자유트리 기반의 그래프마이닝 기법 분석

노영상 윤은일○ 류근호 김명준
충북대학교
ysnoh@cbu.ac.kr, yunei@cbu.ac.kr, khryu@dblab.cbu.ac.kr, mjkim@cbu.ac.kr

Analysis of Graph Mining based on Free-Tree

YoungSang No, Unil Yun, Keun Ho Ryu, Myung Jun Kim
School of Electrical & Computer Engineering, CBITRC, Chungbuk National University

요 약

데이터마이닝은 현재 매우 각광 받고 있는 분야다. 연관규칙탐사는 트랜잭션 데이터베이스에서 일정빈도 이상의 패턴을 찾아내는 작업을 말한다. 그중 빈발서브그래프패턴 마이닝은 최근 관심이 늘어나고 있으며, 그 활용도 또한 매우 높다. 그래프마이닝은 아이템셋마이닝보다 훨씬 더 많은 계산을 필요로 한다. 중복을 최소화 하는 방법이 필요하며, 그중 가장 좋은 성능을 보이는 GASTON 알고리즘을 분석한다.
키워드 : 그래프 마이닝, 빈발패턴마이닝, 패턴확장기법, 자유트리 확장기법

abstract

Recently, there are many research of datamining. On the transaction dataset, association rules is made by finding of interesting patterns. A part of mining, sub-structure mining is increased in interest of and applied to many high technology. But graph mining has more computing time then itemset mining. Therefore, that need efficient way for avoid duplication. GASTON is best algorithm of duplication free. This paper analyze GASTON and expect the future work.
Keyword : graph mining, frequent pattern minnig, pattern growth method, free tree base extension

1. 서론

데이터 마이닝 영역는 현재 많은 관심을 부르는 분야이다. 그 중 그래프 마이닝 영역은 그 관심이 급증하고 있다. 현실세계의 많은 구조들이(분자구조나, 단백질 구조, 인터넷, 등등) 그래프로 모델링 되어 지기 때문이다. 이러한 데이터로부터 정보들을 찾는 시간이 빨라지고 그 질이 향상될 수록, 각 분야들은 더 빠르게 발전하게 된다. 그래프패턴마이닝[1]은 아이템셋마이닝[2]과는 또 다른 효율성을 요구한다. 그래프의 동형판단작업이 지수 함수적 계산시간을 요구하기 때문이다. 이러한 문제를 해결하기 위한 많은 연구가 진행되고 있다. 그중 주류가 되는 기법이 빈발패턴확장탐색기법(pattern growth method)인데, 이는 빈발하는 그래프패턴에서 빈발하는 간선만을 하나씩 확장해 가면서 모든 빈발그래프패턴을 찾아가는 기법이다. 하지만 단순한 확장탐색은 중복생성을 유발하므로, 이를 효과적으로 제어하는 것이 관건이다.

본문에서는 패턴확장기법의 중복을 효과적으로 제어한 GSATON[3] 알고리즘을 분석하고, 성능을 평가하며, 향후과제를 탐색해 본다.

○ : 교신저자

2. 문제의 정의 및 범주

그래프란 노드와 간선으로 구성되어 있는 구조를 말한다. 그래프를 구성하고 있는 노드의 집합을 V로 정하고, 간선의 집합을 $E = \{(v1,v2) \mid v1,v2 \in V \text{ and } v1 \neq v2\}$로 정한다. 간선에는 방향성이 없으므로, 두 간선 (v1,v2), (v2,v1)는 같은 간선으로 본다. 각 노드와 간선에는 우선순위를 가지는 레이블(label)이 정해져 있다. 레이블의 집합을 $\mathcal{L}$로 정의한다. 레이블링(labeling) 함수 $l : V \cup E \rightarrow \mathcal{L}$ 은 모든 노드나 간선을 그 것의 레이블로 변환한다. G1이 G2의 서브그래프라면($G1 \subseteq G2$) 그래프 G1=(V1, E1, l1), G2=(V2, E2, l2)에 대해서, 다음 두 식을 만족하는 일대일 함수 $f : V1 \rightarrow V2$가 존재한다.

