



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2015-0120143

(43) 공개일자 2015년10월27일

(51) 국제특허분류(Int. Cl.)

G06F 17/30 (2006.01)

(21) 출원번호 10-2014-0045993

(22) 출원일자 2014년04월17일

심사청구일자 2014년04월17일

(71) 출원인

세종대학교산학협력단

서울특별시 광진구 능동로 209 (군자동, 세종대학교)

(72) 발명자

윤은일

서울특별시 성북구 종암로5길 7 (종암동)

편광범

충청남도 공주시 무안길 13 (금학동, 덕성아파트)1동 302호

(뒷면에 계속)

(74) 대리인

정부연

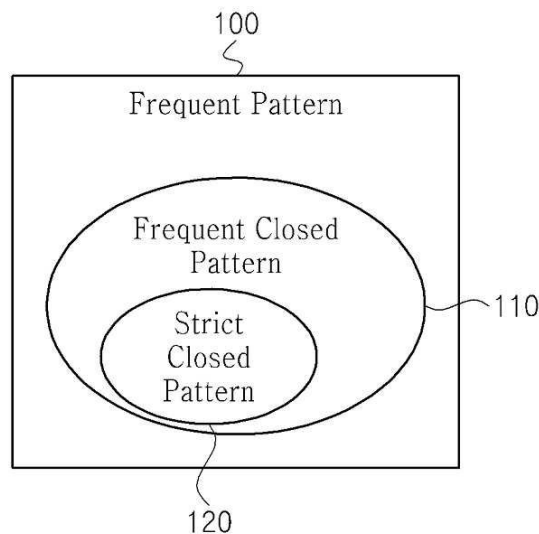
전체 청구항 수 : 총 17 항

(54) 발명의 명칭 빈발 패턴 마이닝 방법 및 이를 저장하는 기록매체

(57) 요약

데이터베이스를 스캔하여 글로벌 FP-tree를 생성하는 단계, 합성 패턴을 생성하는 단계 및 빈발 패턴을 추출하여 top-k 리스트에 삽입하는 단계를 포함하는 것을 특징으로 하는 데이터 마이닝 방법이다.

대표도 - 도1



(72) 발명자

양홍모

충청북도 청주시 상당구 수영로 350 (금천동, 장
자마을1단지부영아파트) 104동 302동

이강인

충청북도 청주시 흥덕구 모충로12번길 4 (개신동,
송학아파트)103동 905호

김지원

충청북도 청주시 상당구 중흥로 142 (용암동)106
동 602호

이 발명을 지원한 국가연구개발사업

과제고유번호 2013R1A1A1A05005682

부처명 미래창조과학부

연구관리전문기관 미래창조과학부-한국연구재단

연구사업명 신진연구지원사업(후속연구지원)

연구과제명 효율적인 스트림 패턴 처리를 위한 마이닝 핵심기술 개발

기 여 율 1/1

주관기관 세종대학교 산학협력단

연구기간 2013.06.01 ~ 2016.05.31

명세서

청구범위

청구항 1

데이터베이스를 스캔하여 글로벌 FP-tree(Frequent Pattern Tree)를 생성하는 단계;

합성 패턴(composite pattern)을 생성하는 단계; 및

빈발 패턴(frequent pattern)을 추출하여 top-k 리스트에 삽입하는 단계를 포함하는 것을 특징으로 하는 데이터 마이닝 방법.

청구항 2

제1항에 있어서, 상기 합성 패턴은

프리픽스(prefix), 강력 폐쇄 패턴(strict closed pattern), 테일(tail), 시작 표시 마크 및 종료 표시 마크를 포함하는 것을 특징으로 하는 데이터 마이닝 방법.

청구항 3

제1항에 있어서, 상기 합성 패턴을 생성하는 단계는

상기 글로벌 FP-tree에서 중간에 위치한 아이템을 선택하는 단계; 및

상기 FP-tree가 단일 경로인지 여부를 확인하는 단계를 포함하는 것을 특징으로 하는 데이터 마이닝 방법.

청구항 4

제3항에 있어서, 상기 합성 패턴을 생성하는 단계는

상기 FP-tree가 단일 경로인 경우, 첫번째 아이템과 상기 아이템의 서포트가 동일한지 여부를 확인하는 단계; 및

동일하다면 강력 폐쇄 패턴에 삽입하고, 동일하지 않다면 테일(tail)에 삽입하는 단계를 포함하는 것을 특징으로 하는 데이터 마이닝 방법.

청구항 5

제3항에 있어서, 상기 합성 패턴을 생성하는 단계는

상기 FP-tree가 단일 경로가 아닌 경우, 새로운 FP-tree를 생성하여 적어도 하나의 노드가 존재하는지 여부를 확인하는 단계; 및

적어도 하나의 노드가 존재하는 경우 상기 노드의 아이템을 프리픽스(prefix)에 삽입하는 단계를 포함하는 것을 특징으로 하는 데이터 마이닝 방법.

청구항 6

제1항에 있어서, 상기 빈발 패턴을 추출하여 top-k 리스트에 삽입하는 단계는

상기 FP-tree가 단일 경로인 경우, 합성 패턴의 서포트가 최소 서포트 기준보다 작은지 여부를 확인하는 단계;

및

상기 합성 패턴의 서포트가 최소 서포트 기준보다 작지 않은 경우 상기 합성 패턴을 top-k 리스트에 삽입하는 단계를 포함하는 것을 특징으로 하는 데이터 마이닝 방법.

청구항 7

제1항에 있어서, 상기 빈발 패턴을 추출하여 top-k 리스트에 삽입하는 단계는

상기 FP-tree가 단일 경로가 아닌 경우, 해당 노드의 아이템이 빈발 패턴인지 여부를 확인하는 단계; 및

빈발 패턴인 경우 해당 패턴을 top-k 리스트에 삽입하는 단계를 포함하는 것을 특징으로 하는 데이터 마이닝 방법.

청구항 8

제1항에 있어서, 상기 top-k 리스트는

빈발 패턴을 삽입하는 단계;

상기 top-k 리스트가 가득 차고 삽입된 패턴의 서포트가 최소 서포트 보다 작는지 여부를 확인하는 단계; 및

상기 top-k 리스트가 가득 차고 삽입된 패턴의 서포트가 최소 서포트 보다 작지 않은 경우, 상기 top-k 리스트의 최소 서포트를 갱신하는 단계를 통해 관리된다.

청구항 9

제1항에 있어서,

합성 패턴을 변환하는 단계를 더 포함하는 것을 특징으로 하는 데이터 마이닝 방법.

청구항 10

제9항에 있어서, 상기 합성 패턴을 변환하는 단계는

시작 표시 마크와 종료 표시 마크 사이에 적어도 하나의 아이템이 존재하는지 여부를 확인하는 단계; 및

시작 표시 마크와 종료 표시 마크 사이에 적어도 하나의 아이템이 존재하는 경우, 상기 시작 표시 마크와 종료 표시 마크 사이에 존재하는 아이템을 서로 결합하는 단계를 포함하는 것을 특징으로 하는 데이터 마이닝 방법.

청구항 11

데이터베이스를 스캔하여 글로벌 FP-tree를 생성하는 단계;

합성 패턴을 생성하는 단계; 및

빈발 패턴을 추출하여 N-itemset top-k 리스트에 삽입하는 단계를 포함하는 것을 특징으로 하는 데이터 마이닝 방법.

청구항 12

제11항에 있어서, 상기 빈발 패턴을 추출하여 N-itemset top-k 리스트에 삽입하는 단계는

테일의 길이 및 강력 폐쇄 패턴의 길이를 계산하는 단계를 포함하는 것을 특징으로 하는 데이터 마이닝 방법.

청구항 13

제11항에 있어서, 상기 N-itemset top-k 리스트는

N개의 아이템으로 구성된 패턴을 포함하고 동일한 패턴을 중복하여 포함하지 않는 것을 특징으로 하는 데이터 마이닝 방법.

청구항 14

제11항에 있어서, 상기 N-itemset top-k 리스트는

상기 N-itemset top-k 리스트의 크기를 계산하여 k 보다 큰지 여부를 확인하는 단계;

k 보다 큰 경우, 최소 서포트 및 최소 서포트 기준을 갱신하는 단계; 및

빈번하지 않은 패턴을 제거하는 단계를 통해 관리되는 것을 특징으로 하는 데이터 마이닝 방법.

청구항 15

제11항에 있어서,

합성 패턴을 변환하는 단계를 더 포함하는 것을 특징으로 하는 데이터 마이닝 방법.

청구항 16

데이터베이스를 스캔하여 글로벌 FP-tree를 생성하는 기능;

합성 패턴을 생성하는 기능; 및

빈발 패턴을 추출하여 top-k 리스트에 삽입하는 기능을 포함하는 것을 특징으로 하는 데이터 마이닝 방법에 관한 컴퓨터 프로그램을 기록한 기록매체.

청구항 17

데이터베이스를 스캔하여 글로벌 FP-tree를 생성하는 기능;

합성 패턴을 생성하는 기능; 및

빈발 패턴을 추출하여 N-itemset top-k 리스트에 삽입하는 기능을 포함하는 것을 특징으로 하는 데이터 마이닝 방법에 관한 컴퓨터 프로그램을 기록한 기록매체.

발명의 설명

기술 분야

본 발명은 데이터 마이닝 기술에 관한 것으로, 보다 상세하게는, CRM(Combination Reducing Method) 알고리즘과 CRMN(Combination Reducing Method for N-itemset top-k patterns) 알고리즘을 이용하여 효율적으로 빈발 패턴을 마이닝 하는 것을 지원할 수 있는 데이터 마이닝 방법 및 이를 저장하는 기록매체에 관한 것이다.

[0001]

배경 기술

- [0002] 일반적으로, 데이터 마이닝(Data Mining)은 대규모로 저장된 데이터 안에서 체계적이고 자동 적으로 통계적 규칙이나 패턴을 찾아내는 것이며 빈발 패턴 마이닝은 대규모로 저장된 데이터 안에서 자주 나타나는 패턴을 찾아내는 것이다. 데이터를 마이닝 하는 과정에서 얻어진 패턴들은 데이터의 특성을 분석하거나 의사결정을 위해 필요한 정보를 얻는데 활용되고 웹 데이터, 고객 데이터, 제품 데이터 및 교통과 통신 데이터 등 다양한 실제 데이터(Real Data) 분석에도 활용될 수 있다.
- [0003] 한국등록특허 제10-0913027호는 빈발항목 및 빈발 순차패턴 과정에서 메모리 사용량 및 마이닝 작업을 최적화하는 데이터 마이닝 방법을 개시한다. 본 선행기술은 모니터링 래티스에서 상당기간 트랜잭션에서 항목이 출현하지 않는 경우 해당 항목을 모니터링 래티스에서 제거하고 모니터링 래티스에서 관리되고 있는 항목들의 출현 빈도 수를 분석하여 빈발항목을 찾아낼 수 있다.
- [0004] 한국등록특허 제10-1105736호는 밀집 대용량 트랜잭션 데이터 베이스 환경에서 빈발 항목 집합 및 최소 빈발 항목 집합 모두를 효율적으로 탐색하는 데이터 마이닝 방법을 개시한다. 본 선행기술은 패트리샤 빈발 패턴 트리를 구성하여 빈발 항목 집합을 탐색하는 것으로, 원본 트랜잭션 데이터베이스 내의 모든 트랜잭션의 항목을 카운트하여 헤더 테이블을 구성하고, 헤더 테이블을 이용하여 압축된 패트리샤 빈발 패턴 트리를 구성한 후 CFP-Tree를 이용하여 빈발 항목 집합을 탐색할 수 있다.
- [0005] 상기 선행기술들은 빈발 패턴을 마이닝 하는 방법에 대해서 각각 효율적인 방법으로서 모니터링 래티스를 통한 방법 및 패트리샤 트리를 통한 방법을 개시하고 있지만, 빈발 패턴을 찾는 과정에서 수행되는 조합의 경우의 수에 대해서는 효율적인 방안을 개시하고 있지 않아 이와 관련하여 마이닝 시간과 메모리 효율 측면에서 문제점이 있다.

선행기술문헌

특허문헌

- [0006] (특허문헌 0001) 한국등록특허 제10-0913027호
(특허문헌 0002) 한국등록특허 제10-1105736호

발명의 내용

해결하려는 과제

- [0007] 본 발명의 일 실시예에는 프리픽스(prefix), 강력 폐쇄 패턴 및 테일(tail)로 이루어진 합성 패턴을 생성하여 조합의 경우의 수를 최소화하고, 이를 통해서 실행 시간을 줄일 수 있는 데이터 마이닝 방법을 제공하고자 한다.
- [0008] 본 발명의 일 실시예에는 합성 패턴을 생성하고 패턴의 길이에 상관없이 빈발 패턴을 추출하여 조합의 경우의 수를 최소화하고, 이를 통해서 실행 시간을 줄일 수 있는 데이터 마이닝 방법을 제공하고자 한다.
- [0009] 본 발명의 일 실시예에는 합성 패턴을 생성하고 패턴의 길이에 의존하여 각각의 길이에 대한 빈발 패턴들을 조합의 경우의 수를 최소화하고, 이를 통해서 실행 시간을 줄일 수 있는 데이터 마이닝 방법을 제공하고자 한다.
- [0010] 본 발명의 일 실시예에는 top-k 리스트에 포함되어 있는 합성 패턴을 일반 패턴으로 변환하여, 최종적으로 top-k 빈발 패턴을 구할 수 있는 데이터 마이닝 방법을 제공하고자 한다.

과제의 해결 수단

- [0011] 실시예들 중에서, 데이터베이스를 스캔하여 글로벌 FP-tree를 생성하는 단계, 합성 패턴을 생성하는 단계 및 빈발 패턴을 추출하여 top-k 리스트에 삽입하는 단계를 포함한다.

