BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÀNH PHỐ HỒ CHÍ MINH

NGUYỄN TẤN PHÚC

NGHIÊN CỨU PHƯƠNG ÁN TỈA ỨNG VIÊN TRONG KHAI THÁC TẬP HỮU ÍCH CAO

LUẬN VĂN THẠC SĨ

Chuyên ngành: Công nghệ thông tin

Mã ngành: 60480201

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

TP. HỒ CHÍ MINH – Tháng 12 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
(Ghi rõ họ, tên, học hàm, học vị và chữ ký)

Luận văn Thạc sĩ được bảo vệ tại Trường Đại học Công nghệ TP. HCM ngày 17 tháng 12 năm 2016

Thành phần Hội đồng đánh giá Luận văn Thạc sĩ gồm:
(Ghi rõ họ, tên, học hàm, học vị của Hội đồng chấm bảo vệ Luận văn Thạc sĩ)

TT	Họ và tên	Chức danh Hội đồng
1	GS. TS. Phan Thị Tươi	Chủ tịch
2	TS. Phạm Thị Thiết	Phản biện 1
3	TS. Trần Đức Khánh	Phản biện 2
4	TS. Nguyễn Thị Thúy Loan	Ủy viên
5	TS. Cao Tùng Anh	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..... tháng..... năm 20.....

NHIỆM VỤ LUẬN VĂN THẠC SĨ

Họ tên học viên: **Nguyễn Tấn Phúc**.....Giới tính: **Nam**.....
Ngày, tháng, năm sinh: **21/10/1982**.....Nơi sinh: **Khánh Hòa** ...
Chuyên ngành: **Công nghệ thông tin**.....MSHV: **1441860034** .....

I- Tên đề tài:
NGHIÊN CỨU PHƯƠNG ÁN TỈA ỨNG VIÊN TRONG KHAI THÁC TẬP HỮU ÍCH CAO.

II- Nhiệm vụ và nội dung:
Nghiên cứu các thuật toán về khai thác tập hữu ích cao, tập trung tìm hiểu các phương pháp thực nghiệm từ các bài báo tham khảo. Tìm hiểu và đánh giá các thuật toán khai thác tập hữu ích cao từ đó phát triển thuật toán mới hiệu quả hơn.

III- Ngày giao nhiệm vụ: 23/01/2016
IV- Ngày hoàn thành nhiệm vụ: 30/12/2016
V- Cán bộ hướng dẫn: PGS.TS Võ Đình Bảy

CÁN BỘ HƯỚNG DẪN
(Họ tên và chữ ký)

KHOA CÔNG NGHỆ THÔNG TIN
(Họ tên và chữ ký)

LỜI CAM ĐOAN

Tôi xin cam đoan rằng luận văn "Nghiên cứu phương án tỉa trong khai thác tập hữu ích cao" 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 không có sản phẩm/ nghiên cứu nào của người khác được sử dụng trong luận văn này mà không được trích dẫn theo đúng quy định.

Học viên thực hiện Luận văn

Nguyễn Tấn Phúc

LỜI CÁM ƠN

Lời đầu tiên tôi xin gửi lời cảm ơn chân thành và biết ơn sâu sắc tới PGS.TS. Võ Đình Bảy – Trường Đại học Công nghệ TP. Hồ Chí Minh đã tận tình chỉ bảo và hướng dẫn tôi trong suốt quá trình nghiên cứu khoa học và thực hiện luận văn này.

Tôi xin chân thành cảm ơn sự dạy bảo, giúp đỡ, tạo điều kiện và khuyến khích tôi trong quá trình học tập và nghiên cứu của các thầy cô giáo, cán bộ quản lý của Trường Đại học Công nghệ TP. Hồ Chí Minh.

Tôi cũng xin chân thành cám ơn Trường Cao đẳng Sư phạm Nha Trang, nay là Đại học Khánh Hòa đã tạo điều kiện về thời gian và công tác để tôi tham gia và hoàn thành khóa đào tạo chương trình Cao học này.

Và cuối cùng, tôi xin gửi lời cảm ơn tới gia đình, người thân và bạn bè - những người luôn ở bên tôi những lúc khó khăn nhất, luôn động viên tôi, khuyến khích tôi trong cuộc sống và trong công việc.

Tôi xin chân thành cảm ơn!

Tp. Hồ Chí Minh, ngày 30 tháng 12 năm 2016

Tác giả

Nguyễn Tấn Phúc

TÓM TẮT

Trong khi khai thác tập phổ biến chỉ quan tâm đến sự xuất hiện của các sản phẩm trong giao dịch (Nghĩa là chúng có hay không có trong các giao dịch) thì khai thác tập hữu ích cao (HUI - High utility itemset) lại quan tâm đến lợi nhuận thu được khi bán các sản phẩm cùng nhau. Đã có nhiều thuật toán được phát triển nhằm nâng cao hiệu quả khai thác HUI, trong đó EFIM là thuật toán mới nhất áp dụng nhiều kỹ thuật để cải thiện tốc độ và không gian tìm kiếm. Tuy nhiên, EFIM vẫn còn tốn nhiều chi phí để quét các dòng dữ liệu để xác định sự liên quan đến ứng viên đang xét làm giảm hiệu quả của thuật toán, đặc biệt là đối với cơ sở dữ liệu thưa. Trong luận văn này, tác giả đề xuất giải pháp chiếu ngược *P-set* để giảm số lượng giao dịch cần xét trong thuật toán *i*EFIM (thuật toán cải tiến của EFIM) và vì vậy, làm giảm thời gian khai thác HUI. Kết quả thực nghiệm cho thấy thuật toán *i*EFIM cải tiến giảm số lượng giao dịch tham gia nhiều lần và đẩy nhanh tốc độ thuật toán đối với loại dữ liệu thưa.

ABSTRACT

Mining frequent itemsets is only interested in the sets of items that appear in transactions but mining High utility itemsets (HUIs) is interested in profits when selling the sets of items. There have been many developed algorithms for mining HUIs, where EFIM is the latest algorithm which applies several techniques to improve the speed and the search space. However, EFIM still spends a lot of transaction scans to determine relevance candidates, it leads to reduce the efficiency, especially for the sparse databases. This thesis proposes a reverse projection solution *P-set* to reduce the number of transaction scans. An efficient algorithm, named *i*EFIM, has been proposed. Experimental results show that *i*EFIM reduces the number of transactions involves and speeds up several times in sparse databases.

MỤC LỤC

DANH MỤC CÁC CHỮ VIẾT TẮT

Ký hiệu, viết tắt	Ý nghĩa tiếng Việt	Ý nghĩa tiếng Anh
CSDL	Cơ sở dữ liệu	Database
DM	Khai thác dữ liệu	Data Mining
FIM	Tập phổ biến	Frequent Itemset Mining
HTWUI	Tập hữu ích có trọng số giao dịch hữu ích cao	High Transaction – Weighted Utilization Itemset
HUI	Tập hữu ích cao	High-utility itemset
Itemset	Tập mục (gọi tắt là tập)	Itemset
KDD	Khám phá tri thức trong cơ sở dữ liệu	Knowledge Discovery in Databases
k-itemsets	Tập chứa k phần tử	k-itemset
minutil	Ngưỡng hữu ích tối thiểu	Minimum utility
TWDC	Tính chất Bao đóng giảm theo trọng số giao dịch	Transaction – Weighted Downward Closure - TWDC
TWU	Trọng số giao dịch hữu ích	Transaction Weighted Utilization
WIT-Tree	Cây WIT	WIT-Tree

DANH MỤC HÌNH ẢNH

DANH MỤC CÁC BẢNG BIỂU

CHƯƠNG 1. GIỚI THIỆU TỔNG QUAN

Chương đầu tiên sẽ trình bày sơ lược về khai thác dữ liệu, cụ thể là khai thác tập hữu ích cao, những vấn đề gặp phải và các giải pháp đang được sử dụng trong thực tế. Bên cạnh đó, mục tiêu và nội dung của luận văn cũng sẽ được tóm tắt trong chương này.

1.1. Giới thiệu

Chúng ta đang sống trong thời kỳ phát triển khoa học kỹ thuật ngày càng vượt bậc, đặc biệt là lĩnh vực công nghệ thông tin và truyền thông. Với sự phổ biến của Internet và các công cụ kết nối khác, tốc độ bùng nổ nguồn dữ liệu từ các các tổ chức, tập đoàn và từ xã hội ngày càng tăng nhanh.

Thông thường dữ liệu thô được lưu trữ trong CSDL ít khi được sử dụng trực tiếp. Trong thực tế, dữ liệu được biểu diễn theo khuôn dạng của người sử dụng và phù hợp với công việc của họ. Với tập dữ liệu nhỏ, có thể sử dụng các phương pháp thống kê hoặc các công cụ quản trị để phân tích. Tuy nhiên với khối lượng dữ liệu khổng lồ và tốc độ gia tăng nhanh, các công cụ tìm kiếm và phân tích truyền thống không còn khả năng đáp ứng, cần phải có công cụ phân tích tự động mới có thể thực hiện được.

Khám phá tri thức trong cơ sở dữ liệu (Knowledge Discovery in Databases - KDD) là lĩnh vực khoa học mới để giải quyết vấn đề nói trên. Khai thác dữ liệu (Data Mining - DM) là một công đoạn chính trong quá trình khám phá tri thức, nhằm tìm kiếm, phát hiện các tri thức mới, những thông tin hữu ích, tiềm ẩn trong nguồn dữ liệu lớn. Hệ thống KDD/DM không phải là hệ thống phần mềm tổng quát. Chúng được phát triển phụ thuộc vào yêu cầu của người sử dụng để giúp họ tự động phân tích dữ liệu đã được xác định, theo các lĩnh vực ứng dụng riêng biệt.

Khai thác tập phổ biến (FIM - Frequent Itemset Mining) được Agrawal giới thiệu vào năm 1993 khi phân tích mô hình dữ liệu siêu thị [1] và làm cơ sở để mở

rộng thành các bài toán khác trong lĩnh vực khai thác dữ liệu.

Khai thác tập hữu ích cao (high-utility itemset - HUI) là một mở rộng của bài toán khai thác tập phổ biến. Với khai thác tập hữu ích cao, giá trị được sử dụng là lợi ích của một tập mục (itemset, được gọi tắt là tập), ví dụ như tổng lợi nhuận mà doanh nghiệp thu được nếu bán tập đó. Khác với khai thác tập phổ biến, lợi ích của tập không thỏa tính chất bao đóng giảm nên độ phức tạp của bài toán cao

1.2. Lý do chọn đề tài

Đối với khai thác tập phổ biến, các thuật toán sinh ứng viên áp dụng tính chất bao đóng giảm (downward closure property) [2], tăng khả năng tỉa các tập ứng viên thừa. Cụ thể, nếu có một tập không phổ biến m thì thuật toán không xét các tập ứng viên chứa tập m, giả sử bộ dữ liệu chứa n phần tử và tập không phổ biến m chứa k phần tử, thuật toán sẽ không xét $2^{(n-k)}$-2 tập có chứa m.

Sự khác biệt của khai thác tập phổ biến và tập hữu ích cao được minh họa cụ thể qua ví dụ ở bảng 1 với 7 phần tử (item) và 5 giao dịch với hữu ích thể hiện ở dòng Utility. Khi tính độ hỗ trợ, thuật toán tìm tập phổ biến chỉ sự dụng 02 cột đầu của bảng Giao dịch (b) và bỏ qua thông tin hữu ích ở bảng lợi nhuận (a), ... [3]

Bảng 1.1. Dữ liệu bán hàng

Item	a	b	c	d	e	f	g
Lợi nhuận	1	2	1	5	4	3	1

(a) Bảng lợi nhuận

Tid	**Giao dịch**	**Số lượng**
T1	{b,c,d,g}	{1,2,1,1}
T2	{a,b,c,d,e}	{4,1,3,1,1}
T3	{a,c,d}	{4,2,1}
T4	{a,b,d,e}	{5,2,1,2}
T5	{a,b,c,f}	{3,4,1,2}

(b) Bảng giao dịch

Tuy nhiên tập có độ phổ biến cao thì chưa chắc có giá trị hữu ích cao. Ví dụ

với độ phổ biến của *{bc}* là 3, hữu ích là 18, trong khi *{de}* có giá trị lần lượt là 2 và 22. Trong chiến lược kinh doanh, người ta quan tâm nhiều đến vấn đề giá cả/lợi nhuận. Nhưng, thuật toán khai thác tập phổ biến truyền thống cho các tập không chứa thông tin đó.

