분산형 FP트리를 활용한 병렬 데이터 마이닝

조 두 산 김 동 승
고려대학교 전기공학과

Parallel Data Mining with Distributed Frequent Pattern Trees

Doosan Cho and Dongseung Kim
Dept. of Electrical Engineering, Korea University, Seoul, Korea
E-mail : mew26@classic.korea.ac.kr

Abstract

Data mining is an effective method of the discovery of useful information such as rules and previously unknown patterns existing in large databases. The discovery of association rules is an important data mining problem. We have developed a new parallel mining called *Distributed Frequent Pattern Tree* (abbreviated by DFPT) algorithm on a distributed shared nothing parallel system to detect association rules. DFPT algorithm is devised for parallel execution of the FP-growth algorithm. It needs only two full disk data scanning of the database by eliminating the need for generating the candidate items. We have achieved good workload balancing throughout the mining process by distributing the work equally to all processors. We implemented the algorithm on a PC cluster system, and observed that the algorithm outperformed the *Improved Count Distribution* scheme.

Ⅰ. 서론

데이터 마이닝은 데이터베이스로부터 잠재적이며 유용한 정보를 추출하는 방법으로 최근 정보에 대한 가치 인식이 높아지면서 많은 연구가 진행되어 왔다. 데이터 마이닝은 연관규칙 (association rules), 분류 (classification), 클러스터링 (clustering) 등의 방법을 통해 다양하게 구현된다. 본 논문에서는 연관규칙에 관한 병렬 알고리즘을 제안한다. 연관규칙은 구매자의 트랜잭션 (transaction) 데이터로부터 추출되며, 예를 들어 A, B가 연관성이 높다는 규칙은 "상품 A를 사는 경우 상품 B도 같이 구매한다"로 해석되며 전자상거래, 의사결정 지원, 의료 등의 다양한 분야에서 활용될 수 있다.

연관규칙 마이닝 알고리즘 중에서 Frequent Pattern Tree (FPT)를 이용하는 순차 (sequential) 알고리즘은 기존의 알고리즘에 비하여 실행시간이 수배이상 개선됨이 보고 되었다 [4]. 대용량의 데이터베이스를 대상으로 연관규칙 마이닝을 실행할 경우 막대한 시간이 소요되어 실행시간의 단축을 위하여 FPT의 병렬실행이 필요하다. 특히 전체 실행시간에서 디스크 입출력의 비중이 높은 연관규칙 마이닝의 경우 분산 시스템을 써서 병렬로 실행시키는 것이 효과적이다. 따라서 PC 클러스터와 같은 고성능 병렬 시스템에 적합한 알고리즘을 개발하고자 본 논문에서는 분산형 FP트리 (DFPT: Distributed Frequent Pattern Tree)알고리즘을 제안한다. DFPT 알고리즘은 FPT를 병렬화한 형태로서 디스크 입출력과 동기화 및 통신을 최소화하였으며 부하를 균등화하도록 설계하였다.

본 논문의 구성은 다음과 같다. 2장에서는 DFPT 알고리즘에 대하여 상세히 기술한다. 3장에서는 부하 분산 기법에 대하여 기술한다. 4장에서는 알고리즘의 실험결과를 비교, 분석한다. 5장에서는 본 논문의 연구결과를 요약하고, 앞으로의 연구방향에 대해 논한다.

이 논문은 한국과학재단 특정기초연구 (R01-2001-00341)의 지원에 의해 수행되었음.

Ⅱ. 분산형 FP트리

DFPT 알고리즘은 다음 두 단계로 구성된다. 첫 단계에서는 각 프로세서가 지역 (local) 디스크의 데이터베이스로부터 DFPT를 구성하고, 구성된 지역 트리를 대상으로 각 프로세서에서의 계산량을 균등하게 조정한다. 두번째 단계에서는 부하 균등화 과정까지 마친 반복 패턴 트리를 대상으로 각 프로세서가 독립적으로 마이닝 작업을 진행하여 최종 결과인 규칙들을 생성한다. 두번째 단계는 각 프로세서에서 서로 다른 데이터를 대상으로 순차 FP 알고리즘에 의해 데이터 마이닝을 실행한다.

