트랜잭션이 없는 시계열 데이터로 부터 가상 트랜잭션을 이용한 데이터 마이닝

김민수*, 이준섭*, 김응모*
*성균관대학교 전기전자및컴퓨터공학부

e-mail : lasarus@ece.skku.ac.kr

Mining Time Series Data With Virtual Transaction

Min-Soo Kim*, Joon-Sub Lee*, Ung-Mo Kim*
*School of Electronical & Computer Engineering, SungKyunKwan University

요 약

대용량의 데이터들로부터 사용자가 원하는 데이터를 찾기 위하여 많은 데이터 마이닝 기술들이 연구되어 실제 응용프로그램에서 많이 적용되고 있다. 이러한 데이터 마이닝의 기술 중 연관규칙은 항목들의 집합으로 표현되는 트랜잭션에서 각 항목간의 연관성을 찾는데 사용된다. 그러나 실세계에는 트랜잭션이 없이 일련의 이벤트만 시간에 따라서 발생하는 데이터들이 많이 존재한다. 이러한 시계열 이벤트 데이터들로부터 다양한 가상 트랜잭션을 생성하는 기법들을 제시한다. 이러한 가상 트랜잭션 데이터로 변환된 시계열 데이터에 연관규칙 , 순차패턴, 주기패턴과 관련된 여러 가지 알고리즘을 바로 적용 함으로서 유용한 규칙들을 발견해 낼 수 있다..

1. 서론

지식탐사의 핵심적인 엔진 역할을 담당하는 데이터 마이닝은 여러가지 기술들을 사용하여 방대한 양의 다양한 유형의 데이터들 속에서 데이터 상호간의 관련성 및 각 도메인에서 관심을 가지는 중요한 정보를 추출하는 일련의 과정을 말한다.

이러한 데이터마이닝에는 크게 연관규칙, 클래시피케이션, 클러스터링 등과 같은 많은 마이닝 기술들이 있으며 현재에도 활발한 연구가 진행되고 있다.

이 중 연관규칙[1][2]은 항목 집합으로 구성된 트랜잭션에서 각 항목간의 연관성을 반영하는 규칙을 찾는 기법이다. 지지도와 신뢰도를 설정 값으로 트랜잭션내에 존재하는 아이템 집합들간의 연관규칙을 찾는다. X , Y -> Z (X 이고 Y 이면 Z 이다) 는 X,Y,Z 간의 연관규칙을 나타내는 식이다. 예를 들어 "아빠들은 기저기를 사면 맥주를 산다"라는 관련규칙으로부터 슈퍼마겟의 상품배치를 변경 함으로서 보다 좋은 고객 서비스를 하여 매출을 높일 수 있다.

그러나 실제 세계에는 시스템이벤트, 트랜잭션이 구성되어 있지 않은 시간기반의 이벤트 데이터(서버에서의 시스템로그, 전화통신회사에서의 알람)가 많이 존재한다.

빈발 에피소드는 이러한 시간기반의 이벤트 데이터(통신네트워크에서 발생하는 여러 가지 이벤트)들로부터 이벤트간의 연관성을 발견하기 위하여 고안되었다[3] . 여기서 에피소드란 부분적으로 정렬되어 있는 이벤트의 집합을 말하며 패턴과 같은 의미로 사용된다..

순차패턴이란 트랜잭션 데이터로부터 아이템간의 순차적인 연관관계를 찾는 것을 말한다. 이렇게 트랜잭션 데이터로부터 순차패턴을 찾는 것은 시계열 데이터에서 빈발 에피소드를 찾아내는 빈발 에피소드알고리즘과 일치하는 내용이다.

이 논문의 주요 아이디어는 시간정보를 담고 있는 이벤트중심의 시계열 데이터로부터 정해진 시간간격 및 이벤트개수를 이용하여 가상의 트랜잭션을 생성함으로서 기존의 연관규칙에 관한 알고리즘들[1][2][12]을 수정 없이 적용하여 유용한 규칙을 찾을려고 하는 것이다. 그리고 가상트랜잭션들의 트랜잭션 발생시간을 트랜잭션에 속한 이벤트들의 발생시간의 평균시간을 사용 함으로서 이벤트들의 주기가 아닌 패턴들의 주기를 발견할 수 있는 주기패턴 알고리즘[4][7][8][13]과 패턴들의 순서를 발견할 수 있는 순차

패턴 알고리즘[5][9]10][11]을 적용할 수 있도록 하는 데 있다.

