BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HCM

TẠ THẾ VINH

KHAI THÁC TẬP ĐƯỢC ĐÁNH TRỌNG PHỔ BIẾN TRÊN CƠ SỞ DỮ LIỆU TĂNG TRƯỞNG

LUẬN VĂN THẠC SĨ

Chuyên ngành: Công nghệ thông tin

Mã số ngành: 60480201

TP. HỒ CHÍ MINH, tháng 5 năm 2016

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HCM

TẠ THẾ VINH

KHAI THÁC TẬP ĐƯỢC ĐÁNH TRỌNG PHỔ BIẾN TRÊN CƠ SỞ DỮ LIỆU TĂNG TRƯỞNG

LUẬN VĂN THẠC SĨ

Chuyên ngành: Công nghệ thông tin

Mã số ngành: 60480201

CÁN BỘ HƯỚNG DẪN KHOA HỌC: PGS.TS. VÕ ĐÌNH BẢY

TP. HỒ CHÍ MINH, tháng 5 năm 2016

CÔNG TRÌNH ĐƯỢC HOÀN THÀNH TẠI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HCM

Cán bộ hướng dẫn khoa học : **PGS.TS. VÕ ĐÌNH BẢY**

Luận văn Thạc sĩ được bảo vệ tại Trường Đại học Công nghệ TP. HCM ngày 31 tháng 05 năm 2016

Thành phần Hội đồng đánh giá Luận văn Thạc sĩ gồm:

TT	Họ và tên	Chức danh Hội đồng
1	TS. Đặng Trường Sơn	Chủ tịch
2	TS. Vũ Thanh Hiền	Phản biện 1
3	TS. Lư Nhật Vinh	Phản biện 2
4	TS. Cao Tùng Anh	Ủy viên
5	TS. Nguyễn Thị Thúy Loan	Ủy viên, Thư ký

Xác nhận của Chủ tịch Hội đồng đánh giá Luận sau khi Luận văn đã được sửa chữa (nếu có).

Chủ tịch Hội đồng đánh giá LV

TRƯỜNG ĐH CÔNG NGHỆ TP. HCM
PHÒNG QLKH – ĐTSĐH

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – Tự do – Hạnh phúc

TP. HCM, ngày 20 *tháng 05 năm 2016*

NHIỆM VỤ LUẬN VĂN THẠC SĨ

Họ tên học viên: Tạ Thế Vinh — Giới tính: Nam

Ngày, tháng, năm sinh: 01/05/1982 — Nơi sinh: Bến Tre

Chuyên ngành: Công Nghệ Thông Tin — MSHV: 1241860028

I- Tên đề tài:

- Khai thác tập được đánh trọng phổ biến trên cơ sở dữ liệu tăng trưởng.

II- Nhiệm vụ và nội dung:

- Nghiên cứu bài toán khai thác tập được đánh trọng phổ biến
- Nghiên cứu bài toán khai thác tập được đánh trọng phổ biến trên cơ sở dữ liệu tăng trưởng.
- Kết hợp hai thuật toán để giải quyết vấn đề.
- Cài đặt ứng dụng minh họa.

III- Ngày giao nhiệm vụ: 20/01/2016 .....

IV- Ngày hoàn thành nhiệm vụ: 14/05/2016 .....

V- Cán bộ hướng dẫn: PGS.TS. Võ Đình Bảy .....

.....

.....

CÁN BỘ HƯỚNG DẪN — **KHOA QUẢN LÝ CHUYÊN NGÀNH**

PGS.TS. Võ Đình Bảy

LỜI CAM ĐOAN

Tôi xin cam đoan đây là công trình nghiên cứu của riêng tôi. Các số liệu, kết quả nêu trong Luận văn là trung thực và chưa từng được ai công bố trong bất kỳ công trình nào khác.

Tôi xin cam đoan rằng mọi sự giúp đỡ cho việc thực hiện Luận văn này đã được cảm ơn và các thông tin trích dẫn trong Luận văn đã được chỉ rõ nguồn gốc.

Học viên thực hiện Luận văn

Tạ Thế Vinh

LỜI CẢM ƠN

Để hoàn thành luận văn này, em đã được sự giúp đỡ tận tình của các thầy cô giáo cùng sự giúp đỡ của các bạn đồng nghiệp, gia đình và bạn bè. Nhân dịp này em xin bài tỏ lòng biết ơn tới:

Các thầy cô trong khoa đào tạo sau đại học Trường Đại Học Công Nghệ Thành Phố Hồ Chí Minh đã truyền đạt cho em những kinh nghiệm quý báu. Bên cạnh đó, em cũng xin gửi lời cảm ơn đến Ban giám hiệu, Ban chủ nhiệm và các ban ngành Trường Đại Học Công Nghệ Thành Phố Hồ Chí Minh đã tạo mọi điền kiện thuận lợi cho em trong quá trình học tập và hoàn thiện luận văn.

Em xin chân thành cảm ơn thầy **PGS.TS.Võ Đình Bảy**, người đã tận tình hướng dẫn em từng bước trong suốt quá trình thực hiện đề tài này. Trong quá trình làm luận văn thầy đã tận tình hướng dẫn giúp em giải quyết các vấn đề và hoàn thành luận văn.

Xin chân thành cảm ơn các thầy cô trong hội đồng chấm luận văn đã cho em những đóng góp quý báu để luận văn thêm hoàn chỉnh.

Xin chân thành cảm ơn các thầy cô giáo, anh chị đồng nghiệp, bạn bè và gia đình đã giúp em hoàn thành luận văn này.

Em xin chân thành cảm ơn!

TP. Hồ Chí Minh, tháng 5 năm 2016

Người thực hiện

Tạ Thế Vinh

TÓM TẮT

Dữ liệu là tài sản quý giá của doanh nghiệp, nó không phải dữ liệu thông thường mà nó ẩn chứa nhiều thông tin rất có giá trị cho doanh nghiệp, đặc biệt là cơ sở dữ liệu bán hàng trong suốt quá trình hoạt động của doanh nghiệp, nếu chúng ta khai thác đúng cách, sẽ khám phá được những tri thức hữu ích cho doanh nghiệp, từ đó giúp doanh nghiệp định hướng phát triển đúng đắn.

Để khám phá được những thông tin có giá trị trong cơ sở dữ liệu, khai thác luật kết hợp là một trong những phương pháp phổ biến nhất để đạt được mục đích này. Trong đó khai thác tập phổ biến đóng một vai trò quan trọng trong khai thác luật kết hợp. Tập phổ biến thường được khai thác từ các cơ sở dữ liệu nhị phân.

Tuy nhiên, cơ sở dữ liệu nhị phân chỉ quan tâm đến vấn đề khách hàng có mua hay không mua sản phẩm nào đó. Nhưng trên thực tế, mỗi một sản phẩm mà khách hàng mua lại có thể có giá trị khác nhau. Tương tự mỗi một hạng mục trong giao dịch cũng có các trọng số khác nhau tùy theo từng loại cơ sở dữ liệu cụ thể.

Khai thác các tập được đánh trọng phổ biến trên các cơ sở dữ liệu tăng trưởng hiện nay vẫn chưa được phát triển. Vì vậy, việc nghiên cứu các kỹ thuật để khai thác các cơ sở dữ liệu này mang tính thực tiễn rất cao.

Luận văn nghiên cứu về các thuật toán khai thác các tập được đánh trọng phổ biến như thuật toán Apriori, WIT-FWIs, WIT-FWIs-MDIFY, WIT-FWIs-DIFF, dựa vào đó làm nền tảng để tiến hành nghiên cứu bài toán khai thác các tập phổ biến được đánh trọng trên CSDL tăng trưởng, và đề nghị ứng dụng khái niệm pre-large vào khai thác tập được đánh trọng phổ biến trên cơ sở dữ liệu tăng trưởng nhằm hạn chế phải quét lại cơ sở dữ liệu ban đầu khi dữ liệu tăng trưởng, từ đó đề xuất thuật toán INCREMENTAL_WIT_FWI trong khai thác tập phổ biến được đánh trọng số trên dữ liệu tăng trưởng.

ABSTRACT

The data are valuable assets of the business, it's not the usual data but that it hides a lot of information is valuable for business, especially sales database during the operation of the business, if we exploit properly, will discover useful knowledge for the enterprise, helping businesses develop proper orientation.

To discover valuable information in the database, association rules mining is one of the most well know methods to achieve this purpose. In this, frequent itemsets mining play an important part in associative rules mining. Frequent itemsets often mined from the binary database.

However, the binary database are only interested in customers buy or not buy a product. In fact, every product that customer buy can have different values. Similarly, each item of transactions also have different weights depending on the type of specific databases.

Frequent itemsets mining is considered common practice on the basis of current growth data have yet to be developed. Thus the techniques research to mining the database have highly practical.

This thesis is devoted to Frequent itemsets mining algorithm is considered common practice on the basis as Apriori algorithm, WIT-FWIs, WIT-FWIs-MDIFY, WIT-FWIs-DIFF, thank to that is foundation to conduct research Frequent itemsets mining, and proposed the concept of pre-large to exploit large weight is considered common practice in the database to limit growth to scan the initial database when the data is grow, since, the proposed algorithm INCREMENTAL_WIT_FWI in incremental data mining.

MỤC LỤC

DANH MỤC CÁC TỪ VIẾT TẮT

Viết tắt	Tiếng việt	Tiếng Anh
CNTT	Công nghệ thông tin	Information Technology
CSDL	Cơ sở dữ liệu	Database
tw	Trọng số giao dịch	Transaction weight
FWI	Tập phổ biến được đánh trọng số	Frequent weighted itemsets
minsup	Hỗ trợ tối thiểu	Minimum support
WIT	Weighted Itemset-Tidset	Weighted Itemset-Tidset
ws	Trọng số hỗ trợ	Weighted support
AR	Luật kết hợp	Association Rule
WAR	Luật kết hợp có trọng số	Weighted Association Rule

DANH MỤC CÁC BẢNG

Trang

DANH MỤC CÁC BIỂU ĐỒ, HÌNH ẢNH

Trang

PHẦN MỞ ĐẦU

1. Đặt vấn đề

Khai thác dữ liệu đang là một trong các lĩnh vực được rất nhiều nhà nghiên cứu quan tâm, đặc biệt là các nhà quản lý rất cần biết được mối quan hệ của các sản phẩm trong số lượng lớn các sản phẩm, từ đó có thể giúp doanh nghiệp kinh doanh hiệu quả hơn và làm tăng doanh thu cho doanh nghiệp. Khai thác dữ liệu là quá trình phân tích và tìm ra các thông tin có giá trị được ẩn chứa trong cơ sở dữ liệu (CSDL).

Khai thác luật kết hợp là một trong những phương pháp phổ biến nhất mà các nhà nghiên cứu thường hay dùng. Mục đính của việc khai thác luật kết hợp nhằm tìm ra các mối quan hệ giữa các tập trong cơ sở dữ liệu, trong đó khai thác tập phổ biến đóng vai trò quan trọng trong khai thác luật kết hợp. Khai thác tập phổ biến thường được khai thác từ CSDL nhị phân, trong đó từng mục trong giao dịch có thể mang nhiều ý nghĩa khác nhau.

Tuy nhiên, khai thác CSDL nhị phân người ta thường chỉ quan tâm đến số lượng bán ra của một sản phẩm nào đó, chưa quan tâm nhiều đến giá trị và lợi ích của các sản phẩm được bán ra. Gần đây, khai thác tập được đánh trọng phổ biến trên CSDL có các item được đánh trọng được quan tâm, tuy nhiên chưa có công trình nào xem xét việc khai thác tập được đánh trọng trên CSDL tăng trưởng. Luận văn tập trung tìm hiểu bài toán khai thác tập được đánh trọng từ đó phát triển thuật toán khai thác tập được đánh trọng trên CSDL tăng trưởng.

2. Mục tiêu nghiên cứu

Mục tiêu chung

- Nghiên cứu các thuật toán để khai thác tập phổ biến được đánh trọng số trên cơ sở dữ liệu tăng trưởng, nhằm giảm quét lại cơ sở dữ liệu ban đầu khi CSDL tăng trưởng.

Mục tiêu cụ thể

- Nghiên cứu các thuật toán để khai thác tập được đánh trọng phổ biến.
- Nghiên cứu khai thác tập phổ biến trên CSDL tăng trưởng.
- Ứng dụng các thuật toán nghiên cứu vào khai thác tập được đánh trọng phổ biến trên cơ sở dữ liệu tăng trưởng.

3. Đối tượng và phạm vi nghiên cứu

Đối tượng nghiên cứu:

- Các thuật toán khai thác luật kết hợp.
- Các thật toán khai thác tập phổ biến được đánh trọng số.
- Các thuật toán khai khác tập phổ biến trên CSDL tăng trưởng.
- Các CSDL lớn thường xuyên thay đổi (Giới hạn trong trường hợp thêm dữ liệu)

Phạm vi nghiên cứu:

- Luận văn tập trung vào nghiên cứu các thuật toán để khai thác tập phổ biến được đánh trọng và nghiên cứu khai thác tập phổ biến trên CSDL tăng trưởng.

4. Ý nghĩa khoa học và thực tiễn của đề tài

Ý nghĩa khoa học:

- Đề tài tìm kiếm giải pháp cập nhật lại các tri thức đã được khai thác trước đó khi CSDL được thêm vào nhằm làm giảm không gian tìm kiếm và thời gian khai thác.

Ý nghĩa thực tiễn:

- Tiết kiệm nhiều thời gian trong quá trình khai thác luật kết hợp.
- Là phương pháp hiệu quả trong khai thác dữ liệu tăng trưởng.

5. Cấu trúc của luận văn

Luận văn gồm có 03 chương, phần mở đầu và phần kết luận.

Phần mở đầu: Giới thiệu về luận văn gồm các mục: Đặt vấn đề, mục tiêu nghiên cứu, đối tượng và phạm vi nghiên cứu, ý nghĩa khoa học và thực tiễn của đề tài.

Chương 1: Tổng quan lĩnh vực nghiên cứu, các khái niệm, định nghĩa, cơ sở khoa học, các công trình nghiên cứu có liên quan, các phương pháp nghiên cứu và nhận xét ưu khuyết điểm của các phương pháp.

Chương 2: Ứng dụng khái niệm pre-large vào khai thác tập được đánh trọng phổ biến trên cơ sở dữ liệu tăng trưởng .

Chương 3: Trình bày về thực nghiệm bao gồm môi trường thực nghiệm, cơ sở dữ liệu thực nghiêm, đánh giá các kết quả thu được.

Phần kết luận: Trình bày các kết quả đạt được của luận văn, nhận xét ưu khuyết điểm và hướng phát triển của đề tài.

Tổng cộng có 56 trang, trong đó có 30 hình vẽ, 12 bảng số.

CHƯƠNG 1: TỔNG QUAN LĨNH VỰC NGHIÊN CỨU VÀ CƠ SỞ LÝ THUYẾT

1.1 Các khái niệm và định nghĩa

Khai thác luật kết hợp là một yếu tố quan trọng trong lĩnh vực khám phá tri thức từ kho dữ liệu (KDD) [3] được lưu trữ trong suốt quá trình hoạt động của doanh nghiệp hoặc một tổ chức nào đó. Khai thác luật kết hợp là để xác định mối quan hệ giữa các tập mục trong cơ sở dữ liệu (CSDL) giao dịch. Quá trình khai thác thường sử dụng nhiều kỹ thuật khác nhau để tìm ra mối quan hệ giữa các tập mục và phát hiện ra tri thức ẩn chứa bên trong cơ sở dữ liệu.

Tuy nhiên, các phương pháp khai khác trước đây thường chỉ quan tâm đến một sản phẩm hay một mặt hàng nào đó có được bán hay là không và thường không quan tâm vào việc xem xét các lợi ích hoặc tầm quan trọng của sản phẩm đó. Trên thực tế thì mỗi mặt hàng hay sản phẩm nào đó có giá khác nhau và lợi nhuận có được khi bán một sản phẩm nào đó cũng khác nhau. Vì vậy, việc khai thác tập phổ biến được đánh trọng số mang tính thực tiễn cao.

Năm 1998, Ramkumar, Ranka và Tsur [7] đã đề xuất một mô hình để mô tả các khái niệm về việc khai thác luật kết hợp có trọng số và dựa trên giải thuật Apriori để tìm ra các tập phổ biến được đánh trọng số. Từ đó, nhiều kỹ thuật khai thác luật kết hợp có trọng số được đề xuất như: Tao, Murtagh, và Farid [8], Vo, Frans, Le [3].

1.2 Tổng quan về khai thác luật kết hợp

Bài toán tìm luật kết hợp và là bài toán cơ bản trong khai thác dữ liệu [1] gồm hai bước chính là: bước một, tìm ra tất cả các tập phổ biến theo ngưỡng hỗ trợ S_0 cho trước và bước hai là tìm ra các luật kết hợp dựa vào các tập phổ biến đã được tìm thấy.

Trong lĩnh vực khai thác dữ liệu, mục tiêu cuối cùng là tìm ra được các mối quan hệ tiềm ẩn giữa các đối tượng trong cơ sở dữ liệu, để đạt được mục tiêu, khai thác luật kết

hợp là vấn đề cần được tìm hiểu, nội dung cơ bản của luật kết hợp (Association Rule – AR) được mô tả như sau:

Một cơ sở dữ liệu giao dịch (D) được định nghĩa như sau: D là bao gồm một tập hợp các giao dịch $T = \{t_1, t_2, \ldots, t_m\}$ và $I = \{i_1, i_2, \ldots, i_n\}$ là một tập các *item.* Mỗi tập con trong *I* được gọi là một itemset, số lượng của các phần tử trong một itemset được gọi là kích thước của một itemset.

Cho X, Y là các itemset, trong đó X và Y là hai tập không giao nhau khác rỗng. Một luật kết hợp được ký hiệu là X →Y, điều này thể hiện mối quan hệ ràng buộc của tập Y với tập X được hiểu là sự xuất hiện của tập X sẽ kéo theo sự xuất hiện của tập Y trong các giao dịch, một cách dễ hiểu là những người mua các mặt hàng trong tập X cũng thường mua các mặt hàng trong tập Y.

Ví dụ, nếu X = {Bánh Mì, Trứng} và Y = {Sữa, Thịt} và ta có luật kết hợp X→Y thì chúng ta có thể nói rằng những người mua Bánh Mì và Trứng thì cũng thường mua Sữa và Thịt.

Tập X được gọi là xuất hiện trong giao dịch t nếu như nó là tập con của t. Độ hỗ trợ và độ tin cậy là hai tham số được dùng để đo lường luật kết hợp. Thuật toán phổ biến nhất để tìm các luật kết hợp là Apriori.

Định nghĩa 1.1: Độ hỗ trợ (Support)

Độ hỗ trợ của luật kết hợp X →Y là tần suất giao dịch có chứa tất cả các item trong cả hai tập X và Y.

Ví dụ: Độ hỗ trợ của luật kết hợp X →Y là 50%, có nghĩa là 50% các giao dịch X và Y được mua cùng nhau.

Công thức tính độ hỗ trợ của luật (X→Y):

$$\text{Support}(X \rightarrow Y) = P(X \cup Y) = \frac{n(X \cup Y)}{N}$$

Trong đó $n(X \cup Y)$ là số giao dịch có chứa X và Y, N là tổng số giao dịch có trong cơ sở dữ liệu.

