2002년 한국정보처리학회 춘계학술발표논문집 제9권 제1호

가상 조인을 이용한 Eclat 알고리즘의 최적화

김계형, 김민호, R.S. Ramakrishna
광주과학기술원 정보통신공학과
e-mail : kgh4001@kjist.ac.kr

Optimizing Eclat Algorithm by Using Virtual Join

Gye Hyung Kim, Minho Kim, R.S. Ramakrishna
Dept. of Information and Communications, K-JIST

요 약

본 논문에서는 데이터 마이닝의 중요한 기법 중 하나인 연관 규칙 발견을 위한 Eclat 알고리즘의 최적화를 위한 가상 조인을 제안하고자 한다. 연관 규칙 발견을 위한 알고리즘 중 특히 Eclat 알고리즘은 효과적으로 다빈도 항목집합을 발견하는 알고리즘으로 알려져 있고, 가상 조인은 이러한 Eclat 알고리즘의 불필요한 교집합 연산을 미리 피함으로써 성능 향상을 얻을 수 있다. 이는 실험 결과를 통해서도 확인할 수 있다.

1. 서론

지식 탐사의 한 연구 분야인 데이터 마이닝의 기법에는 연관 규칙, 클러스터링, 분류, 유전자 알고리즘 등이 있다. 그 중 연관 규칙 (association rule)은 데이터 마이닝의 중요한 기법중의 하나이다. 연관 규칙은 항목집합에서 각 항목의 연관성을 알아내는 규칙으로 [1]에서 처음 소개 되었다.

데이터베이스의 항목의 집합을 $I = \{i_1, i_2, i_3, i_4, \ldots\}$라고 할 때 k 개로 구성된 아이템들의 집합을 *k-itemset* 이라고 한다. 항목집합 (*Itemset*) X 를 포함하고 있는 전체 데이터베이스에 대한 트랜잭션의 빈도수를 *support*, $sup(X)$ 라 하고, 임의의 두 항목집합 X, Y 에 대한 조건 확률 $sup(X \cup Y)$ / $sup(X)$를 신뢰도 (*Confidence*, $conf(X, Y)$)라 한다. 따라서 $conf(X, Y) = c$ 를 가지는 연관 규칙 $X \Rightarrow Y$ 란, $X, Y \subset I$ 이고 $X \cap Y = \emptyset$인 항목집합 X, Y에 대한 조건적인 암시로써, X가 일어났다는 가정아래 그의 c 에 해당하는 확률 만큼의 Y 가 함께 일어남을 의미한다.

연관 규칙을 위한 데이터 마이닝 단계는 크게 (1) 다빈도 항목집합 (Frequent Itemset)을 찾는 단계와 (2) 다빈도 항목집합 사이에서 규칙을 찾는 단계로 구성되어 있다. 이 중 다빈도 항목집합을 찾는 단계가 알고리즘에서 가장 많은 계산 시간을 요구하는 단계이기 때문에 대부분의 연구가 (1)에 초점을 맞추고 있다. 본 논문에서 다룰 동등 클래스 (Equivalence Class, eqclass) 기반 연관 규칙 마이닝 알고리즘인 Eclat 역시 다빈도 항목집합을 효과적으로 발견하는 방법을 효율적으로 처리하기 위한 것이며, 본 논문도 단계 (1)에 중점을 둔다.

본 논문에서는 이런 Eclat 알고리즘을 최적화 하기 위한 가상 조인 (Virtual Join)에 대해 제안 한다. 가상 조인은 단 한번의 검색 (Lookup)을 통해 비교적 비용이 큰 조인을 막을 수 있다. 이는 실험 결과에서도 그 효율성을 볼 수 있다.

본 논문의 구성은 다음과 같다. 2 장에서는 관련 연구로 연관 규칙의 알고리즘들에 대해 설명하고 있고 3 장은 Eclat 알고리즘에 대해 자세히 기술한다. 4 장에서는 조인에 관해 예를 들어 자세히 설명하고 5 장에서는 제안한 가상 조인에 대해 설명하고 있다. 그리고 6 장에서는 실험결과, 7 장에서는 결론을 제시한다.

2. 관련 연구

대부분의 연관 규칙 마이닝 알고리즘들 (Apriori[2], DHP[3], DIC[4], Partition[5], SEAR[6] 등)은 다빈도 항목집합의 모든 부분 집합은 다빈도 항목집합 이어야 한다는 직관적 사실을 이용하는 Apriori 알고리즘에 기반을 두고 있다. 하지만, Apriori 알고리즘은 데이터베이스와 해쉬 트리(Hash Tree)와 같은 복잡한 데이터 구조체에 대한 반복적인 순환이 필요하다. 그래서, SEAR 와 같은 알고리즘은 복잡한 구조체에서의 계산

량을 줄이기 위한 시도를 하였으며, Partition 이나 DIC 와 같은 알고리즘은 많은 비용을 요구하는 데이터베이스의 스캔을 줄이는 시도를 하였다.