Giống như tập phổ biến, một tập được gọi là tập hữu ích cao khi giá trị hữu ích của tập đó vượt qua một ngưỡng tối thiểu nhất định do người dùng đưa ra. Khi thực hiện theo tính chất bao đóng giảm, độ hỗ trợ của tập ứng viên mới không lớn hơn tập phổ biến sinh ra nó. Nhưng với tập hữu ích cao không tuân theo quy luật đó, cụ thể: các tập *{a}, {ab}, {abc}* có độ hỗ trợ lần lượt là 4,2,1 nhưng giá trị hữu ích thì 16, 26, 21. Nếu lấy ngưỡng là 20 thì ta chọn *{ab}, {abc}* và loại *{a}*. Vì thế không thể áp dụng thuật toán tìm tập phổ biến trong trường hợp này.

Chính vì vậy tôi chọn đề tài **"Nghiên cứu phương pháp tỉa ứng viên trong khai thác tập mục hữu ích cao"** nhằm góp phần rút ngắn thời gian cũng như bộ nhớ sử dụng trong quá trình khai thác.

1.3. Mục tiêu của đề tài

Từ khi bài toán được Yao và các đồng sự phát biểu vào năm 2004 [4] đến nay, đã có nhiều thuật toán khai thác tập hữu ích cao được phát triển nhằm nâng cao hiệu quả khai thác: UMining (2004) [4], UMining-H (2006) [5], Two-Phase (2005) [6], IHUP (2009) [7], TWU-Mining (2009) [8], UP-Growth (2010) [9], DTWU-Mining (2011) [10], EFIM (2015) [11], ... và một số hướng phát triển khác của tập hữu ích cao, điển hình như khai thác tập đóng có CHUD (2011) [12], AprioriCH, AprioriHC-D (2015) [13]; khai thác Top-k HUI có TKU (2012) [14], TKO (2016) [15]; khai thác HUI trên luồng dữ liệu có THUI-Mine (2008) [16], GUIDE (2012) [17], hay khai thác HUI trên dữ liệu không chắc chắn [18].

Hiện nay thuật toán EFIM được xem là thuật toán nhanh nhất. Vì vậy nghiên cứu các ưu, khuyết điểm của EFIM và cải tiến thuật toán là một thách thức của khai thác tập hữu ích cao.

1.4. Nội dung chính cần nghiên cứu

Luận văn tập trung nghiên cứu về bài toán khai thác tập hữu ích cao và nghiên cứu các thuật toán đã nêu trên. Từ đó, rút ra những điểm mạnh của các thuật toán để cải tiến thuật toán EFIM.

1.5. Kết luận

Dựa vào những tìm hiểu đã trình bày, luận văn nghiên cứu giải pháp *P-set* và ứng dụng *P-set* để cải tiến thuật toán EFIM trong khai thác tập hữu ích cao. Giải pháp này, giảm đáng kể số lượng giao dịch cần xử lý vì thế giảm được không gian tìm tập hữu ích cao. Thực nghiệm cho thấy giải pháp hiệu quả nhiều lần với những CSDL được đánh giá thưa.

Luận văn tổ chức thành 5 chương có nội dung như sau:

Chương 1: Giới thiệu tổng quan về bối cảnh thực tiễn và định hình hướng nghiên cứu của luận văn.

Chương 2: Trình bày tổng quan về bài toán khai thác tập hữu ích cao. Chương này cũng đề cập đến các hướng tiếp cận bài toán và các thuật toán liên quan. Phần cuối chương này trình bày các nghiên cứu và cơ sở lý thuyết liên quan đến kết quả luận văn.

Chương 3: Tập trung vào bài toán "nghiên cứu phương án tỉa ứng viên trong bài toán khai thác tập hữu ích cao" với đề xuất thuật toán *i*EFIM được cải tiến từ thuật toán EFIM.

Chương 4: Giới thiệu môi trường thực nghiệm, CSDL dùng để thực nghiệm.

Chương 5: Kết luận và hướng phát triển.

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT VÀ MỘT SỐ THUẬT TOÁN KHAI THÁC TẬP HỮU ÍCH CAO

Chương này trình bày bài toán khai thác tập hữu ích cao, các khái niện cơ sở và các thuật toán khai thác tập hữu ích cao tiêu biểu

2.1. Giới thiệu bài toán khai thác tập mục hữu ích cao

2.1.1. Định nghĩa bài toán

Cho tập *I* gồm n phần tử i_1, i_2, ...i_n là tập các phần tử và cơ sở dữ liệu *D* gồm bảng hữu ích (Utility table) và bảng giao dịch (Transaction table) (như Bảng 1.1). Mỗi phần tử trong *I* có giá trị hữu ích nhất định chứa trong bảng hữu ích được dùng chung cho toàn bộ CSDL. Một giao dịch *T* trong bảng giao dịch được xác định duy nhất bằng *Tid* và chứa tập con của *I* có liên kết với số lượng tương ứng. Tập con đó chứa k phần tử được gọi là tập k phần tử (k-itemsets) [3].

Định nghĩa 2.1. Giá trị hữu ích mở rộng của phần tử *i*, ký hiệu *eu(i),* là giá trị hữu ích của i trong bảng hữu ích của *D*. [3]

Định nghĩa 2.2. Giá trị hữu ích nội bộ của phần tử *i* trong giao dịch *T*, ký hiệu *iu(i,T),* là số đếm giá trị kết hợp của phần tử *i* thuộc *T* trong bảng giao dịch của *D*. [3]

Định nghĩa 2.3. Giá trị hữu ích của phần tử *i* trong giao dịch *T*, ký hiệu *u(i,T),* là phép nhân giữa *iu(i,T)* và *eu(i)* hay *u(i,T) = iu(i,T) × eu(i).* [3]

Ví dụ: theo dữ liệu tại Bảng 1 ta có *eu(e)* = 4, *iu(e,T4)* = 2 và *u(e,T4) = iu(e,T4) × eu(e)* = 2 × 4 = 8

Định nghĩa 2.4. Giá trị hữu ích của tập *X* trong giao dịch *T*, ký hiệu *u(X,T),* là tổng giá trị hữu ích của các phần tử thuộc *X* có trong giao dịch *T* hay $u(X,T) = \sum_{i \in X \wedge X \subseteq T} u(i,T)$ [3]

Định nghĩa 2.5. Giá trị hữu ích của tập *X*, ký hiệu *u(X)*, là tổng giá trị hữu ích của *X* trong tất cả giao dịch *T* có chứa *X* trên *D* hay $u(X) = \sum_{T \in DB \wedge X \subseteq T} u(T)$ [3]

Ví dụ: $u(\{ae\}, T2) = u(a, T2) + u(e, T2) = 4 \times 1 + 1 \times 4 = 8$, và $u(\{ae\}) = u(\{ae\}, T2) + u(\{ae\}, T4) = 8 + 13 = 21$.

Định nghĩa 2.6. Một tập được gọi là *tập hữu ích cao* nếu độ hữu ích của nó không nhỏ hơn một ngưỡng hữu ích tối thiểu *minutil* do người dùng đặt ra. Ngược lại, nó được gọi là *tập hữu ích thấp*. [3]

Ví dụ: tập {ae} ở ví dụ trên có độ hữu ích là 21, nếu *minutil = 20* thì {ae} là tập hữu ích cao và ngược lại nếu *minutil = 22* thì {ae} là tập hữu ích thấp.

2.1.2. Phát biểu bài toán

Cho CSDL *D* có mô tả giống như bảng 1.1 trong đó bảng giao dịch là chi tiết hoá đơn của một hoá đơn bán hàng và một ngưỡng hữu ích tối thiểu *minutil* do người dùng đặt ra theo mục đích, yêu cầu mà người dùng sẽ đưa ra ngưỡng *minutil* phù hợp, khai thác tập hữu ích cao từ *D* tương đương với việc tìm trong *D* tất cả các tập (*tập sản phẩm*) có độ hữu ích không nhỏ hơn ngưỡng *minutil*.

Sau khi định nghĩa và phát biểu bài toán khai thác tập hữu ích cao, ta đề cập đến các công trình nghiên cứu liên quan và các thuật toán có ảnh hưởng đến kết quả nghiên cứu của luận văn này.

2.2. Các nghiên cứu liên quan

Bài toán khai thác tập hữu ích cao do Yao và Hamilton đề xuất vào năm 2004 [4]. Các tác giả đề xuất thuật toán UMining dựa vào chặn trên (upper bound) của độ hữu ích để khai thác HUI. Sau đó UMining-H, một dạng heuristic của UMining do thay đổi cách tính chặn trên độ hữu ích để tỉa ứng viên. Cả UMining và UMining-H đều có khả năng tỉa nhầm các tập HUI. Năm 2005, Liu và các đồng sự đề xuất một chặn trên mới có tên là trọng số giao dịch hữu ích - TWU (Transaction Weighted Utilization) (được trình bày tại định nghĩa 8 mục 2.3.1) dùng cho khai thác HUI [6]. TWU của các tập thỏa tính chất bao đóng giảm nên có thể dựa vào đó để tỉa ứng viên. Vì vậy, các tác giả đề xuất thuật toán Two-Phase dựa trên TWU để tỉa ứng viên. Two-Phase được chia làm hai giai đoạn được trình bày tại mục 2.3.1.

Sau Two-Phase, hầu hết các thuật toán đều vận dụng phương pháp tỉa dựa trên TWU và áp dụng những chiến lược riêng để nâng cao hiệu quả tỉa ứng viên. TWU-Mining và DTWU-Mining của Le và các đồng sự vận dụng, phát triển cấu trúc IT-Tree của Zaki [19] thành cấu trúc WIT-Tree [8] để giảm số lần duyệt cơ sở dữ liệu. Cùng vận dụng FP-Growth [20], IHUP của Ahmed và các đồng sự đề xuất tạo ứng viên trên IHUP-Tree [7], còn UP-Growth và UP-Growth+ [21] của Tseng và các đồng sự thì thực hiện việc tạo ứng viên trên UP-Tree [9] bên cạnh các chiến lược bổ trợ: giảm độ hữu ích của tập không triển vọng trên UP-Tree toàn cục (DGU - Discarding Global Unpromising item), giảm độ hữu ích của nút trên UP–Tree toàn cục (DGN - Discarding Global Node utilities), loại bỏ tập không triển vọng cục bộ (DLU - Discarding Local Unpromising item), giảm độ hữu ích của nút trên UP–Tree cục bộ (DLN - Decreasing Local Node utilities), giảm độ hữu ích của tập không triển vọng cục bộ trên UP-Tree cục bộ (DNU - Discarding local unpromising items and their estimated Node Utilities) và giảm độ hữu ích của nút không triển vọng cục bộ trong UP-Tree cục bộ (DNN - Decreasing local Node utilities for the nodes of local UP-Tree). Sau khi tạo danh sách ứng viên IHUP, UP-Growth và UP-Growth+ đều quét lại CSDL để tính giá trị hữu ích và xem xét việc ứng viên có phải là tập hữu ích cao hay không.

Với HUI-Miner của Liu và Qu đi theo hướng mới, chỉ duyệt CSDL một lần và lưu vào cấu trúc Utility-list do nhóm đề xuất [3], khai thác và tỉa ứng viên trên cấu trúc đó. Tuy nhiên, số lượng Utility-list do HUI-Miner tạo ra khá nhiều nên Fournier-Viger và các đồng sự đề xuất thuật toán FHM (2014) [22] và cấu trúc EUCS (Estimated Utility Co-occurrence Structure) [22] với phương án tỉa EUCP (Estimated Utility Co-occurrence Pruning) [22] để hạn chế việc tạo Utility-list nhằm tăng tốc độ thuật toán. Cùng mục đích với FHM, HUP-Miner [23] của Krishnamoorthy áp dụng thêm 2 chiến lược tỉa theo phân vùng (PA - PArtitioned utility) [23] và tỉa trước (LA - LookAhead utility) [23] bên cạnh chiến lược tỉa theo Utility-list.

