퀀터티가 있는 순차 패턴을 찾는 효율적인 알고리즘*

임종화
한국과학기술원 전산학과
igkst@mozart.kaist.ac.kr

심규석
서울대 전기컴퓨터공학부
shim@ee.snu.ac.kr

김철연°
서울대 전기컴퓨터공학부
cykim@kdd.snu.ac.kr

An Efficient Algorithm for Mining Sequential Patterns with Quantities

Jong-Hwa Lim
Dept. of CS, KAIST

Kyuseok Shim
School of EECS, SNU

Chulyun Kim°
School of EECS, SNU

요 약

순차 패턴을 찾는 것은 데이타 마이닝 응용분야에서 중요한 문제이다. 기존의 순차 패턴 마이닝 알고리즘들은 아이템으로만 이루어진 순차 패턴만을 찾아 주었다. 하지만 아이템과 관련된 퀀터티 정보가 더욱 유용한 정보를 제공해 주는 경우가 많이 있다. 본 논문에서는 퀀터티가 있는 순차 패턴을 찾는 알고리즘을 소개한다. 기존 알고리즘을 초보적으로 확장한 알고리즘은 탐색 공간을 모두 다 검색하여 결과를 얻는 방법을 사용하기 때문에 결과적으로 나쁜 성능을 나타내었다. 이러한 단점을 없애기 위해 여과 과정과 샘플링 기반 알고리즘을 사용하여 검색해야 하는 후보 패턴의 수를 줄여줌으로써 알고리즘의 성능을 개선하였다. 실험 결과는 새로운 방법들이 초보적인 확장을 한 기존 알고리즘보다 훨씬 더 좋은 성능을 나타냄을 보여주었다.

제 1 절 서론

1.1 배경 및 문제점

데이타 마이닝 분야 중 하나인 순차 패턴(sequential pattern)을 찾는 알고리즘은 현재까지 크게 두 가지로 분류할 수 있다. IBM 연구소에서 개발한 Apriori 스타일 알고리즘[1][2][3]과 캐나다의 Simon Fraser대학에서 개발한 PrefixSpan 스타일 알고리즘[5][6]이다. 이들 알고리즘 이외에도 사용자가 원하는 시퀀스만을 정규표현식(regular expression)형태로 입력받아 순차 패턴을 찾는 SPIRIT[4]도 있다.

하지만 지금까지의 순차패턴 마이닝과 연관규칙 마이닝에서는 아이템(item)만을 다루는 마이닝을 주로 연구해 왔다. 하지만 실제의 많은 데이타들은 아이템과 함께 아이템의 개수나 아이템과 관련된 시간에 대한 정보를 담고 있는 경우가 많이 존재 한다. 하지만 이런 정량적인 데이타들이 추가된 마이닝의 경우에 기존의 알고리즘에 비해 탐색 공간(search space)이 늘어나게 된다. 이러한 탐색 공간의 확장을 줄여서 빠르게 수행되는 새로운 알고리즘이 필요하기에 연구를 시작하게 되었다.

제 2 절 초보적인 접근법

2.1 확장된 문제 정의

이 장에서는 기존 문제를 확장 하여 퀀터티를 다루는 문제를 정의한다. 퀀터티를 추가함으로써 기존 문제 정의와 달라지는 부분은 아이템이 아이템과 퀀터티의 쌍(pair) 으로 확장된다는 점이다. $I = \{i_1, i_2, ..., i_m\}$는 모든 아이템의 집합(set of all items)이다. $i \in I$인 아이템 i와 정수의 집합 N[1] 의 원소인 정수 n과의 쌍인 $[i,n]$을 **확장아이템**(extended item)이라고 한다. 이때 i를 확장아이템의 **아이템 부분**, n을 확장아이템의 **퀀터티 부분**(혹은 **정수 부분**)이라고 한다. 확장아이템들의 집합인 EI는 $\{[i,n] | i \in I \wedge n \in N_i\}$로 정의 한다. **확장아이템집합**(extended itemset)은 EI의 부분집합(subset)중에 아이템 부분이 같은 확장아이템이 동시에 존재하지 않는 집합을 말한다. 이러한 확장아이템집합 eis과 EI와의 관계를 $eis \subseteq_e EI$로 정의하자. 확장아이템집합간의 부분집합(subset), 포함집합(superset)을 정의하면 확장아이템집합 $eis_1 = \{a_1, a_2, ..., a_n\}$ 와 확장아이템집합 $eis_2 = \{b_1, b_2, ..., b_m\}$ 이 있을 때, $1 \le p \le n$인 eis_1의 모든 확장아이템 a_p의 (1)a_p의 아이템 부분인 i_{a_p}와 같은 아이템을 갖는 $1 \le q \le m$인 b_q가 존재하며, (2)a_p의 퀀터티 부분인 n_{a_p}가 b_q의 퀀터티 부분인 n_{b_q}보다 작거나 같은 경우, $eis_1 \subseteq eis_2$라고 하고 eis_1이 eis_2의 부분집합(subset), eis_2가 eis_1의 포함집합(superset)이라고 한다.