2.1 지역 DFPT의 구성 단계

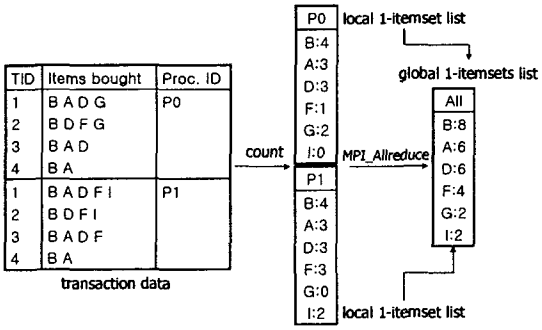

그림 1. 초기화 스캔 단계

그림 1과 같이 먼저 각 프로세서가 독립적으로 지역 DB의 트랜잭션 데이터를 스캔하여 지역적 반복 1-항목 리스트 (Local Frequent 1-Itemset List)를 구성한다. 각 프로세서는 지역적 반복 1-항목 집합을 브로드캐스트하여 모든 프로세서가 전역적 (global) 반복 1-항목 리스트를 갖게 한다. 다음으로 전역적 반복 1-항목 리스트를 항목의 빈도에 의하여 내림차순으로 정렬한다.

반복 패턴 트리를 구성하기 위하여 먼저 루트 노드를 생성하고 지역 DB를 다시 한번 스캔하여 각 트랜잭션 정보를 전역적 반복 1-항목 리스트의 순서에 따라 트리에 삽입한다. 트리의 각 노드로의 접근 (access)을 용이하게 하기 위하여 헤더 테이블을 구성한다. 헤더 테이블은 전역적 반복 1-항목 리스트와 동일한 항목들로 채워져 연결된다. 트리 구성이 완료되면 부하 균등화 과정을 진행한다. 이에 대한 설명은 3장에서 상세히 기술한다.

2.2 분산형 FP트리를 이용한 마이닝

DFPT를 이용한 마이닝은 헤더 테이블의 마지막 항목에서부터 최상위 항목으로 진행하게 된다. DFPT 알고리즘은 다음과 같이 5단계로 진행된다.

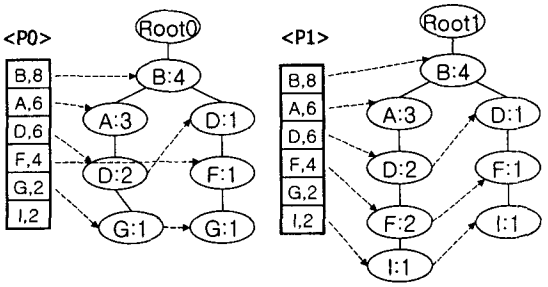

그림 2. DFPT의 구성

1) 초기화 단계 : 각 프로세서는 지역 DB를 한번 스캔하여 지역적 반복 1-항목 리스트를 구성한다. 이것을 브로드캐스트하여 모든 프로세서가 전역적 반복 1-항목 집합을 갖도록 한다.
2) 트리 구성 단계 : 각 프로세서에서 전역적 반복 1-항목 집합을 바탕으로 헤더테이블을 구성한다. 지역 DB를 다시 한번 스캔하여 지역 DFPT를 구성한다.
3) 부하 균등화 단계 : 먼저 DB를 구성하는 트랜잭션의 항목 분포도를 계산한다. 분포도를 참조하여 계산량이 각 프로세서에서 균등해지도록 항목들을 할당한다. 할당된 항목에 관한 정보들을 해당 프로세서에 전송하여 독립적으로 마이닝 작업이 가능하게 된다.
4) 마이닝 단계 : 각 프로세서는 DFPT를 대상으로 독립적으로 마이닝 단계를 수행한다. 즉, 할당된 항목에 대한 조건부 (conditional) DFPT를 구성하여 최종 결과인 규칙들을 저장한다.
5) 결과 출력 단계 : 저장된 최종 결과인 규칙들을 모아 출력한다.