2. 관련연구

시간기반의 데이터에 대한 많은 연구가 진행되었고 현재도 계속되고 있다. 이런 연구들은 커다랗게 시계열 데이터에 대한 연구와 트랜잭션 데이터에 대한 연구로 진행되고 있다.

시계열 데이터에 대한 연구로서는 전화통신회사에서 발생하는 많은 시간기반의 이벤트들로부터 각 이벤트들간의 연관성을 분석해 내는 빈발 에피소드 기법이다[3], 이 알고리즘에서는 시계열 데이터에서 많은 반복을 갖는 패턴들을 "빈발에피소드라"하며 이런 빈발 에피소드를 사용하여 이벤트간의 연관 규칙을 찾는다. 이렇게 찾아진 연관규칙은 이벤트들간의 순차패턴이다. 시간정보를 담고있는 이벤트레코들 집합이 주어지면 , 이벤트 레코드 간격 [t1,t2] 는 시간 t1 부터 t2 까지에서 발생한 이벤트 레코드를 말한다, 간격의 크기는 t2-t1 으로 정의된다. 이렇게 정의된 레코드 즉 이벤트의 집합 을 에피소드라 한다, 이러한 에피소드의 발생회수를 저장하여 알고리즘의 설정 값인 min-fr 보다 큰 것들만 빈발에피소드로 결정해낸다. 그러나 빈발 에피소드 방법은 시계열 데이터로부터 관련규칙을 찾는 데만 국한 되 있다.[3]

트랜잭션데이터들의 순서패턴 찾는 알고리즘은 순서정보를 가지고 있는 트랜잭션 데이터 베이스로부터 트랜잭션내의 데이터에 공통으로 나타나는 순차적인 패턴을 찾는 문제다. 고객의 트랜잭션을 담고 있는 대용량 데이터베이스를 입력 데이터로한다, 여기서 트랜잭션은 고객 ID 와 트래잭션 시간정보를 담고 있어야 한다. 한 개의 트랜잭션에는 한번에 구매하는 아이템들이 담겨져 있다. 예를 들어 비디오 대여점에서 고객들은 "쉬리"를 빌려보는 사람들은 "JSA"를 빌려본다라는 순차패턴을 찾음으로서 "쉬리"를 본 고객들에서 "JSA"를 추천할 수 있다.[5][9][10][11]

대부분의 패턴을 찾는 알고리즘은 짧은 길이의 패턴을 찾는데 효과적이다,그러나 실제 데이터베이스에는 긴길이의 패턴이 많이 포함되 있다. 기존의 Apriory 관련 알고리즘으로는 이러한 긴 패턴을 효과적으로 찾을 수 가없다. 그래서 여기서는 MAX-Miner 알고리즘을 제안했다.[12]

순차패턴을 찾는 다고 해서 이것이 사용자한테 가장 관심있는 룰이 될 수 는 없다. 그리고 대부분의 순서 패턴은 발생빈도를 기준으로 발견하기 때문에 에러검출,웹 사용률 분석 영역에서는 제약사항이 있다. 그래서 이러한 순서패턴이 찾아진 뒤에 보다 관심있는 정보를 찾기 위한 과정으로 연구된 논문이다.[14]

이벤트기반의 시계열데이타로부터 어떤 일,주,월,분기,년 에 해당하는 주기 패턴을 찾기 위한 방법[8]과 각 주기내의 부분적인 주기를 찾는 방법[7][13]들이 있다. 그리고 어떤 이벤트가 발생했을 때 얼마시간이 지난후에는 어떤 이벤트가 발생할 것인가를 예측하는 방법[5][8]들이 많이 연구되고 있다. 이러한 예측은 주식시장에서 주가를 예측하는 곳에 사용되고 있다.

이벤트기반의 시계열 데이터에는 주기를 갖는 패턴도 중요하지만 주기에 어긋나는 패턴이 중요한 관심사항이 될 수 있다. 예를 들어 병원에서 어떤 환자에게 약을 1 일 3 회 투여하고 있다. 그런데 어느날 환자의 상태가 악화되어 약을 좀 빨리 투여하였을 경우, 주기패턴 분석에 의하면 1 일 9 시,13 시,19 시의 주기를 찾을 수 있다.하지만 환자의 상태에 따라 빨리 투여된 약의 경우는 분석이 안된다. 이런 경우의 이벤트 패턴을 찾는 방법이 제안되었다[4].