Mỗi thuật toán đều phát huy hiệu quả chiến lược tỉa ứng viên của mình và đẩy nhanh tốc độ tìm kiếm tập hữu ích cao. Tuy nhiên, trong quá trình khai thác, các thuật

toán vẫn quét các giao dịch rỗng và chưa có phương án xử lý các dòng dữ liệu tương đồng với nhau (giống các phần tử xuất hiện trong giao dịch và chỉ khác số lượng). Vì vậy, EFIM đã đề xuất 3 chiến lược: chiếu trên cơ sở dữ liệu (HDP – High utility Database Projection) [11]để tìm kiếm các phần trùng nhau; chiến lược trộn các giao dịch (HTM – High utility Transaction Merging) [11] để giảm không gian tìm kiếm và các phương pháp tỉa bằng các chặn trên theo giá trị hữu ích cục bộ (Local utility) [11] và giá trị hữu ích trên nhánh phụ (Sub-tree utility) [11] để loại các tập ứng viên không mong đợi.

2.3. Các thuật toán khai thác tập hữu ích cao

Trong tiểu mục này đề cập 03 thuật toán có ảnh hưởng lớn đến kết quả nghiên cứu của luận văn là Two-Phase, TWU-Mining và EFIM.

2.3.1. Thuật toán Two-Phase

Định nghĩa 2.7. Giá trị hữu ích của giao dịch *T*, ký hiệu *tu(T)*, là tổng giá trị hữu ích của các phần có trong *T* hay $tu(T) = \sum_{i \in T} ui(i,T)$ và giá trị hữu ích của *D* là tổng giá trị hữu ích các giao dịch trong *D*. [6]

Ví dụ: độ hữu ích cho từng giao dịch như *tu(T1) = u(b,T1) + u(c,T1) + u(d,T1) + u (g,T1)* = 2 + 2 + 5 + 1 = 10, tương tự độ hữu ích các giao dịch trong *D* được minh họa tại Bảng 2.1 và độ hữu ích của *D* là 79.

Bảng 2.1. Giá trị hữu ích trong từng giao dịch

Tid	**T1**	**T2**	**T3**	**T4**	**T5**
tu	10	18	11	22	18

Định nghĩa 2.8. Trọng số giao dịch hữu ích của tập *X*, ký hiệu *twu(X)*, là tổng giá trị hữu ích của tất cả các giao dịch có chứa *X* trên *D* hay $twu(X) = \sum_{T \in DB \wedge X \subseteq T} tu(T)$ [6]

Ví dụ: Trọng số giao dịch hữu ích của *{a}* là *twu({a}) = tu(T2) + tu(T3) + tu(T4) + tu(T5)*= 18 + 11 + 22 + 18 = 69, tương tự trọng số giao dịch hữu ích các

phần tử minh họa tại Bảng 2.2.

Bảng 2.2 Trọng số giao dịch hữu ích các phần tử trên D

Itemset	{a}	{b}	{c}	{d}	{e}	{f}	{g}
twu	69	68	57	61	40	18	10

Tính chất 2.1. Tính chất bao đóng giảm theo Trọng số giao dịch (*Transaction – Weighted Downward Closure - TWDC*). Với mọi tập X, nếu X không phải tập hữu ích có trọng số giao dịch hữu ích cao (*High Transaction – Weighted Utilization Itemset, viết tắt là HTWUI*) thì bất kỳ tập nào chứa X cũng là tập hữu ích thấp.

Thật vậy, Nếu $X \subseteq X'$ thì $u(X') \leq twu(X') \leq twu(X) < mintil$

Theo tính chất này, tính chất bao đóng giảm có thể được duy trì bằng cách sử dụng trọng số giao dịch hữu ích.

Từ Bảng 2.2 mô tả Trọng số giao dịch tất cả các tập có 1 phần tử. Áp dụng tính chất 2.1, xét với ngưỡng *mintil* = 30, ta có thể tỉa bớt không gian tìm kiếm. Trong trường hợp này, $\{f\}$ và $\{g\}$ bị tỉa và không xét thêm các tập chứa $\{f\}$ và $\{g\}$.

Trên cơ sở định nghĩa 2.7, 2.8 và tính chất 2.1, thuật toán Two-Phase được đề xuất bởi Liu cùng các đồng sự, thuật toán này được chia ra hai giai đoạn chính và gọi là phase 1 và phase 2. Two-Phase sinh ra các tập ứng viên độ dài k phần từ từ các ứng viên có độ dài (k - 1) phần tử và tỉa bớt tập ứng viên bằng tính chất TWDC. Mỗi lần lặp, các tập ứng viên và giá trị TWU được tính bằng cách duyệt CSDL. Sau giai đoạn 1, ta thu được toàn bộ tập ứng viên. Và ở giai đoạn 2, độ hữu ích của các tập ứng viên được tính bằng cách duyệt CSDL gốc thêm lần nữa và từ đó xác định tập hữu ích cao.

Phase 1: Thuật toán Two-Phase triển khai chiến lược tìm kiếm theo chiều rộng để liệt kê tất cả các ứng viên. Giá trị hữu ích theo từng giao dịch TU được

tính theo định nghĩa 2.7 (bảng 2.1), sau đó tiến hành tính Trọng số giao dịch hữu ích TWU theo định nghĩa 2.8 (bảng 2.2). Từ kết quả so sánh TWU với ngưỡng *minutil* để sinh ra các ứng viên có thể là tập hữu ích cao theo tính chất 2.1, trong quá trình này CSDL được duyệt m lần ứng với mỗi lần là k phần tử (*tập 1 phần tử, tập 2 phần tử,....*).

Dựa vào TWU được trình bày trong bảng 2.2, ta tiến hành lặp công việc duyệt CSDL tương ứng với việc tìm tập ứng viên chứa k phần từ với k = 1...m. Với ngưỡng *minutil* = 35 ta có các bước thực hiện như sau:

- Bước 1: Với k = 1 (*tập có 1 phần tử hay 1-itemset*)

Bảng 2.3. Two-Phase với tập chứa 1 phần tử

1-itemset	**TWU**	*minutil* = 35	**1-HTWUI**
{a}	69	✓ →	*{a}*
{b}	68	✓	*{b}*
{c}	57	✓	*{c}*
{d}	61	✓	*{d}*
{e}	40	✓	*{e}*
{f}	10		
{g}	18		

- Bước 2: k = 2 (*2-itemset*): Dùng các tập từ 1 HTWUI được sinh ra từ bước 1 để tạo thành 2-itemset.

Bảng 2.4. Two-Phase với tập chứa 2 phần tử

2- itemset	**TWU**	*minutil* = 35	**2-HTWUI**
{ab}	40	✓ →	*{ab}*
{ac}	29		*{ad}*

Bảng 2.4. Two-Phase với tập chứa 2 phần tử

2- itemset	TWU	*minutil* = 35	2-HTWUI
{ad}	51	✓	*{ae}*
{ae}	40	✓	*{bc}*
{bc}	46	✓	*{bd}*
{bd}	50	✓	*{be}*
{be}	40	✓	*{cd}*
{cd}	39	✓	*{de}*
{ce}	18		
{de}	40	✓	

- Bước 3: k = 3 (*3-itemset*): Dùng các tập được sinh ra từ bước 2 để tạo thành 3-itemset.

Bảng 2.5. Two-Phase với tập có 3 phần tử

3-itemset	TWU	*minutil* = 35	3-HTWUI
{abd}	40	✓ →	*{abd}*
{abe}	40	✓	*{abe}*
{ade}	40	✓	*{ade}*
{bcd}	28		*{bde}*
{bce}	18		
{bde}	40	✓	

- Bước 4: k = 4 (*4-itemset*): Dùng các tập được sinh ra từ bước 3 để tạo thành 4-itemset.

Bảng 2.6. Two-Phase với tập có 4 phần tử

4-itemset	**TWU**	*minutil* = 35	**4-HTWUI**
{abde}	40	✓ →	*{abde}*

- Bước 5: k=5 (5-itemset). Dùng các tập mục được sinh ra từ bước 4 để tạo thành 5-itemset. Bước này duyệt CSDL không tạo ra được tập mục thuộc 5-itemset nên thuật toán dừng giai đoạn 1.

Sau khi thực hiện giai đoạn 1, ta tìm được 18 tập ứng viên là *{a}, {b}, {c}, {d}, {e}, {ab}, {ad}, {ae}, {bc}, {bd}, {be}, {cd}, {de}, {abd}, {abe}, {ade}, {bde}, {abde}.*

Phase 2: Ở giai đoạn này, thuật toán duyệt CSDL gốc thêm lần nữa, xác định độ hữu ích của từng tập ứng viên được tạo ra từ phase 1 và so sánh với *minutil* để kết luận tập hữu ích cao. Kết quả có*{abde}* có độ hữu ích là 37 > *minutil =35* nên là tập hữu ích cao.

Tuy nhiên, Two-Phase sinh ra số lượng ứng viên lớn và duyệt CSDL nhiều lần. Để giải quyết vấn đề sinh ra nhiều ứng viên và duyệt CSDL nhiều lần, sau đây là thuật toán TWU-Mining với nền tảng là cấu trúc cây WIT-Tree sẽ làm giảm số lần duyệt CSDL nhưng vẫn hiệu quả.

2.3.2. Thuật toán khai thác tập mục hữu ích cao TWU-Mining

Thuật toán TWU-Mining của Le và các đồng sự đưa ra năm 2009 cũng thực hiện việc khai thác tập hữu ích cao qua 2 giai đoạn giống như Two-Phase. Khác với Two-Phase, TWU-Mining xây dựng cấu trúc WIT-Tree được phát triển từ cấu trúc IT-Tree và thực hiện việc tìm kiếm và sinh các tập ứng viên trên cấu trúc đó, cấu trúc WIT-Tree chi tiết như sau:

Cấu trúc WIT-Tree

Cấu trúc WIT-Tree gồm nhiều nút lên kết với nhau thành cây, trong đó mỗi nút N bao gồm $N.name$, $N.tidset$, $N.twu$, với $N.name$ là tên tập mục của nút

trong cây, $N.tidset$ là tập số thứ tự của các giao dịch chứa nút N, $N.twu$ là trọng số giao dịch hữu ích của tập phần tử lưu trên N. Giá trị của TWU đã được tính chi tiết theo định nghĩa 2.8.

Nguyên tắc phát sinh WIT-Tree

Xây dựng các k-itemset với k có giá trị từ 1 đến m, với m là số phần tử trong CSDL giao dịch. Tại mỗi bước lặp là tạo ra các tập phần tử có các phần tử chung phải cùng xuất hiện trong giao dịch của CSDL. Độ hữu ích của mỗi nút trên WIT-Tree sẽ là trọng số giao dịch hữu ích TWU của tập đó. Cụ thể như sau:

- Bước 1: Với k = 1 (*1-itemset*)

Bảng 2.7. WIT-Tree với tập có 1 phần tử

1-itemset	TWU	Các giao dịch tham gia
{a}	69	2,3,4,5
{b}	68	1,2,4,5
{c}	57	1,2,3,5
{d}	61	1,2,3,4
{e}	40	2,4
{f}	10	5
{g}	18	1

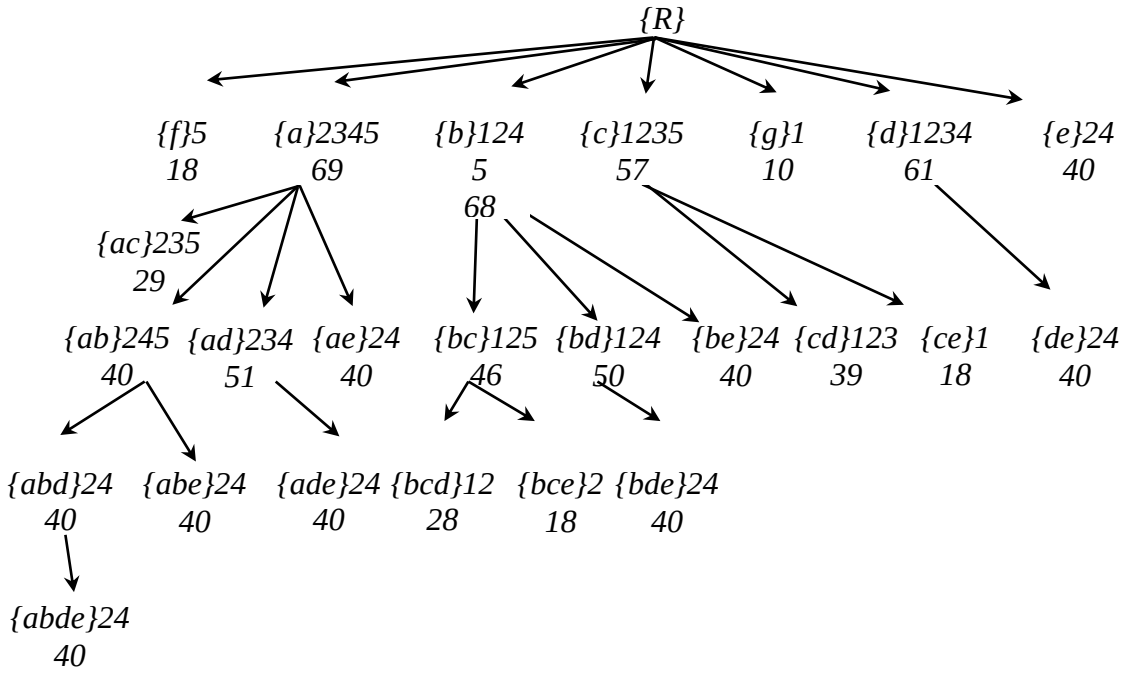
- Bước 2: Với k = 2 (*2-itemset*)

Bảng 2.8. WIT-Tree với tập có 2 phần tử

2-itemset	TWU	Các giao tác tham gia
{ab}	40	2,4,5
{ac}	29	2,3,5
{ad}	51	2,3,4

2-itemset	TWU	Các giao tác tham gia
{ae}	40	2,4
{bc}	46	1,2,5
{bd}	50	1,2,4
{be}	40	2,4
{cd}	39	1,2,3
{ce}	18	1
{de}	40	2,4

- Lặp tương tự tạo ứng viên cho đến khi không thể phát sinh thêm tập ứng viên. Và tạo được cây WIT hoàn chỉnh như hình 2.1



Hình 2.1. Cấu trúc WIT-Tree hoàn chỉnh

Thuật toán TWU-Mining

Thuật toán TWU-Mining được phát triển trên nền tảng WIT-Tree được phát biểu như tại hình 2.2 [8].