(1). $\forall v \in V1 \Rightarrow l1(v)=l2(f(v))$

(2). $\forall (v1,v2) \in E1 \Rightarrow (f(v1),f(v2)) \in E2$ and $l1(v1,v2)=l2(f(v1),f(v2))$

데이터베이스는 그래프 트랜잭션(transaction)으로 구성되어 있다. 각 트랜잭션은 하나의 연결그래프이다. 연결그래프란 그래프의 모든 노드들 사이에 경로가 존재하는 그래프를 말한다. 어떤 그래프 G의 빈도수(frequency)란 데이터베이스 내에서 그래프 G가 서브그래프인 모든 트랜잭션의 개수를 말한다. 또한 그래프 G의 지지도(support)란 전체 데이터베이스 트랜잭션 개수 중 G의 빈도수의 백분율을 말한다. 마이닝 작업은 데이터베이스 내에서 일정한 빈도수/지지도 이상의 트랜잭션에서 나타나는 모든 연결서브그래프패턴를 찾

는 것이다. 일정 빈도수는 사용자의 요구에 의해 결정되며, 이를 최소지지도(minimum support threshold) 라 한다.

3. 관련연구

그래프패턴마이닝의 초기 알고리즘들[4],[5]은 아이템셋마이닝과 같이 그래프의 노드, 혹은 간선의 크기를 한 단계씩 증가 시키며 진행한다. 하지만 이 방식 아이템셋마이닝에서와는 달리 많은 문제점을 가지고 있다. k-1크기의 패턴으로 만들어 낼 수 있는 후보자패턴의 개수가 여러 개이며, 이를 생성하기 위한 동형판단작업이 너무 많기 때문이다.

gSpan[6] 알고리즘은 DFS를 이용하여 패턴확장기법을 효과적으로 사용하였다. 우향확장(right most extension)을 함으로서 중복회피를 하였다. 우향확장은 최우측경로(right most path)위에서만 확장이 가능하도록 하는 것인데, 그 이외의 곳에서 확장되는 간선은 중복된 패턴을 생성하기 때문이다. 하지만 우향확장만으로는 모든 중복을 없앨 수 없었기 때문에 생성된 그래프의 중복생성판단을 해야 한다. gSpan은 확장탐색으로 생기는 동일한 그래프 중 깊이우선탐색에서 생성순서의 최고우선순위를 가지는 그래프만을 선택하도록 했으며 이는 매우 효율적인 중복확인을 가능하게 하였다. 생성된 그래프가 최고우선순위를 가지는지 확인하기만 하면 되기 때문이다. gSpan은 매우 효율적인 방법이며 많은 알고리즘들이 이를 응용하였다. 하지만 gSpan의 DB내에서의 빈도수 계산의 효율성은 개선되어야 할 점이다.

4. GASTON 알고리즘

GASTON[3]은 현재 패턴확장 기법의 중복을 가장 효과적으로 회피한 알고리즘이다. GASTON을 그래프의 확장영역을 경로영역(path), 트리영역(tree), 순환그래프영역(cyclic graph)로 분리 하였다. 그리고 각각의 다른 알고리즘을 적용하여 효율성을 높였다. 특히 트리영역에서의 중복 없는 확장이 알고리즘의 성능을 향상시켰다. 먼저 모든 경로(path)를 중복 없이 생성한다. 트리는 경로(path)에 의존적으로 확장하여 최초의 경로를 변화시키기 않는 범위 내에서 확장을 한다. 이는 경로로써 트리영역을 분할한 것과 같다. 순환그래프영역은 트리에 순환간선(cyclic edge)의 확장으로부터 생성되는데, 한번 순환그래프로 확장되면, 더 이상 노드 확장을 하지 않고, 순환간선확장만을 하여 중복확장을 줄였다. 구체적인 제한을 살펴보도록 한다.

4.1 경로(Path)영역의 중복회피

가장 기본이 되는 것은 경로그래프를 중복 없이 확장하는 것이다. 경로그래프의 영역 안에서 패턴확장을 한다고 하면, 경로그래프는 방향성이 없으므로 두 개의 서브경로그래프로부터 중복확장이 되어 진다. 이중 하나의 서브경로그래프에서만 생성이 가능하도록 제약을 두어 중복생성을 억제하게 된다. 임의의 경로그래프 P 가$v_1e_1v_2e_2...v_{n-1}e_{n-1}v_n$와 같다면, P는 문자열 $l(v_1)l(e_1)...l(v_n)$와 $l(v_n)l(e_{n-1})...l(v_1)$로 표현될 수 있으며, 두 문자열은 사전적 우선순위가 같거나, 또는 둘 중 하나가 높은 사전적 우선순위를 갖는다. 이점을 이용하여 P가 생성될 때, P의 문자열 중 우선순위가 높은 문자열의 뒤쪽 마지막 튜플($l(e')l(v')$)의 확장으로만 P가 생성되도록 하는 것이다.