- [0012] 일 실시예에서, 상기 합성 패턴은 프리픽스(prefix), 강력 폐쇄 패턴, 테일(tail), 시작 표시 마크 및 종료 표시 마크를 포함할 수 있다.
- [0013] 일 실시예에서, 상기 합성 패턴을 생성하는 단계는 상기 글로벌 FP-tree에서 중간에 위치한 아이템을 선택하는 단계 및 상기 FP-tree가 단일 경로인지 여부를 확인하는 단계를 포함할 수 있다.
- [0014] 일 실시예에서, 상기 합성 패턴을 생성하는 단계는 상기 FP-tree가 단일 경로인 경우, 첫번째 아이템과 상기 아이템의 서포트가 동일한지 여부를 확인하는 단계 및 동일하다면 강력 폐쇄 패턴에 삽입하고, 동일하지 않다면 테일에 삽입하는 단계를 포함할 수 있다.
- [0015] 일 실시예에서, 상기 합성 패턴을 생성하는 단계는 상기 FP-tree가 단일 경로가 아닌 경우, 새로운 FP-tree를 생성하여 적어도 하나의 노드가 존재하는지 여부를 확인하는 단계 및 적어도 하나의 노드가 존재하는 경우 상기 노드의 아이템을 프리픽스에 삽입하는 단계를 포함할 수 있다.
- [0016] 일 실시예에서, 상기 빈발 패턴을 추출하여 top-k 리스트에 삽입하는 단계는 상기 FP-tree가 단일 경로인 경우, 합성 패턴의 서포트가 최소 서포트 기준보다 작은지 여부를 확인하는 단계 및 상기 합성 패턴의 서포트가 최소 서포트 기준보다 작지 않은 경우 상기 합성 패턴을 top-k 리스트에 삽입하는 단계를 포함할 수 있다.
- [0017] 일 실시예에서, 상기 빈발 패턴을 추출하여 top-k 리스트에 삽입하는 단계는 상기 FP-tree가 단일 경로가 아닌 경우, 해당 노드의 아이템이 빈발 패턴인지 여부를 확인하는 단계 및 빈발 패턴인 경우 해당 패턴을 top-k 리스트에 삽입하는 단계를 포함할 수 있다.
- [0018] 일 실시예에서, 상기 top-k 리스트는 빈발 패턴을 삽입하는 단계, 상기 top-k 리스트가 가득 차고 삽입된 패턴의 서포트가 최소 서포트 보다 작은지 여부를 확인하는 단계 및 상기 top-k 리스트가 가득 차고 삽입된 패턴의 서포트가 최소 서포트 보다 작지 않은 경우, 상기 top-k 리스트의 최소 서포트를 갱신하는 단계를 통해 관리될 수 있다.
- [0019] 실시예들 중에서, 데이터베이스를 스캔하여 글로벌 FP-tree를 생성하는 단계, 합성 패턴을 생성하는 단계, 빈발 패턴을 추출하여 top-k 리스트에 삽입하는 단계 및 합성 패턴을 변환하는 단계를 포함한다.
- [0020] 일 실시예에서, 상기 합성 패턴을 변환하는 단계는 시작 표시 마크와 종료 표시 마크 사이에 적어도 하나의 아이템이 존재하는지 여부를 확인하는 단계 및 시작 표시 마크와 종료 표시 마크 사이에 적어도 하나의 아이템이 존재하는 경우, 상기 시작 표시 마크와 종료 표시 마크 사이에 존재하는 아이템을 서로 결합하는 단계를 포함할 수 있다.
- [0021] 실시예들 중에서, 데이터베이스를 스캔하여 글로벌 FP-tree를 생성하는 단계, 합성 패턴을 생성하는 단계 및 빈발 패턴을 추출하여 N-itemset top-k 리스트에 삽입하는 단계를 포함한다.
- [0022] 일 실시예에서, 상기 빈발 패턴을 추출하여 N-itemset top-k 리스트에 삽입하는 단계는 테일의 길이 및 강력 폐쇄 패턴의 길이를 계산하는 단계를 포함할 수 있다.
- [0023] 일 실시예에서, 상기 N-itemset top-k 리스트는 N개의 아이템으로 구성된 패턴을 포함하고 동일한 패턴을 중복하여 포함하지 않을 수 있다.
- [0024] 일 실시예에서, 상기 N-itemset top-k 리스트는 상기 N-itemset top-k 리스트의 크기를 계산하여 k 보다 큰지 여부를 확인하는 단계, k 보다 큰 경우, 최소 서포트 및 최소 서포트 기준을 갱신하는 단계 및 빈번하지 않은 패턴을 제거하는 단계를 통해 관리될 수 있다.
- [0025] 실시예들 중에서, 데이터베이스를 스캔하여 글로벌 FP-tree를 생성하는 단계, 합성 패턴을 생성하는 단계, 빈발 패턴을 추출하여 N-itemset top-k 리스트에 삽입하는 단계 및 합성 패턴을 변환하는 단계를 포함한다.
- [0026] 실시예들 중에서, 데이터베이스를 스캔하여 글로벌 FP-tree를 생성하는 단계, 합성 패턴을 생성하는 단계 및 빈발 패턴을 추출하여 top-k 리스트에 삽입하는 단계를 포함하는 것을 특징으로 하는 데이터 마이닝 방법에 관한 컴퓨터 프로그램을 기록한다.
- [0027] 실시예들 중에서, 데이터베이스를 스캔하여 글로벌 FP-tree를 생성하는 기능, 합성 패턴을 생성하는 단계 및 빈발 패턴을 추출하여 N-itemset top-k 리스트에 삽입하는 기능을 포함하는 것을 특징으로 하는 데이터 마이닝 방법에 관한 컴퓨터 프로그램을 기록한다.

발명의 효과

- [0028] 본 발명의 일 실시예에 따른 데이터 마이닝 방법은 프리픽스(prefix), 강력 폐쇄 패턴 및 테일(tail)로 이루어진 합성 패턴을 생성하여 조합의 경우의 수를 최소화하고, 이를 통해서 실행 시간을 줄일 수 있다.
- [0029] 본 발명의 일 실시예에 따른 데이터 마이닝 방법은 합성 패턴을 생성하고 패턴의 길이에 상관없이 빈발 패턴을 추출하여 조합의 경우의 수를 최소화하고, 이를 통해서 실행 시간을 줄일 수 있다.
- [0030] 본 발명의 일 실시예에 따른 데이터 마이닝 방법은 합성 패턴을 생성하고 패턴의 길이에 의존하여 각각의 길이에 대한 빈발 패턴들을 조합의 경우의 수를 최소화하고, 이를 통해서 실행 시간을 줄일 수 있다.
- [0031] 본 발명의 일 실시예에 따른 데이터 마이닝 방법은 top-k 리스트에 포함되어 있는 합성 패턴을 일반 패턴으로 변환하여, 최종적으로 top-k 빈발 패턴을 구할 수 있다.

도면의 간단한 설명

- [0032] 도 1은 본 발명의 일 실시예에 따른 빈발 패턴을 설명하는 블록도이다.
- 도 2는 본 발명의 일 실시예에 따른 FP-growth 알고리즘을 설명하는 흐름도이다.
- 도 3은 본 발명의 일 실시예에 따른 합성 패턴을 설명하는 블록도이다.
- 도 4는 본 발명의 일 실시예에 따른 CRM 알고리즘을 설명하는 흐름도이다.
- 도 5는 본 발명의 일 실시예에 따른 CRM Growth 알고리즘을 설명하는 흐름도이다.
- 도 6는 본 발명의 일 실시예에 따른 Top-k list를 설명하는 흐름도이다.
- 도 7는 본 발명의 일 실시예에 따른 합성 패턴의 변환을 설명하는 블록도이다.
- 도 8는 본 발명의 일 실시예에 따른 CRMN 알고리즘을 설명하는 흐름도이다.
- 도 9는 본 발명의 일 실시예에 따른 CRMN Growth 알고리즘을 설명하는 흐름도이다.
- 도 10는 본 발명의 일 실시예에 따른 N-itemset Top-k list를 설명하는 블록도이다.
- 도 11은 본 발명의 일 실시예에서 CRMN에서의 합성 패턴 변환을 설명하는 알고리즘이다.

발명을 실시하기 위한 구체적인 내용

- [0033] 본 발명에 관한 설명은 구조적 내지 기능적 설명을 위한 실시예에 불과하므로, 본 발명의 권리범위는 본문에 설명된 실시예에 의하여 제한되는 것으로 해석되어서는 아니 된다. 즉, 실시예는 다양한 변경이 가능하고 여러 가지 형태를 가질 수 있으므로 본 발명의 권리범위는 기술적 사상을 실현할 수 있는 균등물들을 포함하는 것으로 이해되어야 한다. 또한, 본 발명에서 제시된 목적 또는 효과는 특정 실시예가 이를 전부 포함하여야 한다거나 그러한 효과만을 포함하여야 한다는 의미는 아니므로, 본 발명의 권리범위는 이에 의하여 제한되는 것으로 이해되어서는 아니 될 것이다.
- [0034] 한편, 본 출원에서 서술되는 용어의 의미는 다음과 같이 이해되어야 할 것이다.
- [0035] "제1", "제2" 등의 용어는 하나의 구성요소를 다른 구성요소로부터 구별하기 위한 것으로, 이들 용어들에 의해 권리범위가 한정되어서는 아니 된다. 예를 들어, 제1 구성요소는 제2 구성요소로 명명될 수 있고, 유사하게 제2 구성요소도 제1 구성요소로 명명될 수 있다.
- [0036] 어떤 구성요소가 다른 구성요소에 "연결되어" 있다고 언급된 때에는, 그 다른 구성요소에 직접적으로 연결될 수도 있지만, 중간에 다른 구성요소가 존재할 수도 있다고 이해되어야 할 것이다. 반면에, 어떤 구성요소가 다른 구성요소에 "직접 연결되어" 있다고 언급된 때에는 중간에 다른 구성요소가 존재하지 않는 것으로 이해되어야 할 것이다. 한편, 구성요소들 간의 관계를 설명하는 다른 표현들, 즉 "~사이에"와 "바로 ~사이에" 또는 "~에 이웃하는"과 "~에 직접 이웃하는" 등도 마찬가지로 해석되어야 한다.
- [0037] 단수의 표현은 문맥상 명백하게 다르게 뜻하지 않는 한 복수의 표현을 포함하는 것으로 이해되어야 하고, "포함하다" 또는 "가지다" 등의 용어는 실시된 특징, 숫자, 단계, 동작, 구성요소, 부분품 또는 이들을 조합한 것이

존재함을 지정하려는 것이며, 하나 또는 그 이상의 다른 특징이나 숫자, 단계, 동작, 구성요소, 부분품 또는 이들을 조합한 것들의 존재 또는 부가 가능성을 미리 배제하지 않는 것으로 이해되어야 한다.

[0038] 각 단계들에 있어 식별부호(예를 들어, a, b, c 등)는 설명의 편의를 위하여 사용되는 것으로 식별부호는 각 단계들의 순서를 설명하는 것이 아니며, 각 단계들은 문맥상 명백하게 특정 순서를 기재하지 않는 이상 명기된 순서와 다르게 일어날 수 있다. 즉, 각 단계들은 명기된 순서와 동일하게 일어날 수도 있고 실질적으로 동시에 수행될 수도 있으며 반대의 순서대로 수행될 수도 있다.

[0039] 본 발명은 컴퓨터가 읽을 수 있는 기록매체에 컴퓨터가 읽을 수 있는 코드로서 구현될 수 있고, 컴퓨터가 읽을 수 있는 기록 매체는 컴퓨터 시스템에 의하여 읽혀질 수 있는 데이터가 저장되는 모든 종류의 기록 장치를 포함한다. 컴퓨터가 읽을 수 있는 기록 매체의 예로는 ROM, RAM, CD-ROM, 자기 테이프, 플로피 디스크, 광 데이터 저장 장치 등이 있으며, 또한, 캐리어 웨이브(예를 들어 인터넷을 통한 전송)의 형태로 구현되는 것도 포함한다. 또한, 컴퓨터가 읽을 수 있는 기록 매체는 네트워크로 연결된 컴퓨터 시스템에 분산되어, 분산 방식으로 컴퓨터가 읽을 수 있는 코드가 저장되고 실행될 수 있다.

[0040] 여기서 사용되는 모든 용어들은 다르게 정의되지 않는 한, 본 발명이 속하는 분야에서 통상의 지식을 가진 자에 의해 일반적으로 이해되는 것과 동일한 의미를 가진다. 일반적으로 사용되는 사전에 정의되어 있는 용어들은 관련 기술의 문맥상 가지는 의미와 일치하는 것으로 해석되어야 하며, 본 출원에서 명백하게 정의하지 않는 한 이상적이거나 과도하게 형식적인 의미를 지니는 것으로 해석될 수 없다.

[0041] 도 1은 본 발명의 일 실시예에 따른 빈발 패턴을 설명하는 블록도이다.

[0042] 도 1을 참조하면, 빈발 패턴(100)은 빈발 폐쇄 패턴(110) 및 강력 폐쇄 패턴(120)을 포함한다.

[0043] 빈발 패턴(100)은 주어진 최소 서포트 기준보다 더 크거나 같은 값의 서포트를 가지는 패턴을 의미한다. 일 실시예에서, 1이 빈발 패턴(100)인 경우, minsup은 사용자가 지정한 최소 서포트 기준이 될 수 있고, L은 P에 포함된 빈발 패턴(100)들의 집합이 될 수 있다. 따라서, L은 아래의 식(1)과 같이 정의될 수 있다.

[0044] 식(1) : $L = \{l \mid \forall l \in P \text{ and } \text{sup}(l) \geq \text{minsup}\}$

[0045] 일 실시예에서, 4가지 트랜잭션들인 {a},{a, c},{b, c} 및 {a, b, c, d}과 함께 데이터베이스 D가 주어진 경우, D로부터 각각의 아이템에 대한 서포트는 {a:3, b:2, c:3, d:1} 가 될 수 있고, minsup이 2인 경우, {{a}, {c}, {b}}, {a,c}, {b,c}}를 제외한 P의 모든 패턴은 위의 식(1)에 따라 빈발하지 않는 패턴이 될 수 있다. 즉, $L = \{\{a\}, \{c\}, \{b\}, \{a,c\}, \{b,c\}\}$ 이 될 수 있다.

[0046] 일 실시예에서, i는 데이터베이스에 속한 아이터일 수 있고 I는 모든 아이터의 집합으로 $I=\{i_1, i_2, \dots, i_n\}$ 와 같이 표시될 수 있다. 트랜잭션이 t로 주어진 경우, m개의 트랜잭션들로 이루어진 데이터베이스 D는 $D=\{t_1, t_2, \dots, t_m\}$ 로 표현될 수 있다. 패턴 p는 적어도 하나의 트랜잭션에 포함되어 질 수 있고 I의 부분집합일 수 있다. P가 D에 있는 p들의 집합인 경우, P는 D로부터 생성 가능한 모든 패턴의 정보를 가질 수 있다. p를 포함하는 t의 개수는 p의 서포트인 $\text{sup}(p)$ 로 표현될 수 있고, |p| 는 p의 아이터의 개수일 수 있다.