Định nghĩa 1.2: Độ tin cậy (Confidence)

Độ tin cậy là xác suất xảy ra Y khi đã biết X.

Ví dụ: Độ tin cậy của {Bánh Mì}→{Sữa} là 70% có nghĩa là 70% khách hàng mua Bánh Mì cũng mua Sữa.

Công thức tính độ tinh cậy của luật (X→Y):

$$\text{Confidence}(X \rightarrow Y) = P(X \mid Y) = \frac{n(X \cup Y)}{n(X)}$$

Trong đó n(X) là số giao dịch có chứa X

Để thu được luật kết hợp người ta thường áp dụng 2 tiêu chí đó là minsup và minconf, trong đó minsup là độ hỗ trợ tối thiểu và minconf là độ tin cậy tối thiểu, là hai giá trị ngưỡng tối thiểu cho trước.

Luật kết hợp $X \rightarrow Y$ được coi là luật kết hợp có giá trị khi thỏa 2 tiêu chí Support $(X \rightarrow Y) \geq$ minsup và Confidence $(X \rightarrow Y) \geq$ minconf

Tập X được gọi là tập phổ biến khi có độ hỗ trợ lớn hơn hay bằng minsup

Định nghĩa 1.3: Lớp tương đương

Lớp tương đương là tập hợp tất cả các itemset có cùng tiền tố X gọi là lớp tương đương, và được ký hiệu là [X].

Cho $X \subseteq I$, ta định nghĩa hàm $p(X,k) = X[1,k]$ gồm k phần tử đầu của X và quan hệ tương đương dựa vào tiền tố sau:

$$\forall X,Y \subseteq I, X \equiv \emptyset_k, Y \leftrightarrow p(X,k) = p(Y,k)$$

Kết nối Galois

Cho $\delta \subseteq I \times T$ có mối quan hệ nhị phân, trong đó I là tập các item còn T là tập các giao dịch chứa trong CSDL cần khai thác. Cho P(S) (tập tất cả các tập con của S) bao gồm các tập con của S. Hai ánh xạ giữa P(I) và P(T) được gọi là kết nối Galois [3] .

Cho $X \subseteq I$ và $Y \subseteq T$, ta có:

I. $t:P(I) \rightarrow P(T)$, $t(X) = \{y \in T \mid \forall x \in X, x\delta y\}$

II. $i:P(T) \rightarrow P(I)$, $i(Y) = \{y \in I \mid \forall y \in Y, x\delta y\}$

Ánh xạ t(X) là tập các giao dịch trong CSDL có chứa X, và ánh xạ i(Y) là một itemset có chứa trong tất cả các giao dịch Y.

Cho X, X1, X2 ∈ P(I) và Y, Y1, Y2 ∈ P(T). Kết nối Galois thỏa mãn các tính chất sau (Zaki, 2004):

i. $X_1 \subset X2 \Rightarrow t(X_1) \supseteq t(X_2)$

ii. $Y_1 \subset Y_2 \Rightarrow i(Y_1) \supseteq i(Y_2)$

iii. $X \subseteq i(t(X))$ và $Y \subseteq t(i(Y))$

1.3 Thuật toán Apriori

Khai thác luật kết hợp là một trong những phương pháp khám phá tri thức từ CSDL và được nhiều nhà nghiên cứu quan tâm và phát triển, Apriori là thuật toán nổi tiếng được tác giả Agrawal cùng các đồng sự đề xuất năm 1994 [12], lúc đầu nó được ứng dụng vào việc khai thác luật kết hợp trong lĩnh vực thương mại, về sau nó được ứng dụng rộng rãi trong các lĩnh vực khác như trong quản lý, y khoa, công nghiệp, v.v.

Ý tưởng chính của thuật toán Apriori là:

Tìm tất cả các tập phổ biến trong cơ sở dữ liệu: k-itemset là tập ứng viên gồm k phần tử, được dùng để tìm k+1 itemset.

Bước đầu tìm tập ứng viên có 1 phần tử được gọi là 1-itemset (được ký hiệu là L_1), tập L_1 được dùng để tìm tập 2-itemset (L_2), L_2 được dùng để tìm tập 3-itemset (L_3) và cứ thế tiếp tục tìm đến khi không có k-itemset nào được tìm thấy.

Từ các tập phổ biến sinh ra các luật kết hợp mạnh, các luật kết hợp phải thỏa minsup và minconf.

Tính chất 1: Mọi tập con của tập phổ biến đều phổ biến, điều này có nghĩa là

$\forall X \subseteq Y$, nếu $Sup(Y) \geq minsup$ thì $Sup(X) \geq minsup$

Tính chất 2: Mọi tập cha của tập không phổ biến đều không phổ biến,điều này có nghĩa là $\forall Y \supseteq X$ nếu $Sup(X) < minsup$ thì $Sup(Y) < minsup$

Mô tả thuật toán:

Bước 1:Tính độ hỗ trợ của mỗi item có kích thước là 1, sau đó lọc ra các item thỏa mãn độ hỗ trợ tối thiểu (minsup) và đặt các item thỏa minsup vừa lọc ra là tập L_1:1-itemset, và chọn tập L_1 là tập nguồn.

Bước 2: Từ tập nguồn L_1 phát sinh ra tập ứng viên có kích thước là 2 được ký hiệu là (C) và tính độ hỗ trợ cho tập ứng viên (C) sau đó lọc ra các tập phổ thỏa minsup và đặt chúng là L_2: 2-itemset và chọn là tập nguồn cho bước tiếp theo.

Bước 3: Lặp lại bước 2 để tìm ra các tập ứng viên có kích thước là (k+1) – itemset, quá trình này sẽ lặp đi lặp lại cho đến khi không tìm ra được tập phổ biến thỏa minsup thì dừng lại.

Thuật toán Apriori:

Đầu vào: Tập các giao dịch D, ngưỡng hỗ trợ tối thiểu minsup

Đầu ra: L các tập phổ biến có trong D

Phương thức:

```
{
Gọi Ck là tập các ứng viên có kích thước là k
Gọi Lk là các tập phổ biến có kích thước là k
L1 là các tập phổ biến có kích thước 1 phần tử thỏa minsup
for (k=2; Lk-1≠ ∅; k++)
{
    Ck = apriori-gen(Lk-1); // New candidates
    forall transactions  t ∈ D
    {
        Ct= subset (Ck, t); // Candidates contained in t
        forall candidate c ∈ Ct
        {
            c.count++;
        }
```

$L_k=\{c \in C_k \mid c.count \geq minsup\}$

}

}

Answer = $U_k L_k$

}

Thuật toán Apriori sử dụng độ hỗ trợ tối thiểu (minsup) để loại bỏ ứng viên không phù hợp. Giá trị minsup do người dùng đưa ra.

Hàm Apriori-gen() có tác dụng sinh ra các tập itemset có kích thước k+1 phần tử từ tập nguồn có kích thước là k trong tập L_k, các itemset được tạo ra bằng cách nối các tập item có cùng tiền tố và áp dụng tính chất 1 để loại bỏ các tập không thỏa.

Bước nối: Bước nối có tác dụng sinh ra các tập L_{k+1}là ứng viên của tập phổ biến có kích thước K+1 bằng cách kết hợp các ứng viên có cùng tiền tố lại với nhau

Bước tỉa: Giữ lại tất cả các ứng viên L_{k+1} thoản mãn tính chất Apriori (tính chất 1) có nghĩa là mọi tập con của tập phổ biến đều phổ biến.

Mô tả giải thuật Apriori:

Ta có cơ sở dữ liệu D gồm 6 giao dịch với các tập item sau:

Bảng 1.1: Cơ sở dữ liệu giao dịch D

Transaction	Item
1	A, B, D, E
2	B, C, E
3	A, B, D, E
4	A, B, C, E
5	A, B, C, D, E
6	B, C, D

Ngưỡng hỗ trợ minsup 0.4

Áp dụng giải thuật Apriori cho CSDL giao dịch D

Bước 1: Thuật toán sẽ quét CSDL D và xác định độ hỗ trợ cho các tập phổ biến 1 itemset. Các tập item nào có độ hỗ trợ nhỏ hơn 40% sẽ bị loại bỏ, chỉ giữ lại các item thỏa ngưỡng minsup, như vậy sau lần quét đầu tiên ta được các tập phổ biến thỏa minsup là:

L_1= {{A},{B},{C},{D},{E}}

Bảng 1.2: Apriori 1-itemset thỏa minsup

Transaction	Item
1	A, B, D, E
2	B, C, E
3	A, B, D, E
4	A, B, C, E
5	A, B, C, D, E
6	B, C, D

→

1-Itemset	Support
{A}	4
{B}	6
{C}	4
{D}	4
{E}	5

Bước 2: Từ các tập phổ biến 1 itemset thỏa minsup vừa tìm được ở bước 1, tạo ra các tập phổ biến có kích thước là 2, quét cơ sở dữ liệu D tìm tập phổ biến có độ hỗ trợ thỏa ngưỡng minsup và loại bỏ các tập không thỏa. Sau bước 2 ta được:

Bảng 1.3: Apriori 2-itemset thỏa minsup

1-Itemset	Support
{A}	4
{B}	6
{C}	4
{D}	4
{E}	5

→

2-Itemset	Support
{A,B}	4
{A,C}	2
{A,D}	3
{A,E}	4
{B,C}	4
{B,D}	4
{B,E}	5
{C,D}	2
{C,E}	3
{D,E}	3

→

2-Itemse	Support
{A,B}	4
{A,D}	3
{A,E}	4
{B,C}	4
{B,D}	4
{B,E}	5
{C,E}	3
{D,E}	3

Bước 3: Từ các tập phổ biến 2 itemset thỏa minsup vừa tìm được ở bước 2, tạo ra các tập phổ biến có kích thước là 3, quét cơ sở dữ liệu D tìm tập phổ biến có độ hỗ trợ thỏa ngưỡng minsup và loại bỏ các tập không thỏa. Sau bước 3 ta được:

Bảng 1.4: Apriori 3-itemset thỏa minsup

2-Itemset	Support
{A,B}	4
{A,D}	3
{A,E}	4
{B,C}	4
{B,D}	4
{B,E}	5
{C,E}	3
{D,E}	3

→

3-Itemset	Support
{A,B,C}	2
{A,B,D}	3
{A,B,E}	4
{A,D,E}	3
{B,C,D}	2
{B,C,E}	3
{B,D,E}	3

→

3-Itemset	Support
{A,B,D}	3
{A,B,E}	4
{A,D,E}	3
{B,C,E}	3
{B,D,E}	3

Bước 4: Từ các tập phổ biến 3 itemset thỏa minsup vừa tìm được ở bước 3, tạo ra các tập phổ biến có kích thước là 4, quét cơ sở dữ liệu D tìm tập phổ biến có độ hỗ trợ thỏa ngưỡng minsup và loại bỏ các tập không thỏa. Sau bước 4 ta được:

Bảng 1.5: Apriori 4–itemset thỏa minsup

3-Itemset	Support
{A,B,D}	3
{A,B,E}	4
{A,D,E}	3
{B,C,E}	3
{B,D,E}	3

→

4-Itemset	Support
{A,B,D,E}	3

Bước 5: Tổng hợp tất cả các tập phổ biến thỏa ngưỡng hỗ trợ minsup = 40%, sau khi kết thúc thuật toán ta tìm được các tập phổ biến:

{{A},{B},{C},{D},{E},{A,B},{A,D},{A,E},{B,C},{B,D},{B,E},{C,E},{D,E}, {A,B,D}, {A,B,E}, {A,D,E}, {B,C,E}, {B,D,E}, {A,B,D,E}}

1.4 Thuật toán Eclat

Eclat là thuật toán được Zaki đề xuất năm 1997 [11], cách tiếp cận IT-tree là dựa vào phần giao nhau của tập các giao dịch trên CSDL giao dịch để tính độ phổ biến và khái niệm lớp tương đương chia không gian xử lý ban đầu thành tập các không gian nhỏ hỗ trợ tìm kiếm nhanh hơn, ngoài ra phương pháp IT-tree còn giúp tính độ phổ biến nhanh hơn bằng cách dựa trên phần khác nhau trên Tidset và làm giảm kích thước bộ nhớ.

Ứng dụng IT-tree khai thác tập phổ biến

Mô tả giải thuật Eclat

Bước 1: Đầu tiên thuật toán sẽ quét CSDL và khởi tạo lớp tương đương rỗng có tiền tố là {} hoặc [∅] chứa tất cả các tập phổ biến có kích thước là 1 thỏa điều kiện lớn hơn hoặc bằng minsup

Bước 2: Gọi hàm ENUMERATE_FREQUENTvới đầu vào là lớp tương đương với tiền tố {}. Thủ tục này sẽ xét mỗi nút $l_i \in [P]$ với $l_j \in [P]$ đứng sau nó, với mỗi cặp (l_i, l_j), thủ tục này sẽ tính $Y=t(l_i \cup l_j) = t(l_i \cap l_j)$, nếu $|Y| \geq$ minsup nghĩa là độ hỗ trợ của $l_i \cup l_j$ thỏa minsup thì thêm nút $X \times Y$ vào lớp tương đương $[P_i]$.

Bước 3: Tiếp tục lặp lại bước 2 cho đến khi không tìm được lớp tương đương nào thỏa minsup

Mô tả giải thuật Eclat

Đầu vào: Cơ sở dữ liệu D và ngưỡng hỗ trợ tối thiểu minsup

Đầu ra: IT-tree chứa tất cả các tập phổ biến từ CSDL D

Phương thức:

Eclat()

1. $[\emptyset] = \{i \in I \wedge \sigma(i) \geq \text{minsup}\}$

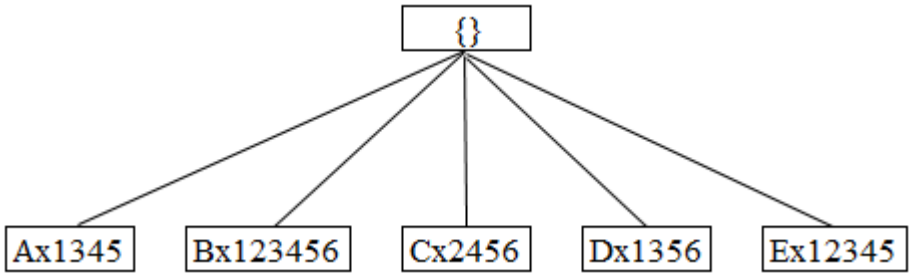
2. **ENUMERATE_FREQUENT**([∅])

ENUMERATE_FREQUENT([P])
3. For all $l_i \in [P]$ do
4. $[P_i]=\{\emptyset\}$
5. For all $l_j \in [P]$, with $j>i$ do
6. $X=l_i \cup l_j$
7. $Y= t(l_i) \cap t(l_j)$
8. If $[Y] \geq$ minsup then
9. $[P_i]=[P_i] \cup \{X \times Y\}$
10. **ENUMERATE_FREQUENT**($[P_i]$)

Đầu tiên, thuật toán khởi tạo lớp tương đương rỗng [∅] chứa toàn bộ các nút có kích thước là 1 gọi là 1-itemsets, và chúng đều thỏa điều kiện ngưỡng minsup. Tất cả các nút ở mức 1 sẽ trở thành một lớp tương đương với tiền tố là [∅] (dòng 1). Sau đó hàm **ENUMERATE_FREQUENT** với biến đầu vào là lớp tương đương rỗng sẽ được thực thi. Thủ tục **ENUMERATE_FREQUENT** sẽ xét mỗi nút $l_i \in [P]$ (dòng 3) với nút $l_j \in [P]$ (dòng 5) đứng sau nó, với mỗi cặp (l_i,l_j), thủ tục này sẽ tính $Y=t(l_i \cup l_j)=t(l_i) \cap t(l_j)$ (dòng 7), nếu $|Y| \geq$ minsup nghĩa là độ hỗ trợ theo số đếm của li ∪ lj thỏa minsup thì thêm nút $X \times Y$ vào lớp tương đương $[P_i]$ (được khởi tạo rỗng ở dòng 4). Sau đó gọi đệ quy thủ tục **ENUMERATE_FREQUENT** để sinh ra các lớp tương đương con cho đến khi không còn lớp tương đương nào được tạo ra (dòng 10)

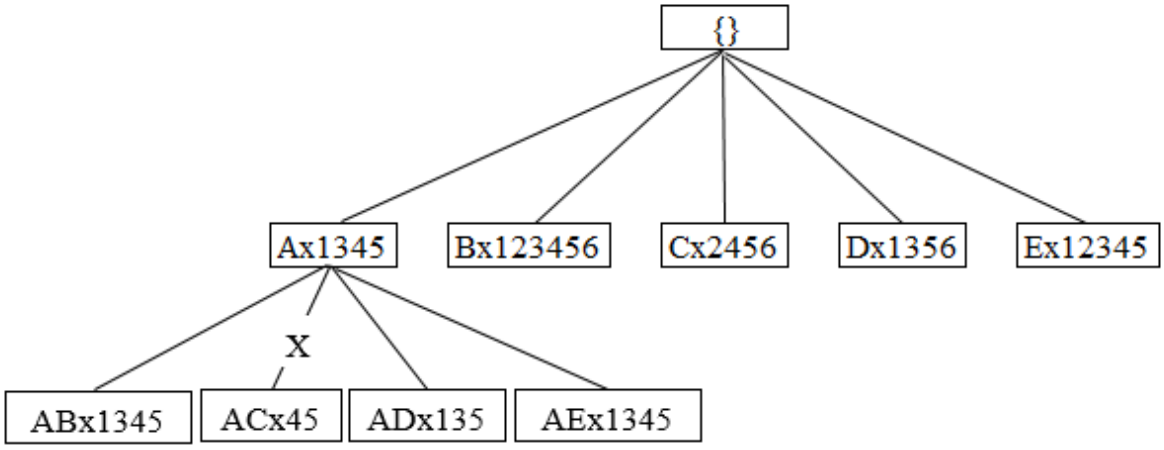
Sử dụng CSDL D (bảng 1.1) để tiến hành khai thác các tập phổ biến có độ hỗ trợ thỏa minsup = 0.4.