시퀀스(sequence)는 $\langle s_1 s_2 ... s_l \rangle$로 나타낼 수 있는데, 여기서 $s_j \subseteq_e EI$, 즉 모든 $1 \le j \le l$를 만족하는 j에 해당되는 s_j는 확장아이템집합이 된다.

2.2 Apriori 알고리즘의 변형

Apriori 알고리즘에서 확장된 문제를 해결하기 위해서는, 후보 패턴 생성 과정과 부분시퀀스 탐색과정을 확장해야 할 것이다. 먼저 후보 패턴 생성 과정에서 확장해야 할 점은 단순히 아이템이 같은 지를 살피는 과정 대신 아이템과 아이템의 퀀터티까지도 같은 지를 체크하면 된다. 그 외의 과정은 기존의 과정과 동일하다. 다음으로 부분시퀀스 탐색 과정에서도 후보 패턴이 사용자 시퀀스에 포함되는지의 여부를 확

*본 연구는 첨단정보기술 연구센터를 통하여 과학재단의 지원을 받았음.

[1] 실수로 확장될 수도 있으나, 문제정의에서는 정수의 집합으로 한다.

장된 문제 정의의 포함관계를 사용하여 판별하면 된다. 앞으로 이 알고리즘을 초보적인(naive) 확장 Apriori 알고리즘이라고 부르기로 한다.

제 3 절 통합된 접근법

3.1 시퀀스 여과과정

퀀터티가 있는 순차 패턴 마이닝의 결과들을 관찰하면 퀀터티가 있는 순차 패턴 마이닝을 하기 전에, 퀀터티가 없는 결과를 먼저 구한 후 그 결과를 이용할 수 있게 된다. 이렇게 퀀터티 없이 얻은 결과를 이용하는 것을 **여과과정**(filtered counting)이라고 하기로 한다.

길이가 m인 패턴의 경우를 생각해 보자. 패턴 c의 n번째 아이템을 $c.i_n$으로 나타내기로 하자. 퀀터티 없는 후보 패턴의 집합을 C라고 하고, 퀀터티 없는 빈번한 패턴의 집합을 F라고 하고, 앞에서와 같이 아이템 i와 관련된 퀀터티는 L_i개라고 하면, 여과과정이 없는 경우는 $\sum_{c\in C}\prod_{1\le n\le m} L_{c.i_n}$개의 후보 패턴을 통해 결과를 얻게 되지만, 여과과정을 넣은 경우는 여과과정에서 $|C|$개의 후보 패턴, 그 이후의 과정에서 $\sum_{c\in F}\prod_{1\le n\le m} L_{c.i_n}$개의 후보 패턴을 만들게 되어, 총 $|C| + \sum_{c\in F}\prod_{1\le n\le m} L_{c.i_n}$개의 후보 패턴을 만들게 된다. 이 경우에도 $|F| \ll |C|$이기 때문에, 여과과정이 있는 경우의 후보 패턴의 수는 여과과정을 사용하지 않은 경우의 후보 패턴의 수보다 항상 작거나 같다.

정리 1 *여과과정을 사용한 경우에 만들어지는 후보 패턴의 수는 여과과정을 사용하지 않은 경우의 후보 패턴의 수보다 항상 작거나 같다.*

증명: *[8] 참조* □

3.2 샘플링 기반 알고리즘

샘플링 기반 알고리즘은 퀀터티가 있는 순차 패턴 마이닝에서 최대의 빈번한 패턴만을 얻기 위한 방법이다. 즉, 빈번한 패턴중에서 다른 빈번한 패턴의 부분시퀀스가 되지 않는 패턴들을 얻기 위한 방법이다.

기존 방법에서는 모든 퀀터티에 대해 후보 패턴을 만들었지만, 샘플링 기반 알고리즘에서는 퀀터티마다 일정 간격(interval)를 두고, 그 간격에 해당하는 후보 패턴만을 만든 후, 그 후보 패턴들중 빈번한 패턴들을 얻은 후, 그 다음 단계에서 간격 사이의 후보 패턴을 모두 만드는 방법을 사용한다. 즉, 1단계에서는 듬성듬성한(sparse grained) 후보 패턴을 만들어서 결과를 얻게 되고, 1단계에서 얻은 빈번한 패턴중에 다른 패턴의 진 부분시퀀스(proper subsequence[2])인 것들을 지운 뒤, 남아 있는 패턴들을 2단계에서 세밀한 후보 패턴을 만드는데에 사용한다.