[0047] 빈발 폐쇄 패턴(110)은 패턴의 확대 집합(superset)이 minsup을 만족하는 패턴들의 서포트의 관점에서 패턴과 동일하지 않은 경우의 특정 패턴이다. 일 실시예에서, c가 빈발 패턴(100)의 집합에 포함된 빈발 폐쇄 패턴(110)이고, c_{super} 가 c의 폐쇄 확대 집합(closed superset)들인 경우, $\text{sup}(c)$ 와 $\text{sup}(c_{\text{super}})$ 는 동일하지 않을 수 있다. 일 실시예에서, $L = \{\{a\}, \{c\}, \{b\}, \{a,c\}, \{b,c\}\}$ 인 경우 패턴의 서포트 값은 {{a:3}, {c:3}, {b:2}, {a, c:2}, {b, c:2}}이 될 수 있다. 빈발 패턴(100) 집합은 빈발 폐쇄 패턴(110)인 {{a:3}, {c:3}, {a, c:2}, {b, c:2}}로 변환될 수 있다. 여기에서, 폐쇄 패턴이 하위 패턴의 모든 정보를 가지고 있는지 여부를 알 수 있고, 폐쇄 패턴 기술은 단일경로의 개수를 줄일 수 있으며, CRM 및 CRMN의 최종 결과는 빈발 패턴(100)으로 생성된다.

[0048] 강력 폐쇄 패턴(120)은 먹집합이 모두 동일한 값의 서포트를 가진 패턴을 의미한다. 일 실시예에서, C가 L에서 얻어진 빈발 폐쇄 패턴(110)의 집합이고 sc가 C에서 얻어진 강력 폐쇄 패턴(120)이고, λ 가 sc의 각각의 아이터이며, sc_{sub} 는 sc의 부분집합이고, SC가 scs의 집합인 경우, SC는 아래의 식(2)를 만족할 수 있다.

[0049] 식(2) : $SC = \{sc \mid sc = \{\lambda_1, \lambda_2, \dots, \lambda_n\}, sc \in C, \text{ and } \text{sup}(\lambda_1)=\text{sup}(\lambda_2)= \dots =\text{sup}(\lambda_n)=\text{sup}(sc_{\text{sub}})\}$

- [0050] SC는 빈발 폐쇄 패턴(110) C에 포함 되어 있을 수 있고 모든 SC의 부분집합은 반드시 같은 서포트를 가질 수 있다. 패턴의 아이템들이 같은 서포트를 가지고 있고 폐쇄되어 있는 경우, 해당 패턴은 강력 폐쇄 패턴(120)에 속한다. 강력 폐쇄 패턴(120)이 일반 서포트를 가지는 강력 폐쇄 패턴(120)의 부분집합을 만들기 때문에, 단일 경로 상태의 특정 트리에서 첫번째에 위치하는 아이템의 서포트와 동일한 서포트를 가지는 아이템들은 해당 강력 폐쇄 패턴(120)으로 변환될 수 있다. 마이닝하는 과정이 끝날 때, 강력 폐쇄 패턴(120)을 일반 빈발 패턴(100)으로 변환하는 것을 통해 top-k의 빈발패턴은 적은 연산으로 마이닝될 수 있다. 트리의 단일 경로에서 알고리즘은 강력 폐쇄 패턴(120)에 관한 조합을 생략할 수 있다. 또한, 강력 폐쇄 패턴(120)은 모든 패턴의 부분집합의 서포트 정보를 포함하고 있기 때문에 쉽게 강력 폐쇄 패턴(120)을 빈발 패턴(110)으로 변환할 수 있다.
- [0051] 일 실시예에서, 단일 경로에서 강력 폐쇄 패턴(120)과 같은 서포트를 가진 아이템의 집합이 생성될 수 있다. 대부분의 단일 경로는 조건부 데이터 베이스에서 만들어 진다. 즉, 모든 단일 경로로부터 조합된 패턴은 그 자신의 프리픽스(prefix)를 포함한다.
- [0052] 이하, 상기 패턴들 이외의 본 발명을 실시하는데 필요한 패턴들에 대하여 설명한다.
- [0053] N-itemset 패턴은 N개의 아이템으로 구성되고 중복되는 아이템은 가지고 있지 않다. 일 실시예에서, 4가지 트랜잭션들인 {a},{a, c},{b, c} 및 {a, b, c, d}과 함께 데이터베이스 D가 주어진 경우, P는 {{a}, {b}, {c}, {d}, {a,b}, {a,c}, {a,d}, {b,c}, {b,d}, {c,d}, {a,b,c}, {a,b,d}, {a,c,d}, {b,c,d}, {a,b,c,d}}가 될 수 있다. 또한, 1-itemset 패턴은 {{a}, {b}, {c}, {d}}, 2-itemset 패턴은 {{a,b}, {a,c}, {a,d}, {b,c}, {b,d}, {c,d}}, 3-itemset 패턴은 {{a,b,c}, {a,b,d}, {a,c,d}, {b,c,d}} 및 4-itemset 패턴은 {{a,b,c,d}}가 될 수 있다.
- [0054] top-k 패턴들은 모든 패턴들 중에서 최상위부터 k번째 위치까지에 존재하는 패턴들이다. 일 실시예에서, x가 top-k 패턴들 중 하나인 경우, x의 서포트는 모든 top-k 패턴들 중에서 가장 작은 서포트값 보다 항상 작지 않을 수 있다. 일 실시예에서, x_k 가 top-k 번째 서포트 값을 가지는 패턴인 경우, x의 집합인 X는 아래의 식(3)와 같이 정의될 수 있다.
- [0055] 식(3) : $X = \{x | \forall x \in P \text{ and } \sup(x) \geq \sup(x_k)\}$
- [0056] N-itemset top-k 패턴들은 각 패턴의 길이인 $j(1 \leq j \leq N)$ 에 따라 분류된 top-k개 패턴들로 정의된다. 일 실시예에서, j-itemset을 길이가 j인 패턴이고, x_j 는 j-itemset top-k 패턴이며, $x_{j,k}$ 가 j-itemset top-k 패턴인 경우, x_j 의 집합인 X_j 는 아래의 식(4)과 같이 표현될 수 있다.
- [0057] 식(4) : $X_j = \{x_j | \forall x_j \in P \text{ and } \sup(x_j) \geq \sup(x_{j,k})\}$
- [0058] 일 실시예에서, 4가지 트랜잭션들인 {a},{a, c},{b, c} 및 {a, b, c, d}과 함께 데이터베이스 D가 주어지고, D로부터 top-5 패턴들을 찾은 경우, 패턴의 길이에 관계없이 가장 큰 값을 가지는 서포트 P에서 패턴들의 일부가 고려될 수 있다. P의 원소들 중에서 top-5 패턴들은 5번째의 패턴인 {b, c}의 서포트보다 크거나 같기 때문에 {{a}, {b}, {c}, {a,c}, {b,c}}가 될 수 있다. 실제로 마이닝을 하는 상황인 경우, 완전한 top-k 패턴을 한번에 찾는 것은 어려우므로, top-k를 마이닝하는 것은 패턴 마이닝을 계속하고 정확한 패턴 결과를 얻을 때까지 top-k 리스트들을 업데이트할 수 있다.
- [0059] 다른 일 실시예에서, 4가지 트랜잭션들인 {a},{a, c},{b, c} 및 {a, b, c, d}과 함께 데이터베이스 D가 주어지고, D로부터 3-itemset top-2 패턴들을 찾은 경우, top-2 패턴들은 각각의 길이가 j ($1 \leq j \leq 3$)가 되는 경우에서 고려된다. 즉, 1-itemset top-2 패턴들은 {{a}, {c}}, 2-itemset top-2 패턴들은 {{a,c}, {b,c}}, 3-itemset top-2 패턴들은 {{a,b,c}, {a,b,d}, {a,c,d}, {b,c,d}}가 될 수 있다. 여기에서, 3-itemset top-2 패턴들의 집합이 2개가 아닌 4개의 원소를 가지는 이유는 두번째 패턴인 {a, b, d}와 같은 서포트를 가지는 2개의 패턴 {a, c, d}, {b, c, d}가 더 있기 때문이다.
- [0060] 도 2는 본 발명의 일 실시예에 따른 FP-growth 알고리즘을 설명하는 흐름도이다.
- [0061] FP-growth는 빈발 패턴(100)을 분할-정복(divide-and-conquer) 접근법에 따라 순환적으로 호출을 하여 빈발 패턴(100)을 마이닝한다. 여기에서, 알고리즘은 작은 크기의 트랜잭션 데이터를 저장하고 마이닝하는 작업을 효율적으로 수행하기 위해 특수 트리 구조인 FP-tree를 이용한다. 알고리즘의 전체적인 과정은 다음과 같다.

- [0062] FP-growth는 주어진 데이터베이스를 제1 스캔한다(단계 S201a). 제1 스캔을 통해 데이터베이스를 구성하는 아이
템들의 서포트를 계산한다. 여기에서, 내림차순으로 아이터의 정보를 얻을 수 있다.
- [0063] FP-growth는 글로벌(global) FP-tree를 생성하기 위해서 주어진 데이터베이스를 제2 스캔한다(단계 S202a). 그
리고나서, 서포트의 크기를 기준으로 내림차순으로 정렬된 데이터베이스의 각각의 트랜잭션에 속한 아이터들을
정렬하고(단계 S203a), 주어진 최소 서포트 기준보다 낮은 서포트를 가진 아이터를 제거한다(단계 S204a).
- [0064] 이 과정을 거친 트랜잭션들은 글로벌 FP-tree에 하나씩 삽입되고(단계 S205a), 글로벌 FP-tree가 생성된다(단계
S206a). 이를 통해 FP-tree는 빈발패턴(100) 마이닝을 위한 모든 정보를 가지게 된다.
- [0065] 상기 과정을 동안, 글로벌 트리를 위한 헤더 테이블이 생성되고, 헤더 테이블은 아이터-이름, 아이터-서포트,
노드-링크 정보를 포함하고 있다. 노드-링크는 트리 안에서 같은 아이터-이름을 가진 노드를 연결하고 있는 것
으로, 같은 아이터-이름을 가진 노드를 쉽고 효율적으로 찾게 한다.
- [0066] 글로벌 FP-tree를 생성한 후에, 알고리즘은 bottom-up 방식으로 헤더 테이블의 각 아이터에 대하여 빈발 패턴을
마이닝하는 작업을 수행한다(단계 S207a). 보다 구체적으로, 헤더 테이블의 마지막 아이터가 첫번째로
선택되고, 선택된 아이터는 프리픽스(prefix)(330)가 되며 알고리즘은 프리픽스(330)의 노드-링크 정보를 사용
하는 글로벌 FP-tree로부터 현재 프리픽스(330)를 포함하는 경로를 탐색한다. 발견된 경로들은 프리픽스(330)를
포함하는 모든 패턴에 관한 정보를 가진다. 발견된 경로들은 서포트에 관해 내림차순으로 정렬되고, 진행된 경
로의 집합이 현재 프리픽스(330)에 대해 조건부 데이터베이스를 호출하는 경우, 각 경로에 대하여 빈발하지 않
는 아이터들은 버려진다. 다음으로, 알고리즘은 조건부 데이터 베이스의 각각의 트랜잭션(혹은 경로)을 프리픽
스(330)에 대한 조건부 FP-tree인 새로운 FP-tree에 삽입하고, 생성된 조건부 트리가 단일 경로를 가지는지 여
부를 확인한다.
- [0067] 일 실시예에서, 조건부 트리가 단일 경로를 갖지 않는 경우, FP-growth는 조건부 트리의 헤더 테이블에서
bottom-up 방식으로 아이터를 다시 선택하고, 선택된 아이터는 이전의 프리픽스(330)에 추가된다. 다음으로, 알
고리즘은 다시 동일한 과정을 반복하여 새로운 프리픽스(330)를 위한 조건부 데이터베이스, 트리 및 헤더테이블
을 생성한다. 다른 일 실시예에서, 조건부 트리가 오직 하나의 경로만을 가지고 있다면, 즉 단일 경로라면, 조
건부 데이터베이스, 트리 및 헤더 테이블이 다시 만들어질 필요가 없기 때문에 FP-growth는 지금까지 추가된 프
리픽스(330)와 단일 경로를 구성하는 아이터를 조합한 빈발 패턴(100)들을 추출한다. 즉, 글로벌 트리의 헤더
테이블에 있는 모든 아이터에 관하여 위의 과정을 반복한 후, 주어진 데이터베이스에서 완전한 빈발 패턴(100)
의 집합이 얻어질 수 있다.
- [0068] 도 3은 본 발명의 일 실시예에 따른 합성 패턴을 설명하는 블록도이다.
- [0069] 도 3를 참조하면, 합성 패턴(300)은 강력 폐쇄 패턴(120), ω (310), $-\omega$ (320), 프리픽스(prefix)(330) 및 테일
(tail)(340)을 포함한다
- [0070] 합성 패턴(300)은 단일 경로에 포함된 모든 패턴들을 마이닝하기 위해서 강력 폐쇄 패턴(120)과 남아있는 아이
터들을 조합하고, 합성 패턴(300)은 강력 폐쇄 패턴(120)과 일반 패턴으로 구성된다. 일 실시예에서, N개 아이
터의 단일 경로인 $T = \{i_1, \dots, i_m, \dots, i_n\}$ 에 대하여, 다른 아이터들은 다른 서포트들을 가지고 있는 반면 i_1 부터
 i_m 까지 아이터들은 같은 서포트를 가질 수 있다. 즉, $\sup(i_1) = \sup(i_2) = \dots = \sup(i_m)$ 인 경우, 이는 강력 폐쇄
패턴(120)이 된다. 다시 말해서, 연속적인 아이터들의 집합은 아이터들이 같은 서포트를 가지고 서포트가 특정
단일 경로의 첫번째 아이터의 값과 같은 경우, 강력 폐쇄 패턴(120)이 된다.
- [0071] 합성 패턴(300)에 포함된 강력 폐쇄 패턴(120)은 단일 경로의 첫번째 아이터와 동일한 서포트를 가진 아이터의
집합이다. 일 실시예에서, 단일 경로에 n개 아이터가 있을 때 i_1 을 단일 경로의 첫번째 아이터(가장 빈발한 아이
터)이고, $\{i_2, i_3, \dots, i_n\}$ 은 남은 아이터들인 경우, i_1 과 같은 서포트를 가진 아이터들의 집합은 $1 \leq s \leq n$ 에
대하여 $\{\sup(i_1) = \sup(i_2) = \dots = \sup(i_s)\}$ 으로 나타낼 수 있다. 일 실시예에서, B가 $\{i_1, i_2, \dots, i_s\}$ 으로부
터 합성된 패턴의 집합인 경우, B는 $B = \{p_1, p_2, \dots, p_b\}$ 으로 나타낼 수 있다. 즉, $\sup(i_1) = \sup(i_2) = \dots = \sup(i_s)$
)이기 때문에 아래의 식(5)도 유지된다.
- [0072] 식(5) : $\sup(i_s) = \sup(\{i_1, i_2\}) = \sup(\{i_1, i_3\}) = \dots = \sup(\{i_1, i_2, i_3\}) = \sup(\{i_1, i_2, i_4\}) = \dots =$

$$\sup(\{i_1, i_2, \dots, i_s\})$$

[0073] 여기에서, B의 모든 원소들도 $\{i_1, i_2, \dots, i_s\}$ 의 부분집합들이므로 $p_1 = \{i_1\}, p_2 = \{i_2\}, \dots, p_b = \{i_1, i_2, \dots, i_s\}$ 으로 나타낼 수 있다.