Bước 1: Khởi tạo lớp[∅] chứa tất cả các tập phổ biến có kích thước là 1



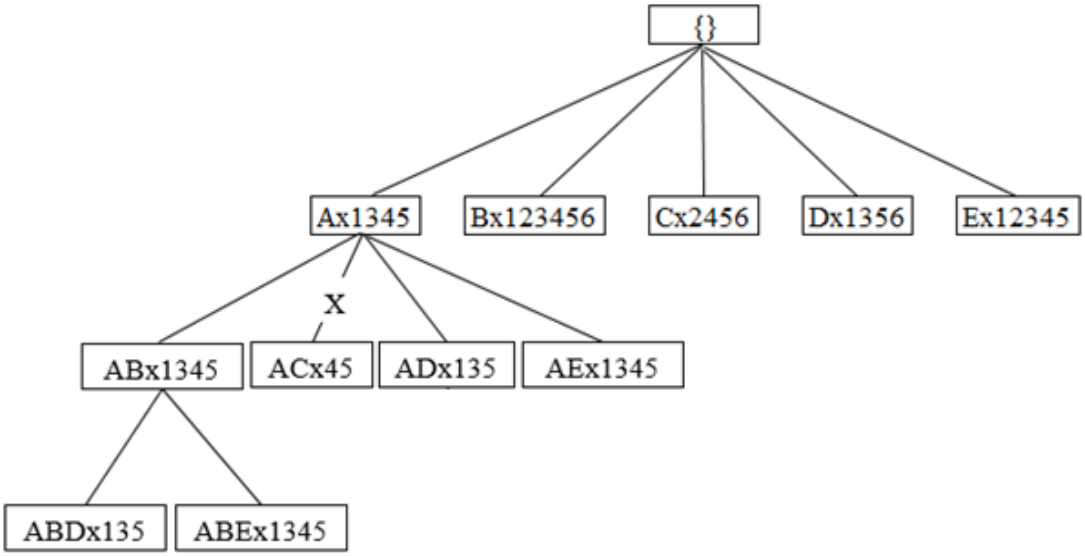
Hình 1.1: Khởi tạo lớp tương đương rỗng trên cây IT-tree

Bước 2: Với mỗi nút ở mức 1, ta tiến hành khởi tạo các lớp tương đương của các tập phổ biến có kích thước là 2 và loại bỏ các tập không thỏa điều kiện minsup.



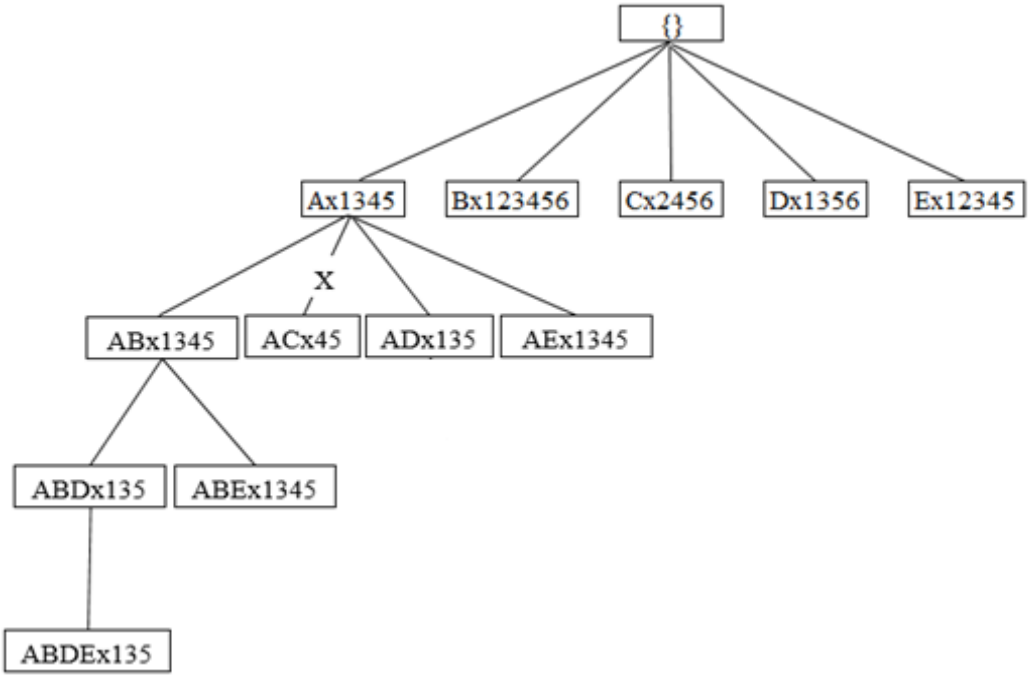
Hình 1.2: Cây IT-tree với lớp tương đương ở mức 2

Bước 3: Tiến hành lặp lại bước 2, khởi tạo các lớp tương đương cho các tập phổ biến có kích thước là 3, với tiền tố là từ các tập ở lớp tương đương mức 2, loại bỏ các tập không thỏa điều kiện minsup.



Hình 1.3: Cây IT-tree với lớp tương đương ở mức 3

Bước 4: Tạo ra các tập phổ biến có kích thước là 4, từ các tập ở mức 3, và loại bỏ các tập không thỏa điều kiện minsup.



Hình 1.4: Cây IT-tree với lớp tương đương ở mức 4

Bước 5: Thuật toán tiếp tục tính ở các nút còn lại cho đến khi không thể sinh ra được các tập thỏa điều kiện minsup thì thuật toán dừng lại. Kết thúc giải thuật ta thu được tập F gồm các tập phổ biến thỏa điều kiện minsup.

F={{A},{B},{C},{D},{E},{AB},{AD},{AE},{BC},{BD},{BE},{CE},{DE},{ABD},{ABE},{ADE},{BCD},{BCE},{BDE},{ABDE}}

1.5 Định nghĩa và tính chất của tập được đánh trọng số

Cơ sở dữ liệu giao dịch của tập được đánh trọng số bao gồm một tập hợp các giao dịch $T=\{t_1,t_2,\ldots,t_m\}$; một tập các item $I=\{i_1,i_2,\ldots,i_n\}$ và tập hợp các trọng số $W=\{w_1,w_2,\ldots,w_n\}$tương ứng với từng item trong I.

Ví dụ 1.1: Xét cơ sở dữ liệu được trình bày ở bảng 1.6 và bảng 1.7, ở bảng 1.6 trình bày cơ sở dữ liệu giao dịch gồm có 6 giao dịch $T=\{t_1,t_2,t_3,t_4,t_5,t_6\}$ và 6 item I={A,B,C,D,E}. Trọng số của các item được trình bày trong bảng 1.7, W={0.6,0.1,0.3,0.9,0.2}

Bảng 1.6: Cơ sở dữ liệu giao dịch

Transaction	Item
1	A, B, D, E
2	B, C, E
3	A, B, D, E
4	A, B, C, E
5	A, B, C, D, E
6	B, C, D

Bảng 1.7: Trọng số giao dịch của từng item

Item	Weight
A	0.6
B	0.1
C	0.3
D	0.9
E	0.2

Bảng 1.8: Trọng số giao dịch của từng giao dịch ở bảng 1.6

Transaction	tw
1	0.45
2	0.2
3	0.45
4	0.3
5	0.42
6	0.43
Sum	2.25

1.6 Khai thác tập phổ biến được đánh trọng số

Để khai thác tập phổ biến được đánh trọng số dựa vào hai thông số là trọng số giao dịch của một giao dịch được ký hiệu là tw và trọng số hỗ trợ của một itemset được ký hiệu là ws.

Định nghĩa 1.1: Trọng số giao dịch (tw) của một giao dịch t_k được định nghĩa

$$tw(t_k)= \frac{\sum_{i_j \in t_k} w_j}{|t_k|}$$

Định nghĩa 1.2: Trọng số hỗ trợ của một itemset được định nghĩa như sau

$$ws(X)= \frac{\sum_{t_k \in t(X)} tw(t_k)}{\sum_{t_k \in T} tw(t_k)}$$

Trong đó, T là danh sách các giao dịch trong cơ sở dữ liệu

Ví dụ 1.2: Dựa vào bảng 1.6, bảng 1.7 và định nghĩa 1.1 chúng ta có thể tính được trọng số giao dịch của giao dịch t_1được ký hiệu $tw(t_1)$:

Ta có T1={A,B,D,E} tương ứng với W_A= 0.6,$W_{B=}$ 0.1,$W_{D=}$ 0.9,$W_{E=}$0.2 nên ta có $tw(t_1)$ được tính như sau:

$$tw(t_1) = \frac{W_A+W_B+W_D+W_E}{4}=\frac{0.6+0.1+0.9+0.2}{4} = 0.45$$

Bảng 1.8 trình bày tất cả trọng số giao dịch của các giao dịch có trong bảng 1.6

Từ dữ liệu ở bảng 1.6, bảng 1.8 và định nghĩa 1.2 chúng ta có thể tính giá trị trọng số hỗ trợ ws(A) như sau: Bởi vì A xuất hiện trong các giao dịch $\{T_1,T_3,T_4,T_5\}$ vì vậy ws(A) được tính như sau:

$$ws(A) = \frac{tw(t_1)+tw(t_3)+tw(t_4)+tw(t_5)}{Sum} = \frac{0.45+0.45+0.3+0.42}{2.25} = 0.72$$

Bảng 1.9: Bảng trọng số hỗ trợ cho tập phổ biến 1 phần tử

item	Weighted support (ws)
A	0.72
B	1
C	0.6
D	0.78
E	0.81

Việc khai thác tập phổ biến được đánh trọng số đòi hỏi chúng ta cần xác định được tất cả các trọng số hỗ trợ thỏa ngưỡng trọng số hỗ trợ tối thiểu (minws) tương ứng với từng tập phổ biến do người dùng đặt ra.

$$FWI = \{X \subseteq I \mid ws(X) \geq minws\}$$

Định lý 1.1: "Mọi tập con khác rỗng của tập phổ biến cũng là tập phổ biến và mọi tập chứa tập không phổ biến đều là tập không phổ biến" có nghĩa là nếu cho $X \subset Y$ thì khi đó $ws(X) \geq ws(Y)$.

1.7 Cấu trúc WIT-tree

Để khai thác các luật kết hợp có trọng số, đầu tiên chúng ta phải tìm tất cả các tập được đánh trọng số thỏa điều kiện ngưỡng trọng số tối thiểu minws. Việc khai thác các tập được đánh trọng số được xem là quá trình quan trọng nhất trong việc khai thác các luật kết hợp có trọng số. Ramkumar và các đồng sự [7] đã trình bày giải thuật khai thác các tập được đánh trọng số dựa trên mô hình thuật toán Apriori.

Nhược điểm chính của các giải thuật dựa trên thuật toán Apriori là việc phải quét cơ sở dữ liệu nhiều lần để tìm ra các tập phổ biến, dẫn đến việc sẽ phát sinh chi phí lớn.

Khai thác tập phổ biến trên CSDL giao dịch có trọng số người ta quan tâm đến trọng số của các item. Năm 2003, Tao và các đồng sự đề xuất phương pháp khai thác WAR [8]. Thuật toán được đề nghị sử dụng một biến thể của thuật toán Apriori cho khai thác các FWI. Năm 2013, Vo và các đồng sự đề xuất một phương pháp khai thác nhanh FWI sử dụng WIT-tree và phát triển các tính chất trên WIT-tree để tính nhanh ws của các itemset [3] .

Giải thuật khai thác tập phổ biến được đánh trọng số dựa trên cấu trúc WIT-tree, cấu trúc này là phần mở rộng của cấu trúc IT-tree. Mỗi nút trên WIT-tree gồm 3 thành phần:

I. X: là một itemset
II. t(X): là tập các giao dịch có chứa X
III. ws: là trọng số giao dịch của X

Mỗi nút được ký hiệu như là một bộ ba (X, t(X), ws).

Trọng số hỗ trợ của mỗi nút được tính dựa theo công thức ở phần định nghĩa 1.2. Trọng số hỗ trợ được tính dựa trên các Tidset. Các nút được tạo bằng cách liên kết các nút ở mức k gọi là X, các nút được tạo ra bằng cách liên kết các nút ở mức k+1 gọi là Y.

Nút gốc (root) của cây WIT-tree chứa tất cả các nút có kích thước là 1 gọi là 1-itemset. Tất cả các nút ở mức 1 sẽ trở thành lớp tương đương với tiền tố là {} (hay [Ø]). Mỗi nút trong mức 1 sẽ kết nối với nhau trở thành các nút ở mức 2 thộc lớp tương đương mới, và sử dụng các item ở mức 1 như là một tiền tố. Mỗi nút có chung tiền tố ở mức 2, nó sẽ kết hợp với các nút phía sau nó để tạo ra các lớp tương đương mới. Quá trình này sẽ được thực hiện đệ quy để tìm ra các lớp tương đương ở các mức cao hơn.

1.8 Thuật toán WIT-FWI

WIT-FWI là phương pháp mới để khai thác nhanh tập phổ biến được đánh trọng số sử dụng WIT-tree đã được Bay Vo và cộng sự đề xuất năm 2013[3], thuật toán sử dụng một ngưỡng minws, gọi là ngưỡng trọng số hỗ trợ dùng để tỉa bớt các nút mà không phải là phổ biến giúp khai thác nhanh tập phổ biến được đánh trọng số.

Mô tả giải thuật WIT-FWI:

Bước 1: Cho tập L_r chứa tất các các tập được đánh trọng có kích thước là 1 và trọng số hỗ trợ của chúng thỏa điều kiện ngưỡng trọng số hỗ trợ tối thiểu minws (dòng 1).

Bước 2: Sắp xếp các nút chứa trong L_r theo thứ tự tăng dần dựa vào trọng số hỗ trợ của từng nút (dòng 2).

Bước 3: Khởi tạo FWI và gán nhãn null (dòng 3)

Bước 4: Gọi hàm **FWI-EXTEND** với tham số là L_r

Hàm **FWI-EXTEND**: xem xét mỗi một nút l_i có trong L_r với các nút phía sau nó để tạo ra một tập những nút mới là L_i (dòng 5 và 7).

Cách để tạo ra L_i như sau: Đầu tiên, cho X= l_i.itemset ∪ l_j.itemset và tính toán Y= t(X) = t(l_i) ∩ t(l_j) (dòng 8). Nếu ws(X) (được tính toán thông qua t(X), dòng 9) thỏa điều kiện minws (dòng 10). Nút mới <X,Y,ws(X)> tạo ra sẽ được thêm vào trong tập L_i (dòng 11). Sau khi tạo thành L_i, hàm FWI-EXTEND sẽ được gọi đệ quy với biến đầu vào là L_i (dòng 13) nếu số lượng các nút có trong tập L_i lớn hơn 1. Hàm COMPUTE-WS(Y) được sử dụng để tính trọng số hỗ trợ của tập X dựa trên giá trị trọng số giao dịch của từng giao dịch được tính toán trước đó với Y = t(X) (dòng 12).

Mô tả giải thuật WIT-FWI:

Đầu vào: Cơ sở dữ liệu D và một ngưỡng trọng số hỗ trợ tối thiểu minsup

Đầu ra: Tập phổ biến được đánh trọng số thỏa ngưỡng minsup

Phương thức:

WIT-FWI()

{

1. L_r = tất cả các item mà có trọng số hỗ trợ ws thỏa ngưỡng minsup
2. Sắp xếp các nút trong L_r theo thứ tự tăng dần của ws
3. Khởi tạo tập FWI=∅
4. Gọi hàm **FWI-EXTEND** với tham số truyền vào là L_r

FWI-EXTEND(L_r)

5. Với mỗi nút l_i trong L_r thực hiện việc
6. Thêm (l_i.itemset, l_i.ws) đến FWI
7. Tạo một tập L_i bằng cách nối l_i với tất cả l_j theo sau nó trong L_r
8. Cho X=l_i.itemset ∪ l_j.itemset và Y =t(l_i) ∩ t(l_j)
9. ws(X)=COMPUTE-WS(Y)
10. Nếu ws(X) thỏa ngưỡng minsup khi đó
11. Thêm nút <X,Y,ws(X)> vào trong L_i
12. Nếu số lượng nút trong Li ≥ 2 khi đó
13. Gọi đệ quy hàm **FWI-EXTEND** với tham số là L_i

}

Thực hiện giải thuật WIT-FWI trên cơ sở dữ liệu ví dụ mẫu

Để hiểu rõ giải thuật WIT-FWI ta sẽ tiến hành khai thác tập phổ biến được đánh trọng số trên cơ sơ dữ liệu mẫu ở ví dụ 1.1 (bảng 1.6 và 1.7) với ngưỡng trọng số hỗ trợ tối thiểu là 0.4.

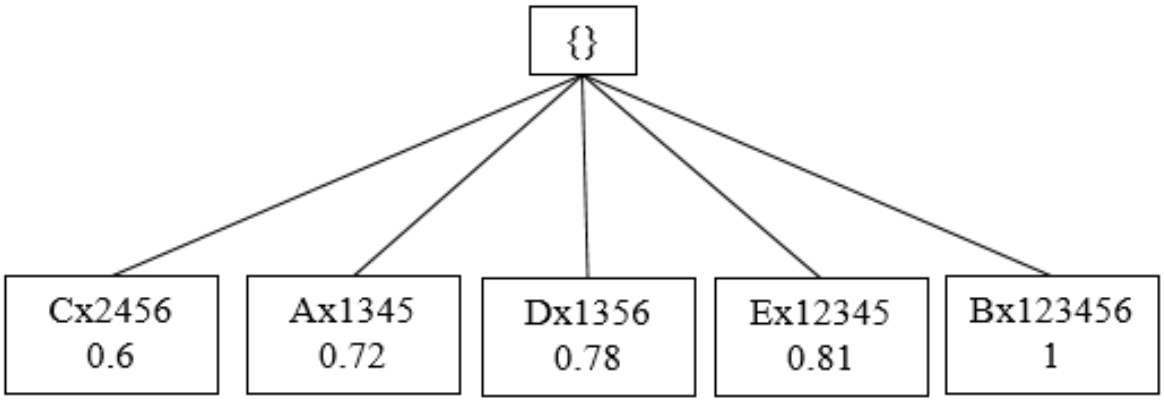
Bước 1: Tính trọng số hỗ trợ của các tập có kích thước là 1 ta được các tập như sau: ws(A) = 0.72, ws(B) = 1, ws(C) = 0.6, ws(D)= 0.78, ws(E) = 0.81, tất cả các tập này đều thỏa ngưỡng minws, khởi tạo tập L_r. Ta được tập L_r như sau:

L_r={<A,1345,0.72>, <B,123456,1.0>, <C,2456,0.6>, <D,1356,0.78>,<E,12345,0.81>}

Sau đó tiến hành sắp xếp các tập trong L_r theo thứ tự tăng dần của trọng số hỗ trợ, sau khi sắp xếp tăng dần ta được.

L_r={<C,2456,0.6>, <A,1345,0.72>, <D,1356,0.78>, <E,12345,0.81>,<B,123456,1.0>}

Tiến hành khởi tạo lớp {} chứa các tập phổ biến thỏa ngưỡng minws có kích thước là 1



Hình 1.5: Khởi tạo lớp tương đương rỗng cho WIT-tree

Bước 2: Tiến hành kết nối các itemset có kích thước là 1 tạo ra các nút thuộc lớp tương đương mới.