이와는 달리 Apriori 기반 알고리즘들과는 다른 독특한 알고리즘이 제안되었는데, 바로 동등 클래스에 기반을 둔 알고리즘이다 [7]. 이 알고리즘은 탐색 공간을 독립적인 eqclass 라는 메모리에 충분히 들어갈 수 있는 작은 단위로 분할함으로써 미리 계산된 독특한 데이터베이스 (Vertical Tid-list DB)에 대해 단 한번의 스캔만으로 모든 다빈도 항목집합을 찾을 수 있다. 또한 이 알고리즘은 단순한 교집합 연산만이 요구되기 때문에 Apriori 기반 알고리즘에서 사용되는 복잡한 데이터 구조체로 인해 야기되는 공간과 계산상의 과부하를 피할 수 있다.

3. Eclat

Eclat 은 접두사 (*prefix*) 기반 동등 클래스를 단위로 각 동등 클래스 (eqclass)에 포함된 모든 다빈도 항목집합을 찾는 알고리즘이다. 그림 1 은 Eclat 알고리즘을 적용한 예이다. 이 그림은 2-itemset 이 F_2 에서 생성된 길이 1 의 접두사 기반 동등 클래스 단위로 처리되고 있음을 보여준다. F_2 에서 생성된 길이 1 의 접두사 기반 동등 클래스는 [A] = {AB, AC, AD, AE}, [B] = {BC, BD, BE}, [C] = {CE} 이다. 그림에서 직사각형 안에 있는 항목집합은 다빈도 항목집합을 나타내며, k-itemset 을 생성하기 위해 결합된 (k-1)-itemset 들을 나타내기 위해 실선을 이용하여 표시하였다. 앞에서 Eclat 은 F_2 에서 생성된 eqclass 단위로 처리하는 알고리즘이라 했는데, 이것은 최초로 분리된 eqclass 단위로 포함된 다빈도 항목집합을 찾는 것을 의미한다. 예를 들어 그림 1 에서 eqclass [A]에 포함된 모든 다빈도 항목집합을 완전히 다 찾은 다음에 eqclass [B]에서 찾고, 다음으로 eqclass [C]에서 찾게 된다.

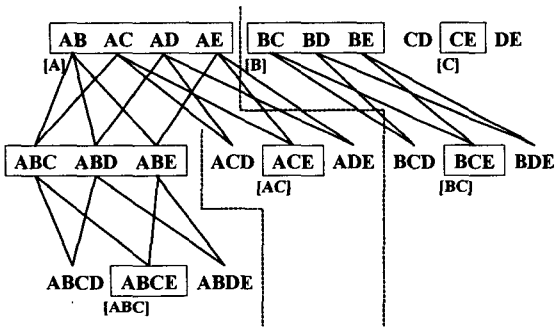


그림 1. Eclat 에 의한 다빈도 항목집합의 나열

다빈도 항목집합의 가능성이 있는 새로운 후보 항목집합 (Candidate Itemset)의 생성은 동일한 eqclass 내로 한정한다. 왜냐하면 서로 다른 eqclass 내에 포함되어 있는 항목집합을 결합하여 생성한 후보 항목집합은 동일 eqclass 내에서 만들어진 후보 항목집합과 중복되기 때문이다. 예를 들어, eqclass [A]에 포함된 AB 와 eqclass [B]에 포함된 BC 를 결합하여 생성한 ABC 는 eqclass [A]에 포함된 AB 와 AC 를 결합하여 생성한 ABC 와 중복된다. 마지막으로 각 eqclass 에 대한 탐색은 더 이상 새로운 다빈도 항목집합을 생성할 수 없을 때 종료 된다.

4. 조인 (Join)

조인은 두 (k-1)-itemset 을 결합하여 새로운 다빈도 k-itemset 을 결정하기 위한 전체 과정을 의미한다. 그 첫 번째 단계에서는 임의의 eqclass 내에 있는 두 (k-1)-itemset 을 결합하여 새로운 후보 k-itemset 을 생성한다. 다음으로 후보 항목집합의 트랜잭션 리스트 (Tid-list)를 결정하기 위해 두 (k-1)-itemset 의 트랜잭션 리스트들의 교집합을 구한다. 마지막으로 후보 항목집합의 지지도(support), 즉, 새로 생성된 트랜잭션 리스트의 크기를 최소 지지도(minimum support)와 비교해서, 최소 지지도보다 크거나 같은 경우 빈번하기 때문에 다음 단계를 위해 저장되고, 그렇지 않을 경우 다음 조인을 하게 된다. 다음은 그 예제이다.