4.2 트리영역의 중복회피

트리란 모든 노드들 사이에 단지 한 개의 경로만 존재하는 구조를 말 한다. 어떤 특정 노드를 루트로 하지 않고, 단지 트리의 구조적 측면을 생각하는 트리를 자유트리(free tree)라고 부른다. 자유트리는 최대 길이의 경로로에 의한 특별한 성질을 가진다. 어떤 자유트리 T의 어떤 최대길이경로를 P라 한다면,($P=\{v_1...v_m\}\subseteq V_T$), T의 모든 최대길이경로들은 반드시 P위의 1개의 노드를 지나거나, 아니면 1개의 간선을 지나게 된다. (P의 노드의 개수(m)가 홀수이면, 모든 최대길이경로들은 v_i를 지나고, 이 노드를 T의 중심노드(centre)라 한다. 한편, P의 노드의 개수(m)가 짝수라면, 모든 최대길이경로들은 v_i, v_{i+1}를 잇는 간선을 지나고, 이 노드들을 T의 이중중심노드(bicentre)라 한다.)

루트로 중심으로 시작되는 최대길이경로들 중 가장 우선순위가 가장 높은 2개의 문자열을 정하여 이를 백본문자열(backbone string)이라고 한다. (이중중심노드-자유트리에서는 각각의 루트를 중심으로 하는 트리로 나누어 가장우선순위가 높은 문자열 1개씩을 정한다.) 이 두 개의 문자열로 구성되는 최대길이경로를 자유트리의 백본이라 한다. 모든 자유트리는 백본을 가지고 있으므로, 모든 트리영역을 같은 백본을 가지는 트리들의 영역으로 구분하는 것이다. 자유트리의 확장은 이 백본을 변화시키지 않는 범위 내에서만 확장된다. 자유트리의 실제 확장은 중심노드를 루트로 하고, 깊이가 $\lceil \frac{m}{2} \rceil - 1$인 루트트리처럼 운용하게 된다. 단, 이중중심노드일 경우는 각각의 중심노드별로 나누어진 각각의 루트트리처럼 사용한다. 확장도중 백본이 변하지 않기 위해서는 확장이 최대길이 경로길이를 넘지 않게 해야 한다. 또한, 루트로부터 새로 생긴 노드까지의 경로의 레이블들의 우선순위가 백본의 우선순위를 넘지 않아야 한다. 구체적으로, 중심노드트리 T에서 $\lceil \frac{m}{2} \rceil - 1$ 깊이에 노드가 확장될 때에는, 루트로부터의 새로 생긴 노드까지 문자열의 우선순위는 최대 두 번째 백본문자열의 우선순위와 같거나 낮다. 단, 새로 생긴 문자열이 첫 번째 백본문자열 위에 있을 경우는 첫 번째 백본문자열 보다 같거나 낮다. 이중중심노드트리에서 $\lceil \frac{m}{2} \rceil - 1$ 깊이에 노드가 확장될 때에는, 새로 생겨난 경로의 문자열의 우선순위는 그것이 생긴 루트트리의 백본문자열의 우선순위와 같거나 낮아야 한다.

백본을 변화시키지 않는 범위 내에서의 중복 없는 모든 확장을 위해서는 Depth Sequence를 사용한다. 그런데 이를 자유트리에 적용하기 위해서는 몇 가지 세심한 기술이 필요하다. 먼저, 백본 위의 일부 튜플(노드확장$l(e')l(v')$)의 DFS 우선순위보다도 높은 우선순위를 갖는 튜플 들도 백본을 변화시키지 않는다면 확장에 이용되어야 하기 때문에, GASTON에서는 이를 피하기 위해 백본을 가장 큰 우선순위로 만드는 임의의 함수를 사용한다. 또 다른 점은, 백본의

두 문자열이 서로 같을 경우에, 일반 깊이우선탐색은 중복된 확장을 야기함으로 이점에 대해서도 신경 써야 한다. 자유트리가 이중중심노드트리라고 한다면, 먼저 한쪽의 트리에서 가능한 모든 확장을 하며, 나머지 한쪽의 트리의 확장은 각각 첫 번째 트리의 Depth Sequence를 넘지 않는 범위내로 확장을 함으로서 중복을 피할 수 있다.