[0074] 즉, 아래의 식(6)이 성립하고, 모든 패턴들의 서포트는 B의 서포트들과 동일하기 때문에 B는 강력 폐쇄 패턴(120)이 될 수 있다. 결과적으로, $SC = \{i_1, i_2, \dots, i_s\}$ 이다.

$$[0075] \text{식(6) : } \sup(i_s) = \sup(\{i_1, \dots, i_s\}) = \sup(p_1) = \sup(p_2) = \dots = \sup(p_b)$$

[0076] 일 실시예에서, 단일 경로의 첫번째 서포트와 동일한 서포트를 가진 아이템들이 강력 폐쇄 패턴(120)을 형성하는 경우, 모든 강력 폐쇄 패턴(120)에 있는 부분집합들은 같은 서포트를 가질 수 있다. 따라서, 합성 패턴(300)이 빈발 패턴(100)으로 변환될 때, 정확한 서포트 정보가 계산될 수 있다.

[0077] 일 실시예에서, 강력 폐쇄 패턴(120)에 포함되지 않은 아이템들인 $\{i_{m+1}, \dots, i_n\}$ 은 태일 itemset인 TI가 될 수 있고, TI의 먹집합인 $P(TI)$ 는 $P(TI) = \{\text{태일}_1, \text{태일}_2, \text{태일}_3, \dots, \text{태일}_x\}$, $x = 2^{(n-m)}$ 으로 표현될 수 있다. 각 태일은 강력 폐쇄 패턴(120)과, 프리픽스(prefix)(330)가 있는 경우에는 프리픽스(330)도 추가되어 조합된다. 또한, 강력 폐쇄 패턴(120)을 구성하는 아이템들이 아닌 $P(TI)$ 만을 고려하여 단일 경로 트리의 아이템들을 조합할 수 있다.

[0078] 합성 패턴(300)의 서포트는 다음과 같다. 일 실시예에서, n개 아이템의 단일 경로에서 아이템들의 집합인 T에 대하여 T의 합성 패턴(300)인 CP의 서포트인 $\sup(CP)$ 는 $TI = \{i_{m+1}, i_{m+2}, \dots, i_n\}$, $m \leq n$ 인 경우에 아래의 식(7) 및 식(8)과 같이 계산될 수 있다. 식(7) 및 식(8)은 주어진 아이템들의 값 중에서 최소 서포트를 반환한다.

$$[0079] \text{식(7) : } \text{MINsup}(F) = \min(\sup(f_1), \sup(f_2), \dots, \sup(f_k)), \forall f \in F$$

$$[0080] \text{식(8) : } \sup(CP) = \text{MINsup}(CP \cap TI), \text{ if } (CP \cap TI) \neq \emptyset$$

$$[0081] \sup(CP) = \sup(i_1), \text{ otherwise}$$

[0082] 일반 top-k 빈발 패턴(100)을 얻기 위해서, 합성 패턴(300)은 조합 이후에 반드시 변환되어야 한다. 합성 패턴(300)에서 강력 폐쇄 패턴(120)을 구분하기 위해 시작 표시 마크인 ω (310) 및 종료 표시 마크인 $-\omega$ (320)가 추가된다. 여기에서, ω (310) 및 $-\omega$ (320)는 합성 패턴(300)을 일반 패턴으로 변환하는 데 도움을 줄 수 있다.

[0083] 일 실시예에서, 단일 경로인 트리가 주어진 경우, 합성 패턴(300)은 다음과 같이 생성될 수 있다. 우선, 강력 폐쇄 패턴(120)을 생성하고, 강력 폐쇄 패턴(120), 프리픽스(330) 및 $P(TI)$ 의 각각의 원소와 패턴 조합을 실시한다. 여기에서, 패턴을 구분하기 위해서 ω (310)를 강력 폐쇄 패턴(120)의 시작점에 표시하고, $-\omega$ (320)를 끝나는 부분에 표시한다.

[0084] 합성 패턴(300)은 프리픽스(330), 강력 폐쇄 패턴(120) 및 태일(340)로 구성된다. 여기에서, 강력 폐쇄 패턴(120)을 제외한 남아있는 아이템들은 프리픽스(330)와 태일(340)에 속하고 프리픽스(330)는 고정된다. 합성 패턴(300)인 CP는 아래의 식(9)와 같은 구조를 갖는다.

$$[0085] \text{식(9) : } CP = \text{prefix} \cup \text{sc} \cup \text{태일}$$

[0086] 일 실시예에서, p는 T로부터 조합 가능한 빈발 패턴(100)이고 $R = \{p_1, p_2, \dots, p_{2^n-1}\}$ 은 가능한 모든 ps의 집합인 경우, 프리픽스(330)와 p_0 의 조합은 프리픽스(330) 자체이므로 T에 있는 공집합인 p_0 는 R에 포함되지 않을 수 있고, 단일 경로를 생성하기 전에는 top-k 리스트에 삽입되지 않을 수 있다. 따라서, R에 있는 원소의 개수는 2^n 이 아니라 2^n-1 이 된다. 일 실시예에서, $\text{sc} = \emptyset$ 및 $TI = T$ 인 경우, 태일(340)은 TI의 부분집합 중 하나이므로, $\text{태일} \in R$ 이 될 수 있다. 즉, 합성 패턴(300)은 R에 있는 모든 패턴을 표현할 수 있다.

[0087] 일 실시예에서, sc가 T에 있는 모든 아이템을 가지고있는 경우(예를 들어, $SC=T$), 단일 경로로부터 조합될 수 있는 모든 패턴들은 SC의 부분집합이 될 수 있다. 즉 아래의 식(10)을 만족한다.

$$[0088] \text{식(10) : } \forall \text{sc}_{\text{sub}} \in R, \text{ if } \text{sc} = T$$

- [0089] 즉, $sc=TI$ 인 경우, sc 로 R 에 있는 모든 패턴들은 표현될 수 있다. 일 실시예에서, 합성 패턴(300)이 sc 와 TI 의 조합으로 구성되는 경우, $sc=\{i_1, i_2, \dots, i_m\}$, $1 \leq m < n$ 및 $TI = \{i_{m+1}, i_{m+2}, \dots, i_n\}$ 으로 나타낼 수 있다. sc 는 폐쇄 패턴의 특징들을 가지고있기 때문에 SC 는 $\{i_1, i_2, \dots, i_m\}$ 로부터 만들어질 수 있는 패턴들을 포함한다. 반면, 다른 아이템들인 $\{i_{m+1}, i_{m+2}, \dots, i_n\}$ 는 TI 에 속하고 $\{i_{m+1}, i_{m+2}, \dots, i_n\}$ 로부터 생성될 수 있는 패턴들은 $P(TI)$ 와 같다. 그래서, SC 와 $P(TI)$ 태일(340)의 각 원소의 조합인 합성 패턴들은 R 에 있는 모든 패턴들을 포함한다.
- [0090] 강력 폐쇄 패턴(120)의 정의에 따르면 모든 sc 의 부분집합들인 sc_{sub} 는 SC 와 동일한 서포트를 가진다. 일 실시예에서, $SC = \{i_1, i_2, \dots, i_m\}$, $m < n$ 및 $TI = \{i_{m+1}, i_{m+2}, \dots, i_n\}$ 인 경우, R 에 포함되어 있는 p 의 서포트인 $sup(p)$ 는 $sup(i_k)$ 와 같다. 여기에서, i_k 는 p 에서 가장 낮은 서포트를 가지는 아이터이다. 일 실시예에서, CP 가 p 로부터 생성된 합성 패턴(300)인 경우, i_k 는 태일(340) 또는 SC 에 속한다.
- [0091] 합성 패턴(300)의 서포트에 대한 정의로부터 $sup(CP)=sup(i_k)$, if $i_k \in$ 태일 이다. 또한, $sup(CP)=sup(i_k)=sup(p)$ 이다. 일 실시예에서, CP 는 top-k 빈발 패턴(100)을 정확하게 마이닝 할 수 있게 하는 원래의 패턴 p 와 동일한 서포트를 가지고 있다. 일 실시예에서, $i_k \in SC$ 인 경우, $sup(CP) = sup(SC)$ 일 수 있다. 그러므로 $sup(SC) = sup(SC_{sub})$ 및 $sup(SC_{sub}) = sup(i_k)$ 이기 때문에 강력 폐쇄 패턴(120)의 정의에 따라 $sup(CP) = sup(p)$ 이다. 이런 이유로 CP 와 p 는 모든 경우에 있어서 같은 서포트를 갖는다. 즉, R 에 있는 모든 패턴들의 서포트들은 정확하게 계산될 수 있다.
- [0092] 도 4a는 본 발명의 일 실시예에 따른 CRM 알고리즘을 설명하는 흐름도이다.
- [0093] CRM(Combination Reducing Method)은 패턴의 길이에 상관없이 단일 경로에서 조합의 개수를 줄이는 top-k의 패턴 마이닝 접근법이다. CRM은 서포트의 크기를 기준으로 내림차순으로 정렬되어 있는 FP-tree를 검색하면서 top-k의 패턴을 찾아낸다.
- [0094] 일 실시예에서, 트리의 현재 상태가 단일 경로가 아닌 경우, 알고리즘은 헤더테이블에서 중간 아이터들을 먼저 선택하고 각 노드의 아이터에서 루트까지 노드를 이동하면서 선택된 아이터에 해당하는 조건적 데이터베이스와 트리를 생성한다. 보다 구체적으로, 프리픽스(330)를 선택하는 과정은 다음과 같다. 헤더 테이블에서 중간 아이터가 선택된다. 선택된 프리픽스를 위한 모든 작업이 끝난 경우, 중간 아이터 바로 위에 있는 아이터가 선택된다. 위의 작업들이 가장 위에 위치한 아이터까지 반복한 후에도 top-k의 패턴들을 찾지 못한 경우에는 중간에 있는 아이터의 바로 아래에 있는 아이터를 선택하고 top-k의 패턴의 집합이 모두 찾아질 때까지 패턴을 찾아내는 것을 계속한다. 마이닝 하는 단계에서 찾아진 빈발 패턴(100)들은 top-k의 리스트에 삽입된다. 현재 트리가 단일 경로인 경우, CRM은 합성 패턴(300)을 만들고, 생성된 합성 패턴(300)이 태일(340)을 포함하는 경우, 알고리즘은 복수의 합성 패턴(300)들을 생성하는 것에 의하여 프리픽스(330), 강력 폐쇄 패턴(120) 및 태일(340) 아이터 집합의 각 부분집합을 결합한다. 여기에서, 각각의 합성 패턴(300)에 대한 아이터들의 가장 작은 서포트는 각각의 합성 패턴(300)의 서포트 값이 된다. 일 실시예에서, 생성된 패턴들의 서포트값이 현재 최소 서포트 기준보다 작지 않은 경우, 생성된 패턴들은 top-k의 리스트에 추가될 수 있다. 마이닝 과정 후에, 리스트는 합성 패턴(300)을 포함하여 top-k 패턴들을 포함한다. 따라서, 작업들을 개시하는 일련의 합성 패턴(300)들을 통해 합성 패턴(300)들을 일반 빈발 패턴(100)들로 바꾸고 이를 통해 의도된 top-k 패턴들을 얻을 수 있다.
- [0095] 전체적인 CRM 알고리즘은 아래와 같이 수행된다.
- [0096] 먼저, 데이터베이스를 스캔한다(단계 S401). 보다 구체적으로, 데이터 베이스를 스캔하는 과정을 통하여 각각의 아이터에 대하여 서포트들을 계산하고 모든 아이터들을 서포트의 크기를 기준으로 내림차순으로 정렬한다. 그 이후에, 알고리즘은 초기 FP-tree와 정렬된 아이터들을 사용하는 글로벌 FP-tree를 생성한다(단계 S402). 여기에서, 처음에는 제거하는 작업이 없기 때문에 생성된 글로벌 FP-tree는 데이터베이스의 모든 정보를 포함한다. 또한, 트리에 대한 헤더 테이블이 동시에 생성된다. 알고리즘은 top-k의 패턴들을 저장하고 관리하기 위하여 Top-k 리스트를 초기화하고(단계 S403), 글로벌 FP-tree와 Top-k 리스트를 사용하는 CRM Growth 방법을 시작하고(단계 S404), CRM Growth 방법으로부터 빈발 패턴을 추출한다(단계 S405). 여기에서, top-k 빈발 패턴들은 Top-k 리스트에 위치하고, 일반 패턴들과 합성 패턴들이 섞여 있다. 이후에, 알고리즘은 Top-k 리스트의 패턴들을 탐색하여 합성 패턴들을 일반 top-k 빈발 패턴들로 바꾸기 위한 과정들을 수행한다(단계 S406).

[0097] 도 4b는 본 발명의 일 실시예에 따른 CRM 알고리즘이다.

