fast Algorithms for frequent itemset mining using FP-Trees," IEEE Transactions on Knowledge and Data Engineering, Vol.17, No.10, Oct., 2005, pp.1347-1362.

[11] J. Han, H. Cheng, D. Xin, X. Yan, "Frequent pattern mining: current status and future directions," Data Mining and Knowledge Discovery, Vol.15, 2007, pp.55-86.

[12] A. Metwally, D. Agrawal, A. E.Abbadi, "An Integrated Efficient Solution for Computing Frequent and Top-*k* Elements in Data Streams," ACM Transactions on Database Systems (TODS), Vol.31, No.3, 2006, pp.1095-1133.

[13] N. Jiang and L. Gruenwald, "Research Issues in Data Stream Association Rule Mining," SIGMOD Record, Vol. 35, No. 1, Mar., 2006, pp.14-19.

[14] C. K. -S. Leung, Q. I. Khan, "DSTree: A Tree structure for the mining of frequent Sets from Data Streams," Proc. of the 6th Int. Conf. on Data Mining (ICDM'06), 2006, pp.928-932.

[15] J.-L. Koh, S.-F. Shieh, "An efficient approach for maintaining association rules based on adjusting FP-tree structures," Proc. of the DASFAA'04, 2004, pp.417-424.

[16] C. K.-S. Leung Q.I. Khan, Z. Li and T. Hoque, "CanTree: a canonical-order tree for incremental frequent-pattern mining," Knowledge and Information Systems, Vol.11, No.3, 2007, pp.287-311.

[17] U. Yun, "Mining lossless closed frequent patterns with weight constraints," Knowledge-Based Systems, Vol.210, 2007, pp.86-97.

[18] S.K. Tanbeer, C.F. Ahmed, B.-S. Jeong and Y.-K. Lee, "CP-tree: A tree structure for single pass frequent pattern mining," Proc. of the 12th Pacific-Asia Conf. on Knowledge Discovery and Data Mining (PAKDD'08), 2008.

정 병 수

e-mail : jeong@khu.ac.kr

1983년 서울대학교 컴퓨터공학과(학사)

1985년 한국과학기술원 전산학과(석사)

1995년 Georgia Institute of Technology, College of Computing(박사)

1985년~1989년 한국 데이터통신㈜ 선임연구원

1995년~1996년 Georgia Institute of Technology, College of Computing, PostDoc

1996년~현 재 경희대학교 컴퓨터공학과 교수

관심분야 : 데이터베이스, 데이터 마이닝, 모바일 컴퓨팅

Ahmed Farhan

e-mail : farhan@khu.ac.kr

2004년 방글라데시 다카 대학 컴퓨터공학과(학사)

2006년 방글라데시 다카 대학 컴퓨터공학과(석사)

2006년~2007년 방글라데시 다카 대학 컴퓨터공학과 전임강사

2007년~현 재 경희대학교 컴퓨터공학과 박사과정

관심분야 : 데이터 마이닝, 패턴 탐색, 모바일 컴퓨팅