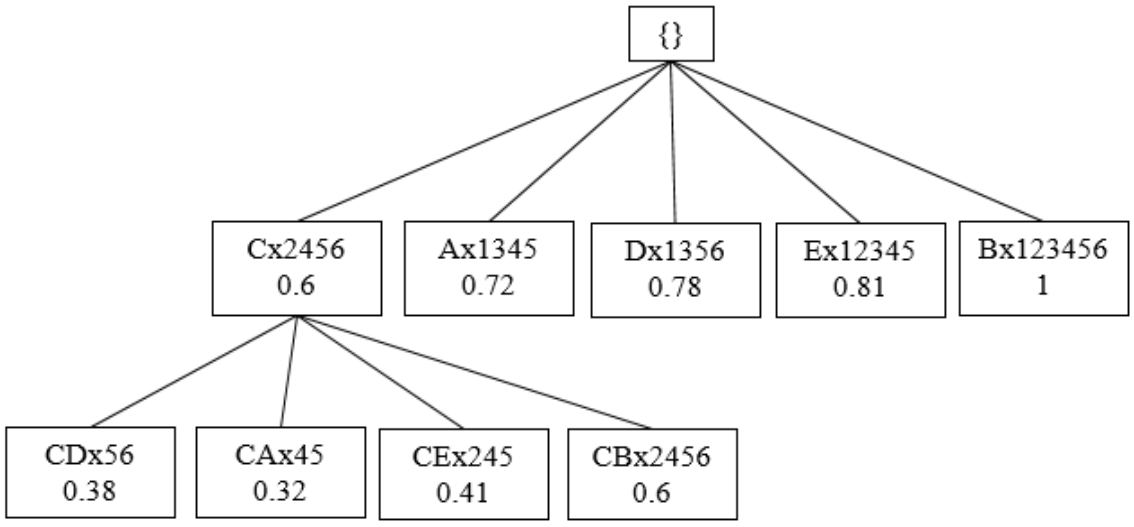
C kết nối với A ta được itemset mới CA với t(CA)=45 và ws(CA)=0.32, vì ws(CA) không thỏa minws nên không được thêm vào tập L_C.

C kết nối với D ta được itemset mới CD với t(CD)=56 và ws(CD)=0.38, vì ws(CD) không thỏa minws nên không được thêm vào tập Lc.

C kết nối với E ta được itemset mới CE với t(CE)=245 và ws(CE)=0.41, vì ws(CE) thỏa minws nên thêm vào tập L_C.

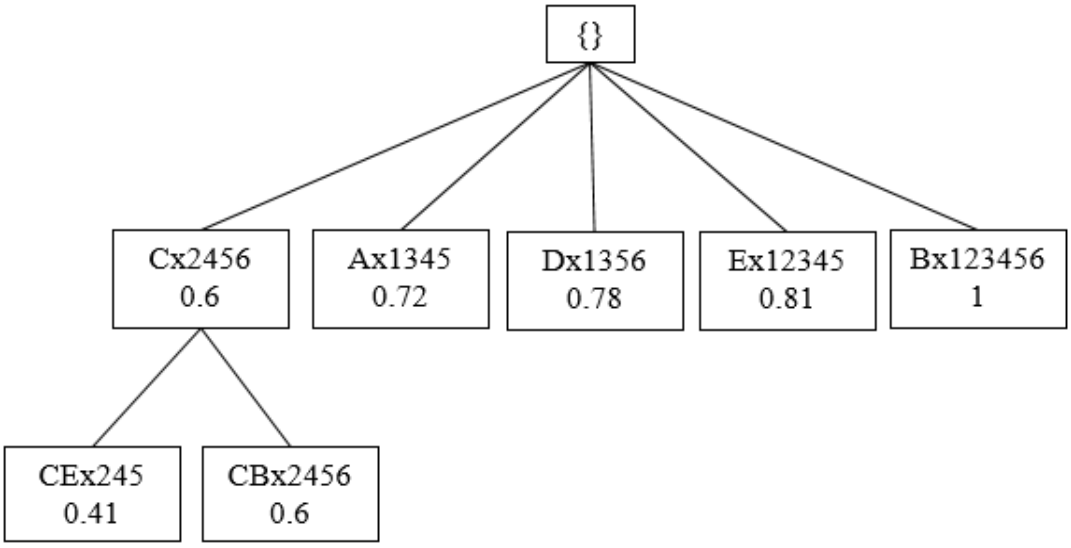
C kết nối với B ta được itemset mới CB với t(CB)=2456 và ws(CB)=0.60, vì ws(CB) thỏa minws nên thêm vào tập L_C.

Kết quả cuối cùng sau khi kết nối các nút ta có tập L_C như hình sau:



Hình 1.6: Tập L_c của cây WIT-tree

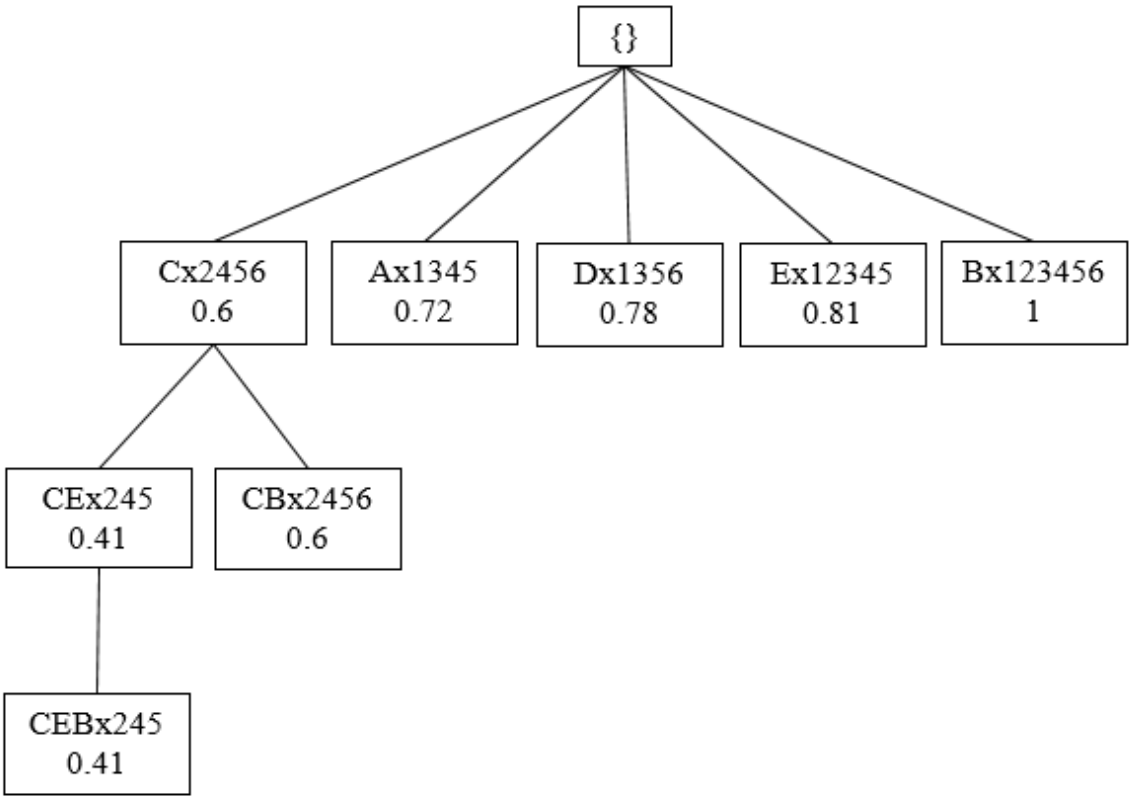
Trong tập L_c có các nút CD và CA có minws không thỏa nên tiến hành loại bỏ



Hình 1.7: Cây WIT-tree sau khi loại bỏ tập không thỏa minws trong L_c

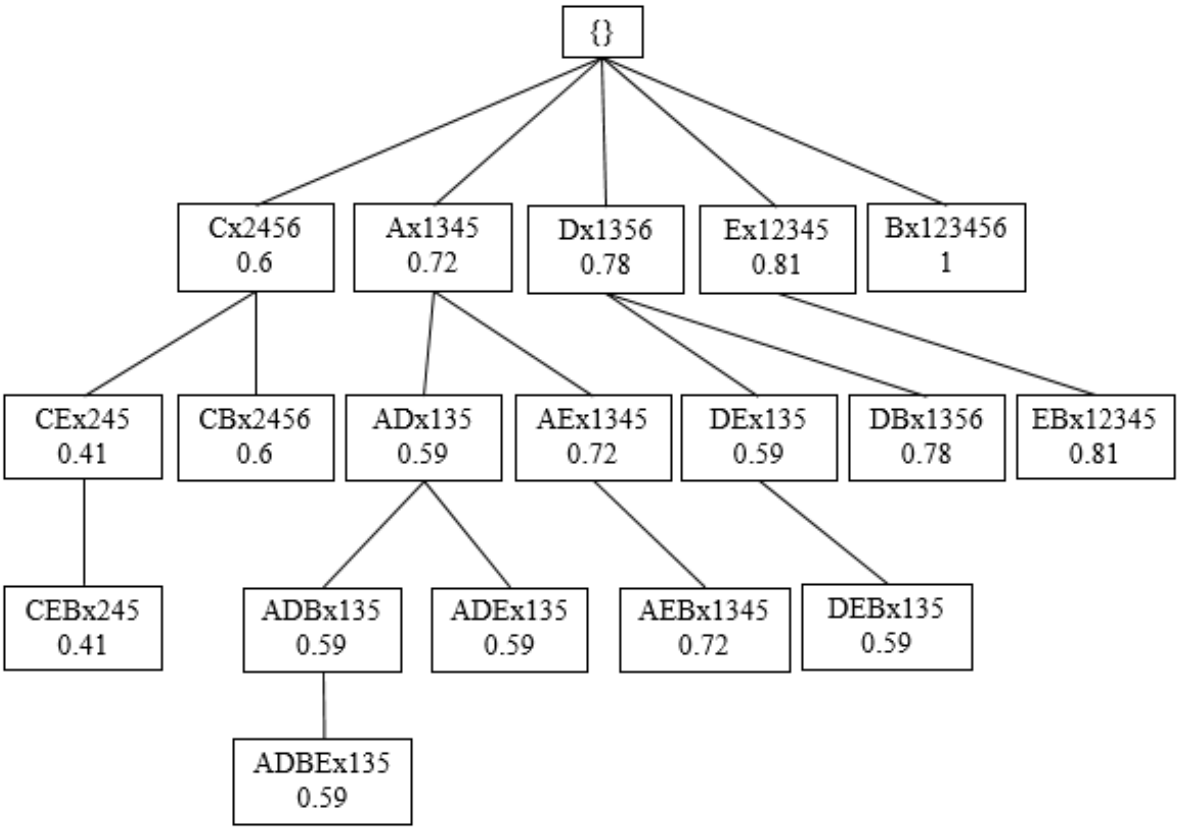
Bước 3: Vì số lượng các nút trong L_C nhiều hơn 1 tiến hành gọi đệ quy hàm FWI-EXTEND để tạo ra các nút con của tập L_C.

Ta kết nối CE với CB ta được nút mới CEB với t(CEB)=245, ws(CEB)= 0.41, ta thêm tập CEB vào trong tập L_{CE} ={CEB}



Hình 1.8: Cây WIT-tree với tập L_{CE}

Bước 4: Ta tiến hành thực hiện tiếp các nút còn lại để tìm tất cả các tập phổ biến có trọng số thỏa ngưỡng minws.



Hình 1.9: Cây WIT-tree hoàn chỉnh với minws = 0.4

Kết quả tập phổ biến được đánh trọng được tìm thấy:

FWI={{C}, {CE},{CB},{CEB},{A},{AD}, {ADB},{ADBE}, {ADE}, {AE}, {AEB},{AB},{D},{DA},{DE}, {DEB},{DB}, {E},{EB},{B}}

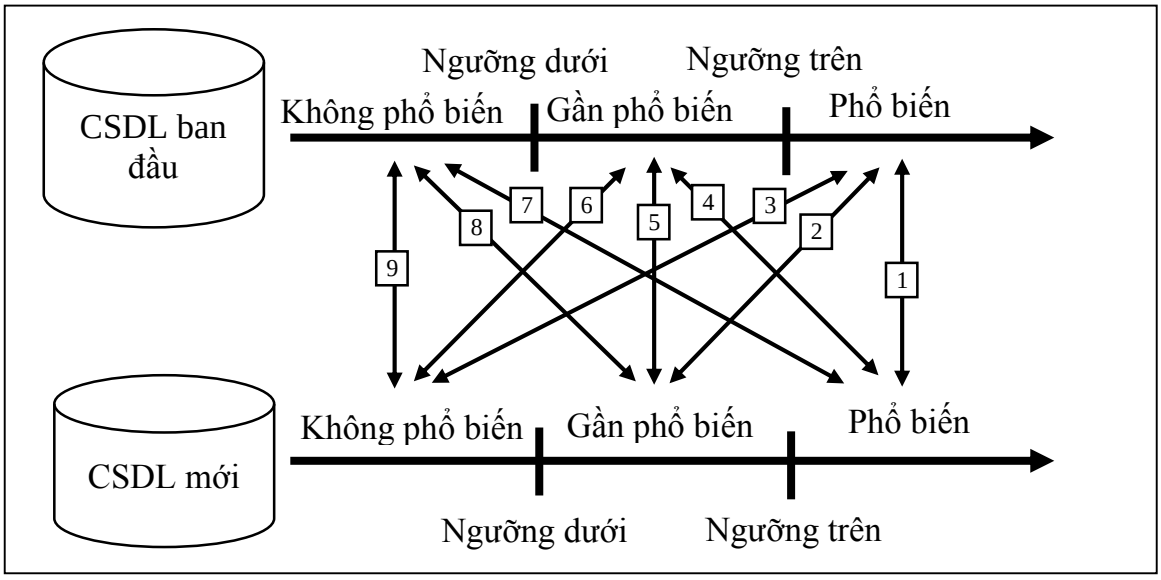
1.9 Khái niệm PRE-LARGE trong khai thác dữ liệu tăng trưởng

Khái niệm pre-large được Hong, Wang, Tao đề xuất năm 2001[5]. Nó dựa trên một ngưỡng an toàn f để giảm nhu cầu quét lại cơ sở dữ liệu ban đầu và duy trì hiệu quả luật kết hợp. Số lượng các giao dịch mới được thêm vào thuộc ngưỡng an toàn f được tính theo công thức sau:

$$f=\left\lfloor \frac{(S_U - S_L)\times|D|}{1-S_U} \right\rfloor$$

Trong đó S_U là ngưỡng trên, S_L là ngưỡng dưới và |D| là số lượng các giao dịch trong cơ sở dữ liệu ban đầu. Khi số lượng các giao dịch mới được thêm vào bằng hoặc nhỏ hơn ngưỡng an toàn f thì thuật toán không cần phải quét lại cơ sở dữ liệu ban đầu.

Khi hai ngưỡng được sử dụng thì mỗi itemset có 3 trường hợp có thể xảy ra là: phổ biến, gần phổ biến và không phổ biến. Phân chia itemsets trong cơ sở dữ liệu ban đầu và cơ sở dữ liệu mới thêm vào thành 9 trường hợp.



Hình 1.10: 9 trường hợp xảy ra khi thêm dữ liệu mới vào dữ liệu ban đầu

Trường hợp 1, 5, 6, 8 và 9 ở trên sẽ không làm ảnh hưởng đến kết quả cuối cùng của các luật kết hợp. Các trường hợp 2 và 3 có thể loại bỏ các luật kết hợp hiện có, và các trường hợp 4 và 7 có thể thêm các luật kết hợp mới.

Nếu chúng ta giữ lại tất cả các tập gần phổ biến và các tập phổ biến thì trường hợp 2,3 và 4 sẽ được xử lý một cách dễ dàng.

Ngoài ra trong giai đoạn duy trì luật kết hợp, tỷ lệ giữa giao dịch mới vào giao dịch cũ thường rất nhỏ. Điều này có thể nhận thấy rõ hơn khi khai thác dữ liệu lớn. Một tập phổ biến trong trường hợp 7 không thể là phổ biến cho toàn bộ dữ liệu được cập nhật, miễn là số lượng trong giao dịch mới là nhỏ so với số lượng giao dịch trong cơ sở dữ liệu ban đầu. Điều này đã được Hong và các cộng sự chứng minh.

1.10 Khai thác tập phổ biến trên cơ sở dữ liệu tăng trưởng

Các thuật toán khai thác tập phổ biến trước đây chủ yếu xử lý trên cơ sở dữ liệu được xác định trước. Ta biết rằng, ngày nay công nghệ thông tin phát triển, các cửa hàng giao dịch trực tuyến ngày càng phổ biến và phát triển rất nhanh, nhu cầu mua hàng trực tuyến ngày càng trở thành nhu cầu cần thiết của con người, điều này đã kéo theo dữ liệu giao dịch ngày càng tăng trưởng theo thời gian, vì vậy mà các tập phổ biến và các luật kết hợp đã được tính toán không còn giá trị, khi cần khai thác tập phổ biến thì cần phải tính lại từ đầu.

Để khắc phục điều này, nhiều nhà nghiên cứu đã tìm hiểu và phát triển thuật toán tăng trưởng, trong đó có tác giả Nguyễn Xuân Huy và cộng sự đề xuất thuật toán tăng trưởng[1] được đăng trên tạp chí khoa học công nghệ năm 2007, Vo và cộng sự đề xuất phương pháp tiếp cận hiệu quả để duy trì dàn tập phổ biến dựa trên khái niệm pre-large [4].

Ý tưởng cơ bản của thuật toán như sau:

Cho cơ sở dữ liệu giao dịch ban đầu, lúc đầu thuật toán tính độ hỗ trợ của tất cả các tập mục có trong cơ sở dữ liệu giao dịch ban đầu và lưu trữ vào trong tập K. Theo thời gian, số lượng các giao dịch tăng dần, thuật toán chỉ tính toán với các dữ liệu được thêm mới vào mà không cần phải tính lại từ đầu.

Để tính độ hỗ trợ của các tập mục dữ liệu, không cần phải tính tất cả các tập mục mà chỉ cần tính cho các tập mục dữ liệu xuất hiện trong các giao dịch mới.

Thuật toán tăng trưởng INCREMENTAL-FIL()

Mô tả thuật toán:

Bước 1: Cho cơ sở dữ liệu đầu vào D, ngưỡng an toàn f dựa trên D, dữ liệu tăng trưởng D', ngưỡng hỗ trợ trên S_U và ngưỡng hỗ trợ dưới S_L.

Bước 2: Nếu dữ liệu ban đầu rỗng thì thực hiện gọi hàm **FIL** để xây dựng dàn sử dụng ngưỡng S_L để xây dựng dàn tập phổ biến và tính lại ngưỡng an toàn.

Bước 3: Nếu số lượng giao dịch trên D' lớn hơn f thì gọi hàm **FIL** xây dựng dàn tập phổ biến cho D + D' sử dụng ngưỡng hỗ trợ S_L.

Bước 4: Ngược lại, nếu số lượng giao dịch mới thêm vào bé hơn hoặc bằng f thì thực hiện xóa các thông tin tidset, cập nhật lại thông tin ở mức 1 và đánh dấu những nút có thông tin thay đổi ngưỡng trọng số hỗ trợ.

Bước 5: Gọi hàm cập nhật dàn tập phổ biến để cập nhật tất cả các nút trên dàn với tham số L_1.