Ⅲ. 부하 균등화 단계의 기술

DFPT가 완성되면 부하균등화 단계를 시작한다. 먼저 각 프로세서는 데이터 집합 항목의 분포도를 수집한다. 분포도에 관한 식은 [3]에 데이터 왜곡 (data skewness)에 관한 식으로 기술되어 있으며 본 논문에서는 이 값을 'B'로 나타낸다. 다음으로 헤더 테이블의 각 항목들을 분포도를 참조하여 해당 항목을 상대적으로 많이 포함한 프로세서에게 그 항목처리를 전담시킨다. 이렇게 하여 할당된 항목을 포함하는 가지 (branch) 정보를 전송할 때 통신의 양이 최소가 된다. 각 프로세서는 할당된 항목에 관하여 독립적으로 마이닝 작업을 할 준비가 된다.

다음은 구체적 진행과정을 예를 들어 설명한다. 그림 3은 두개의 프로세서가 참여한 부하균등화 단계를 나타낸다. P0와 P1이 각각 <B,D,G>와 <A,F,I>를 할당받았다. P0의 트리에서 어두운 노드들이 F-패스를 나타내고, P1에서 어두운 노드들이 D-패스를 나타내고 있다. 그림은 두 프로세서가 할당받은 항목 <D,F>의 정보를 주고받는 것을 나타낸다.

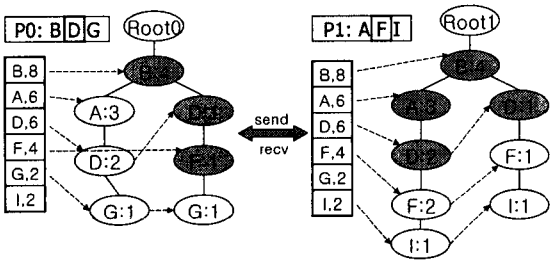

그림 3. 부하 균등화 과정

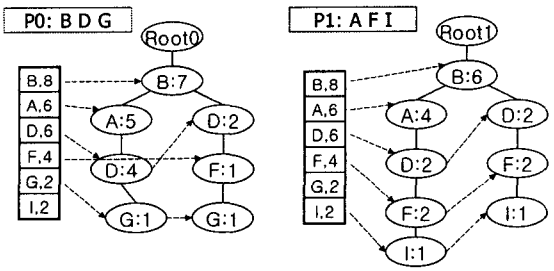

그림 4. 부하 균등화가 완료된 반복 패턴 트리

그림 4는 부하 균등화가 완료된 상태이다. P0가 P1에게서 D-패스 <D:1,B:1>, <D:2,A:2,B:2>를 받아 카운트를 증가시켜 <D:4,A:5,B:6>, <D:2,B7>이 됨을 확인할 수 있다. 끝으로 부하 균등화가 완료된 반복 패턴 트리를 써서 각 프로세서는 독립적으로 마이닝 과정을 진행한다. 표 1은 그 결과를 표시한다.

표 1. 마이닝 단계에서의 결과

Proc.	Items	Conditional Pattern Base	Conditional DFP-Tree
P0	B	{0}	{0}
	D	{B:2} {A:4, B:4}	{B:6, A:4}/D
	G	{D:1, A:1, B:1} {F:1, D:1, B:1}	{B:2, D:2, A:1, F:1}/G
P1	A	{B:6}	{B:6}/A
	F	{D:2, B:2} {D:2, A:2, B:2}	{B:4, D:4, A:2}/F
	I	{F:1, D:1, A:1, B:1} {F:1, D:1, B:1}	{B:2, D:2, F:2, A:1}/I

그림 2를 보면 항목 G는 전체 DB에서 발생빈도가 2임을 확인할 수 있다. P0의 트리에서 항목 G와 함께 구입된 항목들은 G항목의 상위 노드들로 나타나있다. 표 1에 정리되어 있는 것처럼 G항목과 함께 구매된 <D:1,A:1,B:1>, <F:1,D:1,B:1>가 항목 G의 조건부 패턴 베이스 (conditional pattern base)이다. 이것은 G항목과 함께 <B:2,D:2,A:1,F:1>가 구입되었다는 것을 의미한다. 각 프로세서는 조건부 (conditional) DFPT를 구성하여 <GB, GD, GA, GF>등의 연관규칙들이 최소지지도를 만족하는지를 검사한 후 만족하는 결과들을 저장한다. 각 프로세서는 분산형 FP트리를 이용하여 마이닝한 결과인 연관규칙들을 출력한 후 알고리즘을 종료한다.