길이가 m인 패턴의 경우를 생각해 보자. 기존의 방법을 사용할 경우는 $\sum_{c\in C}\prod_{1\le n\le m} L_{c.i_n}$개의 후보 패턴을 통해 결과를 얻게 되지만, 샘플링 기반 알고리즘을 사용하면 1단계에서 $k^m|C|$개의 후보 패턴을 만들게 되고, 진 부분시퀀스 제거과정을 거친 후의 결과의 집합이 $F_1(0 \le |F_1| \le k^m|C|)$라면 2단계에서는 $\sum_{c\in F_1}(\prod_{1\le n\le m}\frac{L_{c.i_n}}{k} - 1)$개의 후보 패턴을 만들게 된다. 따라서 총

$$k^m|C| + \sum_{c\in F_1}(\prod_{1\le n\le m}\frac{L_{c.i_n}}{k} - 1)$$

개의 후보 패턴을 만들게 된다. 이 역시 기존의 방법에서의 후보 패턴의 수보다는 항상 작게 된다.

정리 2 *샘플링 기반 알고리즘을 사용한 경우에 만들어지는 후보 패턴의 수는 샘플링 기반 알고리즘을 사용하지 않은 경우의 후보 패턴의 수보다 항상 작거나 같다.*

증명: *[8] 참조* □

3.2.1 정확성(correctness)증명

정리 3 *1단계에서 찾아진 빈번한 패턴중에 진 부분시퀀스를 지워도 최대의 빈번한 패턴을 잃지 않는다.*

증명: *[8] 참조* □

하지만 1단계의 결과중에 진 부분시퀀스가 아닌 부분시퀀스의 관계에 있는 빈번한 패턴[3]은 지우지 말아야 한다. 그 패턴들을 이용해서 2단계에서 만들게 되는 후보 패턴들은 최대의 패턴이 될 가능성이 있기 때문이다.

3.2.2 여과과정과 샘플링 기반 알고리즘을 같이 사용하는 경우

정리 4 *여과과정과 샘플링 기반 알고리즘을 동시에 사용하는 경우 만들어지는 후보 패턴의 수는 여과과정만을 사용한 경우나 샘플링 기반 알고리즘만을 사용한 경우보다 항상 작거나 같다.*

증명: *[8] 참고*

제 4 절 성능 평가

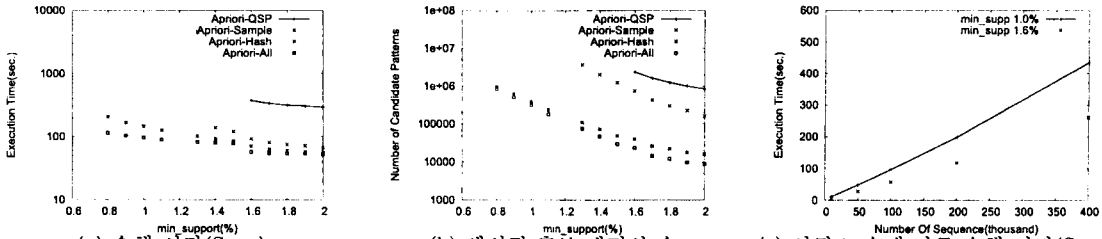
모든 실험은 LINUX 운영체제에 메모리 512MB인 Pentium 4 PC에서 실행되었다.

합성 데이타를 만드는 데에는 IBM 연구소에서 만든 순차 패턴 마이닝과 연관 규칙 마이닝을 위한 데이타 생성기를 수정하여 사용하였다.[4] 기존의 데이타 생성기는 퀀터티 부분을 생성해 주지 않았으므로, 주어진 확률분포에 따라 퀀터티 부분을 추가해 주도록 수정 하였으며, 본 논문에서 사용된 실험 데이타 들은 시퀀스당 평균 트랜잭션 수 10, 트랜잭션당 평균 아이템 수 2.5, 총 아이템 수 10000개의 파라미터로 생성된 데이타들이다.

[2] 아이템 부분은 모두 같고 퀀터티 부분은 모두 작은 부분시퀀스

[3] 모든 아이템부분은 같고, 퀀터티부분도 같은 경우가 존재하는 부분시퀀스

[4] http://www.almaden.ibm.com/cs/quest/syndata.html



(a) 수행 시간(Secs.) (b) 생성된 후보 패턴의 수 (c) 시퀀스 수에 따른 수행 시간(Secs.)