3. 접근방법

시계열데이타로부터 가상트랜잭션집합을 생성해내는데 있어서 두가지의 접근방법을 시도한다. 한가지는 고정타임윈도우를 이용하는 방법이고 다른 한가지는 고정이벤트윈도우를 이용하는 방법이다. 위의 두 가지 기법에 "겹침깊이(Overlapping Depth)"를 사용하여 가상트랜잭션을 생성한다. 이 방법으로 만들어진 가상트랜잭션 데이터 집합은 연관규칙 알고리즘의 입력데이터로 사용하여 이벤트 패턴을 분석해 낼 것이다. 이어서 가상트랜잭션 데이터의 각각의 트랜잭션을 새로운 이벤트로 설정하고 가상트랜잭션들의 평균 시간을 계산하여 signified 가상 트랜잭션데이타 집합을 생성할 것이다. 이 집합은 패턴간의 주기 및 순차패턴의 입력데이타로 사용되어 패턴들간의 주기와 순처패턴을 분석할 것이다. 그림 1 에 전체적인 접근 구조가 나와있다.

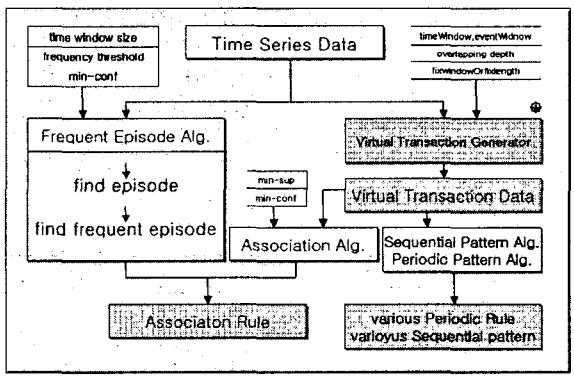

그림 1

4. 이벤트와 가상트랜잭션

이벤트는 순차적으로 발생한다고 가정하고 발생시간정보와 기타정보를 포함할 수 있다.

E : 이벤트유형의 집합 = {A,B,C , , , ...}

□ : (A, t) = 시간 t 에 발생한 이벤트 A

S : 발생한 이벤트(□)들의 집합= (Ts,Te)
= { (A1,t1), (A2,t2), . . . , (An,tn) }
여기서 Ai ∈ E , i=1,2,3,⋯ , n
Ts : start time of event sets
Te : end time of event sets
Ts ≤ ti ≤ Te
LengthTime(S) = Te □ Ts
CountEvent(S) = # of Event in S

가상트랜잭션: 규칙에 의해서 합쳐진 이벤트집합.
VT(Ai,ts,te) = { (A1,t1), (A2,t2), . . . , (An,tn) } :
고정시간윈도우에서의 가상트랜잭션
VT(Ai,ts,len) = { (A1,t1), (A2,t2), . . . , (Alen,tlen) } :
고정이벤트윈도우에서의 가상트랜잭션
CVTS : 후보가상트랜잭션집합.
SVT : SVT(Sigi,tm) : Signed 가상트랜잭션.
tm : 가상트래잭션에속한 ti 의 평균시간.
Si: 가상트랜잭션안의 이벤트 패턴들을 변환한 코드값.
SIGN :패턴들이 새롭게 변환될 코드집합= (S1, S2,..)

5. 타임윈도우길이 와 이벤트윈도우 길이

얼마만큼의 시간간격으로 가상 트랜잭션을 유지할 것인가. 몇 개의 이벤트로 가상트랜잭션을 유지할것인가가 중요한 문제이다. 여기에 겹침깊이(Overlapping depth)를 사용하여 트랜잭션을 구축한다.

w :∑ (Ai,ti) , i=ts..te ,ts ≥ Ts , te ≤ Te: timewindow
lengthT(w) : ts – te – time window size
countT(w) : # of evnet in a time window w
ℓ :∑ (Ai,ti), i=Os..Oe :eventwindow
Os : 트랜잭션의 시작이벤트순서
lengthL(ℓ) : Oe`s t- Oe`s t
countL(ℓ) : # of evnet in a time window w
overlapping depth : 가상트랜잭션간의 겹침정도.
Dt :overapping depth for timewindow, time unit
Dℓ :overlapping depth for eventwindow,event unit
아래 예를 통해서 타임윈도우,이벤트윈도우를 이해할 수 있다.