[0098] 도 5a는 본 발명의 일 실시예에 따른 CRM Growth 알고리즘을 설명하는 흐름도이다.

[0099] 모든 아이템의 위치를 계산하여 중간 아이템을 선택하고(단계 S501), 현재 FP-tree가 단일 경로인지 여부를 확인한다(단계 S502).

[0100] 일 실시예에서, 현재 FP-tree가 단일 경로인 경우, 알고리즘은 CRM을 사용한다. 합성 패턴(300)을 만들기 위하여 첫번째 아이템과 서포트가 동일한지 여부를 확인한다(단계 S503). 동일한 서포트 값을 가지는 아이템들은 아이템의 집합인 SC에 삽입하고(단계 S504), 나머지 아이템들은 테이블에 포함된다(단계 S505). 여기에서, 알고리즘은 아이템들을 분류하기 위하여 패턴에 ω (310) 및 $-\omega$ (320) 을 표시하고, 분류된 정보는 조합하는 과정에서 사용된다. 다음으로, 합성 패턴(300)의 서포트가 최소 서포트 기준보다 작지 않은지 여부를 확인하고(단계 S506), 최소 서포트 기준보다 더 낮은 서포트 값을 갖지 않는다면, 해당 패턴은 Top-k 리스트에 삽입된다(단계 S507).

[0101] 다른 일 실시예에서, FP-tree가 단일경로가 아닌 경우, 헤더 테이블에서 가운데에 있는 아이템부터 아이템들을 검색한다(단계 S508). 그리고 나서 bottom-up 방식을 사용하여 선택된 노드 링크를 따라 해당 아이템들을 이동하면서 방문한다(단계 S509). 여기에서, 시작 아이템 위에 있는 아이템으로부터 마이닝하는 과정이 수행된다. 그리고 나서, 빈발 패턴(100)인지 여부를 확인하고(단계 S510), 빈발 패턴(100)인 경우 패턴을 Top-k리스트에 삽입한다(단계 S511). 다음으로, 새롭게 생성된 FP-tree가 적어도 하나의 노드를 가지는지 여부를 확인하고(단계 S512), FP-tree가 적어도 하나의 노드를 가진다면 선택된 아이템인 i_c 를 프리픽스(330)에 포함시키고(단계 S513), 확장된 프리픽스를 위해 FP-tree를 사용하는 것을 재귀적으로 호출한다(단계 S514). Growth 부분 이후에 i_c 는 i_c 에 대한 모든 과정이 충분히 수행되었기 때문에 제거된다(단계 S515).

[0102] 도 5b는 CRM Growth 알고리즘의 일 실시예이다.

[0103] 예를 들어, 표2는 표1에 있는 아이템 'e' 에 대한 조건부 데이터베이스를 나타낸다.

[0104]

표 1

TID	Items
100	a b c
200	a b c d g
300	b d f g
400	b d
500	a b c d e f g
600	a b c
700	b d f g
800	a b c d e
900	a b c
1000	b c g

[0105]

표 2

TDI	Conditional Items
1	b c a d g f
2	b c a d

[0106]

여기에서, 아이템 'e' 는 프리픽스(330)가 될 수 있다. 최소 서포트가 1일 때, 표에 있는 아이템들은 도 5b에 있는 단일경로인 FP-tree를 생성할 수 있다. 그림은 강력 폐쇄 패턴(120)을 사용하는 것에 의하여 압축된 패턴들이 주어진 것에 대한 과정을 반영할 수 있다. 트리는 아이템 {b,c,a,d,g,f}로 이루어져 있고 서포트들은 {b:2,c:2,a:2,d:2,g:1,f:1}일 수 있다. 조건부 데이터베이스는 아이템 'e'를 선택하는 것에 의하여 얻어졌기 때문에 프리픽스(330)는 {e}로 설정될 수 있다. 일반적인 조합 방법을 사용하기 위하여, 모든 아이템들은 조합되어야 한다. 반면에, CRM 접근법을 사용하면 패턴 {b,c,a,d}는 내부 아이템들이 같은 서포트 값 2를 갖기 때문에

강력 폐쇄 패턴(120)이 될 수 있다. 패턴 $\{b, c, a, d\}$ 의 길이는 4이고, 합성 패턴(300) 결과들은 $\{e, \omega, b, c, a, d, -\omega\}$, $\{e, \omega, b, c, a, d, -\omega, g\}$, $\{e, \omega, b, c, a, d, -\omega, f\}$ 및 $\{e, \omega, b, c, a, d, -\omega, g, f\}$ 일 수 있다. 또한, 패턴 $\{b, c, a, d\}$ 는 강력 폐쇄 패턴(120)이고 $\{e\}$ 는 프리픽스(330)이며, $\{g, f\}$ 는 태일(340) 아이템 집합이 될 수 있다.

[0107] 도 5c는 본 발명의 일 실시예에 따른 CRM Growth 알고리즘이다.

[0108] 도 6a는 본 발명의 일 실시예에 따른 Top-k list를 설명하는 흐름도이다.

[0109] CRM은 패턴들의 길이를 고려하지 않으므로, CRM은 top-k 후보 패턴들을 길이에 상관없이 Top-k리스트에 삽입한다. 여기에서, 삽입된 결과들은 일반 패턴들과 합성 패턴(300)들로 이루어진다. Growth 단계들이 완료되었을 때, 모든 top-k패턴들은 Top-k리스트에서 찾을 수 있다. Top-k리스트는 효율적인 패턴 제거 작업을 위하여 지속적으로 최소 서포트를 갱신한다. 패턴들을 제거하는데 사용되어 지는 최소 서포트는 Top-k리스트에 있는 k번째 패턴의 서포트가 된다. 그래서 최소 서포트는 빈발패턴(100)들이 리스트에 추가되는 것으로서 계속 증가 된다. CRM에서 일반 빈발 패턴(100)들과 합성 빈발 패턴들 모두 리스트 안에 함께 저장되어 있기 때문에, 마이닝 과정의 진행에 의존하는 일반 top-k 패턴 마이닝 접근법과 다른 최소 서포트값을 가진다. 그러나, 패턴의 최소 서포트는 항상 일반 방법들에 의한 최소 서포트보다 작거나 같기 때문에 CRM은 패턴의 손실 없이 모든 의도된 top-k 패턴들을 찾을 수 있다.

[0110] 보다 구체적으로, minsup_s 가 합성 패턴(300)들을 포함하는 Top-k 리스트를 사용하는 CRM에 의한 최소 서포트이고, minsup_0 이 일반 top-k 마이닝에 의한 최소 서포트인 경우, $\text{minsup}_s \leq \text{minsup}_0$ 이므로 CRM은 의도된 모든 top-k 패턴들을 추출할 수 있다. minsup_s 와 minsup_0 은 CRM과 일반 top-k마이닝에 의한 k번째 패턴의 서포트들이다. 일반 방법의 각각의 패턴인 p 및 p 의 top-k 집합인 P 가 $P=\{p_1, p_2, \dots, p_k\}$ 로 주어지고, $\text{minsup}_s=\sup(p_k)$ 이다. Top-k 리스트가 합성 패턴(300)들을 하나도 갖지 않는 경우, 동일한 top-k패턴들이 저장되었기 때문에 $\text{minsup}_s=\text{minsup}_0$ 이다. 일 실시예에서, $\Gamma=\{u_1, u_2, \dots, u_k\}=\{p_1, p_2, \dots, p_k\}$ 이고 $\text{minsup}_s=\sup(p_k)$ 이다. 패턴들의 집합인 $\{p_a, \dots, p_b\}$ 이 마이닝 과정에서 새롭게 찾아지고 합성 패턴(300)이 n 로 표현된다고 가정하면, P 와 Γ 는 $P=\{p_1, p_2, \dots, p_k, \dots, p_{k+b-a}\}$ 와 $\Gamma=\{u_1, u_2, \dots, n, \dots, u_k\}$ 로 나타내어질 수 있다. 즉, $\text{minsup}_0=\sup(p_k)$ 이고, $\text{minsup}_s=\sup(u_{k-1})$ 이다. p_{k+b-a} 는 u_k 와 같기 때문에 $\sup(u_{k-1})$ 은 $\sup(u_k)$ 보다 작거나 같다. 이러한 이유에서 $\text{minsup}_s \leq \text{minsup}_0$ 이다.

[0111] 예를 들어, 표1에 있는 데이터를 가지고 top-k 패턴들을 마이닝하고, k 는 4일 때, Top-k 리스트에 저장된 첫번째 패턴은 $\{d\}$ 이고 이것의 서포트는 5이다. $\{d\}$ 에 대한 조건부 FP-tree는 단일 경로인 $\{b, c, a\}$ 를 가진다. 여기에서, 노드들의 서포트들은 5, 3 및 3이다. 그리고 합성 패턴(300)은 생성되지 않고 Top-k 리스트에 프리픽스(330), $\{d\}$ 및 단일 경로인 $\{d, b\}$, $\{d, c\}$, $\{d, a\}$, $\{d, b, c\}$, $\{d, b, a\}$ 및 $\{d, b, c, a\}$ 로부터 모든 부분집합들 사이의 모든 조합들은 삽입한다. 그리고 나서, k번째 패턴인 $\{d, a\}$ 의 서포트가 3이기 때문에 현재 최소 서포트는 3으로 설정된다. 다음 단계에서, $\{a\}$ 는 프리픽스(330)로 선택되고 단일 경로 $\{b, c\}$ 인 조건부 FP-tree는 만들어진다. 여기에서, $\{b\}$ 및 $\{c\}$ 는 같은 서포트 값 6을 가진다. 그래서 그들은 강력 폐쇄 패턴(120)이 되고, 프리픽스(330)와 함께 조합하는 것은 합성 패턴인 $\{a, \omega, b, c, -\omega\}$ 으로 표시된다. 그것을 리스트로 삽입한 결과는 $\{a\}$, $\{a, \omega, b, c, -\omega\}$, $\{d\}$, 및 $\{d, b\}$ 이고, 최소 서포트는 다시 5로 설정된다. 그러나 만약 우리가 합성 패턴 기술을 사용하지 않는다면, Top-k리스트는 $\{a\}$, $\{a, b\}$, $\{a, c\}$ 및 $\{a, b, c\}$ 패턴의 집합을 가지고 최소 서포트는 6이 된다.

[0112] 보다 구체적으로, Top-k리스트는 아래와 같이 진행한다. Top-k리스트는 처음에 리스트에 빈발 패턴(100) 또는 합성패턴(300)을 삽입하고(단계 S601), 리스트가 가득 차고 삽입된 패턴의 서포트가 최소 서포트보다 작지 않은지 여부를 확인한다(단계 S602). 리스트가 가득 차고 삽입된 패턴의 서포트가 최소 서포트보다 작지 않은 경우, 서포트의 크기를 기준으로 내림차순으로 리스트를 정렬하고(단계 S603), k번째 패턴 서포트를 새로운 최소 서포트로 정한다(단계 S604).

- [0113] 도 6b는 본 발명의 일 실시예에 따른 Top-k 리스트의 알고리즘이다.
- [0114] 도 7a는 본 발명의 일 실시예에 따른 합성 패턴의 변환을 설명하는 블록도이다.
- [0115] 마이닝 하는 과정들 이후에, Top-k리스트는 일반 빈발 패턴(100)들뿐만 아니라 합성 패턴(300)들도 포함하게 되나, 패턴들은 효과적인 마이닝을 위해서 압축되어졌기 때문에 합성 패턴(300)들로부터 실제 빈발 패턴(100)들을 직접적으로 얻지 못한다. 즉, 실제 빈발 패턴(100)들을 얻기 위해서 CRM은 합성 패턴(300)들을 해당 일반 패턴들로 다시 바꾸어야 한다. Top-k리스트에는 일반 빈발 패턴(100)(704a) 및 합성 패턴(300)(701a, 702a 및 703a)에 해당하는 두가지 종류의 패턴들이 있다. 여기에서, 합성 패턴(300)들은 첫번째 세가지 형식들처럼 (705a, 706a 및 707a) 각 합성 패턴(300)(701a, 702a 및 703a)들에 대하여 세가지 경우로 다시 세분화되어 진다. 일반 빈발 패턴(100)(704a)들은 추가적인 작업 없이 직접적으로 추출될 수 있다.
- [0116] 도 7b는 본 발명의 일 실시예에 따른 합성 패턴의 변환을 설명하는 흐름도이다.
- [0117] CRM이 마이닝 과정을 종료했을 때, Top-k리스트에 있는 합성 패턴(300)들은 아래의 단계에 따라 일반 패턴들로 바뀐다.
- [0118] CRM은 아이템들이 시작 표시인 ω (310)와 종료 표시인 $-\omega$ (320) 사이에 존재하는지 여부를 확인한다(단계 S701b). 아이템들이 시작 표시인 ω (310)와 종료 표시인 $-\omega$ (320) 사이에 존재하는 경우, 찾아진 아이템들은 서로 결합 된다(단계 S702b). 일 실시예에서, 가능한 모든 부분집합들은 아이템들로부터 생성된다.
- [0119] 다음으로, top-k 패턴을 추출한다(단계 S703b). 보다 구체적으로, 강력 폐쇄 패턴(120)이 특정 합성 패턴(300)에 존재하는 경우, 패턴의 모든 부분집합들은 바로 top-k 빈발 패턴들이 될 수 있고, 프리픽스(330)가 강력 폐쇄 패턴(120)과 함께 합성 패턴(300)들에 포함되어 진 경우, 강력 폐쇄 패턴(120)의 각각의 부분집합과 함께 프리픽스(330)의 모든 조합들은 유효한 패턴들이 될 수 있다. 또한, 태일(340) 아이템 집합이 프리픽스(330)와 함께 있는 경우, CRM은 프리픽스(330)와 태일(340) 각각의 부분집합과 함께 프리픽스(330)를 조합할 수 있다. 즉, top-k 패턴들이 된다.
- [0120] 알고리즘은 k개의 빈발 패턴(100)들을 찾을 때까지 위의 과정들을 반복하고, k개의 빈발 패턴(100)이 다 찾아지면(단계 S704b), 리스트에 남아있는 패턴들은 더 이상 필요없기 때문에 제거된다(단계 S705b).
- [0121] 마이닝하는 과정들이 종료할 때, 합성 패턴(300)들을 포함하는 Top-k리스트를 빈발 패턴(100)으로 바꾸는 것이 일반적인 top-k마이닝 접근법의 결과와 동일하다면, CRM은 어떠한 패턴의 손실 또는 에러 없이 top-k 패턴들의 마이닝을 완료하는 것으로 간주 될 수 있다. 일 실시예에서, 일반적인 top-k 마이닝에서의 패턴이 α 인 경우, $L=\{\alpha_1, \alpha_2, \dots, \alpha_k\}$ 를 패턴들의 집합이고, CRM에서 합성 패턴들을 포함하는 β 에 대하여 $M=\{\beta_1, \beta_2, \dots, \beta_k\}$ 는 패턴들의 집합이고, γ 는 M 을 바꾼 것인 경우 $R=\{\gamma_1, \gamma_2, \dots, \gamma_k\}$ 는 패턴들의 집합일 때, $R=L$ 이 될 수 있다. 보다 구체적으로, 특정 데이터베이스에 대하여, 서포트를 기준으로 내림차순으로 정렬되어 있는 n개의 빈발 패턴(100)들의 집합이 주어졌을 때, $P=\{\alpha_1, \alpha_2, \dots, \alpha_k\}$ 는 top-k 빈발 패턴(100)들의 집합이고, $L=L=\{\alpha_1, \alpha_2, \dots, \alpha_k\}$ 으로 나타내어 질 수 있다. $\{\alpha_e, \dots, \alpha_f\}(1 \leq e < f \leq k \leq n)$ 은 합성 패턴(300)들로 변환할 수 있는 패턴들의 집합이고, M 은 CRM에서 합성 패턴(300)들을 방출하기 전에 top-k 패턴들의 집합인 경우, $M=\{\beta_1, \beta_2, \dots, \beta_k\} = \{\alpha_1, \alpha_2, \dots, \alpha_{e-1}, \beta_e, \alpha_{f+1}, \dots, \alpha_{f+2}, \dots, \alpha_k, \dots, \alpha_{k+f-e}\}$ 로 표현될 수 있다. 여기에서, β_e 는 합성 패턴(300)이다. $\{\beta_1, \dots, \beta_{e-1}\}$ 은 합성 패턴(300)이 되지 않는 부분이기 때문에 $\{\alpha_1, \dots, \alpha_{e-1}\}$ 과 동일하다. 게다가 CRM의 최종 결과 때문에 R 은 $\{\alpha_e, \dots, \alpha_f\}$ 대신에 β_e 를 저장하고, L 과 비교하여 추가적으로 f-e개의 패턴들을 가진다. $\{\beta_{e+1}, \dots, \beta_k\}$ 는 $\{\beta_{e+1}, \dots, \beta_k\}$ 가 합성 패턴(300)이 아니기 때문에 $\{\alpha_{f+1}, \dots, \alpha_{f+2}, \dots, \alpha_k, \dots, \alpha_{k+f-e}\}$ 와 동일하다. $R=\{\gamma_1, \gamma_2, \dots, \gamma_k\}$ 은 M 의 β_e 를 원래의 패턴들로 변환한 것이고 k번째 패턴 이후의 패턴들을 삭제하기 때문에 R 은 L 과 동일하다.
- [0122] 예를 들어, 표3은 표1에 기반하여 CRM에 의한 top-10 패턴들을 계산한 후의 결과이다.