Trong Phase 1, thuật toán duyệt CSDL tạo được WIT-Tree, sau đó trong phase 2, duyệt WIT-Tree để phát sinh tập ứng viên và tìm ra các tập HUI thoả ngưỡng *minutil*. Cũng như Two-Phase, TWU-Mining phát sinh được 18 tập ứng viên và tìm được tập hữu ích cao là {abde} nhưng với số lần duyệt CSDL ít hơn.

TWU-Mining()

HUIs = ∅

[∅**]** = {$i \times t(i) \mid i \in I \wedge twu(i) \geq min\_util$}

TWU-Mining-Extend(**[**∅**]** , *min_util*)

TWU-Mining-Extend (**[***P***]**, *min_util*)

// Phase 1

1. for all $l_i \in$ **[***P***]** do
2. **[***P*$_i$**]** = ∅
3. for all $l_j \in$ **[***P***]**, with j > i do
4. $X = l_i \cup l_j$
5. $Y = Tidset(l_i) \cap Tidset(l_j)$
6. if $twu(X) \geq min\_util$ then
7. **[***P*$_i$**]** = **[***P*$_i$**]** $\cup \{\frac{X \times Y}{twu(X)}\}$
8. **TWU-Mining-Extend([***P*$_i$**]**, *min_util*)

// Phase 2

9. foreach itemset *s* in **[***P***]** do
10. If $u(s) \geq min\_util$ then
11. HUIs = HUIs ∪ s

Hình 2.2. Thuật toán TWU-Mining

2.3.3. Thuật toán EFIM

Như đã trình bày ở phần cuối mục 2.2, Zida và các đồng sự đề xuất thuật toán EFIM năm 2015 để khắc phục một số khiếm khuyết của các thuật toán trước đó. EFIM đã đề xuất 3 chiến lược: chiếu trên cơ sở dữ liệu để tìm kiếm các phần trùng

nhau; chiến lược trộn các giao dịch để giảm không gian tìm kiếm và đề xuất các giải pháp tỉa bằng các phép tính chặn trên mới. Các chiến lược và giải pháp của EFIM được khái quát qua các định nghĩa sau:

Định nghĩa 2.9. Cho tập các phần tử *I* được xếp thứ tự theo trọng số giao dịch hữu ích TWU $\succ$, và tập X, tập các phần tử mở rộng của X được định nghĩa như sau $E(X) = \{ z \mid z \in I \wedge z \succ x \forall x \in X \}$ [11].

Định nghĩa 2.10. Cho giao dịch T và tập X, phép chiếu của tập X trên giao dịch T được xác định là $T_X = \{ i \mid i \in T \wedge i \in E(X)\}$ [11].

Ví dụ: cho $X = \{b\}$, xét phép thứ tự $a \succ b \succ c \succ d \succ e$ thì $T1_X = \emptyset$, $T2_X = \{a\}$.

Định nghĩa 2.11. Cho CSDL D và tập X, phép chiếu của tập X trên D được định nghĩa như sau $D_X = \{ T_X \mid T \in D \wedge T_X \neq \emptyset\}$ [11].

Ví dụ: cho $X = \{c\}$, xét phép thứ tự $a \succ b \succ c \succ d \succ e$, $D_X = \{ T1_X, T2_X, T3_X, T5_X\} = \{ \{b\}, \{a, b\}, \{a\}, \{a, b\}\}$

Định nghĩa 2.12. Cho hai giao dịch T_a, T_b chứa các phần tử tương ứng $\{i_1, i_2, \ldots, i_m\}$ và $\{j_1, j_2, \ldots, j_n\}$. T_a và T_b được gọi là đồng nhất hay $T_a = T_b$ nếu thỏa các điều kiện: n = m và $\forall k \in [1, n]$, $i_k = j_k$ [11].

Ví dụ: Xét tiếp ví dụ ở định nghĩa 2.11, thì $T2_X$ và $T5_X$ được xem là đồng nhất vì có cùng kết quả là $\{a, b\}$.

Định nghĩa 2.13. cho các giao dịch đồng nhất $Tr_1 = Tr_2 = \ldots = Tr_m$ trên D, các giao dịch trên được trộn lại bởi giao dịch T_m trong đó $\forall i \in T_m, iu(i, T_{m)} = \sum_{k=1\ldots m} iu(i, T_{k)}$ [11].

Ví dụ: Giả sử $T2_X$ và $T5_X$ ở định nghĩa 2.12 là 2 giao dịch độc lập, thì hai giao dịch này được thay bằng $T2'_X$ có giá trị hữu ích nội bộ $iu(\{a\}, T2'_X) = iu(\{a\}, T2_X) + iu(\{a\}, T5_X) = 4 + 3 = 7$ và $iu(\{b\}, T2'_X) = iu(\{b\}, T2_X) + iu(\{b\}, T5_X) = 1 + 4 = 5$

Định nghĩa 2.14. Phép chiếu kết hợp trộn các giao dịch đồng nhất: khi chiếu tập *X* lên *D*, các giao dịch đồng nhất được trộn bằng một giao dịch mới, ký hiệu cD_X. [11].

Ví dụ: phép chiếu kết hợp phép trộn tiếp theo ví dụ định nghĩa 2.11.

Tid	Giao dịch	Số lượng
$T1_X$	{b}	{1}
$T2_X$	{a,b}	{4,1}
$T3_X$	{a}	{4}
$T5_X$	{a,b}	{3,4}

(a) Dữ liệu đầy đủ của D_X với X={c}

Tid	Giao dịch	Số lượng
$T1_X$	{b}	{1}
$T2'_X$	{a,b}	{7,5}
$T3_X$	{a}	{4}

(b) Dữ liệu của cD_X với X={c}

Hình 2.3. Minh họa phép chiếu X={c} trên CSDL và phép trộn kết hợp

Định nghĩa 2.15. Gọi $\succ$ là phép xếp thứ tự các phần tử của tập *I* theo *TWU*. Giá trị hữu ích còn lại của *X* trong giao dịch *T*, ký hiệu *ru(X,T)* là tổng giá trị hữu ích các phần tử sau *X* trong *T*, hay là $ru(X,T) = \sum_{i \in T \wedge i \succ x \, \forall \, x \in X} u(i,T)$. [3]

Ví dụ: giả sử các phần tử ở giao dịch T3 đã được sắp xếp, *ru({a},T3) = u({c}, T3) + u({d},T3)* = 2 + 5 = 7

Định nghĩa 2.16. Cho tập *X*, phần tử $z \in E(X)$ và giá trị hữu ích cục bộ của *(X,z)* được tính như sau $lu(X,z)=\sum_{T \supset (X \cup \{z\})}[u(X,T) + ru(X,T)]$ [11]

Ví dụ. Giả sử các giao dịch đã sắp xếp, với *X* ={a}, *lu(X,c) = (u(X,T2) + ru(X,T2)) + (u(X,T3) + ru(X,T3)) + (u(X,T5) + ru(X,T5))* = 18 + 11 + 18 = 47

Tính chất 2.2. Cho tập *X*, $z \in E(X)$, nếu $lu(X,z) < minutil$ thì tất cả các tập mở rộng của tập *X* với *z* đều không thể là tập hữu ích cao. [11]

Thuật toán EFIM

input : *D*: CSDL cần khai thác, *minutil*: ngưỡng tối thiểu

output: các tập hữu ích cao

1. $X = \emptyset$;

2. Duyệt *D*, tính *lu*(*X*, *i*) cho tất cả $i \in I$;

3. *Secondary*(*X*) = $\{i | i \in I \wedge lu(X, i) \geq minutil\}$;

4. Sắp xếp tăng dần *Secondary*(*X*) theo giá trị *lu*;

5. Duyệt D để xóa các phần tử $i \notin Secondary(X)$ ra khỏi các giao dịch và xóa các giao dịch rỗng;

6. Sắp xếp các giao dịch T tăng dần;

7. Duyệt *D* tính *su*(*X*,*i*) cho từng phần tử $i \in Secondary(X)$;

8. *Primary*(*X*) = $\{i | i \in Secondary(X) \wedge su(X, i) \geq minutil\}$;

9. Search (*X*, *D*, *Primary*(*X*), *Secondary*(*X*), *minutil*);

Hình 2.4. Thuật toán EFIM

Định nghĩa 2.17. Cho tập *X* và phần tử $z \in E(X)$), giá trị hữu ích trên nhánh phụ z và tập *X* là $su(X, z) = \sum_{T \supset (X \cup \{z\}))} [u(\alpha, T) + u(z, T) + \sum_{i \in T \wedge i \in E(\alpha \cup \{z\})} u(i, T)]$ [11]

Ví dụ. Cho $X = \{a\}$, $su(X,c) = (u(\{a\},T2) + u(\{c\},T2) + u(\{d\},T2) + u(\{e\},T2)) + (u(\{a\},T3) + u(\{c\},T3) + u(\{d\},T3)) + (u(\{a\},T5) + u(\{c\},T5) + u(\{f\},T5))$ = 16 + 11 + 10 = 37

Tính chất 2.3. Cho tập X và $z \in E(X)$, nếu $su(X, z) < minutil$ thì tất cả các tập mở rộng của tập *X* với *z* đều không thể là tập hữu ích cao. [11]

Định nghĩa 2.18. Cho tập *X*, phần tử chính và phần tử phụ (Primary, Secondary item) được định nghĩa như sau $Primary\ (X) = \{\ z\ | z \in E(X)\ \wedge su(X, z) \geq minutil\}$ và $Secondary\ (X) = \{\ z\ | z \in E(X)\ \wedge lu(X, z) \geq minutil\}$. [11]

Ví dụ. Tiếp tục ví dụ tại định nghĩa 2.16 và 2.17, nếu xét *minutil = 40* thì $X=\{a\}$ là 1 phần tử phụ nhưng không phải là phần tử chính, nhưng với *minutil = 30* thì $X=\{a\}$ vừa là phần tử chính vừa là phần tử phụ.

Thuật toán EFIM thủ tục Search được mô tả ở hình 2.4 và tại hình 2.5: [11]

Thủ tục Search

input : X: tập phần tử đang xét, cD_X: Các giao dịch được chiếu và trộn bởi X, *Primary*(X): các phần tử chính X, *Secondary*(X): các phần tử mở rộng của X, ngưỡng *minutil*

output: các tập hữu ích cao mở rộng từ *x*

1 foreach *item* $i \in Primary\ (X)$ **do**

2 $\beta = X \cup \{i\}$;

3 Duyệt D_X để tính $u(\beta)$ and xây dựng D_β; // dùng phép trộn giao dịch

4 **if** $u(\beta) \geq minutil$ **then** xuất β;

5 Duyệt D_β tính $su(\beta, z)$, $lu(\beta, z)$ cho tất cả $z \in Secondary(X)$ sau i ;

6 $Primary(\beta) = \{z \in Secondary(X) | su(\beta, z) \geq minutil\}$;

7 $Secondary(\beta) = \{z \in Secondary(X) | lu(\beta, z) \geq minutil\}$;

8 *Search* (β, D_β, *Primary*(β), *Secondary*(β), *minutil*);

9 end.