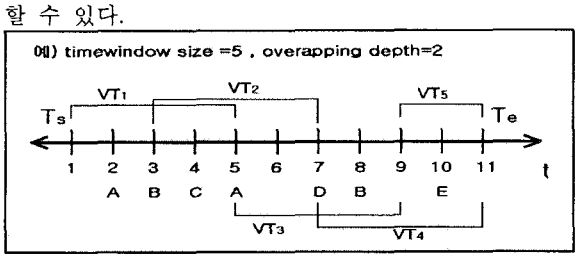

그림 2

VT1 = { (A,2), (B,3), (C,4), (A,5) }
VT2 = { (B,3), (C,4), (A,5), (D,7) }

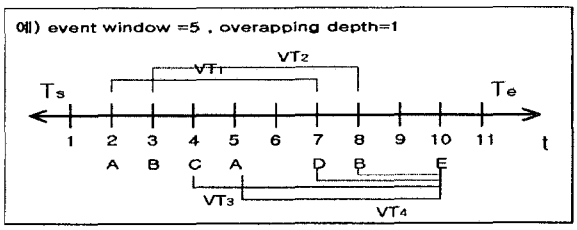

그림 3

VT1 = { (A,2), (B,3), (C,4), (A,5), (D,7) }
VT2 = { (B,3), (C,4), (A,5), (D,7), (B,8) }

6. 알고리즘

Algorithm Main1 for time window
Input : S – 발생한 시계열 이벤트 집합
windth – time window size
depth – 가상 트랜잭션 overlapping depth
Output:VTS - Virtual Transactions
SVTS – Signified Virtual Transactions
Method
//find event type from S
E= findEventtype(S);
//find candidate Virtual transactions
CVTS=CVTransactionWin(E,S,width,depth)
//split CVTS ,if it has a cyclic events
VTS=SplitCyclicEvent(CVTS);
//write VTS for association Alg.
WriteVTS(VTS);
//signify event pattern & pluning single event
SVTS=SignifyVTS(VTS)
//wtite SVTS for periodic,sequential Alg.
WriteSVTS(SVTS);
//wtire mapping information, SIGN&pattern
WriteSignPattern();

Algorithm Main2 for event window
Input : S – 발생한 시계열 이벤트 집합
windth – event window size
depth – 가상 트랜잭션 overlapping depth
Output:VTS - Virtual Transactions
SVTS – signified Virtual Transactions
Method
//find event type from S
E= findEventtype(S);
//find candidate Virtual transactions
CVTS=CVTransactionEvent(E,S,width,depth)
-- same logic above Main1`s Alg.
아래 그림 4 에는 위의 알고리즘을 도표화한것이다.

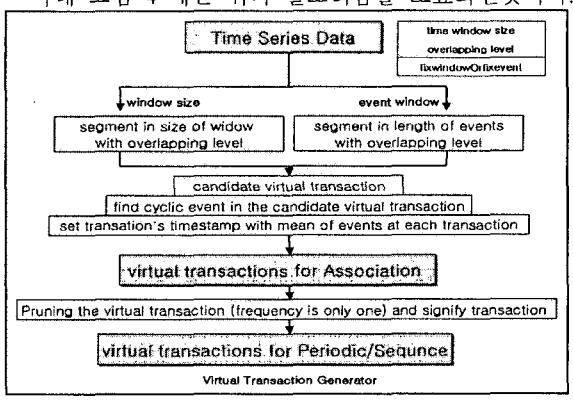

그림 4

7. 실험방법

샘플데이타로서 두가지를 사용한다, 한가지는 정해진 룰에 의해 생성된 데이터이고 나머지는 UCI 의 파이오니아 데이터 자료이다. 우선 두 가지 종류의 데이타를 사용해서 빈발에피소드 알고리즘과 가상트랜잭

션 알고리즘 + 연관규칙알고리즘을 이벤트패턴의 정확성과 속도를 비교한다. 그리고 가상트랜잭션 생성 알고리즘에서 파라미터 값 들을 바꿔가면서 파라미터에 따른 결과를 비교한다.

두 가지의 샘플에 대해서 패턴들의 순차패턴과 주기패턴을 분석한다.

이 실험을 위해 생성하는 샘플 데이타는 샘플 생성기를 사용하여 생성하고 ,UCI 에서 제공되는 이동 로봇인 파이오니아-I 의 센서에 의한 시간 데이터[15]는 시간에 따라 이벤트가 발생하고 그 이벤트에 포함되는 많은 정보들이 포함되어 있기 때문에 실험에 필요한 데이터로 가공 아래 그림 5 처럼 Format Fitter 프로그램을 만들어 사용한다.