표 3

Sequence	Items	Support
1	b	10
2	c	8
3	c b	8
4	a	6
5	a ω c b $-\omega$	6
6	d	5
7	g	5
8	d b	5
9	g b	5
10	g d	4
11	g d b	4

[0123]

[0124]

패턴 변환 과정은 top-k리스트에서 가장 큰 서포트 값을 가지는 패턴부터 시작한다. 알고리즘은 일반 패턴들인 1~4번째 패턴들을 버리고, 5번째 패턴은 합성 패턴(300)이므로 변환한다. 여기에서, 알고리즘은 프리픽스 아이템 {a}를 버리고, 시작 ' ω '부터 종료 ' $-\omega$ ' 까지 강력 폐쇄 패턴(120)을 먼저 고려한다. 즉, 아이템들 패턴에 있는 {c,d}을 조합하고, 조합 결과들은 프리픽스와 조합된다. 그리고 나서, 완성된 패턴들은 3개의 일반 패턴들로 변환되고 10번째 패턴은 {d,b}가 된다. 게다가, 패턴 {g,b}는 서포트가 {d,b}와 동일하기 때문에 패턴 {g,b} 또한 유효하다. 즉, $\sup(\{d,b\}) > \sup(\{g,d\})$ 이기 때문에 12번째부터 마지막까지 존재하는 아이템들은 제거된다. 최종 top-k패턴들은 {{b}, {c}, {c,b}, {a}, {a,c}, {a,b}, {a,c,b}, {d}, {g}, {d,b}, {g,b}} 이 될 수 있다.

[0125]

도 7c는 본 발명의 일 실시예에 따른 합성 패턴의 변환에 대한 알고리즘이다.

[0126]

도 8a는 본 발명의 일 실시예에 따른 CRMN 알고리즘을 설명하는 흐름도이다.

[0127]

CRMN(Combination Reducing Method for N-itemset top-k patterns)은 패턴의 길이에 의존하여 top-k 빈발 패턴들을 마이닝하기 위하여 다수의 Top-k리스트를 가진다. CRMN의 마이닝 과정의 기본적인 작동은 CRM과 유사하나 CRMN 알고리즘은 아이템-길이에 대한 조건이 있고 길이가 1인 것부터 사용자가 지정한 길이까지 각각의 길이에 대한 top-k 빈발 패턴들을 마이닝한다.

[0128]

알고리즘은 먼저 주어진 데이터베이스를 스캔하고(단계 S801) 길이가 1인 아이템의 서포트를 카운팅한다(단계 S802). 그리고 나서, 아이템들은 서포트의 크기를 기준으로 내림차순으로 정렬되고(단계 S803), 알고리즘은 데이터베이스와 아이템 순서에 대한 정보를 이용하여 글로벌 FP-tree를 생성한다(단계 S804). 이후에, 알고리즘은 N-itemset Top-k 리스트와 프리픽스(330)를 초기화 하고(단계 S805), CRMN과 FP-tree를 사용하여 CRMN Growth 과정을 수행한다(단계 S806). CRMN Growth와 패턴 변환 단계들 이후에, 완성된 N-itemset top-k 빈발 패턴들의 집합이 얻어진다.

[0129]

도 8b는 본 발명의 일 실시예에 따른 CRMN 알고리즘이다.

[0130]

도 9a는 본 발명의 일 실시예에 따른 CRMN Growth 알고리즘을 설명하는 흐름도이다.

[0131]

현재 FP-tree가 단일 경로인지 여부를 확인한다(단계 S901). 일 실시예에서, FP-tree가 단일 경로인 경우, 단일 경로로부터 강력 폐쇄 패턴(120)과 태일(340)을 찾고(단계 S902), 합성 패턴을 생성한다(단계 S903). 그리고 나서 r과 u는 계산된다(단계 S904). 여기에서, r은 태일(340)의 크기이고 u는 SC의 길이로서, r과 u의 길이에 따라 합성 패턴(300)이 적절한 N-itemset Top-k 리스트들에 삽입될 수 있다. 즉, 합성 패턴(300)을 만들기 위해 프리픽스(330), 태일(340) 및 SC를 조합한 후에, 알고리즘은 r과 u를 사용하여 합성 패턴(300)을 적절한 N-itemset Top-k 리스트에 위치시킨다(단계 S905). 일 실시예에서, 트리에 강력 폐쇄 패턴(120)이 없다면, 태일(340)와 프리픽스(330)의 조합들은 길이에 따라 N-itemset Top-k 리스트들에 삽입될 수 있다.

[0132]

다른 일 실시예에서, 현재 트리가 단일 경로가 아닌 경우, 알고리즘은 트리를 이동해다니고(단계 S906), 조건부

FP-tree를 생성하고(단계 S907) 순환적으로 CRMN Growth를 호출한다.

[0133] 도 9b는 본 발명의 일 실시예에 따른 CRMN Growth 알고리즘이다.

[0134] 도 10a는 본 발명의 일 실시예에 따른 N-itemset Top-k list를 설명하는 블록도이다.

[0135] CRMN에서 u 는 합성 패턴(300)에 있는 어떠한 강력 폐쇄 패턴(120)의 아이템들의 개수이고, 패턴으로부터 얻어진 모든 하위 패턴들은 1부터 u 까지의 길이를 가진다. 일 실시예에서, 합성 패턴(300)의 특정 일반 패턴(예를 들어, 프리픽스(330)와 테일(340) 패턴들)의 길이가 r 로 주어진 경우, 최종 결과 패턴들은 r 부터 $r+u$ 까지의 길이를 가질 수 있다. 여기에서, 최종 결과 패턴들은 합성 패턴(300) 형식으로서 길이에 따라 분류된 N-itemset Top-k 리스트들에 삽입된다. 즉, 패턴은 다른 길이의 하위 패턴들을 가지고 있기 때문에 합성 패턴(300)은 다양한 위치에 추가될 수 있다. 예를 들어, CRMN은 도5b에 있는 단일 경로를 합성 패턴(300)들로 변환하고, 길이에 따라 적절한 Top-k리스트들에 저장한다. 여기에서, N-itemset은 5로 설정된다. $\{e, \omega, b, c, a, d, -\omega\}$ 에서 r 및 u 는 각각 1과 5이다. 즉, 길이가 1에서 5 사이인 패턴들을 포함하고, 1-itemset Top-k 리스트부터 5-itemset Top-k 리스트까지 삽입된다. $\{e, \omega, b, c, a, d, -\omega, g\}$ 의 r 및 u 는 2 및 4이고 패턴들은 2에서부터 6까지의 길이를 갖기 때문에, 2-itemset Top-k리스트부터 5-itemset Top-k리스트까지 추가된다. 마찬가지로, $\{e, \omega, b, c, a, d, -\omega, f\}$ 및 $\{e, \omega, b, c, a, d, -\omega, g, f\}$ 는 2-itemset Top-k리스트 부터 5-itemset Top-k리스트까지 및 3-itemset Top-k 리스트부터 5-itemset Top-k리스트 까지 각각 저장된다. N-itemset 이 5로 설정되었기 때문에 6 이상의 itemset Top-k리스트들 및 패턴들을 고려되지 않는다.

[0136] 보다 구체적으로, 특정 N-itemset Top-k 리스트에 k 개 이상의 패턴들이 있는 경우, N-itemset의 최소 서포트는 갱신된다. 여기에서, 갱신된 서포트 값은 리스트에서 k 번째 패턴의 서포트 값이 된다. 모든 N-itemset의 최소 서포트들 중에서, 가장 작은 값은 최소 서포트 기준 값이 된다. 여기에서, SKC(Short K-pattern Counter)가 적용되고, SKC는 합성 패턴(300)으로부터 얻어진 패턴들의 개수를 저장하고 저장된 숫자를 이용하여, 최소 서포트를 정하는 타겟 패턴의 위치를 계산한다. 일 실시예에서, ψ 는 N-itemset Top-k 리스트에 포함된 합성 패턴(300)들의 개수이고, o 는 합성 패턴(300)들로부터 얻어질 수 있는 패턴들의 개수인 경우, 최소 서포트를 결정하는 패턴의 위치는 $k+\psi-\sum(\text{from } i=1 \text{ to } \psi) o_i$ 가 될 수 있다. 일 실시예에서, 합성 패턴(300)인 $\{a, \omega, b, c, / \omega, e\}$ 이 합성 패턴(300)을 갖지 않는 N-itemset Top-k 리스트에 추가된 경우, N-itemset Top-k 리스트는 $\{a, b, e\}$, $\{a, c, e\}$, $\{a, d, e\}$, $\{a, b, c, e\}$, $\{a, b, d, e\}$, $\{a, c, d, e\}$ 및 $\{a, b, c, d, e\}$ 인 패턴들을 포함할 수 있다. 따라서 합성 패턴(300)은 3-itemset Top-k리스트부터 5-itemset Top-k 리스트까지 각각 저장될 수 있다. 보다 구체적으로, 3-itemset Top-k리스트에 $\{a, b, e\}, \{a, c, e\}, \{a, d, e\}$ 가 저장되고, 4-itemset Top-k리스트에 $\{a, b, c, e\}, \{a, b, d, e\}, \{a, c, d, e\}$ 가 저장되며, 5-itemset Top-k 리스트에 $\{a, b, c, d, e\}$ 가 저장된다. 즉, 비록 하나의 패턴(합성 패턴(300))이 각 리스트에 입력되었을지라도, SKC는 각 길이에 대한 합성 패턴(300)으로부터 얼마나 많은 패턴들이 생성되었는지를 고려하여 N-itemset 최소 서포트들을 계산한다. 예를 들어, 특정 N-itemset Top-k 리스트에 3개의 유효한 패턴들을 포함하는 합성 패턴(300)이 존재한다면, ψ 는 1이고 o_1 은 3이기 때문에, N-itemset의 최소 서포트를 결정하는 위치는 $k-2$ 가 된다.

[0137] 예를 들어, Top-k는 5이고 3-itemset Top-k리스트는 세개의 패턴들 $\{d, b, a\}, \{d, b, c\}$ 및 $\{d, a, c\}$ 이 있고 각각의 서포트는 5, 4 및 3이고, 서포트가 2인 합성 패턴 $\{a, \omega, b, c, d, -\omega, e\}$ 이 리스트에 삽입되는 것을 고려하자. 만약 SKC가 사용되지 않는다면, 네개의 패턴들 $\{d, b, a\}, \{d, b, c\}, \{d, a, c\}$ 및 $\{a, \omega, b, c, d, -\omega, e\}$ 는 리스트에 포함되고, 5번째 패턴이 없기 때문에 3-itemset의 최소 서포트는 0이 된다. 반면에, CRMN은 추가적으로 $\{a, \omega, b, c, d, -\omega, e\}$ 로부터 얻어진 길이가 3인 3개의 패턴들을 고려한다. 이 경우, k -두번째 패턴의 서포트는 2이고 3-itemset의 최소 서포트가 된다. 계속해서, SKC를 사용하는 CRMN은 최소 서포트의 값이 빠르게 증가하기 때문에 SKC를 사용하지 않는 CRMN보다 더 빨리 top-k패턴들을 마이닝 할 수 있다.

[0138] 도 10b는 본 발명의 일 실시예에서 N-itemset Top-k리스트를 설명하는 흐름도이다.