Mô tả thuật toán INCREMENTAL-FIL()

Đầu vào:

- Dữ liệu ban đầu D
- Ngưỡng an toàn f được tính dựa trên D
- Dữ liệu tăng trưởng D'
- Ngưỡng hỗ trợ trên S_U
- Ngưỡng hỗ trợ dưới

Đầu ra: Dàn Tập phổ biến

Phương thức:

INCREMENTAL-FIL()

1. Nếu dữ liệu ban đầu D = 0 thuật toán sẽ thực hiện
2. Gọi hàm FIL để xây dựng dàn cho D' sử dụng ngưỡng S_L
3. Tính lại ngưỡng hỗ trợ f theo công thức $f=\left\lfloor \frac{(S_U-S_L)\times|D^{|\prime})}{1-S_U} \right\rfloor$

4. Nếu tổng trọng số giao dịch trong dữ liệu D' > f thuật toán sẽ thực hiện
5. Gọi hàm FIL để xây dựng cây cho D+D' sử dụng ngưỡng S_L
6. Tính lại ngưỡng hỗ trợ $f=\left\lfloor \frac{(S_U-S_L)\times|D|+|D'|}{1-S_U} \right\rfloor$
7. Ngược lại thuật toán sẽ thực hiện
8. Xóa thông tin tidset ở mỗi nút trên dàn
9. Cập nhật lại thông tin các nút ở mức đầu tiên L_1, và đánh dấu toàn bộ các nút được thay đổi thỏa ngưỡng hỗ trợ S_L
10. Gọi thủ tục UPDATE-PFIL để cập nhật lại tất cả các nút trên dàn với tham số L_1
11. Tính lại ngưỡng an toàn f = f – |D'|
12. Cập nhật lại dữ liệu D = D + D'

Thực hiện thuật toán tăng trưởng INCREMENTAL-FIL trên dữ liệu mẫu

Cho dữ liệu ban đầu D_1 có 6 giao dịch, ngưỡng hỗ trợ dưới S_L=50%, ngưỡng hỗ trợ trên S_U=65%

Transaction	Items
1	A,C,T,W,Z
2	C,D,W
3	A,C,T,W
4	A,C,D,W
5	A,C,D,T,W
6	C,D,T

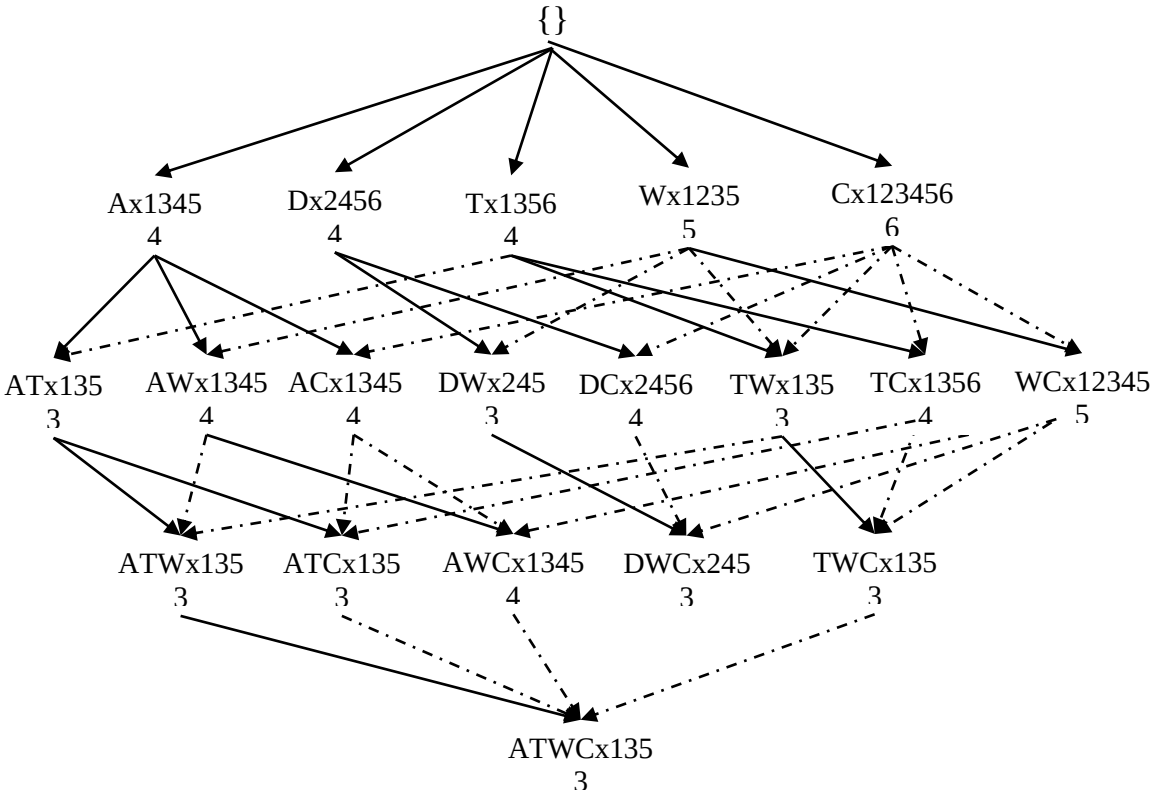
Dữ liệu D_2 tăng trưởng lần 1

Transaction	Items
7	A,T,W,Z
8	C,T,W,Z

Dữ liệu D_3 tăng trưởng lần 2

Transaction	Items
9	A,D,T,W,Z

Đầu tiên (dữ liệu D_1 có 6 giao dịch) thuật toán thực hiện gọi hàm FIL để xây dựng dàn tập phổ biến dựa vào cơ sở dữ liệu ban đầu D_1, sau đó thực hiện tính ngưỡng an toàn $f=\left\lfloor\frac{(0.65-0.5)\times(6)}{1-0.65}\right\rfloor=2$ và tiến hành cập nhật lại số lượng giao dịch D = $\emptyset + D1$

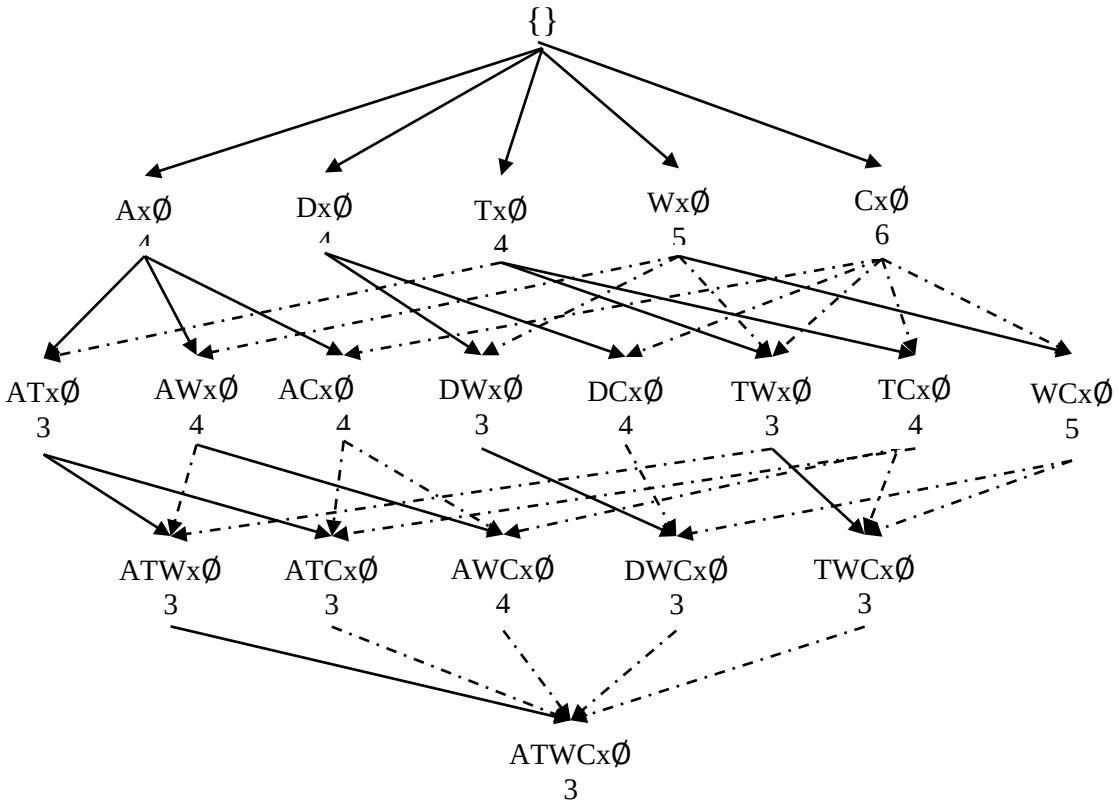


Hình 1.11: FIL cho dữ liệu D1 với minsup = 50% sử dụng TFIL

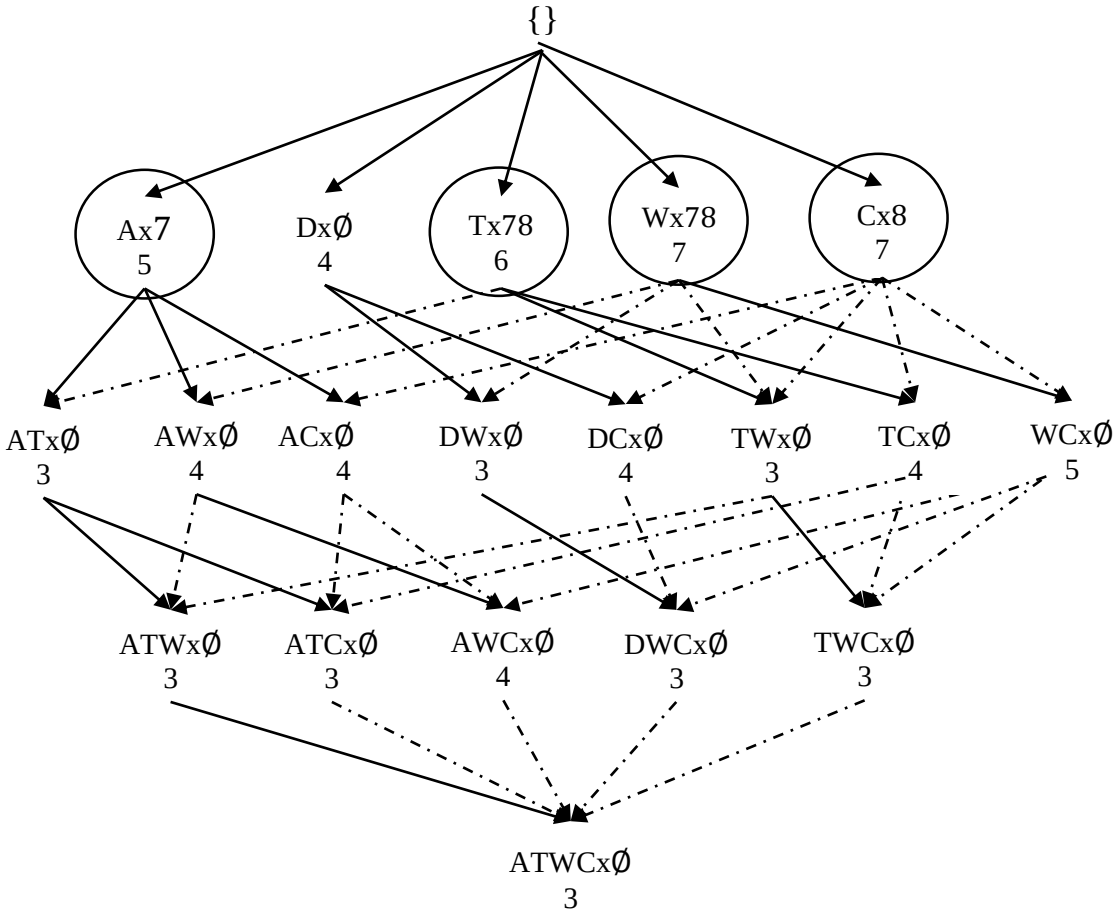
Kế tiếp (dữ liệu D_2 có 2 giao dịch) số lượng giao dịch trong D_2 còn nằm trong ngưỡng an toàn f nên thuật toán không quét lại toàn bộ dữ liệu mà thực hiện cập nhật thông tin như sau:

Thật toán sẽ xóa thông tin tidset kết hợp với tất cả các nút trên dàn.

Thuật toán chèn vào thông tin tidset (chỉ trong các giao dịch 7 và 8) kết hợp với tập phổ biến 1-itemset trên dàn và đánh dấu những nút đã cập nhật.



Hình 1.12: Xóa thông tin tidset trên dàn

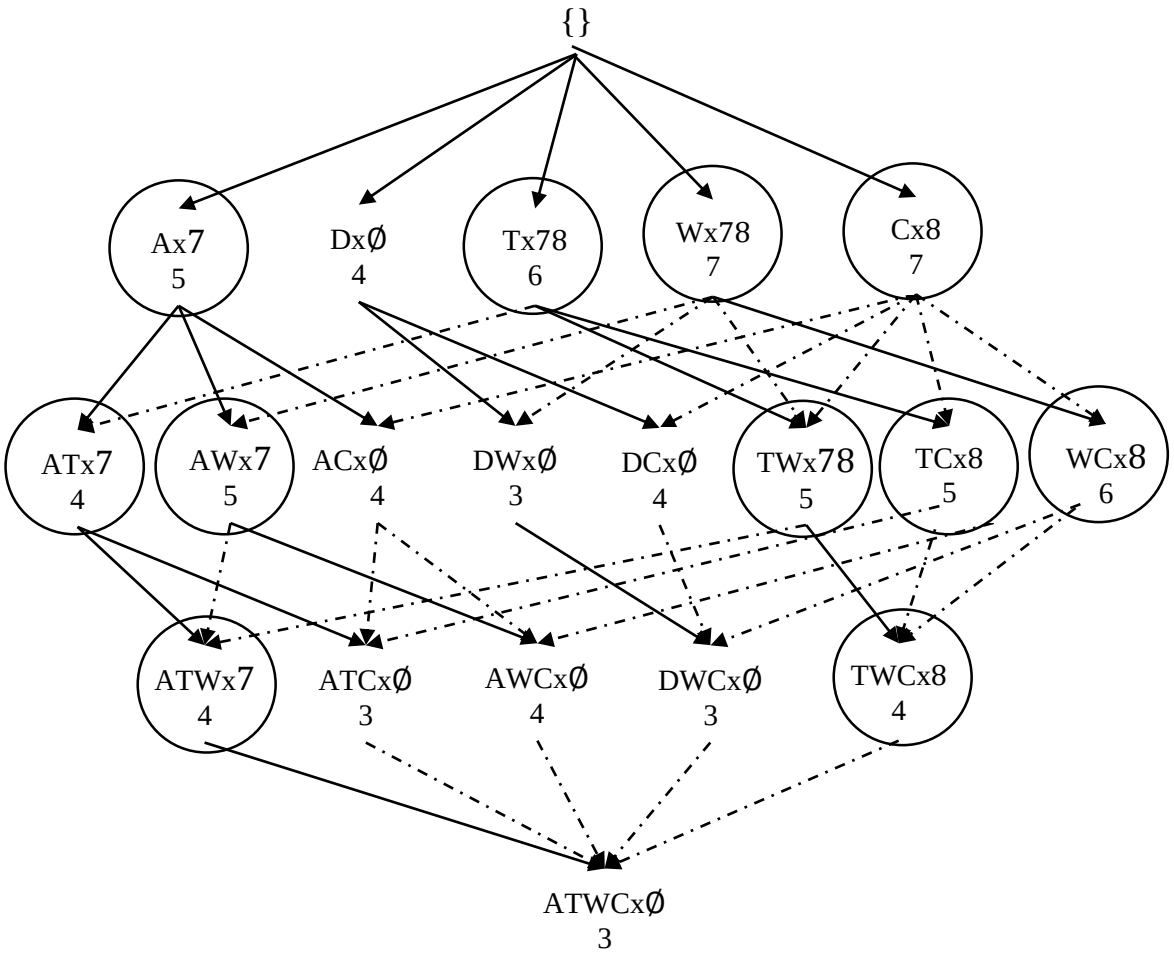


Hình 1.13: Cập nhật thông tin tidset 1-itemset

Sau đó thuật toán tiến hành gọi đệ quy hàm UPDATE-PFIL để cập nhật thông tin tidset các nút kết hợp với 1-itemset trên dàn.

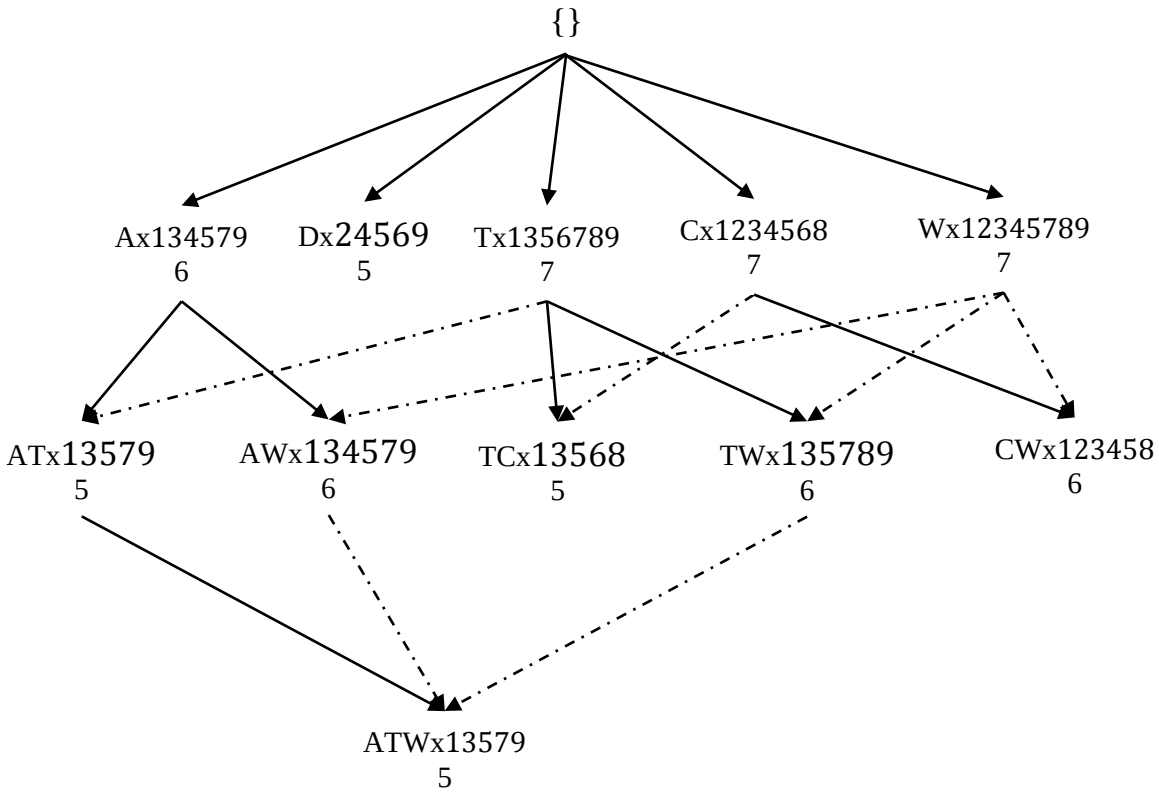
Tính lại ngưỡng an toàn f = f-|D_2| = 2-2 =0

Cập nhật lại dữ liệu D = D_1 + D_2



Hình 1.14: Gọi UPDATE-PFIL cập nhật thông tin tidset kết hợp với 1-itemset

Kế tiếp tiến hành tăng trưởng dữ liệu D_3 (D_3 có 1 giao dịch), lúc này ngưỡng an toàn f = 0 nên khi có bất kỳ tăng trưởng nào thuật toán sẽ phải quét lại toàn bộ dữ liệu để tính lại thông tin các nút trên dàn.