Ⅳ. 실험 결과

성능을 실험적으로 평가하기 위해 리눅스 운영체제 하에서 16 노드로 구성된 PC 클러스터 시스템에 고안된 알고리즘을 구현하였다. 각 노드는 AMD Athlon 1Ghz로 구성되어 있으며 256MB의 메인 메모리와 30GB의 하드디스크를 부착하고 100Mbps 이더넷 스위치로 연결되어 있다. 프로세서들 사이의 통신을 위해서 MPI (Message Passing Interface)를 사용하였다. 사용된 데이터 집합은 [1]에서 사용한 것을 수정한 합성 데이터이며 평균 트랜잭션 사이즈 |T|=25, 평균 최대 잠재적 반복 항목 사이즈 |I|=10, 전체 트랜잭션 개수 |D|=10K~80K, 전체 항목의 수 N=1000으로 구성되어 있다. 사용자에 의해 설정되는 최소 지지도 (minimum support)는 $0.02\% \le |S| \le 0.005\%$의 범위를 사용하였다. 반복항목은 최소지지도를 넘는 빈도를 가지므로 최소지지도가 낮을수록 계산량이 많아지게 된다. 실행성능 비교를 위하여 Improved Count Distribution (ICD) 알고리즘을 이용하였다 [2].

그림 5에서 ICD와 DFPT 알고리즘을 나타내는 막대그래프의 왼쪽은 항목의 분포가 각 프로세서에서 균등한 상태(B=0)에서 실험결과이다. 프로세서 개수 4개, |D|=40K로 고정시킨 상황에서 최소지지도를 낮춤에 따라 늘어나는 계산량에 대한 확장성(scalability)을 관찰한 결과이다. ICD 알고리즘이 최소지지도가 낮아짐에 따라 실행시간이 지수증가 하는 것을 확인할 수 있다. 이것은 ICD 알고리즘에서 반복 패턴의 후보자 생성시에 검사대상이 되는 후보자 집합의 수가 지수적으로 증가하기 때문이다. DFPT는 후보자 집합 생성단계를 완전히 제거하였기 때문에 실행시간이 선형적으로 증가됨을 확인하였다. 그림 5에서 ICD와 DFPT 알고리즘을 나타내는 막대그래프의 오른쪽은 데이터 분포도 B=0.7인 상태에서 프로세서 수 4개, |D|=40K로 고정시키고 |S|를 0.02%~0.005%까지 변화시켜서 실험한 결과이다. 이것은 항목들의 분포가 불규칙성이 심화됨에 따라 부하 불균형이 커지는 현상을 실행시간의 변화를

통해 관찰한 것이다. ICD 알고리즘은 B=0인 상황에 비하여 평균 31.3% 성능이 저하되었으며, DFPT 알고리즘의 경우 평균 7% 성능이 저하 되었다. ICD 알고리즘은 각 단계마다 동기화 과정을 거치기 때문에 부

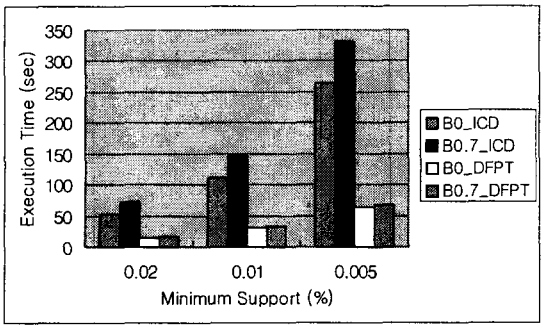

그림 5. 항목들이 불균등하게 분포할 때의 확장성

하 불균형에 따른 오버헤드가 크게 작용하게 된다. DFPT 알고리즘의 경우 동기화 단계 수를 최소화하여 부하 불균등에 따른 지연시간이 ICD에 비하여 평균 75% 적다. 부하 불균형인 조건에서 ICD알고리즘에 비하여 DFPT 알고리즘이 평균 4배 이상 빠른 실행성능을 나타내었다.