Hình 2.5. Thủ tục Search của thuật toán EFIM

Sau đây là kết quả chạy từng bước thuật toán EFIM với *minutil = 35*

Bước 1. Khởi tạo $X = \emptyset$

Bước 2. Tính $lu(X, i)$ $với$ $i \in I$ $và$ $X = \emptyset$

Itemset	{a}	{b}	{c}	{d}	{e}	{f}	{g}
lu	69	68	57	61	40	18	10

Hình 2.6. Kết quả tính lu(X,i) với i∈I và X=∅

Bước 3. Từ $lu(X, i)$ ta có $Secondary(X) = \{a, b, c, d, e\}$

Bước 4. Sắp xếp $Secondary(X)$ tăng dần theo $lu(X, i)$ $Secondary(X) = \{e, c, d, b, a\}$

Bước 5. Quét lại dữ liệu và xóa các phần tử không thuộc tập $Secondary(X)$

Bước 6. Sắp xếp các giao dịch của D (xem kết quả tại hình 2.7)

Tid	Giao dịch	Số lượng
T1	{c,d,b}	{2,1,1}
T5	{c,b,a}	{1,4,3}
T3	{c,d,a}	{2,1,4}
T4	{e,d,b,a}	{2,2,1,5}
T2	{e,c,d,b,a}	{1,3,1,1,4}

Hình 2.7. Kết quả của EFIM sau khi sắp xếp CSDL

Bước 7. Tính $su(X,i)$ $với$ $i \in Secondary(X)$ và $X = \emptyset$ (xem hình 2.8)

Itemset	{e}	{c}	{d}	{b}	{a}
su	40	53	37	28	16

Hình 2.8. Kết quả tính su(X,i) với i∈I và X=∅

Bước 8. Xét $Primary(X) = \{e, c, d\}$

Bước 9. Gọi đệ quy hàm *Search* $(X = \emptyset,\ D, Primary(X) = \{e, c, d\},$ $Secondary(X) = \{e, c, d, b, a\}, minutil = 35)$

- $\beta = X \cup \{e\} = \{e\}$

- Lần lượt duyệt qua các giao dịch T1, T5, T3, T4, T2 để tính $u(\beta) = 8 + 4 = 12$ và tạo phép chiếu trên CSDL D_β. Kết quả thể hiện tại hình 2.9.

- Tính $lu(\beta, i)$ và $su(\beta, i)$ với $i \in Secondary(X)$

Tid	Giao dịch	Số lượng
T4	{d,b,a}	{2,1,5}
T2	{c,d,b,a}	{3,1,1,4}

(a) CSDL khi chiếu $\beta = \{e\}$

Item	*c*	*d*	*b*	*a*
lu	18	40	40	40
su	18	37	27	21

(b) Kết quả tính $lu(\beta, i), su(\beta, i)$

Hình 2.9. Kết quả thuật toán EFIM khi β= {e}

- Xét ngưỡng *minutil=35*, suy ra $Primary(\beta) = \{d\}$ và $Secondary(\beta) = \{d, b, a\}$

+ Tương tự gọi đệ quy hàm *Search* ($X = \{e\}$, $D_X, Primary(X) = \{d\}$, $Secondary(X) = \{d, b, a\}, minutil = 35$) ta có $u(\beta)$ = 22 với $\beta = \{ed\}$ thể hiện ở hình 2.10.

Tid	Giao dịch	Số lượng
T4	{b,a}	{1,5}
T2	{b,a}	{1,4}

(a) CSDL khi chiếu $\beta = \{ed\}$

Item	*b*	*a*
lu	37	37
su	37	31

(c) Kết quả tính $lu(\beta, i), su(\beta, i)$

Tid	Giao dịch	Số lượng
T1’	{b,a}	{2,9}

(b) Kết quả sau khi trộn giao dịch

$\boldsymbol{Primary(\beta)}$	*{b}*
$\boldsymbol{Secondary(\beta)}$	*{b, a}*

(d) Các tập mở rộng của β

Hình 2.10. Kết quả thuật toán EFIM khi β= {ed}

+ Thực hiện tương tự đệ quy hàm Search để tiếp tục mở rộng tập *{ed}* ta có được 1 tập hữu ích cao *{edba}* với *u({edba}) = 37.*

- Tương tự duyệt lần lượt 5 giao dịch cho từng trường hợp mở rộng với $\beta = \{c\}$ có *u{c} = 7*, $\beta = \{d\}$ có *u({d})=20* và ta không tìm thêm được tập hữu ích cao

với 2 trường hợp này. Xem hình 2.11 và 2.12

Tid	Giao dịch	Số lượng
T1	{d,b}	{1,1}
T5	{b,a}	{4,3}
T3	{d,a}	{1,4}
T2	{d,b,a}	{1,1,4}

(a) CSDL khi chiếu với $\beta = \{c\}$

Item	*d*	*b*	*a*
lu	34	39	41
su	34	25	17

(b) Kết quả tính $lu(\beta, i), su(\beta, i)$

$\boldsymbol{Primary(\beta)}$	∅
$\boldsymbol{Secondary(\beta)}$	*{b, a}*

(c) Các tập mở rộng của β

Hình 2.11. Kết quả thuật toán EFIM khi β= {c}

Tid	Giao dịch	Số lượng
T1	{b}	{1}
T3	{a}	{4}
T4	{b,a}	{1,5}
T2	{b,a}	{1,4}

(a) CSDL khi chiếu với $X = \{d\}$

Tid	Giao dịch	Số lượng
T1	{b}	{1}
T3	{a}	{4}
T4’	{b,a}	{2, 9}

(b) Kết quả sau khi trộn giao dịch

Item	*b*	*a*
lu	32	34
su	32	28

(c) Kết quả $lu(\beta, i), su(\beta, i)$

$\boldsymbol{Primary(\beta)}$	∅
$\boldsymbol{Secondary(\beta)}$	*{a}*

(d) Các tập mở rộng của β

Hình 2.12. Kết quả thuật toán EFIM khi β= {d}

2.4. Kết luận

Qua khảo sát và nghiên cứu các thuật toán khai thác tập hữu ích cao và chi tiết 3 thuật toán Two-Phase, TWU-Mining và EFIM. Mỗi thuật toán có những chiến lược

phát huy hiệu quả tỉa ứng viên của mình để đạt hiệu quả cao nhất. Một số nhận xét sau:

- Với Two-Phase, ưu điểm lớn nhất là khởi xướng chặn trên TWU để vận dụng tính chất bao đóng giảm để đưa ra các ứng viên có khả năng là tập hữu ích cao. Khuyết điểm chính là duyệt CSDL quá nhiều lần nên hiệu quả chưa cao với CSDL lớn.

- Với TWU-Mining, các tác giả đã cải tiến Two-Phase bằng cách vận dụng ưu điểm của IT-Tree và xây dựng cấu trúc WIT-Tree chứa Tid các giao dịch liên quan đến tập ứng viên nên giảm được số lần duyệt CSDL trong quá trình khai thác tập hữu ích cao so với Two-Phase.

- Với EFIM, dù đã khắc phục được nhiều khiếm khuyết với các thuật toán trước đó và cải tiến các chặn trên để đẩy nhanh hiệu quả khai thác. Nhưng quá trình thực hiện phép chiếu, thuật toán quét qua các giao dịch không liên quan đến ứng viên nên ít nhiều ảnh hưởng đến tốc độ khai thác tập hữu ích cao.

CHƯƠNG 3: THUẬT TOÁN EFIM CẢI TIẾN

Như nhận xét ở trên, EFIM tốn quá nhiều chi phí cho việc tạo phép chiếu trên tập *X* trên vùng giao dịch đang xét để dự toán sự triển vọng của các tập mở rộng.

Xét tập *X* và $z \in Secondary\ (X)$, và vùng dữ liệu cD_X là các giao dịch cần xét khi mở rộng phần tử. Xét phép chiếu *z* lên vùng cD_X, EFIM buộc phải quét lại toàn bộ cD_X một lần nữa, trong khi có thể xác định được vùng chiếu này khi tìm tập phần tử phụ $Secondary\ (X)$.

Nhằm hạn chế số lượng giao dịch cần duyệt khi thực hiện phép chiếu của *X* với một phần tử phụ, vận dụng hiệu quả cải tiến của TWU-Mining với Two-Phase, luận văn cải tiến thuật toán EFIM thành thuật toán *i*EFIM.

3.1. Thuật toán *i*EFIM

Với TWU-Mining, WIT-Tree đã lưu *Tid* các giao dịch có liên quan đến nút đang xét, vận dụng và kế thừa hiệu quả của việc lưu trữ các *Tid* vào EFIM, được gọi là *P-set* với các khái niệm như sau:

Định nghĩa 3.1. ***Phép chiếu ngược của tập X trên D***

Cho cơ sở dữ liệu *D* và tập *X*, *P-set* là phép chiếu ngược của tập *X* trên *D* được xác định như sau $P - set(X) = \{\ T.id |\ T \in D \wedge X\ \subseteq T\}$.

Ví dụ: Xét *X={e}, P-set(X) ={T2,T4}.*

Định nghĩa 3.2. ***Phép chiếu ngược mở rộng của tập X với i trên D***

Cho CSDL *D* và tập *X*, *P-set-ex(X,i)* là phép chiếu ngược mở rộng của tập *X* với phần tử *i* trên *D* được xác định như sau $Pex - set\ (X, i) = \{\ T'.id |\ T' \in cD_X \wedge i\ \subseteq T'\}$.

Mệnh đề 3.1. giá trị hữu ích của tập *X* không đổi khi áp dụng *P-set* và *Pex-set(X,i)* trên *D.*

Giả sử chia CSDL *D*, theo định nghĩa 2.4 ta có $u(X) = \sum_{T \in D1 \wedge X \subseteq T} u(X, T)$

hay $u(X) = \sum_{T \in D \wedge T.id \in P-set(X)} u(X,T)$ (theo định nghĩa 3.1). Vì vậy, khi áp dụng *P-set*, giá trị hữu ích các tập không thay đổi. Áp dụng thêm định nghĩa 2.14 và 3.2, chứng minh tương tự với *Pex-set(X,i)*, ta có:

$$u(X) = \sum_{T \in D \wedge T.id \in Pex-set(X,i)} u(X,T) \quad (1)$$

Ngoài ta, theo định nghĩa 2.11 và 2.14, ta có: $|cD_X| \leq |D_X| \leq |D|$ (2)

Áp dụng định nghĩa 3.2, ta có $|Pex\text{-}set(X,i)| \leq |cD_X|$ (3)

Kết hợp (2) và (3) suy ra $|Pex\text{-}set(X,i)| \leq |D_X|$ (4)

Từ (1) và (4) cho thấy hiệu quả *Pex-set* tỉ lệ nghịch với độ phổ biến của các tập *X* và phần tử mở rộng *i* trên vùng dữ liệu tương ứng cD_X.

Thuật toán *i*EFIM

input: *D*: CSDL cần khai thác, *minutil*: ngưỡng tối thiểu

output: các tập hữu ích cao

1. $X = \emptyset$;

2. Duyệt *D* tính *lu*(*X*, *i*) cho tất cả $i \in I$;

3. $Secondary(X) = \{i | i \in I \wedge lu(X, i) \geq minutil\}$;

4. Sắp xếp tăng dần *Secondary*(*X*) theo giá trị *lu*;

5. Duyệt D để xóa các phần tử $i \notin Secondary(X)$ ra khỏi các giao dịch và xóa các giao dịch rỗng;

6. Sắp xếp các giao dịch T tăng dần;

7. Duyệt *D* tính *su*(*X*,*i*) và *Pex-set(X,i)* cho từng phần tử $i \in Secondary(X)$;

8. $Primary(X) = \{i | i \in Secondary(X) \wedge su(X, i) \geq minutil\}$;

9. Search (*X*, *D*, *Primary*(*X*), *Secondary*(*X*), *minutil*,*Pex-set(X,i)*);

Hình 3.1. Thuật toán iEFIM

Ví dụ: xét *X={e}, P-set(X) ={T2,T4},* khi cần tính độ hữu ích của *X*, ta trực tiếp đến *T2* và *T4* để tính thay vì duyệt cả 5 giao dịch Và hiển nhiên hiệu quả khi sử dụng *P-set({a})* thấp hơn của *P-set({e})* do *{a}* xuất hiện trong nhiều giao dịch hơn *{e}*.