[0139] N-itemset Top-k리스트 과정은 N-itemset Top-k 리스트를 관리하기 위하여 호출된다. SKC기술을 적용하기 위하여, N-itemset Top-k 리스트는 패턴 정보와 SKC 값들을 저장한다. 알고리즘은 합성 패턴(300)들로부터 조합될 수 있는 N-itemset들의 개수를 계산한다. 여기에서, aC_b 는 길이가 a 인 itemset에서 길이가 b 인 itemset들을 선택하는 경우들의 개수이다. 또한, 'a'는 강력 폐쇄 패턴(120)에서 아이템들의 개수이고, 'b'는 N으로부터 테일

(340)과 프리픽스(330)에 있는 아이템들의 개수를 제외한 수로서, 합성 패턴(300)으로부터 N-itemset 패턴들을 성공적으로 얻기 위해 요구된다. 패턴은 N-itemset top-k리스트에 삽입되고 리스트는 서포트 값을 기준으로 내림차순으로 정렬된다.

[0140] SKC를 이용하여 N-itemset Top-k 리스트의 사이즈를 계산한 후에(단계 S1001), 리스트의 사이즈가 k보다 큰지 여부를 확인한다(단계 S1002). 리스트의 사이즈가 k보다 큰 경우, 알고리즘은 N-itemset의 최소 서포트를 리스트에 있는 k번째 패턴의 서포트 값으로 갱신한다(단계 S1003). 그 이후에, N-itemset들의 최소 서포트를 중에서 가장 작은 값을 최소 서포트 기준으로 변경하고(단계 S1004) N-itemset Top-k 리스트들로부터 빈번하지 않은 패턴들을 제거한다(단계 S1005).

[0141] 도 10c는 본 발명의 일 실시예에서 N-itemset Top-k리스트를 설명하는 알고리즘이다.

[0142] 도 11은 본 발명의 일 실시예에서 CRMN에서의 합성 패턴 변환을 설명하는 알고리즘이다.

[0143] CRMN에서 top-k패턴들을 위하여 변환하는 과정은 다음과 같다. 마이닝하는 과정을 종료한 후에, N-itemset Top-k 리스트는 각 패턴의 길이인 $j(1 \leq j \leq N)$ 에 대하여 top-k 빈발 패턴(100)들을 포함한다. 리스트는 일반 패턴들 뿐만 아니라 합성 패턴(300)들도 가지기 때문에, 합성 패턴(300)들을 일반 패턴들로 변환하는 것이 필요하다. N-itemset Top-k 리스트에서 조합된 패턴들로부터 유효한 빈발 패턴(100)들을 구하기 위하여, ω 부터 $-\omega$ 까지 (예를 들어, 강력 폐쇄 패턴(120)들) 다른 아이템들(예를 들어, 일반 패턴들)과 함께 아이템들을 조합한다. CRM의 경우로서, 패턴들을 변화하는 것에 대한 세가지 경우들이 있다. CRMN에서 패턴을 변환하는 단계의 기본적인 방법은 CRM과 같다. 그러나 아래와 같은 차이점이 있다. 위에서 언급했듯이, 합성 패턴(300)은 복수의 N-itemset Top-k리스트들에 속할 수 있기 때문에 각 리스트에 대한 중복을 고려하여 패턴들을 변환하여야 한다. CRM에서, 주어진 합성 패턴(300)으로부터 모든 가능한 부분 집합들을 조합한다. 반면에, CRMN에서는 각 N-itemset Top-k리스트에 대한 길이 조건을 만족하는 부분 집합의 부분만을 사용한다. 이를 통해, 중복 패턴들을 생성하는 것은 방지될 수 있고, CRMN은 일반적인 N-itemset Top-k 마이닝 방법들과 비교하여 정확하게 같은 개수의 결과물들을 갖는다.

[0144] 보다 구체적으로, CRMN은 마이닝 과정 이후에 합성 패턴(300)들을 포함하는 N-itemset Top-k의 패턴들을 일반 빈발 패턴(100)들로 변환한다. N-itemset Top-k 리스트들의 각 길이에 대하여 알고리즘은 ω (310) 와 $-\omega$ (320)를 사용하는 합성 패턴(300)들로부터 강력 폐쇄 패턴(120)들과 일반 패턴들(예를 들어, Pr)로 분리한다. 만약 SC와 Pr사이의 각 조합이 N-itemset에 속한다면, N-itemset에 속하는 각 조합은 출력되고, 만약 출력된 패턴들의 개수가 k개보다 많다면 이 과정들은 종료된다.

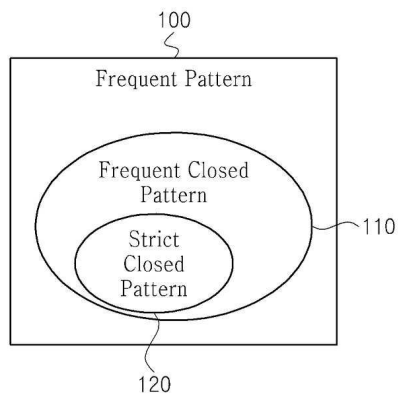
[0145] 상기에서는 본 출원의 바람직한 실시예를 참조하여 설명하였지만, 해당 기술 분야의 숙련된 당업자는 하기의 특허 청구의 범위에 기재된 본 발명의 사상 및 영역으로부터 벗어나지 않는 범위 내에서 본 출원을 다양하게 수정 및 변경시킬 수 있음을 이해할 수 있을 것이다.

부호의 설명

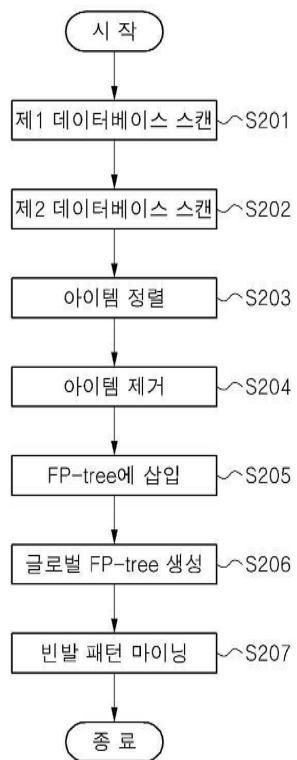
[0146] 100: 빈발 패턴
110: 빈발 폐쇄 패턴 120: 강력 폐쇄 패턴
300: 합성 패턴
310: 시작 표시 마크 320: 종료 표시 마크
330: 프리픽스(prefix) 340: 테일(tail)

도면

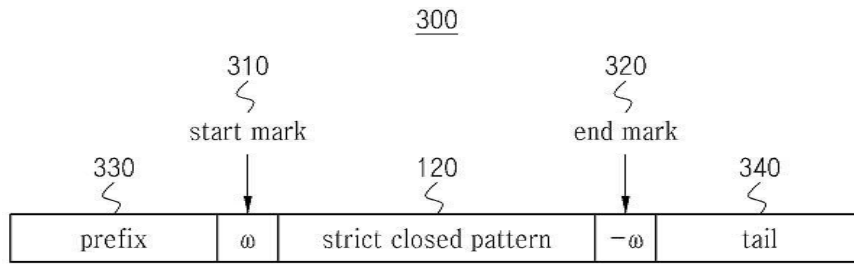
도면1



도면2



도면3



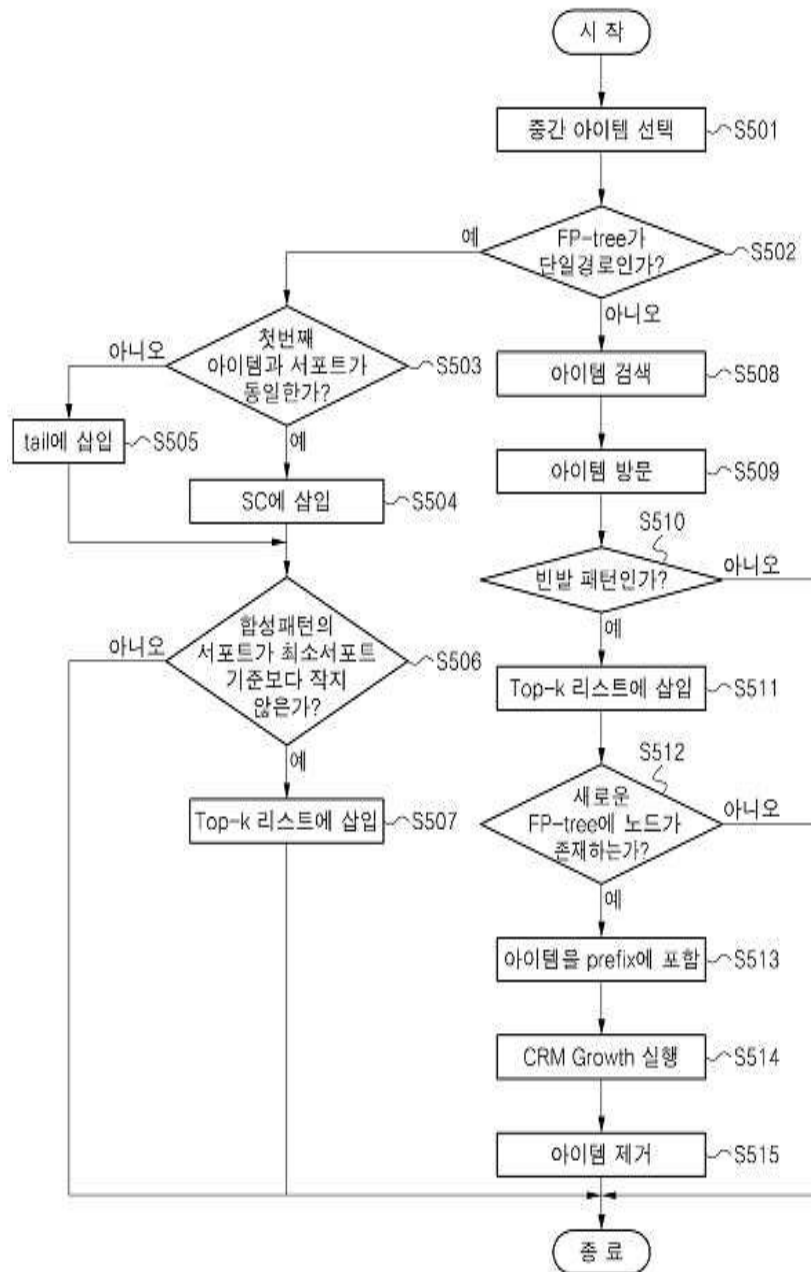
도면4a



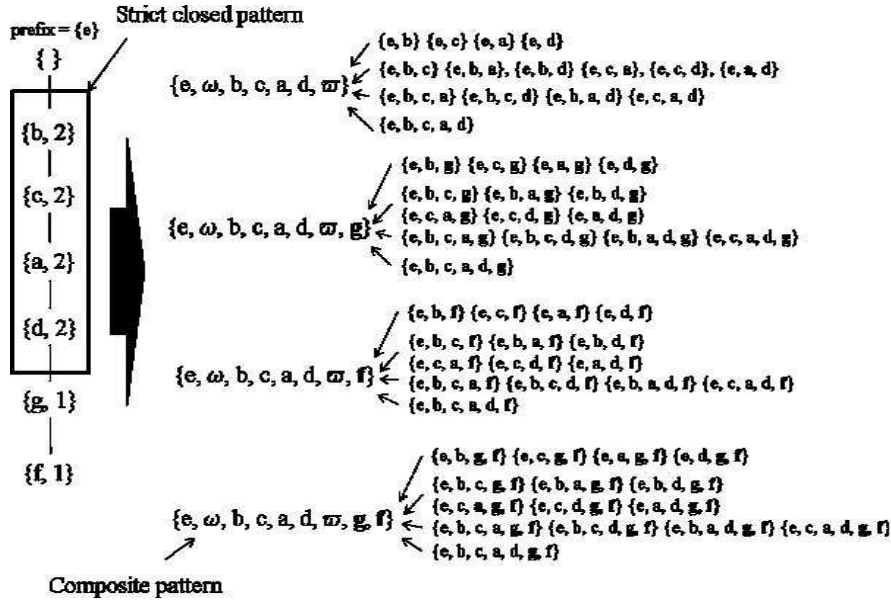
도면4b

Procedure CRM algorithm
D : a transaction database
t : a transaction
prefix : a prefix itemset
order : item sequence sorted in support descending order
k : the number of extracted Top-k patterns
Input : **D**, **k**
Output : Top-k patterns
Begin
 1. for all transition $t \in D$ do
 2. Count item supports for 1-items
 3. **order** $\leftarrow$ support descending order
 4. FP-tree $\leftarrow$ Build global FP-tree(**D**, **order**)
 5. Top-k list $\leftarrow$ Initialize Top-k list()
 6. **prefix** $\leftarrow \emptyset$
 7. CRM Growth(**prefix**, FP-tree, Top-k list)
 8. Releasing Composite Patterns(Top-k list)
End procedure