Hình 1.15: Kết quả sau khi tăng trưởng D_3

Sau khi thực hiện tăng trưởng D_3, thuật toán sẽ tính lại ngưỡng an toàn $f=\left\lfloor \frac{(0.65-0.5)\times(9)}{1-0.65} \right\rfloor=3$

Lúc này thực hiện tăng trưởng D_4, nếu số lượng các giao dịch trong $D_4 \leq 3$ thì thuật toán sẽ tiến hành cập nhật lại dàn tập phổ biến mà không cần phải quét lại toàn bộ dữ liệu.

CHƯƠNG 2: KHAI THÁC TẬP PHỔ BIẾN ĐƯỢC ĐÁNH TRỌNG SỐ TRÊN CƠ SỞ DỮ LIỆU TĂNG TRƯỞNG

2.1 Khai thác tập phổ biến được đánh trọng số

Việc khai thác tập phổ biến được đánh trọng số trên cơ sở dữ liệu tăng trưởng chưa được nhiều nhà nghiên cứu quan tâm phát triển, trong khi đó các thuật toán khai thác tập phổ biến được đánh trọng số đã được nhiều nhà nghiên cứu phát triển, trong đó Vo và đồng sự đã đề xuất thuật toán WIT-FWI dựa vào cấu trúc cây WIT-tree [3]. sau đó thuật toán đã được cải tiến để tăng tốc độ tính toán và giảm bớt việc sử dụng bộ nhớ như là WIT-FWI-MODIFY, WIT-FWI-DIFF đã mang lại hiệu suất tối ưu trong việc khai thác dữ liệu được đánh trọng số. Tuy nhiên việc khai thác này thường là trên cơ sở dữ liệu được xác định trước, trong khi ngày nay cơ sở dữ liệu thường tăng trưởng theo thời gian. Đặc biệt là cơ sở dữ liệu bán hàng, các món hàng sẽ được mua và thay đổi theo thời gian. Việc ứng dụng thuật toán WIT-FWI trong khai thác cơ sở dữ liệu tăng trưởng tồn tại nhược điểm là mỗi khi có món hàng được mua, việc tính toán sẽ phải quét lại toàn bộ cơ sở dữ liệu và cần phải tốn khá nhiều thời gian cho việc tính toán.

Để khắc phục nhược điểm phải quét lại toàn bộ cơ sở dữ liệu mỗi khi có món hàng được mua, đề tài nghiên cứu và ứng dụng thuật toán WIT-FWI [3][4] kết hợp với khái niệm pre-large [5] trong khai thác dữ liệu tăng trưởng trong đó sử dụng hai ngưỡng WS_u và WS_L được gọi là ngưỡng trọng số hỗ trợ trên và ngưỡng trọng số hỗ trợ dưới hoạt động như là bộ nhớ đệm nhằm tăng tốc độ tính toán, trong đó có sử dụng một ngưỡng an toàn f nhằm hạn chế việc phải quét lại toàn bộ dữ liệu ban đầu.

2.2 Khai thác tập phổ biến được đánh trọng số trên dữ liệu tăng trưởng.

Ứng dụng khái niệm pre-large của Hong và cộng sự đề xuất năm 2001 [5] trong khai thác dữ liệu tăng trưởng kết hợp với thuật toán WIT-FWI trong khai thác dữ liệu được đánh trọng số được Vo và đồng sự đề xuất năm 2013 [3], nghiên cứu đề xuất thuật toán INCREMENTAL_WIT_FWI trong khai thác tập được đánh trọng phổ biến trên CSDL tăng trưởng nhằm hạn chế việc phải quét lại cơ sở dữ liệu ban đầu khi có dữ liệu mới được thêm vào.

2.3 Các bước của thuật toán tăng trưởng INCREMENTAL-WIT-FWI()

Bước 1: Cho cơ sở dữ liệu đầu vào D, dữ liệu tăng trưởng D', ngưỡng trọng số hỗ trợ trên WS_U và ngưỡng trọng số hỗ trợ dưới WS_L và tập trọng số của từng item, ngưỡng an toàn f dựa trên D và WS_U, WS_L.

Bước 2: Nếu dữ liệu ban đầu rỗng thì thực hiện gọi hàm **WIT-FWI** để xây dựng cây cho dữ liệu D, sử dụng ngưỡng WS_L và tính lại ngưỡng an toàn f.

Bước 3: Nếu số lượng giao dịch trên D' lớn hơn ngưỡng an toàn f thì gọi hàm **INCREMENTAL-WIT-FWI()** xây dựng cây cho D + D' sử dụng ngưỡng trọng số hỗ trợ WS_L.

Bước 4: Ngược lại, nếu số lượng giao dịch mới thêm vào bé hơn hoặc bằng ngưỡng an toàn f thì thực hiện xóa các thông tin tidset ở các nút trên cây, cập nhật lại thông tin ở mức đầu tiên L_1, và đánh dấu toàn bộ các nút được thay đổi thỏa ngưỡng hỗ trợ WS_L.

Bước 5: Gọi hàm cập nhật cây để cập nhật tất cả các nút trên cây với tham số L_1.

Bước 6: Tính lại ngưỡng an toàn f và cập nhật lại dữ liệu D

Bước 7: Duyệt cây và lọc ra những tập thỏa ngưỡng hỗ trợ WS_U

2.4 Mô tả thuật toán INCREMENTAL_WIT_FWI

Đầu vào:

- Cơ sở dữ liệu ban đầu D
- Dữ liệu tăng trưởng D'

- Ngưỡng trọng số hỗ trợ trên WS_U
- Ngưỡng trọng số hỗ trợ dưới WS_L
- Ngưỡng an toàn f được xác định từ dữ liệu ban đầu D và ngưỡng WS_L và WS_U
- Dữ liệu trọng số

Đầu ra: Tập phổ biến được đánh trọng số thỏa ngưỡng WS_U

INCREMENTAL_WIT_FWI()

1. Nếu dữ liệu ban đầu D = 0 thuật toán sẽ thực hiện
2. Gọi hàm WIT_FWI để xây dựng cây cho D' sử dụng ngưỡng WS_L
3. Tính lại ngưỡng hỗ trợ f theo công thức

$$f=\left(\frac{(WS_U-WS_L)\times Sum(D)}{1-WS_U}\right)$$

4. Nếu tổng trọng số giao dịch trong dữ liệu D' > f thuật toán sẽ thực hiện
5. Gọi hàm WIT_FWI để xây dựng cây cho D+D' sử dụng ngưỡng WS_L
6. Tính lại ngưỡng hỗ trợ $f=\left(\frac{(WS_U-WS_L)\times Sum(D)+Sum(D')}{1-WS_U}\right)$
7. Ngược lại thuật toán sẽ thực hiện
8. Xóa thông tin tidset ở mỗi nút trên cây WIT_FWI
9. Gọi hàm UpdateLevel1() cập nhật lại thông tin ở mức đầu tiên L_1, và đánh dấu toàn bộ các nút được thay đổi thỏa ngưỡng hỗ trợ WS_L
10. Gọi thủ tục UpdateFWI() để cập nhật lại tất cả các nút trên cây với tham số L_1
11. Tính lại ngưỡng an toàn f = f – Sum(D')
12. Cập nhật lại dữ liệu D = D + D'
13. Duyệt cây và lọc ra những tập thỏa ngưỡng hỗ trợ WS_U

Trong đó, Sum(D) là tổng trọng số hỗ trợ của các giao dịch trong dữ liệu D, Sum(D') là tổng trọng số hỗ trợ của các giao dịch dữ liệu D', Sum(D)+Sum(D') là tổng trọng số hỗ trợ của các giao dịch trong D và D'

2.5 Thực thiện thuật toán tăng trưởng trên dữ liệu mẫu

Để hiểu rõ giải thuật **INCREMENTAL_WIT_FWI()** ta sẽ tiến khành khai thác tập phổ biến được đánh trọng số trên cơ sơ dữ liệu mẫu ở bảng 1.6 (gọi là D_1) và bảng 1.7 với ngưỡng trọng số hỗ trợ dưới là WS_L=0.4 và ngưỡng trọng số hỗ trợ trên là WS_U=0.6 và bảng dữ liệu tăng trưởng D_2 (bảng 2.1) và D_3 (bảng 2.2).

Bảng 2.1: Bảng dữ liệu tăng trưởng D_2

Transaction	Item
7	A, D

Bảng 2.2: Bảng dữ liệu tăng trưởng D_3

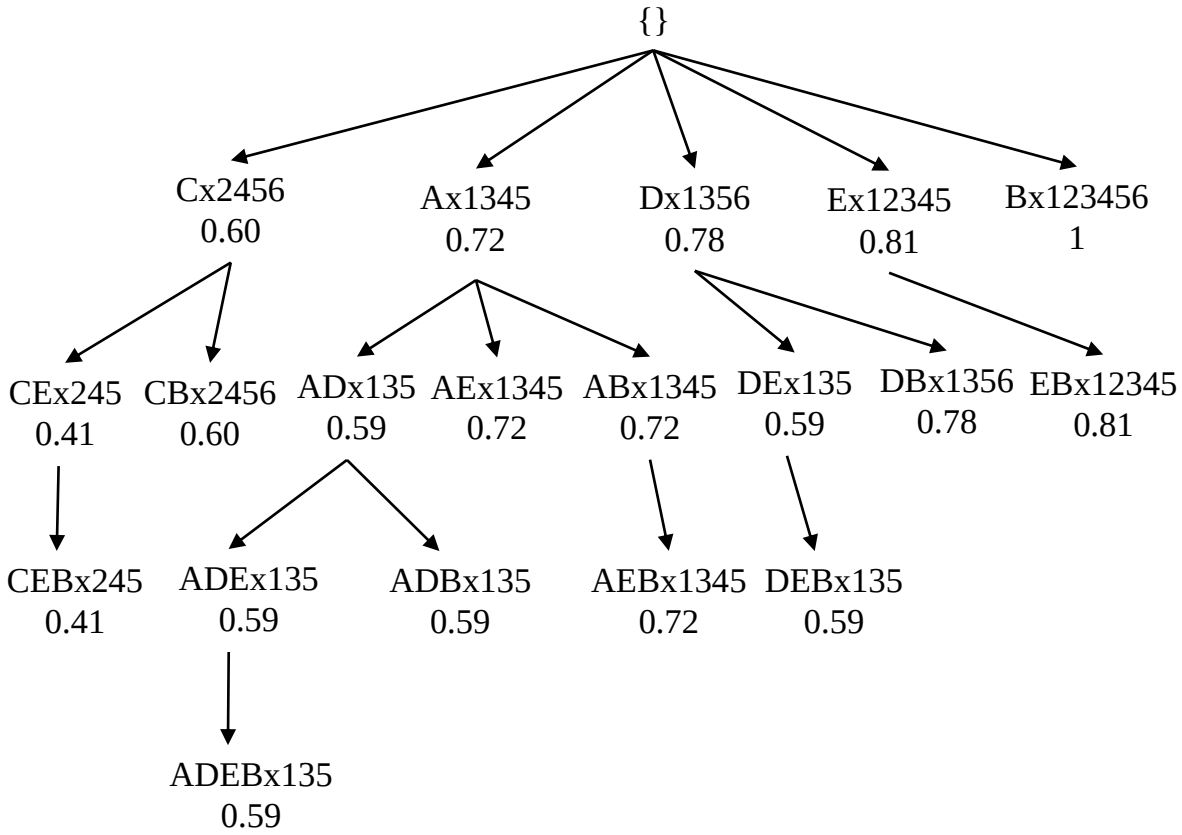
Transaction	Item
8	A, C, D

Trước tiên thuật toán nhận được các giá trị

- D: là cơ sở dữ liệu ban đầu
- Ngưỡng trọng số hỗ trợ trên WS_U =0.6
- Ngưỡng trọng số hỗ trợ dưới WS_L= 0.4
- Tập dữ liệu trọng số (bảng 1.7)
- Giá trị ngưỡng an toàn f được tính từ dữ liệu D

Lần đầu tiên chạy chương trình dữ liệu D = 0, trong trường hợp hày thuật toán sẽ gọi hàm **WIT-FWI** sử dụng ngưỡng hỗ trợ dưới WS_L để xây dựng cây, và tính lại ngưỡng an toàn f = $((WS_U - WS_L)*Sum(D)/(1-WS_U))$ = ((0.6-0.4)*2.25)/(1-0.6) = 1.1.

Cập nhật lại dữ liệu D= $\emptyset + D_1$



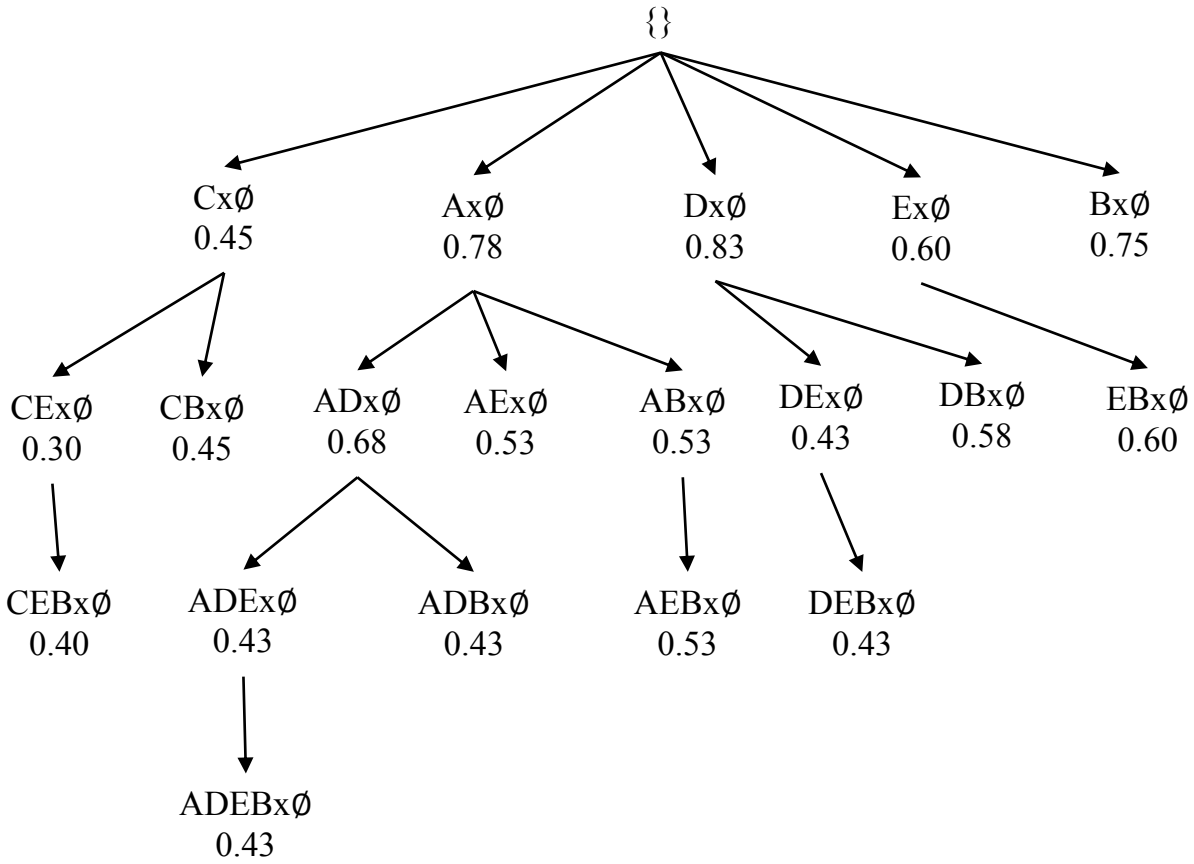
Hình 2.1: Cây D_1 với ngưỡng trọng số hỗ trợ WS_L 40%

Sau khi có cây D_1 ta tiến hành thêm dữ liệu D_2 (bảng 2.1) để tính dữ liệu tăng trưởng.

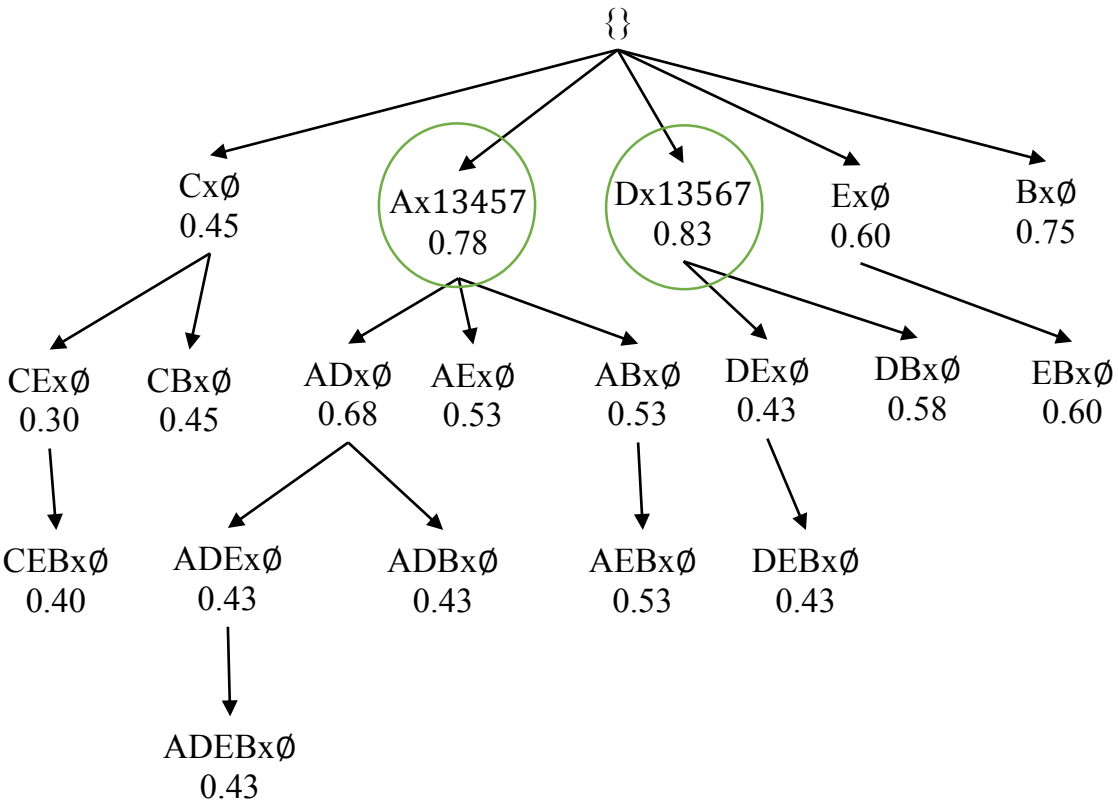
Lần này dữ liệu D đã khác 0, trong bảng 2.1 có 1 giao dịch, thuật toán sẽ tính tổng trọng số của các giao dịch trên D_2.và so sánh với ngưỡng an toàn f.