Với việc sử dụng *Pex-set*, thuật toán *i*EFIM thay đổi tại dòng 7 tính *Pex-set(X,i)* song song với *su*(*X, i*), là tổng giá trị hữu ích của *X, i* và giá trị hữu ích còn lại của *i* trong các giao dịch chứa *X* và *i* và tập các phần tử *i* có *su*(*X, i*) lớn hơn ngưỡng được gọi là phần tử chính *Primary*(*X*) và tại dòng 3, 5 của thủ tục Search (hình 3.1 và 3.2).

Do *Pex-set(X,i)* là tập chứa *T.id* của phép chiếu D_X-*ex* nên thực hiện đồng thời với việc tính giá trị hữu ích trên nhánh phụ *su*(*X,i*) không làm tăng độ phức tạp của thuật toán. Tương tự như thế với dòng 5 tại thủ tục Search. Hiệu quả của *Pex-set(X,i)* được thể hiện rõ tại dòng 3 của thủ tục Search, nó chỉ xét các giao dịch có *tid* thuộc *Pex-set(X,i)* thay vì quét toàn bộ D_X.

Thủ tục Search

input : *X*: tập phần tử đang xét, cD_X: Các giao dịch được chiếu và trộn bởi *X*, *Primary*(*X*): các phần tử chính *X*, *Secondary*(*X*): các phần tử mở rộng của *X*, ngưỡng *minutil*,*Pex-set(X,i):* tid các giao dịch dùng mở rộng *X* với *{i}*

output: các tập hữu ích cao mở rộng từ *x*

1 foreach *item* $i \in Primary(X)$ **do**

2 $\beta = X \cup \{i\}$;

3 ***Dùng Pex-set(X,i) để d*uyệt D_X để tính *u*(*β*) and xây dựng D_β; // dùng phép trộn giao dịch**

4 **if** $u(\beta) \geq minutil$ **then** xuất β;

5 **Duyệt D_β tính *su*(*β, z*), *lu*(*β, z*) và *P-set-ex*(*β,z*) cho tất cả $z \in$ *Secondary*(*X*) sau *i* ;**

6 $Primary(\beta) = \{z \in Secondary(X) | su(\beta, z) \geq minutil\}$;

7 $Secondary(\beta) = \{z \in Secondary(X) | lu(\beta, z) \geq minutil\}$;

8 *Search* (β, D_β, *Primary*(β), *Secondary*(β), *minutil*,*Pex-set(β,z)*);

9 end.

Hình 3.2. Thủ tục Search của iEFIM

3.2. Ví dụ minh họa thuật toán *i*EFIM

Sau đây là kết quả chạy từng bước thuật toán iMEFIM với *minutil = 35*

Bước 1. Khởi tạo $X = \emptyset$

Bước 2. Tính $lu(X, i)$ $với$ $i \in I$ và $X = \emptyset$

Bước 3. Từ $lu(X, i)$ ta có $Secondary(X) = \{a, b, c, d, e\}$

Bước 4. Sắp xếp $Secondary(X)$ tăng dần theo $lu(X, i)$ $Secondary(X) = \{e, c, d, b, a\}$

Bước 5. Quét lại dữ liệu và xóa các phần tử không thuộc tập $Secondary(X)$

Bước 6. Sắp xếp các giao dịch của D

Bước 7. Tính $su(X, i)$ $với$ $i \in Secondary(X)$ và $X = \emptyset$, đồng thời xây dựng *P-set* (xem kết quả thể hiện ở hình 3.3)

Itemset	{a}	{b}	{c}	{d}	{e}	{f}	{g}
lu	69	68	57	61	40	18	10

(a) Kết quả $lu(X, i)$

Itemset	{e}	{c}	{d}	{b}	{a}
su	40	53	37	28	16

(c) Kết quả $su(X, i)$

Tid	**Giao dịch**	**Số lượng**
T1	{c,d,b}	{2,1,1}
T5	{c,b,a}	{1,4,3}
T3	{c,d,a}	{2,1,4}
T4	{e,d,b,a}	{2,2,1,5}
T2	{e,c,d,b,a}	{1,3,1,1,4}

(b) CDSL sau khi xóa phần tử và sắp xếp

Item	**P-set**
e	{T4, T2}
c	{T1, T5, T3, T2}
d	{T1, T3, T4, T2}
b	{T1, T5, T4, T2}
a	{T5, T3, T4, T2}

(d) P-set dùng mở rộng ứng viên

Hình 3.3. Kết quả thuật toán iEFIM khi X=Ø

Bước 8. Xét $Primary(X) = \{e, c, d\}$

Tid	Giao dịch	Số lượng
T4	{d,b,a}	{2,1,5}
T2	{c,d,b,a}	{3,1,1,4}

(a) CSDL khi chiếu $\beta = \{e\}$

Item	*c*	*d*	*b*	*a*
lu	18	40	40	40
su	18	37	27	21

(b) Kết quả tính $lu(\beta, i), su(\beta, i)$

Item	Pex-set
c	{ T2}
d	{T4, T2}
b	{ T4, T2}
a	{T4, T2}

(c) Pex-set dùng mở rộng ứng viên

Hình 3.4. Kết quả phép chiếu, tính lu, su và Pex-set của iEFIM khi β={e}

Bước 9. Gọi đệ quy hàm *Search* $(X = \emptyset,\ D, Primary(X) = \{e, c, d\}$, $Secondary(X) = \{e, c, d, b, a\}, mintil = 35, P - set)$

- $\beta = X \cup \{e\} = \{e\}$

- Dùng *P-set({e})* duyệt 02 giao dịch *T4*, *T2* tính $u(\beta) = 12$ và tạo phép chiếu trên CSDL D_β. Kết quả thể hiện tại hình 3.4

- Tính $lu(\beta, i)$, $su(\beta, i)$ và xây dựng *Pex-set*(β, i)với $i \in Secondary(X)$

- Với *minutil=35*, suy ra $Primary(\beta) = \{d\}$ và $Secondary(\beta) = \{d, b, a\}$

+ Tương tự gọi đệ quy hàm *Search* $(X = \{e\},\ D_X, Primary(X) = \{d\}$, $Secondary(X) = \{d, b, a\}, minutil = 35, Pex - set)$ ta có $u(\beta)$ = 22 với $\beta = \{ed\}$ thể hiện ở hình 3.5.

Tid	Giao dịch	Số lượng
T4	{b,a}	{1,5}
T2	{b,a}	{1,4}

(a) CSDL khi chiếu $\beta = \{ed\}$

Item	*b*	*a*
lu	37	37
su	37	31

(c) Kết quả lu, su

Item	Pex-set
b	{T1'}
a	{T1'}

(d) Pex-set

Tid	Giao dịch	Số lượng
T1'	{b,a}	{2,9}

(b) Kết quả sau khi trộn giao dịch

$\boldsymbol{Primary(\beta)}$	*{b}*
$\boldsymbol{Secondary(\beta)}$	*{b, a}*

(e) Các tập mở rộng của β

Hình 3.5. Kết quả của iEFIM khi β={ed}

+ Thực hiện tương tự đệ quy hàm Seach để tiếp tục mở rộng tập *{ed}* ta có được 1 tập hữu ích cao *{edba}* với *u({edba}) = 37.*

Tid	Giao dịch	Số lượng
T1	{d,b}	{1,1}
T5	{b,a}	{4,3}
T3	{d,a}	{1,4}
T2	{d,b,a}	{1,1,4}

(a) CSDL khi chiếu với $\beta = \{c\}$

$\boldsymbol{Primary(\beta)}$	$\emptyset$
$\boldsymbol{Secondary(\beta)}$	*{b, a}*

(c) Các tập mở rộng của β

Item	*d*	*b*	*a*
lu	34	39	41
su	34	25	17

(b) Kết quả tính $lu(\beta, i), su(\beta, i)$

Item	Pex-set
d	{T1, T3, T2}
b	{ T1, T5, T2}
a	{T5, T3, T2}

(d) Pex-set dùng mở rộng ứng viên

Hình 3.6. Kết quả của iEFIM khi β={c}

- Tương tự duyệt các giao dịch theo *P-set* cho từng trường hợp mở rộng với $\beta = \{c\}$ có *u{c} = 7*, $\beta = \{d\}$ có *u({d})=20* và ta không tìm thêm được tập hữu ích cao với 2 trường hợp này. Xem hình 3.6 và 3.7

Tid	Giao dịch	Số lượng
T1	{b}	{1}
T3	{a}	{4}
T4	{b,a}	{1,5}
T2	{b,a}	{1,4}

(a) CSDL khi chiếu với $X = \{d\}$

Tid	Giao dịch	Số lượng
T1	{b}	{1}
T3	{a}	{4}
T4’	{b,a}	{2, 9}

(b) Kết quả sau khi trộn giao dịch

Item	*b*	*a*
lu	32	34
su	32	28

(c) Kết quả $lu(\beta, i), su(\beta, i)$

Item	Pex-set
b	{ T1, T4’}
a	{T3, T4’}

(d) Pex-set dùng mở rộng ứng viên

$\boldsymbol{Primary(\beta)}$	$\emptyset$
$\boldsymbol{Secondary(\beta)}$	*{a}*

(e) Các tập mở rộng của β

Hình 3.7. Kết quả thuật toán iEFIM khi β= {d}

3.3. Hiệu quả *P-set* của *i*EFIM

Để minh họa hiệu quả của *P-set, Pex-set* của *i*EFIM so với EFIM, ta tiếp tục xét và so sánh 2 ví dụ minh họa của 2 thuật toán được trình bày ở phần 2.3.3 và 3.2.

Với mục tiêu hạn chế số giao dịch quét thừa của EFIM, với cấu trúc *P-set* và *Pex-set* dùng mở rộng tập *X* với phần tử *i* khác, ta thấy số giao dịch 2 thuật toán phải quét qua để tạo phép chiếu ở dòng 3 của thủ tục *Search* (cụ thể xem bảng 3.1). Qua đó, ta thấy số giao dịch thuật toán cần duyệt để tạo phép chiếu của *i*EFIM giảm hơn 26% so với EFIM, nếu không xét phép chiếu mở rộng, *i*EFIM giảm 33% số giao dịch (cụ thể chỉ xét phép chiếu của *{e}, {c}, {d}* là 10 với *i*EFIM và 15 với EFIM).

3.4. Kết luận

Qua chứng minh của mệnh đề 3.1, trên cơ sở lý thuyết ta thấy được hiệu quả của *P-set* và *Pex-set*. Thông qua ví dụ minh họa và bảng so sánh (Bảng 10), thể hiện được khả năng giảm số lượng giao dịch phải duyệt trong quá trình khai thác tập hữu ích cao. Tuy nhiên, hiệu quả thực nghiệm của *P-set* và *Pex-set* được cụ thể ở chương 4.

Bảng 3.1. So sánh số giao dịch phải duyệt khi tạo phép chiếu của iEFIM và EFIM

Thao tác	**EFIM**	***i*EFIM**
- Tạo phép chiếu với $X=\{e\}$	5	2
- Tạo phép chiếu khi mở rộng $X=\{e\}$ với d	2	2
- Tạo phép chiếu khi mở rộng $X=\{ed\}$ với b	1	1
- Tạo phép chiếu khi mở rộng $X=\{edb\}$ với a	1	1
- Tạo phép chiếu với $X=\{c\}$	5	4
- Tạo phép chiếu với $X=\{d\}$	5	4
Tổng cộng	***19***	***14***

CHƯƠNG 4: KẾT QUẢ THỰC NGHIỆM

4.1. Môi trường và dữ liệu thực nghiệm

Thuật toán *i*EFIM được cài đặt và tiến hành chạy thực nghiệm so sánh với thuật toán EFIM trên các CSDL được lấy từ thư viện mở SPMF: An Java Open-Source Data Mining Library tại địa chỉ http://www.philippe-fournier-viger.com/spmf/ [24]. Các thuật toán được thực hiện trên môi trường Java sử dụng hệ điều hành Windows 8.1, 64 bit, RAM 4GB, CPU Core i3 M350.

Ngoài ra, các CSDL Retail, T10I4D100K, T40I10D100K được phát sinh ngẫu nhiên từ 1 đến 10 các giá trị: độ hữu ích của từng phần tử và số lượng trong từng giao dịch, các bộ dữ liệu thực nghiệm chuẩn được mô tả tại bảng 4.1.