도면5a



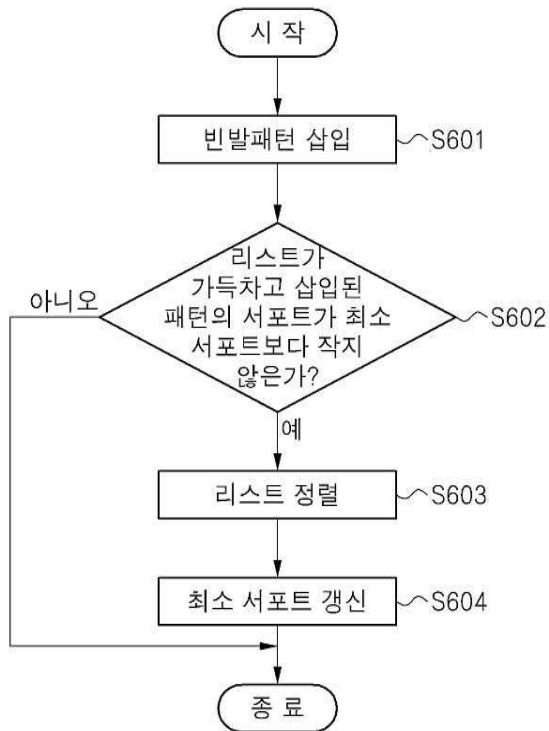
도면5b



도면5c

Procedure CRM Growth(prefix, tree, Top-k list)
SC : a strict closed pattern
Begin
 1. $n \leftarrow$ the number of items
 2. $start \leftarrow \lfloor n/2 \rfloor$
 3. if(single path) /* single path */
 4. for each item, $i_c \in \{i_1, i_2, \dots, i_n\}$
 5. if($sup(i_0) = sup(i_c)$)
 6. $SC \leftarrow SC \cup i_c$ /* Finding strict closed pattern */
 7. Else break
 8. $tail \leftarrow \{i_c, \dots, i_n\}$
 9. if(size of SC > 2) /* the strict closed pattern exists */
 10. for each subset, $tail_{sub}$ in tail
 11. $cp \leftarrow prefix \cup 'w' \cup SC \cup 'w' \cup tail_{sub}$ /* making composite strict closed pattern */
 12. insert cp into Top-k list such that $sup(cp)$ is not smaller than $minsup$
 13. Else for combine tail
 14. $p \leftarrow prefix \cup tail$
 15. insert p into Top-k list
 16. Else /* multiple path */
 17. for each item, $i_c \in \{i_{start}, i_{start-1}, \dots, i_1\}$ /* upper items */
 18. if ($sup(i_c) \geq minsup$)
 19. insert $prefix \cup i_c$ into Top-k list
 20. search from nodes with i_c to the root
 21. generate a conditional pattern base(data base) for i_c
 22. $order \leftarrow$ sorting items in support descending order
 23. construct a conditional FP-tree for i_c
 24. if (new FP-tree $\neq \emptyset$)
 25. $prefix \leftarrow prefix \cup i_c$
 26. CRM Growth(prefix, new FP-tree, Top-k list)
 27. remove i_c in the prefix
 28. for each $i_c \in \{i_{start+1}, \dots, i_n\}$ /* lower items */
 29. conduct the same operations as the lines 18~27
End procedure

도면6a



도면6b

Procedure Top-k list(pattern, support)
 Γ : Top-k list
 $|\Gamma|$: size of Top-k list
 p : a pattern of Top-k list
 k : the expected number of top-k patterns
Begin
 1. $\Gamma \leftarrow \Gamma \cup \{\text{pattern} | \text{support} \geq \text{minsup}\}$
 2. sort Γ in support descending order
 3. if($|\Gamma| \geq k$)
 4. if($\text{sup}(p_k) > \text{minsup} // p_k$ is the k -th pattern in the Top-k list
 5. $\text{minsup} \leftarrow \text{sup}(p_k)$
 6. remove $p_k \sim p_{|\Gamma|} //$ remove infrequent top-k patterns
End procedure

도면7a

Strict closed pattern $\leadsto 701a$

$\{\omega, a, b, c, \varpi\} \rightarrow \begin{cases} \{a\} \{b\} \{c\} \leadsto 705a \\ \{a, b\} \{a, c\} \{b, c\} \leadsto 706a \\ \{a, b, c\} \leadsto 707a \end{cases}$

combination

prefix + Strict closed pattern $\leadsto 702a$

$\{e, \omega, a, b, c, \varpi\} \rightarrow \begin{cases} \{e, a\} \{e, b\} \{e, c\} \\ \{e, a, b\} \{e, a, c\} \{e, b, c\} \\ \{e, a, b, c\} \end{cases}$

combination

prefix + Strict closed pattern + tail itemset $\leadsto 703a$

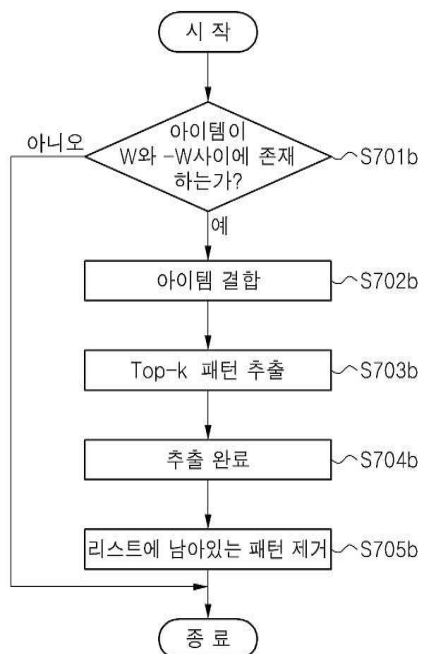
$\{e, \omega, a, b, c, \varpi, g\} \rightarrow \begin{cases} \{e, a, g\} \{e, b, g\} \{e, c, g\} \\ \{e, a, b, g\} \{e, a, c, g\} \{e, b, c, g\} \\ \{e, a, b, c, g\} \end{cases}$

combination

Frequent pattern $\leadsto 704a$

$\{a, b, c, e\} \rightarrow \{a, b, c, e\}$

도면7b



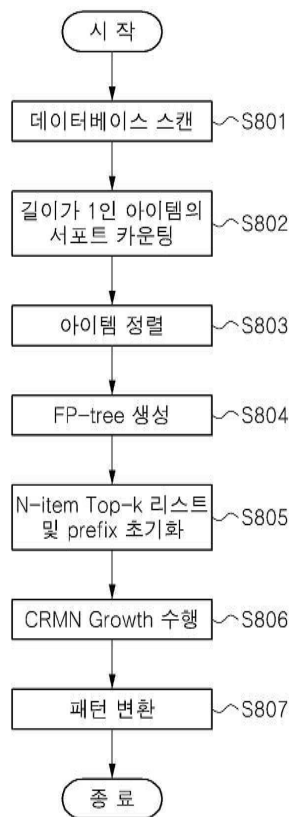
도면7c

Procedure Releasing Composite patterns(Top-k list)
Itemset P_r : a set of items except for a strict closed pattern

1. For each $p \leftarrow$ a pattern of Top-k list
2. For each item, $c \in p$
3. if(c is included in items between ' ω ' and ' ϖ ' in p)
4. $SC \leftarrow SC \cup c$
5. else
6. $P_r \leftarrow P_r \cup c$
7. for each subset, SC_{sub} in SC
8. $L \leftarrow P_r \cup SC_{sub}$
9. print(L)
10. count++
11. if(count > k)
12. break

End procedure

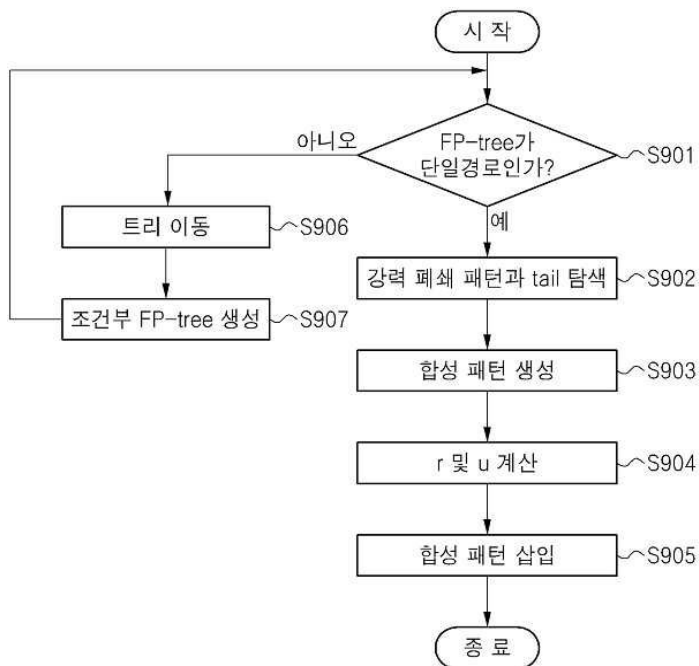
도면8a



도면8b

Procedure CRMN algorithm
D: a transaction database
t: a transaction
prefix: a prefix itemset
order: item sequence sorted in support descending order
k: the number of extracted top-k patterns
N: the maximum length of N-itemset
Input: D, k, N
Output: top-k patterns or N-itemset top-k patterns
Begin
 1. for all transition $t \in D$ do
 2. Count item supports for 1-items
 3. $order \leftarrow$ support descending order
 4. FP-tree $\leftarrow$ Build global FP-tree(D, $order$)
 5. N-itemset Top-k list $\leftarrow$ Initialize N-itemset Top-k list()
 6. $prefix \leftarrow \emptyset$
 7. CRMN Growth($prefix$, FP-tree, N-itemset Top-k list)
 8. Releasing N-itemset Composite Patterns(N-itemset Top-k list)
End procedure

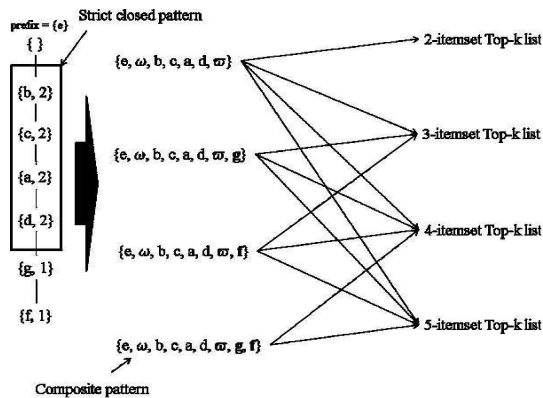
도면9a



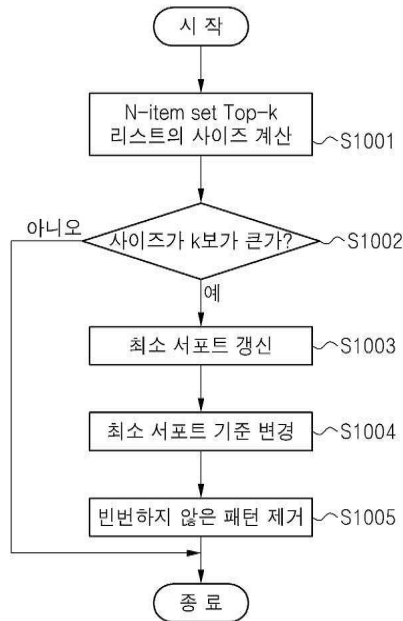
도면9b

Procedure CRMN Growth(*prefix*, *tree*, *N*-itemset Top-k list)
SC: a strict closed pattern
Begin
 1. n = the number of items
 2. $start \leftarrow \lceil n/2 \rceil$
 3. if(single path) /* single path */
 4. for each item $i_c \in \{i_1, i_2, \dots, i_n\}$
 5. if($\text{sup}(i_0) = \text{sup}(i_c)$)
 6. $SC \leftarrow SC \cup i_c$ /* Finding strict closed pattern */
 7. Else break
 8. $tail \leftarrow \{i_c, \dots, i_n\}$
 9. $r \leftarrow \text{size of } tail$
 10. $u \leftarrow \text{size of } SC$
 11. if($m > 2$) /* the strict closed pattern exists */
 12. for each subset, $tail_{sub}$ in $tail$
 13. $cp \leftarrow prefix \cup ' \omega ' \cup SC \cup ' \varpi ' \cup tail_{sub}$ /* making composite strict closed pattern */
 14. for $h \leftarrow r$ to $r + u$
 15. insert cp into h -itemset Top-k list
 16. Else for each subset, $tail_{sub}$ in $tail$
 17. $p \leftarrow prefix \cup tail_{sub}$
 18. insert p into $|p|$ -itemset Top-k list /* $|p|$ means the length of p */
 19. Else /* multiple paths */
 20. for each $i_c \in \{i_{start}, i_{start-1}, \dots, i_1\}$ /* upper items */
 21. if ($\text{sup}(i_c) \geq \text{minsup}$)
 22. insert $prefix \cup i_c$ into N -itemset Top-k list
 23. search from nodes with i_c to the root
 24. generate a conditional pattern base(data base) for i_c
 25. $order \leftarrow$ sorting items in support descending order
 26. construct a conditional FP-tree for i_c
 27. if (new FP-tree $\neq \emptyset$)
 28. $prefix \leftarrow prefix \cup i_c$
 29. CRMN Growth($prefix$, new FP-tree, N -itemset Top-k list)
 30. remove i_c in the $prefix$
 31. for each $i_c \in \{i_{start+1}, \dots, i_n\}$ /* lower items */
 32. conduct the same operations as the lines 21 ~ 30
End procedure

도면10a



도면10b



도면10c

Procedure N-itemset Top-k list(pattern, support, N, |SC|, |prefix ∪ tail|)
N: size of N-itemset
SKC: short k-pattern counter
 Γ : N-itemset Top-k list (list has pattern and pattern's SKC)
|SC|: number of item in strict closed pattern |**prefix ∪ tail**|: number of item in prefix and tail
p: a pattern of N-itemset Top-k list
k: the expected number of top-k patterns
 1. $a \leftarrow |SC|$
 2. $b \leftarrow N - |prefix \cup tail|$
 3. $count \leftarrow_a C_b(aC_b$ is denoted by 'a' choose 'b')
 4. $\Gamma[N].pattern \leftarrow \Gamma[N].pattern \cup pattern$
 5. $\Gamma[N].SKC \leftarrow \Gamma[N].SKC \cup count$
 6. sort Γ in support descending order
 7. $\alpha \leftarrow 0, \beta \leftarrow 0$
 8. for each $p \in \Gamma[N]$
 9. $\alpha \leftarrow \alpha + SKC$ of p
 10. if($\alpha \geq k$)
 11. if($sup(p_\beta) > N$ -itemset's minsup // p_β is the β -th pattern in the N-itemset Top-k list
 12. N-itemset's minsup $\leftarrow sup(p_\beta)$
 13. minsup = min{2-itemset's minsup, ..., N-itemset's minsup}
 14. if($sup(p) < N$ -itemset's minsup) then remove p // remove infrequent top-k pattern
 15. $\beta++$
End procedure

도면11

Procedure Releasing N-itemset composite patterns(N-itemset Top-k list)

1. For each $n \leftarrow 1$ to N
2. For each $p \leftarrow$ a pattern of n-itemset Top-k list
3. For each item $c \in p$
4. if(c is included in items between ' ω ' and ' ϖ ' in p)
5. $SC \leftarrow SC \cup c$
6. else
7. $P_r \leftarrow P_r \cup c$
8. for each subset, SC_{sub} in SC
9. $L \leftarrow P_r \cup SC_{sub}$
10. if(size of $L = n$)
11. print(L)
12. count++
13. if(count > k)
14. break

End procedure