4.3 순환그래프영역의 중복회피

순환그래프에 대한 중복확장방지방법은 아직 없다. 그래서 확장된 그래프가 중복되었는지 아닌지를 효과적으로 판단하는 방법에 대한 연구가 필요한 부분이기도 하다. GASTON에서는 동형판독을 위해 그래프의 정규형(canonical form)을 사용한다. 그래도 GASTON에서는 이 영역의 중복확장을 억제하기 위하여 특별한 제한을 두었다. 경로나 트리가 순환그래프를 만드는 순환간선확장을 한 이후부터는 노드확장없이 순환간선확장만을 허용하는 것이다. 순환간선들 간에는 우선순위를 정할 수 있다. 간선이 연결하는 노드의 확장생성순서와 간선의 라벨을 가지고 그 순환간선확장을 튜플 $l(v_i)l(v_j)l(e_m)$, ($v_i,v_j \in V_T$, e_m은 생성되는 순환간선)로 표현할 있고, 이로써 튜플의 우선순위를 정할 수 있다. 이 하나의 트리 안에서의 이 튜플들은 아이템 셋과 같이 운용될 수 있으며, 중복확장을 제어할 수 있게 된다.

표 1. **GASTON 알고리즘**

```
Find-Paths(A path P, a set of leg L)
  for L중 중복된 path를 생성하지 않는 모든 l에 대하여 do
     G′ = P에 l을 확장한 그래프;
     if l이 노드확장이면 do
        L′ = Extend(l) ∪ {Join(l,l′)|l′(≠l)∈L};
        if G′가 path라면 then
           Find-Paths(G′,L′) ;
        else
           Find-Trees(G′,L′);
     else
        L′ = L′ ∪ {Join(l,l′)|l′(≠l)∈L};
        Find-CyclicGraphs(G′,L′);

Find-Trees(A tree T, a set of leg L)
  for L중 백본을 변화시키지 않는 모든 l에 대하여 do
     G′ = T에 l을 확장한 그래프;
     if l이 노드확장이면 do
        //백본제약과 중복제어를 하며 확장한다.
        L′ = Restricted-Extend(l);)
        L′ = L′ ∪ {Join(l,l′)|G′에 의해 허락되는 l′∈L};
        Find-Trees(G′,L′);
     else
        L′ = L′ ∪ {Join(l,l′)|l′(≠l)∈L};
        Find-CyclicGraphs(G′,L′);

Find-CyclicGraphs(A graph G, a set of leg L)
  for L중 확장 가능한 모든 l에 대하여 do
     G′ = G에 l을 확장한 그래프;
     L′ = L′ ∪ {Join(l,l′)|l′(>l)∈L};
     Find-CyclicGraphs(G′,L′);
```

4.4 알고리즘

이를 종합한 GASTON의 대략적인 알고리즘은 표2이다. 확장은 처음 경로그래프로부터 시작하며 점차 순환그래프로 루틴으로 탐색해 간다. Extend()는 확장도중 그래프에서 확장 가능한 모든 간선을 찾는 함수 이며, 그래프는 모든 확장 가능한 간선을 기억해 둔다. Join()은 이 확장 가능한 간선들과 새로 확장된 간선과의 동시에 발생하는 간선들만을 간추리는 작업을 하게 된다. [7]

5. 추후연구

패턴확장기법의 중복을 제어하는 작업은 그래프마이닝의 기본이 되는 작업이므로, 여기에 다른 성질(property)를 적용하여 알고리즘을 개선할 수 있다. 빈발패턴들 중에 자신과 같은 빈도수를 가지는 슈퍼셋(superset, 확장된 그래프)이 없다면 그 패턴을 폐쇄형패턴(closed pattern)이라고 하는데, 이것들을 이용하여 동일하면서도 반복된 작업을 효과적으로 줄일 수 있다.[8] 또한 특정한 제약조건(constraint)이 있는 작업을 하도록 할 수도 있다.[9]

표 2. 실행 시간 비교

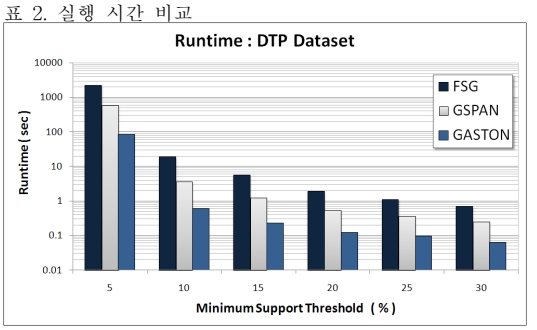
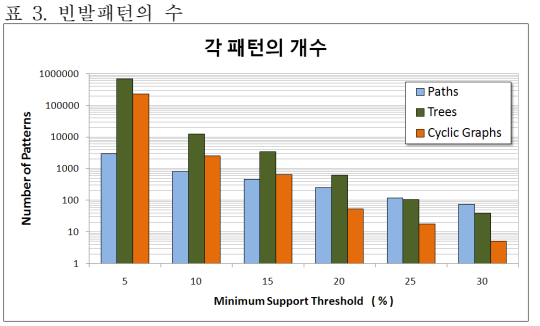