그림 1: 합성 데이타 집합들에 대한 성능 평가

실험에서는 4가지의 알고리즘의 실행시간 및 생성된 후보 패턴의 개수를 비교하였다.

- 초보적인 확장 Apriori 알고리즘(Apriori-QSP)
- 여과과정 기반 알고리즘(Apriori-Hash)
- 샘플링 기반 알고리즘(Apriori-Sample)
- 여과과정+샘플링 기반 알고리즘(Apriori-All)

실험 결과를 통해 본 논문이 제시하는 알고리즘이 초보적으로 확장된 알고리즘에 비해 수십배 이상의 수행시간 단축을 보여 준다는 사실을 알 수 있다.

4.1 합성 데이타 집합에 대한 결과

그림 1의 (a)와 (b)는 각각의 알고리즘들의 최소 지지도별 수행속도와 생성된 후보 패턴의 수를 보여준다. 본 논문에서 제시하는 아이디어의 유무에 따른 수행시간과 생성된 후보 패턴의 수의 차이는 최소 지지도가 낮아질 수록 더욱 커짐을 알 수 있다.

마지막으로 본 논문에서 제시한 알고리즘의 범위성 (scalability)에 관한 결과를 살펴보자. 그림 1의 (c)는 데이타의 특성은 동일하면서, 데이타내의 시퀀스의 수를 늘려가면서 실험을 한 결과이다. 이 실험 결과를 통해 알고리즘은 데이타베이스 내의 시퀀스의 수에 대해 좋은 범위성을 갖는다는 것을 알 수 있다.

제 5 절 결론

본 논문에서는 기존의 순차 패턴 마이닝을 확장하여 퀀터티 정보까지도 포함한 순차 패턴 마이닝 알고리즘을 제시하였다. 이들 퀀터티가 있는 순차 패턴들은 퀀터티가 없는 순차 패턴들에 비해 더 자세한 정보를 나타내 줄 수 있기 때문에 기존의 순차 패턴이 사용되던 마케팅과 같은 분야에서 더욱 세밀한 이용을 가능케 해 준다.

퀀터티가 있는 순차 패턴을 찾는 알고리즘으로 Apriori 알고리즘의 초보적인 확장 알고리즘은 생성해야 하는 후보 패턴의 수를 크게 늘리게 되어 결과적으로 수행 시간도 크게 증가하였다. 따라서 결과를 얻는데에 필요한 후보 패턴의 수를 줄임으로써 수행 시간을 개선할 수 있는 방법을 제안하였다. 즉, 여과과정(filtered counting)과 샘플링 기반 알고리즘(sampling based algorithm)이라는 새로운 아이디어를 사용하여 최대의 패턴이 될 수 없는 후보 패턴들을 미리 전지함으로써 알고리즘의 수행속도를 매우 개선하였다.

참고 문헌

[1] R. Agrawal and R. Srikant. "Fast algorithms for mining association rules," In *Proc. 1994 Int'l Conf. Very Large Data Bases (VLDB'94)*, pages 487-499, Santiago, Chile, September 1994.

[2] R. Agrawal and R. Srikant. "Mining sequential patterns," In *Proc. 1995 Int'l Conf. Data Engineering (ICDE'95)*, pages 3-14, Taipei, Taiwan, March 1995.

[3] R. Agrawal and R. Srikant. "Mining sequential patterns: Generalizations and performance improvements," In *Proc. 5th Int'l Conf. Extending Database Technology (EDBT'96)*, pages 3-17, Avignon, France, March 1996.

[4] M. Garofalakis, R. Rastogi, and K. Shim. "Spirit: Sequential pattern mining with regular expression constraints," In *Proc. 1999 Int'l Conf. Very Large Data Bases (VLDB'99)*, pages 223-234, Edinburgh, UK, September 1999.

[5] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M.-C. Hsu. "Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth," In *Proc. 2001 Int'l Conf. Data Engineering (ICDE'01)*, pages 215-224, Heidelberg, Germany, April 2001.

[6] J. Han, J. Pei, and Y. Yin. "Mining frequent patterns without candidate generation," In *Proc. 2000 ACM-SIGMOD Int'l Conf. Management of Data (SIGMOD'00)*, pages 1-12, Dallas, TX, May 2000.

[7] J.S. Park, M.S. Chen, and P.S. Yu. "An effective hash-based algorithm for mining association rules," In *Proc. 1995 ACM-SIGMOD Int'l Conf. Management of Data (SIGMOD'95)*, pages 175-186, San Jose, CA, May 1995.

[8] Jong-Hwa Lim, "An Efficient Algorithm for Mining Sequential Patterns with Quantities," *M.S. Thesis*, KAIST, Korea, Feb 2002.