Lúc này Sum(D_2) được gọi là tổng các trọng số giao dịch trên dữ liệu D_2 = 0.75

f= 1.1 > Sum(D_2) nên thỏa ngưỡng an toàn, thuật toán sẽ không quét lại toàn bộ dữ liệu mà chỉ thực hiện xóa các thông tin tidset ở các nút trên cây và cập nhật lại thông tin các nút ở mức đầu tiên L_1, sau đó đánh dấu các nút có thông tin thay đổi và và thỏa ngưởng WS_L.

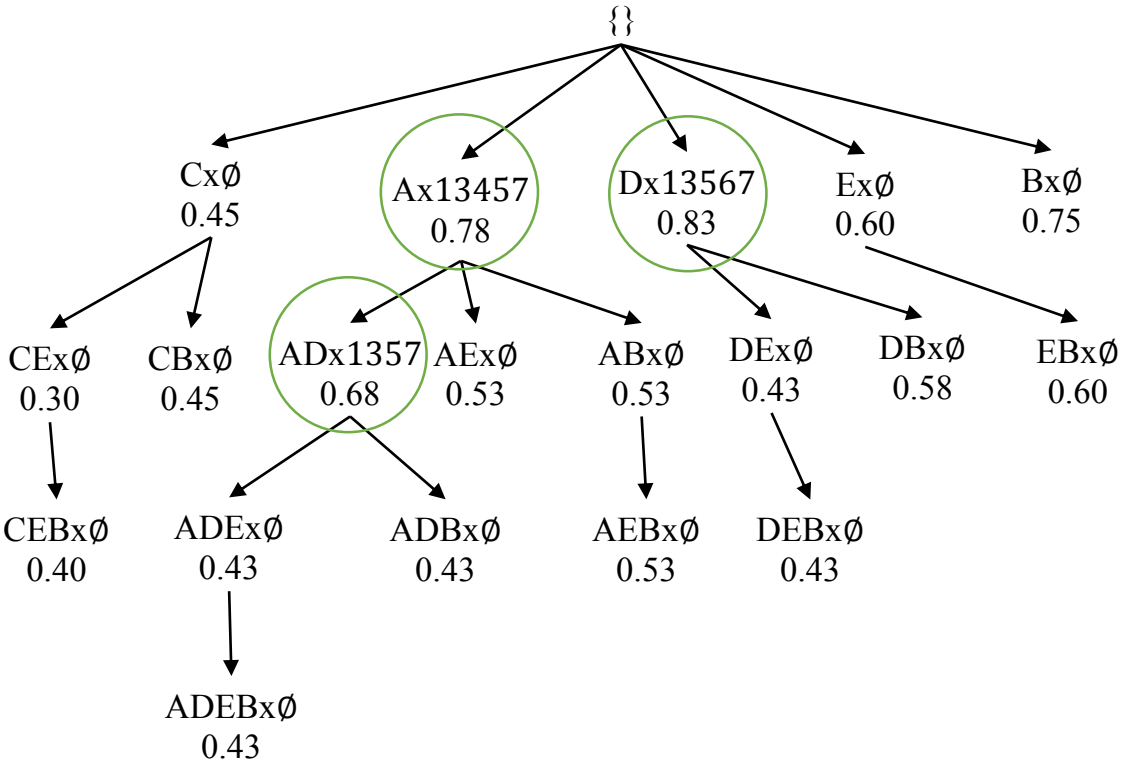


Hình 2.2: Xóa thông tin tidset ở các nút trên cây



Hình 2.3: Cập nhật lại thông tin các nút ở L_1 và đánh dấu các nút thay đổi

Kế tiếp thuật toán gọi hàm **UPDATE-FWI** để cập nhật các nút trên cây với tham số là L_1



Hình 2.4: Cập nhật lại các nút ở mức L_1

Sau khi cập nhật xong thuật toán sẽ tính lại ngưỡng an toàn f và cập nhật lại dữ liệu D

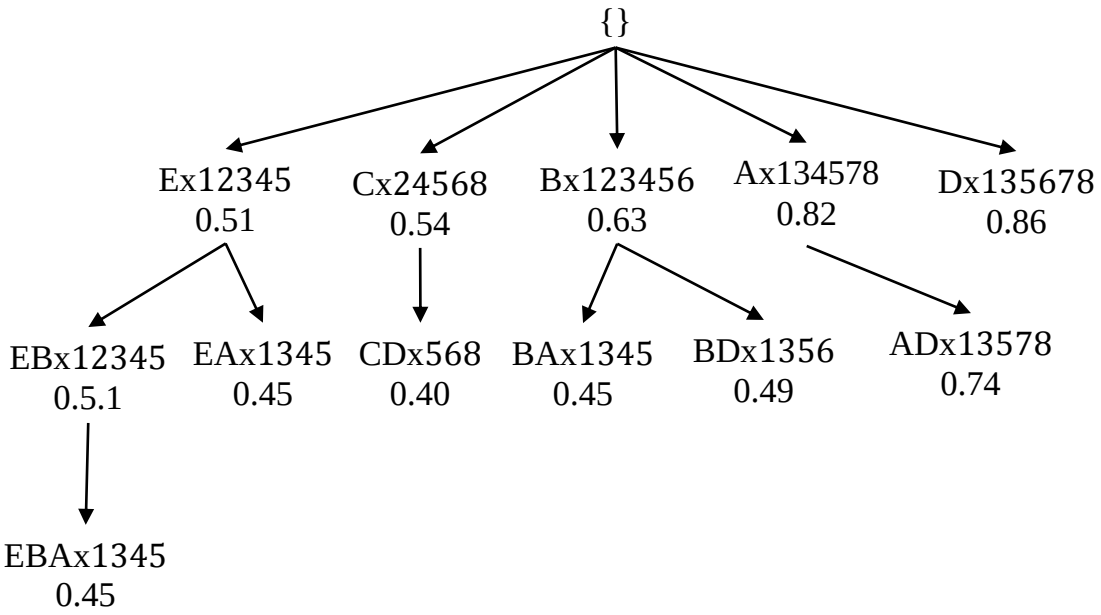
Lúc này dữ liệu D_2 đã được thêm vào thuật toán sẽ tính lại tổng trọng số của các giao dịch Sum(D) = Sum(D_1) + Sum(D_2) = 2.28+0.75 = 3.03

Lúc này f = f – Sum(D_2)= 1.1-0.75 = 0.35

Cập nhật lại dữ liệu D= D_1 + D_2

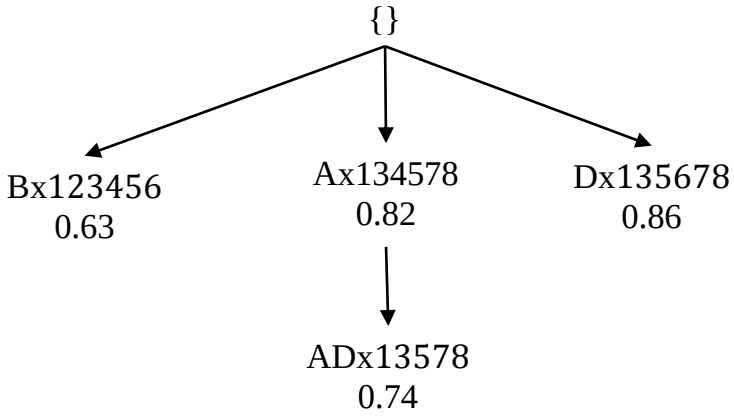
Khi dữ liệu mới được thêm vào mà có tổng trọng số hỗ trợ lớn hơn ngưỡng an toàn f điều này có nghĩa là thuật toán cần phải quét lại toàn bộ dữ liệu để tính toán lại các thông tin trên cây.

Ta tiến hành tăng trưởng dữ liệu D_3 (bảng 2.2) có 1 giao dịch, thuật toán sẽ tiến hành tính tổng trọng số giao dịch của dữ liệu D_3 = 0.6. Tuy nhiên, lúc này ngưỡng an toàn f = 0.35 vượt ngưỡng an toàn nên ta tiến hành quét lại toàn bộ dữ liệu để tính toán lại các thông tin trên cây.



Hình 2.5: Cây sau khi tăng trưởng dữ liệu D_3 với WS_L=0.4

Sau khi kết thúc tăng trưởng D_3 thuật toán sẽ lấy ra danh sách các nút thỏa ngưỡng trọng số hỗ trợ trên (WS_U=0.60).



Hình 2.6: Danh sách các nút thỏa ngưỡng WS_U sau khi tăng trưởng D_3

Ta có thể hiểu được khi dữ liệu mới được thêm vào thuật toán sẽ xem xét ngưỡng an toàn trên dữ liệu mới, nếu không vượt quá ngưỡng an toàn thì không cần phải quét lại toàn bộ dữ liệu mà chỉ tiến hành cập nhật lại thông tin với dữ liệu mới, điều này giúp giảm đáng kể thời gian cho việc khai thác tập phổ biến được đánh trọng số trên cơ sở dữ liệu tăng trưởng.

CHƯƠNG 3: THỰC NGHIỆM VÀ ĐÁNH GIÁ

3.1 Môi trường thực nghiệm

Tất cả các thí nghiệm được trình bày trong phần này được thực hiện trên máy tính cá nhân với thông số cấu hình máy tính CPU Intel i5-5200U 2.20 GHz và 8 GB RAM chạy hệ điều hành Windows 8. Tất cả các chương trình đã được mã hóa trong C# 2013.

3.2 Đặc điểm dữ liệu thực nghiệm

Nguồn cơ sở dữ liệu thực nghiệm được lấy từ trang web Frequent Itemset Mining Dataset Repository: http://fimi.cs.helsinki.fi/data/ với tên cơ sở dữ liệu được sử dụng cho thực nghiệm là: Chess, Mushroom, Connect. Các bộ dữ liệu này được sửa đổi bằng cách thêm vào một bảng để lưu trữ trọng số hỗ trợ của từng item, trọng số này có giá trị từ 1 đến 10 cho mỗi cơ sở dữ liệu.

Dữ liệu thực nghiệm được tiến hành khai thác trên dữ liệu chuẩn ở bảng 3.1

Bảng 3.1: Cơ sở dữ liệu thực nghiệm có chỉnh sửa trọng số hỗ trợ

CSDL	Số Item	Số giao dịch	Tình trạng
CHESS	76	3196	Đã được sửa đổi
MUSHROOM	120	8124	Đã được sửa đổi
CONNECT	129	67557	Đã được sửa đổi

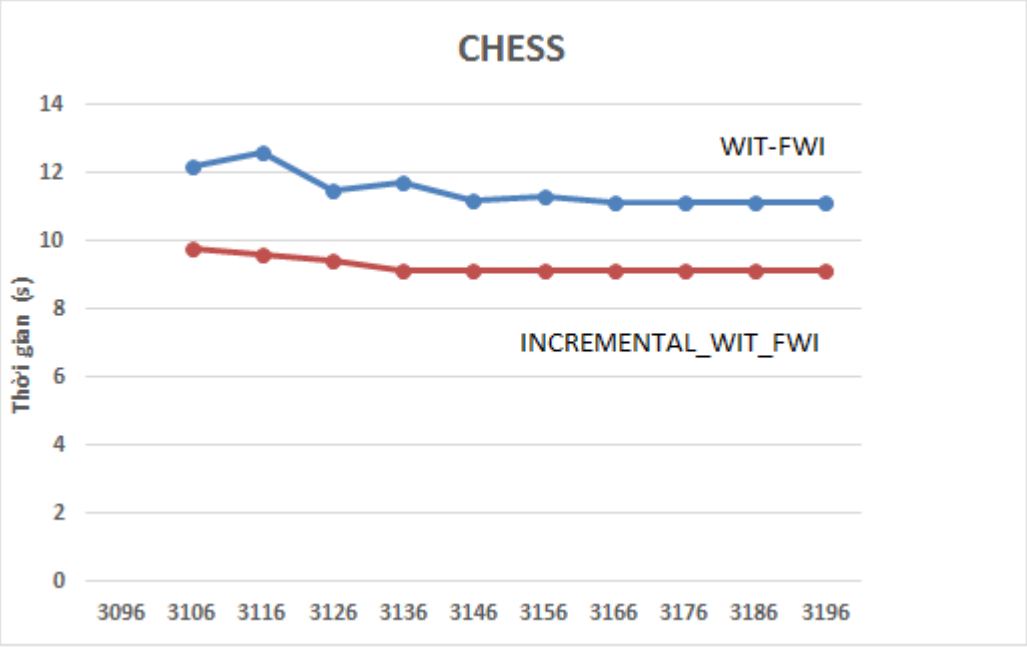
Nội dung thực nghiệm sẽ sử dụng hai thuật toán WIT-FWI và thuật toán tăng trưởng INCREMENTAL_WIT_FWI cùng chạy trên bộ dữ liệu chuẩn và so sánh thời gian thực hiện của hai thuật toán để tìm thuật toán tối ưu hơn trong khai thác dữ liệu tăng trưởng.

3.3 Kết quả thực nghiệm

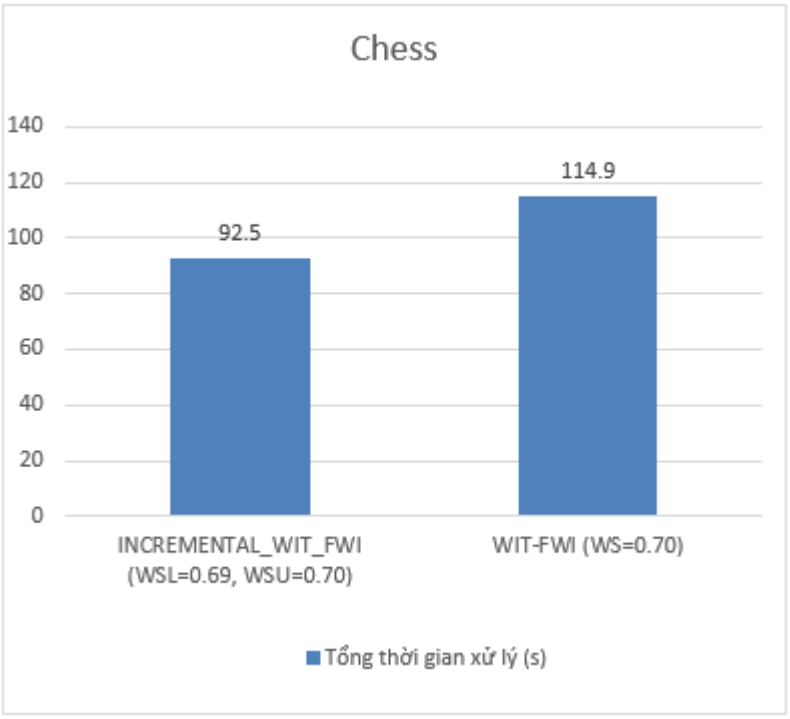
Thời gian thực thi chương trình cho việc tìm kiếm các tập phổ biến được đánh trọng số khác nhau tùy theo từng ngưỡng trọng số hỗ trợ, nếu ngưỡng trọng số hỗ trợ càng thấp thời gian thực thi càng lâu, kết quả tìm kiếm tập FWI được tìm thấy càng nhiều

Khoảng cách giữa ngưỡng trọng số hỗ trợ WS_L và WS_U càng xa thì khả năng quét lại toàn bộ dữ liệu ban đầu càng thấp, tuy nhiên việc này có thể tốn nhiều bộ nhớ và làm chậm thời gian tìm kiếm tập FWI, vì vậy việc chọn ngưỡng trọng số hỗ trợ sao cho phù hợp với dữ liệu là điều quan trọng và sẽ giúp giảm đáng kể thời gian khai thác.

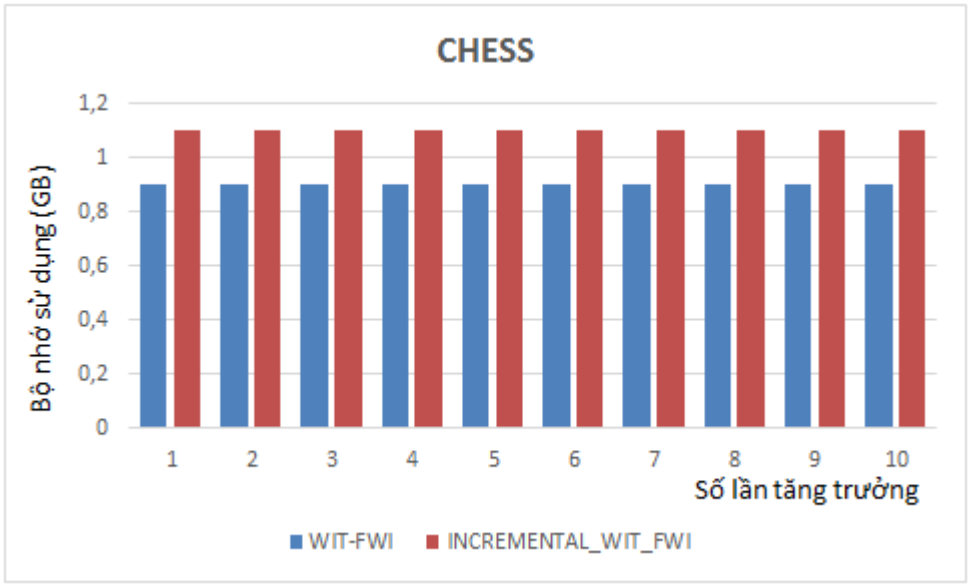
Tiến hành thực nghiệm với dữ liệu Chess, dữ liệu Chess gồm có 3196 giao dịch, được chia làm hai phần, phần 1 gồm 3096 giao dịch làm dữ liệu gốc, phần 2 gồm có 100 giao dịch được chia điều cho 10 lần tăng trưởng (mỗi tăng trưởng 10 giao dịch), kết quả sau 10 lần tăng trưởng tìm thấy được 50649 tập FWI. Ở cả hai thuật toán đều cho ra kết quả như nhau nhưng khác nhau về thời gian thực thi tìm kiếm.