표 3. 빈발패턴의 수



6. 성능시험

실험은 그래프마이너 중 apriori-based 알고리즘인 FSG[5]와 패턴확장알고리즘의 가장 기본적인 알고리즘인 gSpan[6], 그리고 GASTON[2]의 성능을 비교한 것이다. 실험 환경은 OS: fedora9 (linux kernel 2.6), CPU: AMD Athlon64-X2 2.4GHz, Memory: 768MB(DDR2 6400)이다. 실험에 사용한 데이터는 DTP(Compound_422)로 422개의 화합물 그래프데이터로써, gSpan과 함께 제공 되어지는 것을 이용하였다. 지지도를 변화시켜가면서 얻어진 수행시간은 표2과 같다. FSG는 그래프마이닝에서의 apriori의 비효율성으로 인해 가장 낮은 성능을 보이며, gSpan은 패턴확장기법의 효율성으로 FSG보다 좋은 성능을 보인다. GASTON은

gSpan 보다도 6배정도 빠른 수행시간을 보여준다. 이는 트리패턴에서의 효율성이 잘 나타난 결과로 보여진다. 표3는 지지도의 변화에 따라 찾아지는 패턴의 개수를 보인 것이다. 지지도가 작아질 수 록 패턴의 개수가 급격하게 증가하는 것을 보인다. 패턴의 구조 중 트리구조의 패턴의 개수가 가장 많은 비중을 차지하는 것을 볼 수 있다. GASTON의 자유트리확장의 효율성이 더욱 잘 나타나게 한다는 것을 알 수 있다.

7. 결론

본문은 확장탐색기법의 알고리즘 중 가장 좋은 성능을 보이는 GASTON을 분석한 것이다. 확장탐색기법의 중복제어는 패턴마이닝의 하위영역(low-level)의 작업이다. 이 부분의 발전은 다른 알고리즘의 발전에 기반이 된다. GASTON 트리영역에서 효율적 확장을 이용하여 이 부분에서의 성능을 매우 향상시켰다. 하지만 순환그래프 영역에서의 중복은 제어하지 못한 것은 아쉬운 점이며, 앞으로 더 연구가 필요하다.

8. RRFERENCE

[1] 이순행 장민희 도영주 김상욱 "그래프 데이터베이스에서 빈발 서브그래프 마이닝" 데이타베이스연구회 학회지, VOL.23, NO.03, pp.0021~0037, 2007

[2] R. Agrawal, T. Imilienski, and A. Swami. "Mining association rules between sets of items in large database." ACM SIGMOD, Volume 22, Issue 2, pp.207-216, 1993

[3] Siegfried Nijssen, Joost N. Kok "A quickstart in frequent structure mining can make a difference". In Proc. Int'l Conf. on ACM SIGKDD Int'l Conference on Knowledge Discovery and Data Mining, KDD, pp.647-652, 2004.

[4] Akihiro Inokuchi, Takashi Washio, Hiroshi Motoda "An apriori-based algorithm for mining frequent substructures from graph data". In Proc. Int'l Conf. on the European Conference on Principles and Practice of Knowledge Discovery in Databases, PKDD, pp.13-23, 2000.

[5] Michihiro Kuramochi, George Karypis "Frequent subgraph discovery". In Proc. Int'l Conf. on IEEE International Conference on Data Mining, ICDM, pp.313-320, 2001.

[6] Xifeng Yan, Jiawei Han "gSpan: Graph-based substructure pattern mining". In Proc. Int'l Conf. on IEEE International Conference on Data Mining, ICDM, pp.721, 2002

[7] Yun Chi Yirong Yang Muntz, R.R. "HybridTreeMiner: an efficient algorithm for mining frequent rooted trees and free trees using canonical forms" Scientific and Statistical Database Management, 2004. Proceedings. 16th International Conference on, June 2004

[8] Xifeng Yan, Jiawei Han "CloseGraph: Mining Closed Frequent Graph Patterns" In Proc. Int'l Conf. on the International Conference on Knowledge Discovery and Data Mining, ACM SIGKDD, pp.286-295, 2003

[9] Feida Zhu, Xifeng Yan, Jiawei Han, and Philip S. Yu, "gPrune: A Constraint Pushing Framework for Graph Pattern Mining", in Proc. 2007 Pacific-Asia Conf. on Knowledge Discovery and Data Mining (PAKDD'07), Nanjing, China, May 2007