Example: Eqclass [A] = {AC, AD}와 tid-list(AC) = {1, 2, 3, 4, 5, 6}, tid-list(AD) = {5, 6, 7, 8, 9}, tid-list(CD) = {5, 6}가 주어졌다고 가정하자. 그리고 min_sup 가 3 이라고 하자. AC 와 AD 의 조인의 첫 단계로써 새로운 후보 항목집합 ACD 를 생성한다. 다음으로 ACD 의 tid-list 를 구하기 위해 tid-list(AC)와 tid-list(AD)를 교집합 하여 얻는다. 즉, $tid\text{-}list(ACD) = tid\text{-}list(AC) \cap tid\text{-}list(AD) = \{5, 6\}$가 된다. ACD 의 지지도를 min_sup 와 비교하여 min_sup 값 보다 큰지를 판단한다. $support(ACD) = |tid\text{-}list(ACD)| = 2 < min\_sup$ 이므로, 후보 항목집합 ACD 는 빈번하지 않은 항목이다. 따라서 다음 조인으로 넘어간다. 만약 $support(ACD) \geq min\_sup$, 즉, 빈번한 항목인 경우 ACD 는 다음 단계를 위해 eqclass [AC]에 포함시킨다.

4.1. 관찰

조인에 대한 몇 가지 관찰을 해보자. 첫 번째로 교집합 연산은 간단한 연산이긴 하지만 비교적 비용이 높은 연산이다. 특히 트랜잭션 리스트의 크기가 클수록 더 심해진다. 비용이 높은 교집합 연산을 줄일 수 있는 방법으로써, 새로 생성된 후보 항목집합이 다빈도 항목집합 일 가능성이 있는지를 먼저 검사하는 방법이 있다. 이 방법은 잘 알려진 "다빈도 항목집합의 모든 부분 집합 또한 다빈도 항목이다"는 사실을 이용한다. 이것은 임의의 다빈도 항목집합의 모든 부분집합 S 에 대해 $support(S) \geq min\_sup$ 이어야 하며, $support(S) < min\_sup$ 인 S 가 하나라도 존재하면 새로 생성된 후보 항목집합은 다빈도 항목집합이 될 수 없음을 의미한다. 따라서 후보 항목집합의 부분집합 중에 빈도수가 낮은, 즉, $support(S) < min\_sup$ 인 부분집합을 미리 찾는다면 교집합을 하지 않고서도 빈도수가 낮은지를 판단할 수 있게 된다. 예를 들어, 앞의 예제

에서 후보 항목집합 ACD 의 부분 집합 중의 하나인 CD 의 지지도가 2 로써 최소 지지도 보다 작기 때문에 ACD 는 다빈도 항목집합 일 수 없다는 걸 알 수 있다. 이 때 비용이 훨씬 낮은 지지도 테이블의 조사만으로 비용이 높은 교집합을 하지 않아도 된다.

하지만, 이 방법에도 단점이 존재한다. 후보 항목집합의 크기가 커짐에 따라, 그 부분 집합의 수는 기하급수적으로 늘어나기 때문에, 그에 따라 추가 비용도 기하 급수적으로 늘어난다. 이것은 2-itemset 부터 (k-1)-itemset 까지의 모든 부분 집합에 대한 지지도 테이블을 유지해야 하고, 후보 항목집합의 모든 부분 집합을 생성해야 하며, 그들에 대해서 지지도 테이블을 검색해야 하기 때문이다.

이러한 문제점을 해결할 수 있는 방법을 고안하기 위해 또 다른 관찰을 해보자. 후보 항목집합 ACD 가 다빈도 항목집합일 가능성이 있는지를 판단하기 위해 모든 부분 집합의 지지도를 확인할 필요는 없다. 왜냐하면 ACD 를 생성할 때 사용된 AC 와 AD 가 이미 다빈도 항목이라는 것을 알고 있기 때문이다. 따라서 CD 의 지지도만 확인하면 된다.