Hình 3.1: Thời gian thực thi trên dữ liệu Chess

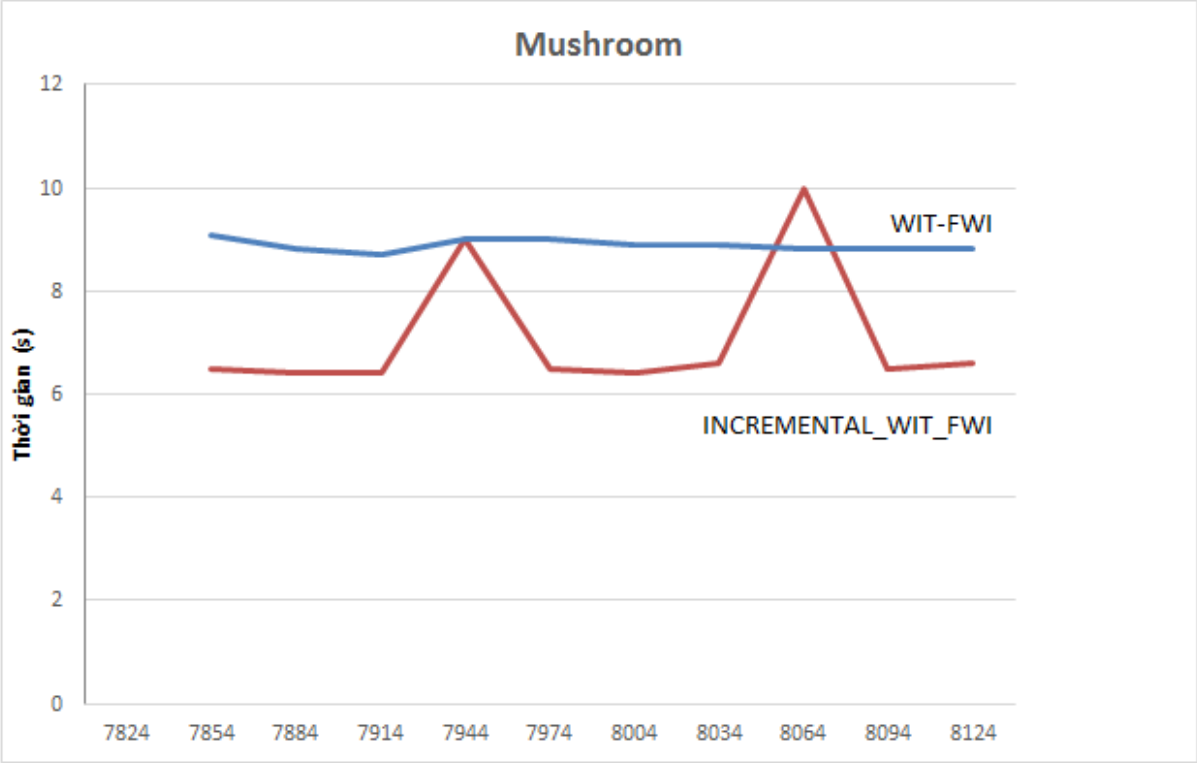


Hinh 3.2: Tổng thời gian thực hiện trên dữ liệu Chess

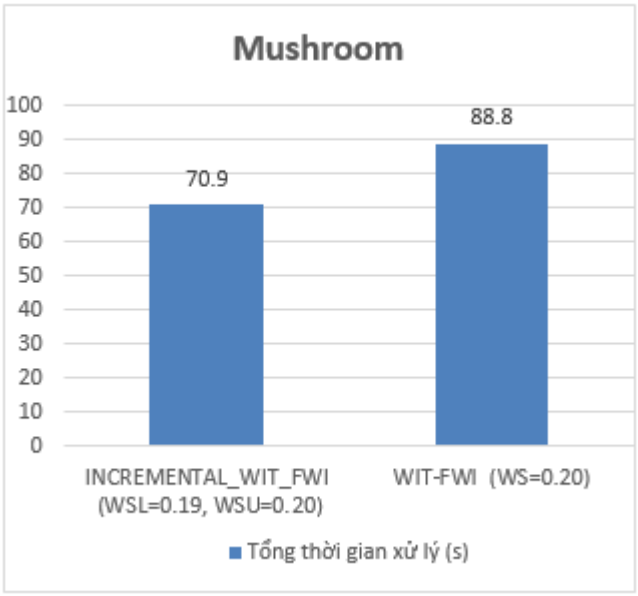


Hinh 3.3: Bộ nhớ sử dụng khi chạy dữ liệu Chess

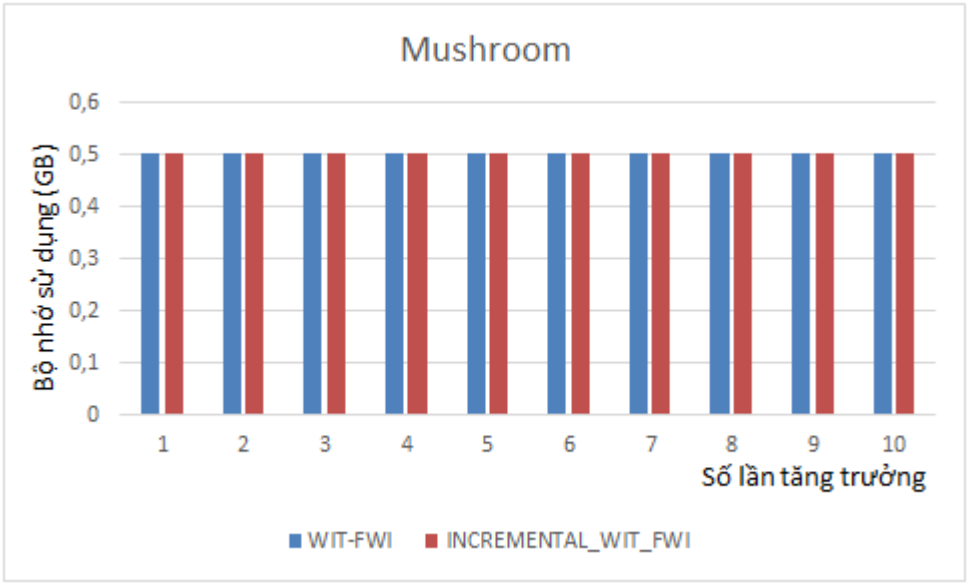
Tiếp đến ta tiến hành thực nghiệm với dữ liệu Mushroom, dữ liệu Mushroom gồm có 8124 giao dịch, được chia làm hai phần, phần 1 gồm 7624 giao dịch làm dữ liệu gốc, phần 2 gồm có 500 giao dịch được chia điều cho 10 lần tăng trưởng (mỗi tăng trưởng 50 giao dịch) , kết quả sau 10 lần tăng trưởng tìm thấy được 54115 tập FWI, ở cả hai thuật toán WIT-FWI và thuật tóa tăng trưởng INCREMENTAL_WIT_FWI đều cho ra kết quả như nhau nhưng khác nhau về thời gian thực thi tìm kiếm.



Hinh 3.4: Thời gian thực thi trên dữ liệu Mushroom

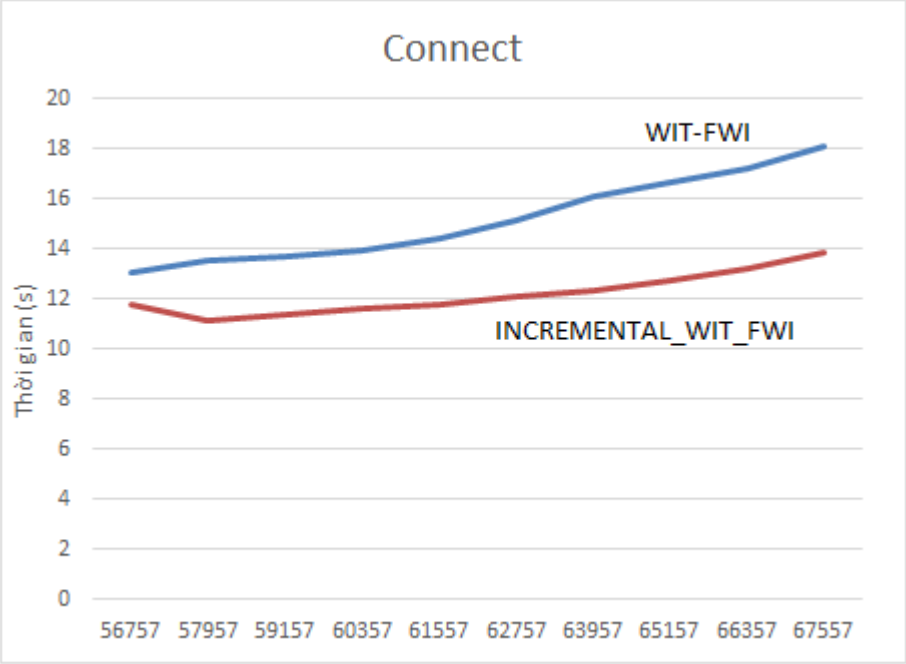


Hinh 3.5: Tổng thời gian thực hiện trên dữ liệu Mushroom

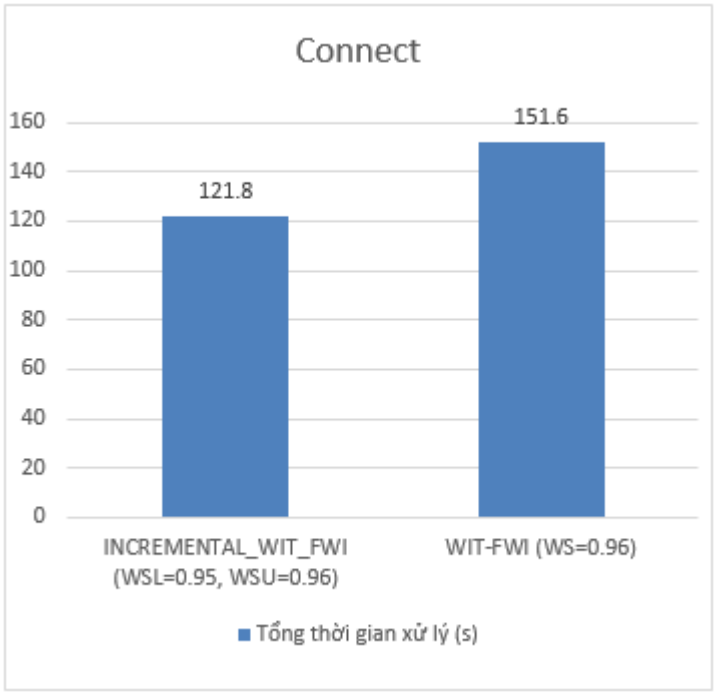


Hinh 3.6: Bộ nhớ sử dụng khi chạy dữ liệu Mushroom

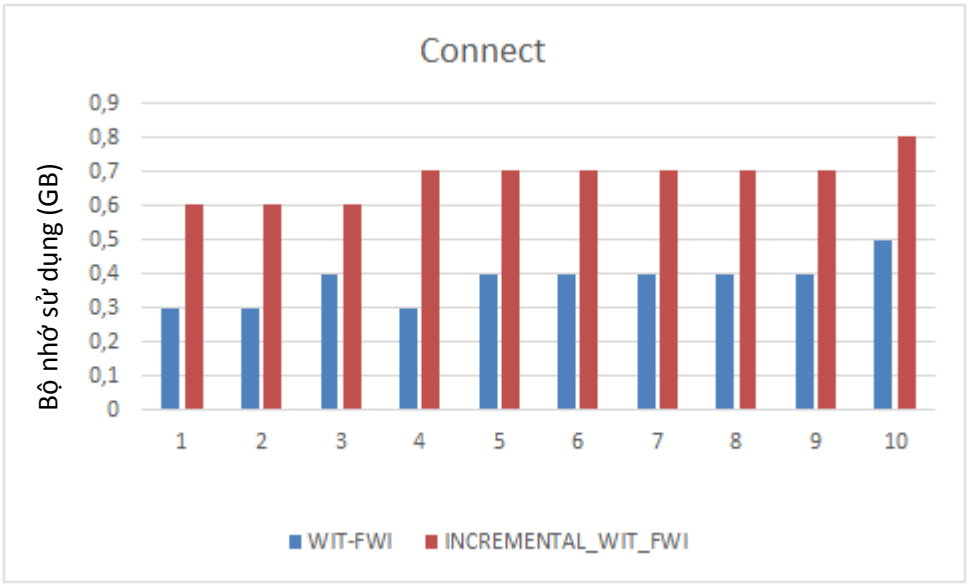
Kế tiếp ta tiến hành thực nghiệm với dữ liệu Connect, dữ liệu Connect gồm có 67557 giao dịch, được chia làm hai phần, phần 1 gồm 55557 giao dịch làm dữ liệu gốc, phần 2 gồm có 1200 giao dịch được chia điều cho 10 lần tăng trưởng (mỗi tăng trưởng 1200 giao dịch) , kết quả sau 10 lần tăng trưởng tìm thấy được 1077 tập FWI, ở cả hai thuật toán WIT-FWI và thuật tóa tăng trưởng INCREMENTAL_WIT_FWI đều cho ra kết quả như nhau nhưng khác nhau về thời gian thực thi tìm kiếm.



Hình 3.7: Thời gian thực thi trên dữ liệu Connect



Hinh 3.8: Tổng thời gian thực thi trên dữ liệu Connect



Hinh 3.9: Bộ nhớ sử dụng khi chạy dữ liệu Connect

Từ những kết quả thực nghiệm ở trên ta có thể thấy thời gian xử lý cho từng dữ liệu khác nhau cần có thời gian xử lý khác nhau, tùy thuộc vào ngưỡng trọng số hỗ trợ do người dùng đặt ra và độ lớn của dữ liệu cần khai thác, tổng số thời gian thực hiện cho 10 lần tăng trưởng, thuật toán tăng trưởng INCREMENTAL_WIT_FWI luôn cho thời gian thực thi nhanh hơn thuật toán WIT-FWI.

Thuật toán tăng trưởng INCREMENTAL_WIT_FWI thực sự mang lại hiệu quả đáng kể trong khai thác dữ liệu tăng trưởng, giúp rút ngắn thời gian khai thác và làm giảm đáng kể số lần phải quét lại toàn bộ dữ liệu, khi khai thác trên cơ sở dữ liệu lớn, lúc này ngưỡng an toàn là rất lớn, trong khi dữ liệu tăng trưởng trong thực tế thường rất nhỏ và thường không vượt quá ngưỡng an toàn. Vì vậy ứng dụng thuật toán tăng trưởng INCREMENTAL_WIT_FWI sẽ mang lại hiệu quả đáng kể cho việc khai thác tập phổ biến được đánh trọng số trên cơ sở dữ liệu tăng trưởng.

PHẦN KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Kết Luận

Đề tài này tập trung nghiên cứu khai thác tập phổ biến được đánh trọng số trên dữ liệu tăng trưởng, đề xuất một thuật toán hiệu quả để khai thác dữ liệu tăng trưởng và duy trì cây WIT-FWI dựa trên khái niệm pre-large. Thông qua quá trình thực hiện đề tài tôi đã thực hiện được các mục tiêu:

- Nghiên cứu cơ sở lý thuyết về các kỹ thuật khai thác tập phổ biến như phương pháp Apriori, IT-tree, WIT-tree.
- Tìm hiểu cơ sở dữ liệu giao dịch có trọng số, trọng số hỗ trợ và các lý thuyết có liên quan.
- Nghiên cứu các thuật toán khai thác các tập phổ biến trên cơ sở dữ liệu giao dịch có trọng số WIT-FWI, WIT-FWI-MODIFY, WIT-FWI-DIF.
- Cài đặt thực nghiệm để khảo sát kết quả của thuật toán đề xuất: tiến hành khai thác tập phổ biến được đánh trọng số trên các cơ sở dữ liệu chuẩn như Chess, Mushroom, Connect.

Từ đó đề xuất thuật toán INCREMENTAL_WIT_FWI để khai thác tập phổ biến được đánh trọng số trên cơ sở dữ liệu tăng trưởng, áp dụng khái niệm pre-large trong khai thác dữ liệu tăng trưởng, giúp hạn chế việc phải quét lại toàn bộ dữ liệu ban đầu khi có dữ liệu mới được thêm vào làm cho việc khai thác được xử lý nhanh hơn. Với thuật toán đề xuất này đã mang lại hiệu quả đáng kể trong khai thác tập phổ biến trên cơ sở dữ liệu tăng trưởng. Từ đó ứng dụng thuật toán này vào trong thực tiễn.

Nhận xét ưu điểm và hạn chế

Ưu điểm:

Trong khai thác dữ liệu tăng trưởng, dữ liệu mới được thêm vào là thường xuyên, mỗi khi có dữ liệu mới được thêm vào thì cần phải quét lại toàn bộ dữ liệu để cập nhật lại trọng số hỗ trợ, việc phải quét lại dữ liệu ban đầu làm tốn khá nhiều thời gian, thuật toán

được đề xuất trong đề tài này đã khắc phục được nhược điểm này, và làm tăng tốc độ xử lý.

Hạn chế:

Bên cạnh những ưu điểm thuật toán cũng tồn tại những mặt hạn chế, thuật toán cần nhiều bộ nhớ để lưu trữ các thông tin phục vụ cho việc tính toán nhanh trọng số hỗ trợ, khi khai thác trên cơ sở dữ liệu lớn cần phải tốn lượng lớn bộ nhớ cho việc lưu trữ.

Lần đầu chạy thuật toán cần nhiều thời gian cho việc xây dựng cấu trúc lưu trữ dữ liệu phục vụ cho việc tính toán nhanh khi dữ liệu tăng trưởng.

Người dùng cần chọn ngưỡng trọng số hỗ trợ phù hợp với dữ liệu cần khai thác.

Hướng phát triển

Tiếp tục nghiên cứu cách khai thác tập phổ biến được đánh trọng số trên cơ sở dữ liệu tăng trưởng một cách hiệu quả hơn.

Nghiên cứu áp dụng kỹ thuật Diffset trong khai thác tăng trưởng.

Nghiên cứu cải tiến thuật toán theo hướng giảm thời gian tính toán.

Nghiên cứu cải tiến thuật toán theo hướng giảm bộ nhớ lưu trữ cho việc tính toán.

TÀI LIỆU THAM KHẢO

Tài liệu tiếng việt

[1] Nguyễn Xuân Huy, Đoàn Văn Ban, Nguyễn Huy Trọng, Huỳnh Văn Đức (2007). Thuật toán khai thác dữ liệu tăng trưởng. Tạp chí khoa học và công nghệ, Tập 45, Số 2 (9-18).

[2] Mai Ngoc Thu (2015), Khai thác TOP-RANK K cho tập đánh trọng trên cơ sở dữ liệu có trọng số. Trường Đại Học Công Nghệ TP.HCM. (1-64)

Tài liệu tiếng anh

[3] B. Vo, F. Coenen, B. Le (2013). A new method for mining Frequent Weighted Itemsets based on WIT-trees. Expert Systems with Applications 40:1256–1264.

[4] B. Vo, F. Coenen, B. Le (2014). An effective approach for maintenance of pre-large-based frequent-itemset lattice in incremental mining, Appl Intell (2014) 41:759–775.

[5] T.P. Hong, C.Y. Wang, Y.H. Tao (2001) A new incremental data mining algorithm using pre-large itemsets. Int Data Anal 5(2):111–129.

[6] B. Vo, B. Le (2009). Mining traditional association rules using frequent itemsets lattice. In: CIE'09 (1401–1406).

[7] G.D. Ramkumar, S. Ranka, S. Tsur (1998) Weigh ted Association Rules Model and Algorithm. In: SIGKDD'98 (661–666).

[8] F. Tao, F. Murtagh, M. Farid (2003). Weighted Association Rule Mining using Weighted Support and Significance Framework. In: SIGKDD'03 (661-666).

[9] T.P. Hong, H.Y. Chen, S.T. Li. (2008) Incrementally Fast Updated Sequential Pattern Trees. (3991-3996).

[10] R. Agrawal, T. Imielinski, A. Swami (1993). Mining Association Rule between sets of items in large databases. ACM SIGMOD Record 22 (2) (207-216).

[11] M.J. Zaki, S. Parthasarathy, M. Ogihara, W. Li. (1997). New algorithms for fast discovery of association rules. In KDD97 (pp. 283-286).

[12] R. Agrawal, R. Srikant (1994). Fast algorithms for mining association rules. In: VLDB’94 (pp. 487–499).