그림 6과 7은 |D|=20K 그리고 프로세서 수를 2개로 고정시킨 상황에서 각 분포도를 B=0.0, 0.7로 변화 시켜 두 알고리즘의 CPU 이용률을 관찰한 결과이다. ICD 알고리즘은 동기화에 의하여 부하 불균형에 의한 오버헤드가 전체성능을 크게 저해하는 것을 확인하였다. 부하 균등화 과정이 효과적으로 적용된 DFPT 알고리즘의 경우 동일조건에서 CPU 이용률이 ICD 알고리즘에 비하여 우수함을 확인할 수 있다.

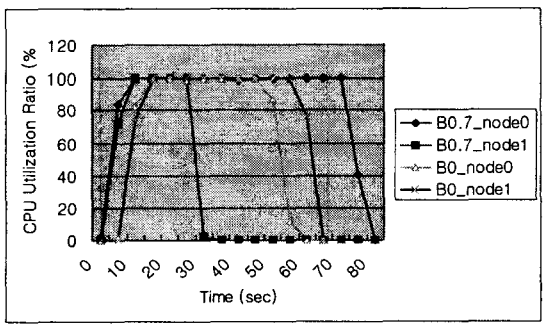

그림 6. ICD 알고리즘의 CPU 이용률

V. 결론

본 논문에서는 새로운 병렬 데이터 마이닝 알고리즘인 DFPT 알고리즘을 제안하였다. Frequent Pattern

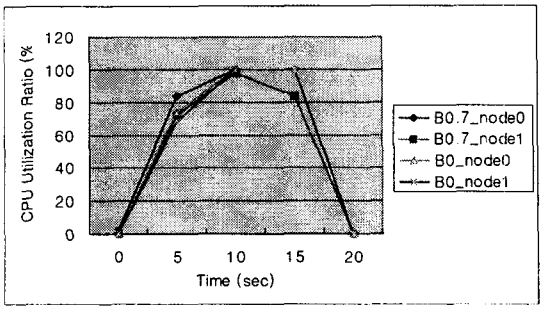

그림 7. DFPT 알고리즘의 CPU 이용률

Growth 알고리즘을 기반으로 하여 기존의 Apriori 알고리즘을 병렬화한 알고리즘의 단점인 반복적인 디스크 입출력과 동기화에 의한 지연증가를 줄였다. 또한 부하 균등화를 효과적으로 이루어 수행시간을 단축하였다. PC 클러스터 시스템을 기반으로 실험한 결과 다양한 조건에서 ICD에 비하여 최소 3배 이상 빠르게 실행됨을 관찰할 수 있었다.

앞으로 다양한 데이터 집합에 대해서도 DFPT의 압축율을 극대화 시킬 수 있도록 트리구조를 보다 최적화된 형태로 개선하는 연구가 필요하다.

참고문헌

[1] R. Agrawal, T. Imielinski, and A. Swami, "Mining Association Rules between Sets of Items in Large Databases", In *Proc. of ACM SIGMOD* Intl. Conf. Management of Data, 1993.

[2] R. Agrawal, J.C. Shafer, "Parallel Mining of Association Rules: Design, Implementation and Experience", In *Technical Report TJ10004*, IBM Research Division, Almaden Research Center, 1996.

[3] D.W. Cheung, S.D. Lee, and Y. Xiao, "Effect of Data Skewness and Workload Balancing in Parallel Data Mining", In *IEEE Trans. Knowledge and Data Engineering*, Vol. 14, No. 3, pp. 498-514, 2002.

[4] J. Han, J. Pei, and Y. Yin, "Mining Frequent Patterns without Candidate Generation", In *Proc. of SIGMOD*, pp. 1-12, 2000.

[5] M.J. Zaki, S. Parthasarathy, and W. Li. "A Localized Algorithm for Parallel Association Mining", In *ACM Symp. Parallel Algorithms and Architectures*, 1997.

[6] M.J. Zaki. "Parallel and Distributed Association Mining: A Survey", In IEEE Concurrency, 1999.