이것을 일반화하면 다음과 같다. Eclat 의 조인은 동등 클래스를 기반으로 이루어지기 때문에 새로운 후보 항목집합의 생성은 $pX+pY=pXY$ 의 형태를 지닌다. 여기에서 pX 와 pY 가 다빈도 항목이기 때문에 그들 각각의 모든 부분 집합은 이미 다빈도 항목이며, pXY 에서 새로 생성되는 크기 2 의 부분 집합은 XY 뿐이다. 마지막으로 Eclat 알고리즘에서는 모든 2-itemset 의 지지도에 대한 검색 테이블이 이미 존재하며 항상 접근이 가능하다.

5. 가상 조인(Virtual Join)

위의 관찰들을 종합해 볼 때, 실제 조인 이전에 후보 항목집합이 다빈도 항목집합일 가능성을 확인하기 위해 크기 2 의 부분 집합만을 미리 고려하는 것은 훌륭한 타협점이며, 이 경우 단 한번의 검색만이 필요하다. 이것이 가상 조인 알고리즘이며 다음과 같이 요약할 수 있다.

(For the join of pX and pY, i.e., $pX+pY=pXY$)

If support(XY) < min_sup,
    goto the next join
else (i.e., support(XY) ≥ min_sup),
    do the real join

그림 2 는 가상 조인을 적용한 결과이다. 그림에서 굵은 실선은 실제 조인을 의미하며 일점 쇄선은 가상 조인을 통해 실제 조인을 막은 경우를 의미한다. 이 예제에서 CD 와 DE 가 빈도수가 낮다는 사실을 이용하여 총 12 번의 조인 중에 6 번 (ACD, ADE, BCD, BDE, ABCD, ABDE)의 실제 조인을 피할 수 있었다.

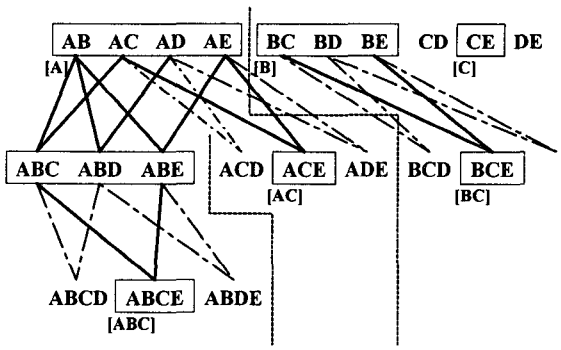
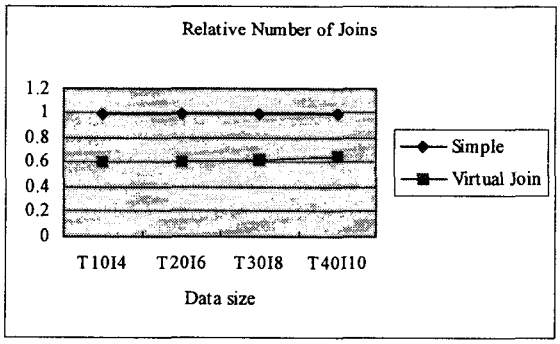


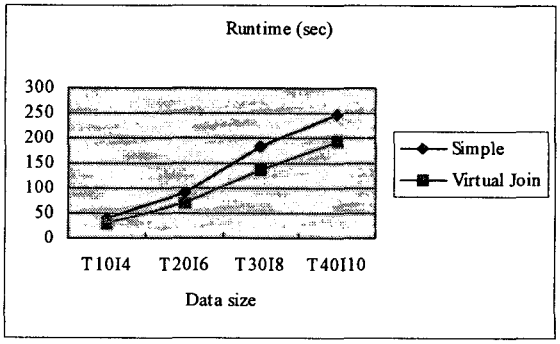
그림 2. 가상 조인을 사용했을 때 Eclat 에서의 다빈도 항목집합의 나열

6. 실험 결과

본 실험은 Pentium III 733Mhz, 256MB PC 를 이용하여 실행하였다. 첫번째 실험에 사용된 데이터 베이스는 T10I4D100K, T206D100K, T30I8D100K, T40I10D100K 이다. 여기서 T 는 평균 트랜잭션 크기를, I 는 평균 항목집합의 크기를, D 는 트랜잭션의 수를 나타낸다. 두번째 실험에서는 T10I4 에 각 트랜잭션 수를 다르게 하여 실험하였다. 모든 실험에서 지지도는 0.1%로 설정하였다.