Bảng 4.1. Bảng mô tả dữ liệu thực nghiệm chuẩn

Loại dữ liệu	Số giao dịch	Số phần tử	Độ dài trung bình	Đánh giá
Accident	340.183	468	33.8	Đặc
BMS	59.601	497	4.8	Thưa
Chess	3.196	75	37	Rất đặc
Foodmart	67.557	129	43	Thưa
Kosarak	990.002	41.270	8.1	Thưa
Retail	87.943	16.465	10.3	Thưa
T10I4D100K	100.000	870	10.1	Thưa
T40I10D100K	100.000	942	39.6	Thưa

Chúng tôi chạy thực nghiệm trên các CSDL trên và ghi lại thời gian thực hiện và số giao dịch được quét để thực hiện phép chiếu để xây dựng vùng dữ liệu mới

dùng mở rộng ứng viên và tính giá trị hữu ích. (Xem bảng 4.2)

Bảng 4.2. Kết quả thực nghiệm trên dữ liệu chuẩn

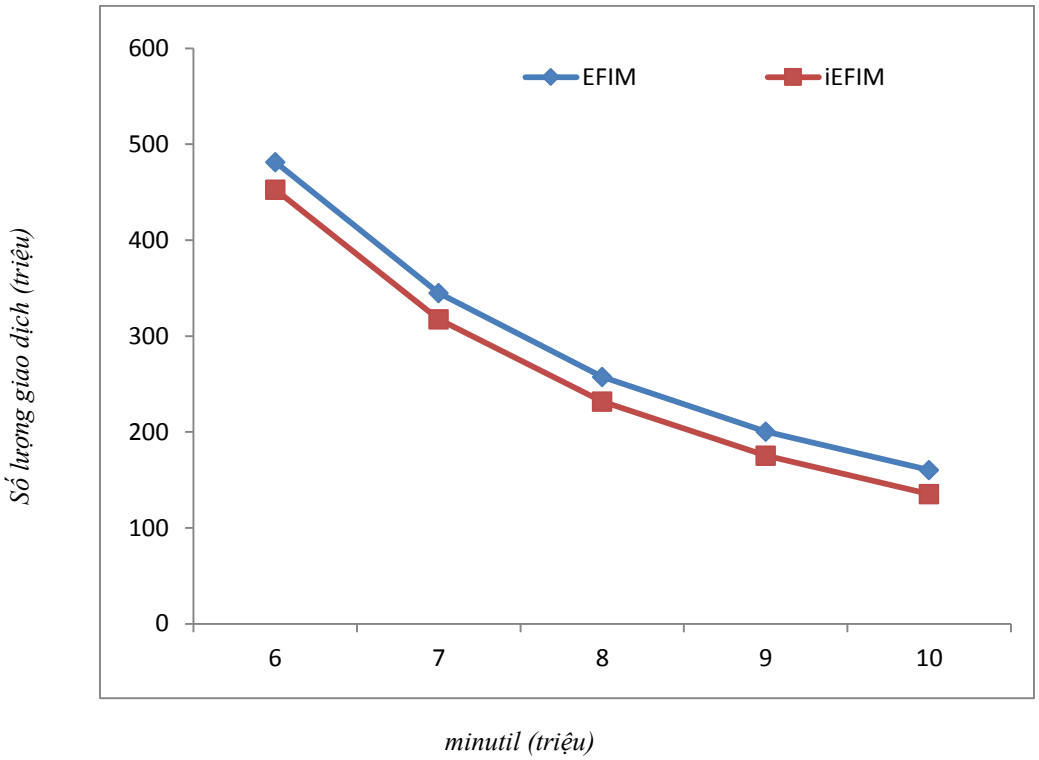
CSDL	minutil	Số giao dịch tham gia thuật toán		Thời gian (ms)	
		EFIM	***i*EFIM**	**EFIM**	***i*EFIM**
Accident	6.000.000	481.384.237	452.623.534	143.258	139.288
	7.000.000	344.980.709	317.552.753	113.927	99.511
	8.000.000	257.400.687	231.598.795	82.386	78.771
	9.000.000	200.530.132	175.370.521	68.358	65.875
	10.000.000	160.275.480	135.444.966	59.285	56.513
BMS	2.090.000	11.092.787	303.528	2.456	559
	2.100.000	10.912.762	179.219	2.399	431
	2.110.000	10.873.767	176.316	2.264	320
	2.130.000	10.763.324	169.615	2.230	277
	2.200.000	10.468.482	154.690	2.159	246
Chess	300.000	45.049.177	44.980.586	14.745	14.861
	320.000	29.693.811	29.628.087	8.755	8.800
	340.000	19.832.587	19.767.928	5.479	5.512
	360.000	13.452.590	13.390.942	3.580	3.574
	380.000	9.304.332	9.242.366	2.376	2.364
Foodmart	1.000	7.105.014	667.514	1.011	666
	1.500	7.031.947	594.447	675	328
	2.000	6.939.388	501.885	619	251
	2.500	6.845.807	408.304	598	205
	3.000	6.756.448	318.937	504	168
Kosarak	1.000.000	1.846.625.089	11.900.258	421.788	26.888
	1.100.000	1.354.952.101	3.037.229	318.566	22.336
	1.200.000	1.030.389.514	8.899.348	249.182	18.103
	1.300.000	804.028.839	8.039.582	192.828	16.316
	1.400.000	656.851.432	7.354.244	161.204	15.411

CSDL	minutil	Số giao dịch tham gia thuật toán		Thời gian (ms)	
		EFIM	*i*EFIM	EFIM	*i*EFIM
Retail	1.000	799.966.359	6.813.471	231.765	3.508
	1.500	686.936.586	4.273.098	160.980	3.140
	2.000	609.734.486	3.297.957	156.972	2.791
	2.500	549.790.860	2.807.803	147.225	2.566
	5.000	493.437.224	2.492.283	133.817	2.580
T10I4D100K	80.000	80.308.480	4.609.892	19.805	2.133
	100.000	77.587.696	3.387.511	19.327	2.007
	120.000	74.921.610	2.718.140	19.180	1.919
	140.000	72.623.626	2.316.566	18.244	1.897
	160.000	70.364.888	2.051.549	17.990	1.892
T40I10D100K	1.500.000	84.287.140	9.089.804	36.704	11.096
	1.600.000	82.542.339	8.535.513	35.991	11.042
	1.700.000	81.134.406	8.218.225	35.433	10.613
	1.800.000	80.208.963	6.958.839	33.763	10.308
	1.900.000	79.550.309	7.822.690	33.637	10.051

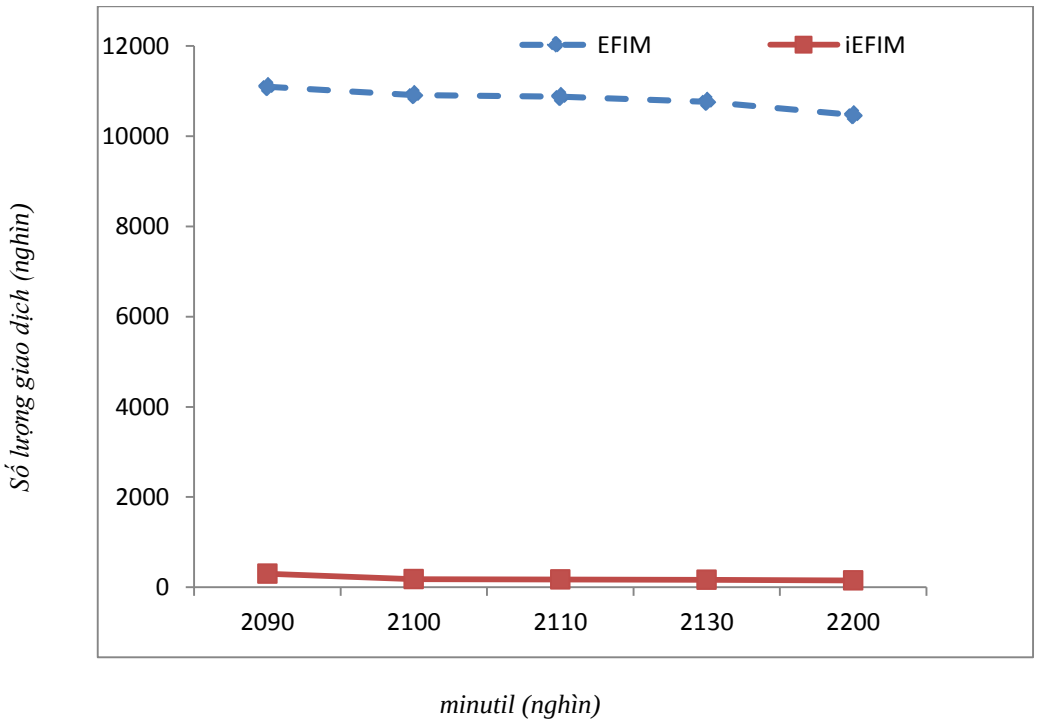
4.2. So sánh về số lượng giao dịch

Từ kết quả thực nghiệm được thể hiện qua bảng 4.2 và các đồ thị so sánh số lượng giao dịch tham gia phép chiếu tạo vùng dữ liệu để mở rộng ứng viên và tính giá trị hữu ích của tập ứng viên (từ hình 4.1 đến hình 4.8) ta có nhận xét, khi áp dụng phương pháp chiếu ngược, thuật toán *i*EFIM giảm số giao dịch giảm từ 9 (như T40I10D100K, hình 4.8) đến 400 lần (như Kosarak, hình 4.5) đối với loại CSDL được đánh giá thưa, và tỉ lệ này giảm dần đối với các loại dữ liệu được đánh giá đặc và rất đặc, cụ thể với Accident và Chess (hình 4.1 và 4.3), số lượng giao dịch được quét giảm không đáng kể.

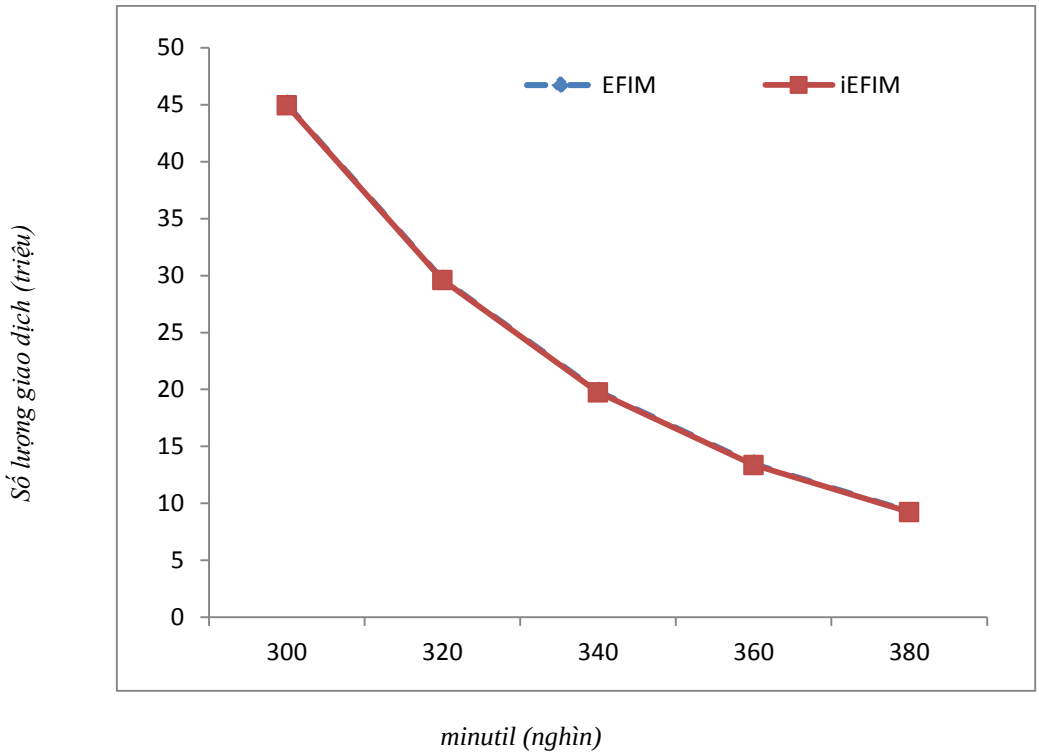
Nguyên nhân: Như mục tiêu đề ra ở mục 3.1, hiệu quả của *P-set* và *Pex-set* phụ thuộc vào sự xuất hiện ít hay nhiều của tập ứng viên đang xét nên loại dữ liệu càng thưa thì hiệu quả càng cao và ngược lại.