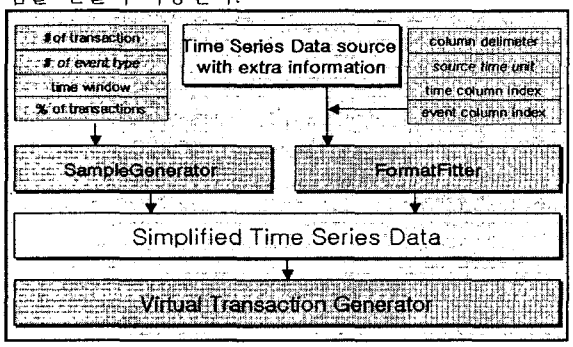

그림 5

8. 결론

여기서 제안된 기법을 사용하여 연관규칙들을 찾아내는데 있어 frequent episode 기법보다 시간적으로 많은 비용을 소비할 수 도 있지만 시계열 데이터에 대한 마이닝에 있어서 새로운 접근 방법으로 가상 트랜잭션을 만들어 논다면 현존하는 연관 규칙 알고리즘을 사용하여 분석할 수 있다는 장점이 있다. 그리고 트랜잭션 데이터처럼 처리되기 때문에 여러가지 알고리즘의 데이터 소스를 적용할 수 있는 잇점을 가지고 있다. 그리고 각 가상트랜잭션의 이벤트들의 평균값을 사용하여 트랜잭션 시간을 설정 함으로서 패턴 주기를 찾는 여러가지 알고리즘을 적용할 수 있다.

참고문헌

[1] Rakesh Agrawal, Tomasz Imielinski, and Arun Swami. Mining association rules between sets of items in large databases. In Proc. of the ACM SIGMOD Conference on Management of Data, pages 207--216, Washington, D.C., May 1993.

[2] Rakesh Agrawal and Ramakrishnan Srikant. Fast Algorithms for Mining Association Rules. In Proc. of the 20th Int'l Conference on Very Large Databases, Santiago, Chile, September 1994.

[3] H. Mannila, H. Toivonen, and A. I. Verkamo. Discovery of frequent episodes in event sequences. Data Mining and Knowledge Discovery, 1(3), 1997.

[4] Jiong Yang, Wei Wang, and Philip Yu, Mining asynchronous periodic patterns in time series data, Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD), pp. 275-279, 2000.

[5] C. Bettini, X. Sean Wang, and S. Jajodia. Mining temporal relationships with multiple granularities in time sequences. Data Engineering Bulletin, 21:32--38, 1998

[7] J. Han, G. Dong, and Y. Yin. Efficient mining of partial periodic patterns in time series database. In Proc. 1999 Int. Conf. Data Engineering (ICDE'99), Sydney, Australia, April 1999.

[8] J. Han, W. Gong, and Y. Yin. Mining segment-wise periodic patterns in time-related databases. In Proc. 1998 Int'l Conf. on Knowledge Discovery and Data Mining (KDD'98), New York City, NY, August 1998.

[9] Rakesh Agrawal and Ramakrishnan Srikant. Mining Sequential Patterns. In Proc. of the 11th Int'l Conference on Data Engineering, Taipei, Taiwan, March 1995.

[10] Srikant, R., & Agrawal, R. (1996). Mining sequential patterns: Generalizations and performance improvements, . Proc. of the Fifth Int'l Conference on Extending Database Technology (EDBT). Avignon, France.

[11] F. Masseglia, F. Cathala, and P. Poncelet. The PSP Approach for Mining Sequential Patterns. In Proceedings of the 2nd European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD'98), LNAI, Vol. 1510, pages 176-184, Nantes, France, September 1998.

[12] R. J. Bayardo. Efficiently mining long patterns from databases. In Proc. 1998 ACM-SIGMOD Int. Conf. Management of Data, pages 85--93, Seattle, Washington, June 1998.

[13] Sheng Ma, Joseph L.Hellerstein. mining partially periodic event patterns,IEEE,2001

[14] Myra Spiliopoulou. Managing interesting rules in sequence mining. In Poster proceedings of the 3rd European Conf. on Principles and Practice of Knowledge Discovery in Databases PKDD'99, number 1704 in LNAI, pages 554-560, Prague, Czech Republic, Sept. 1999. S

[15]http://kdd.ics.uci.edu/databases/pioneer/pioneer.html , online repository of large data sets