(a) 데이터 크기에 따른 조인 횟수

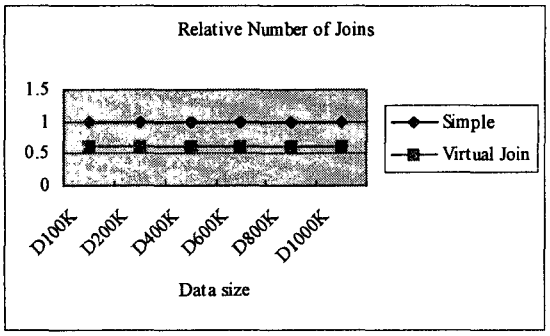


(b) 데이터 크기에 따른 실행 시간

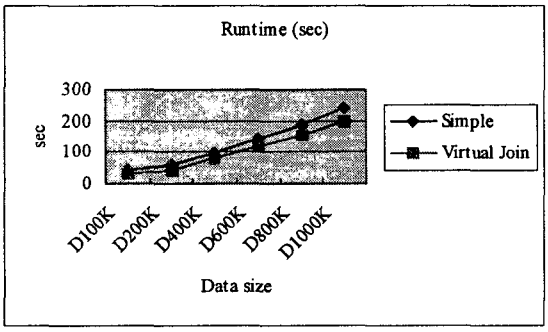
그림 3. Tx 와 Iy 를 변화시킬 때 조인 횟수와 실행 시간의 변화

첫번째 실험은 데이터 베이스에 포함된 트랜잭션의 수를 100K (D100K)로 고정시키고 트랜잭션의 평균 크기 (Tx)와 항목집합 패턴의 평균 크기 (Iy)를 바꿔 가며 실험을 한 것이다. 그림 3 에 그 결과를 보여주고 있다.

두 번째 실험은 트랜잭션의 평균 크기 (Tx)와 항목집합 패턴의 평균 크기 (Iy)를 각각 10 과 4 로 고정시키고 데이터 베이스에 포함된 트랜잭션의 수를 바꿔 가며 실험을 한 것이다.



(a) 트랜잭션의 크기에 따른 조인 횟수



(b) 트랜잭션의 크기에 따른 실행 시간

그림 4. Dz 를 변화 시킬 때 조인의 횟수와 실행 시간의 변화

그림 4 에서 보여진 것처럼, 상대적인 조인의 수는 평균적으로 61%로써 거의 일정하지만, 데이터 베이스의 트랜잭션 수가 증가함에 따라 실행 시간은 더 많이 절약할 수 있었다.

7. 결론

본 논문에서는 Eclat 의 성능 향상을 위한 가상 조인 알고리즘을 제안하였다. 제안된 알고리즘은 조인의 횟수를 줄임으로써 Eclat 알고리즘을 최적화 시켰다. 이는 실험을 통해서도 가상 조인의 적용이 성능 향상의 결과를 보여주는 것을 알 수 있었다. 또한 실험 결과를 통해 가상 조인 알고리즘은 트랜잭션이 더 많은 패턴을 생성할수록,데이터 베이스의 트랜잭션의 수가 많아 질수록, 더 많은 실행 시간을 절약할 수 있음을 알 수 있었다. 제안한 가상 조인 알고리즘은 바이오인포매틱스와 같은 분야에서 생성되는 막대한 데이터에서 연관 규칙을 발견하는데 효과적으로 이용될 수 있을 것이다.

감사의 글

본 연구는 교육부 두뇌한국 21(BK21) 정보기술사업단의 지원에 의한 것입니다.

참고문헌

[1] R. Agrawal, T. Imielinski, and A.Swami. "Mining Association Rules Between Sets of Items in Large Databases," Proc. ACM SIGMOD Conf. Management of Data, pp. 207-216, 1993.

[2] R. Agrawal, R. Srikant. "Fast Algorithms for Mining Association Rules," Proc. the 20th Int'l Conf .VLDB, 1994.

[3] Bing Liu, Wynne hsu and Yiming Ma. "Mining Association Rules with Multiple minimum Supports," Proc. ACM KDD, 1999.

[4] S. Brin, R. Motwani, J.D. Ullman, S. Tsur. "Dynamic Itemset Counting and Implication Rules for Market Basket Data," Proc. ACM SIGMOD Conf. Management of Data, pp. 255-264, 1997.

[5] J.S. Park, M. Chen, and P.S. Yu. "An Effective Hash based Algorithm for Mining Association rules," ACM SIGMOD Int'l. Conf. Management of Data, 1995.

[6] A. Mueller. "Fast Sequential and Parallel Algorithm for Association Rule Mining: A Comparison," Tech. Report CS-TR-3515, 1995.

[7] M.J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li. "New Algorithms for Fast Discovery of Association Rules," Proc. the 3rd Int'l. Conf. KDD, pp. 283-286, 1997.