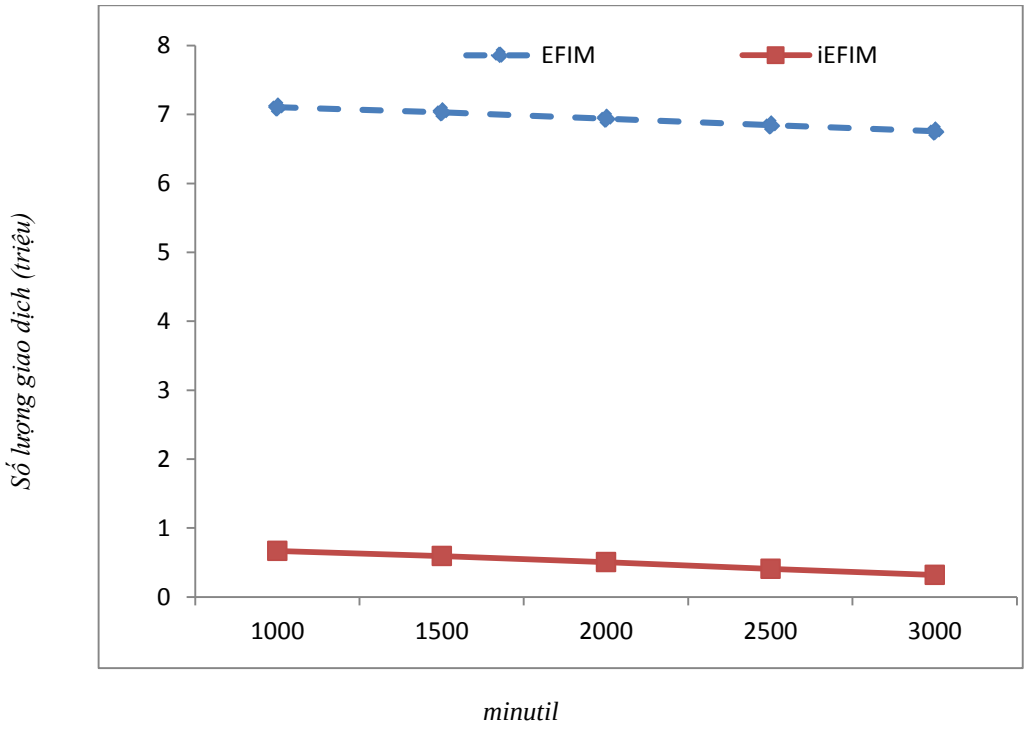
Hình 4.1. Đồ thị so sánh số lượng giao dịch CSDL Accident



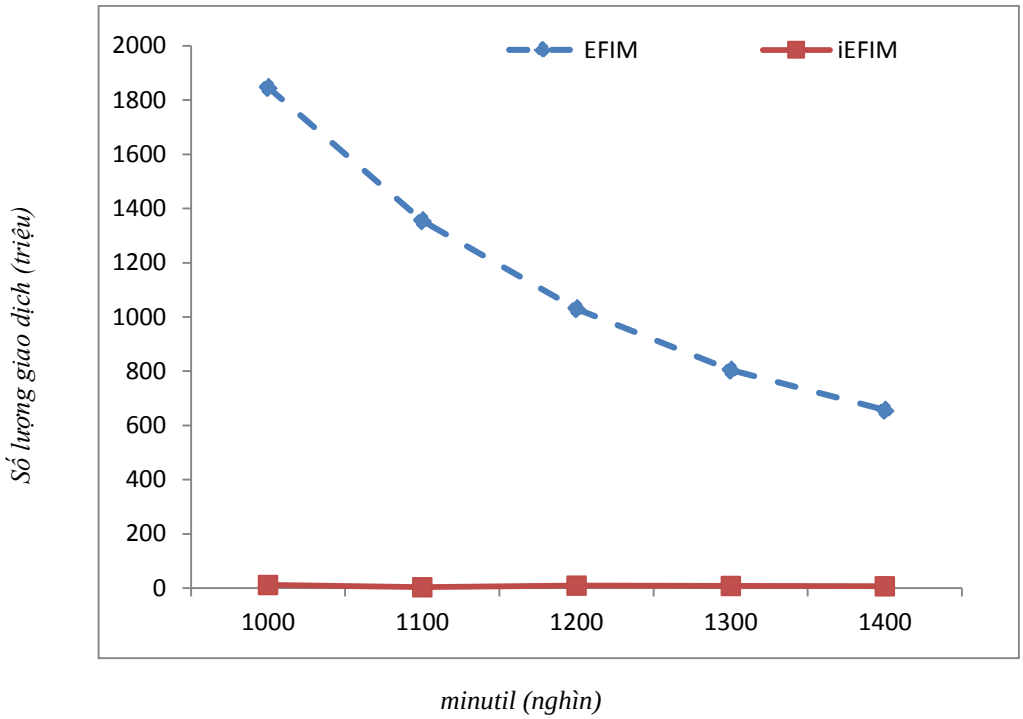
Hình 4.2. Đồ thị so sánh số lượng giao dịch CSDL BMS



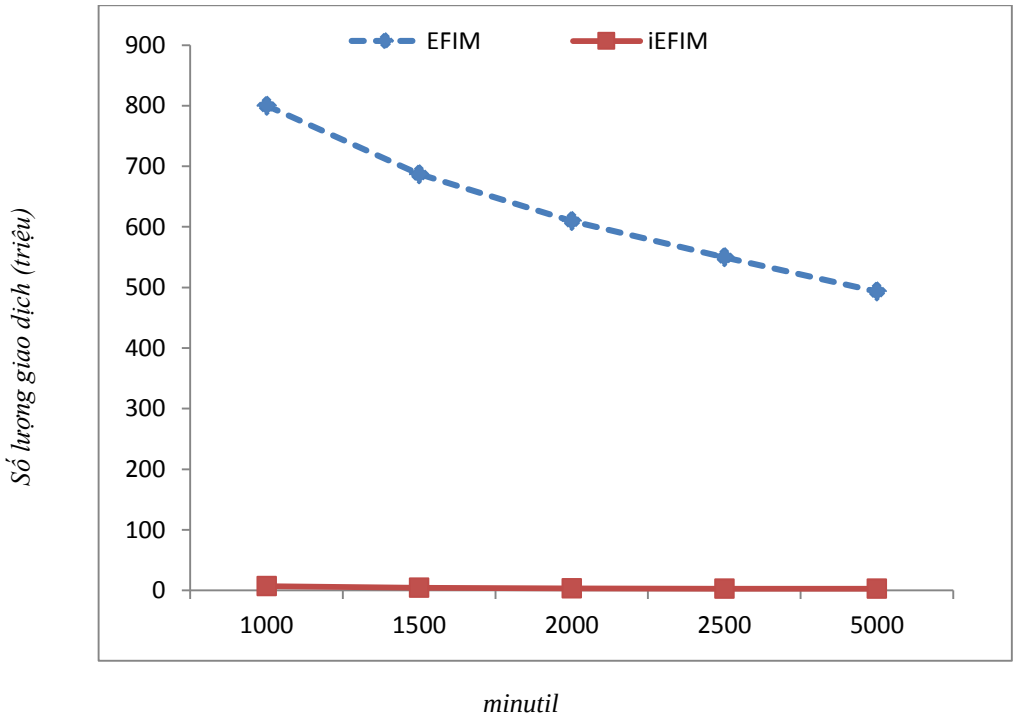
Hình 4.3. Đồ thi so sánh số lượng giao dịch CSDL Chess



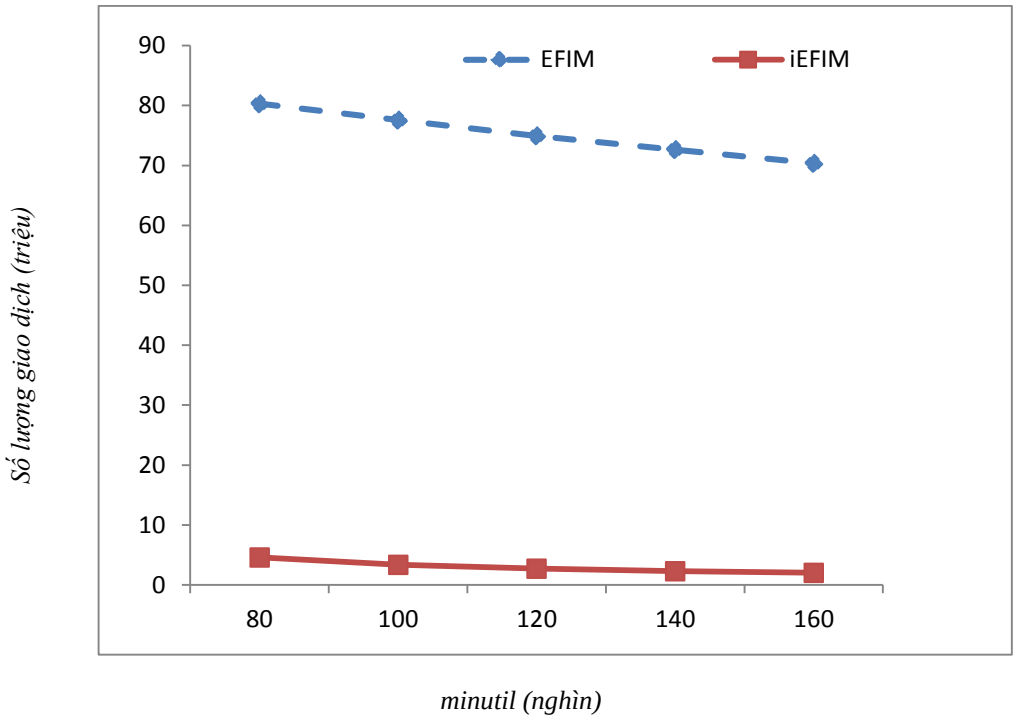
Hình 4.4. Đồ thị so sánh số lượng giao dịch CSDL Foodmart



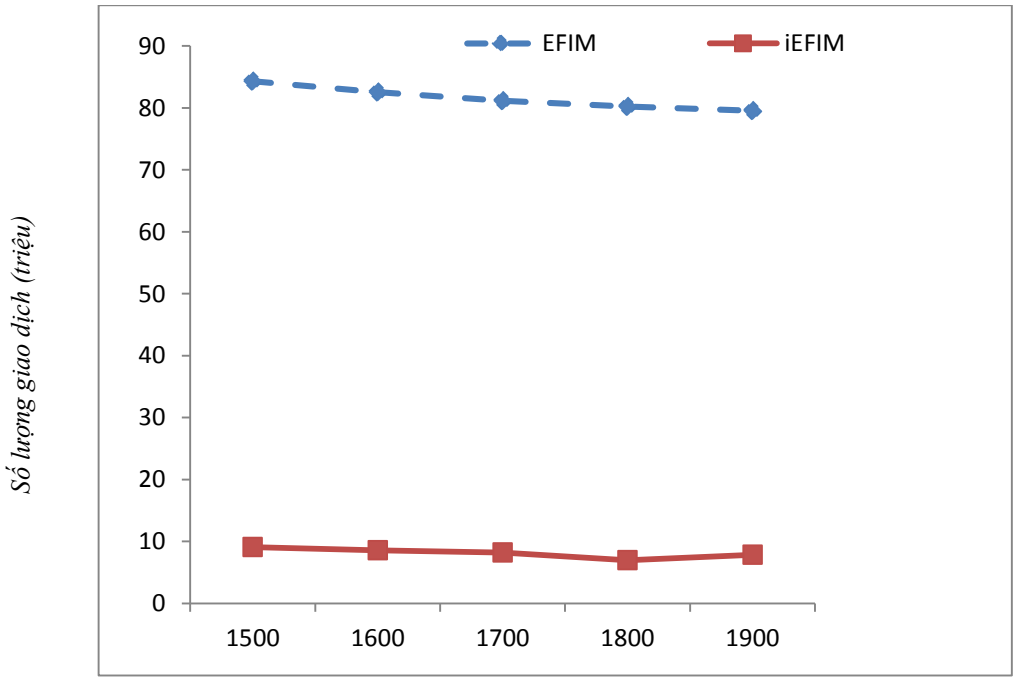
Hình 4.5. Đồ thị so sánh số lượng giao dịch CSDL Kosarak



Hình 4.6. Đồ thị so sánh số lượng giao dịch CSDL Retail

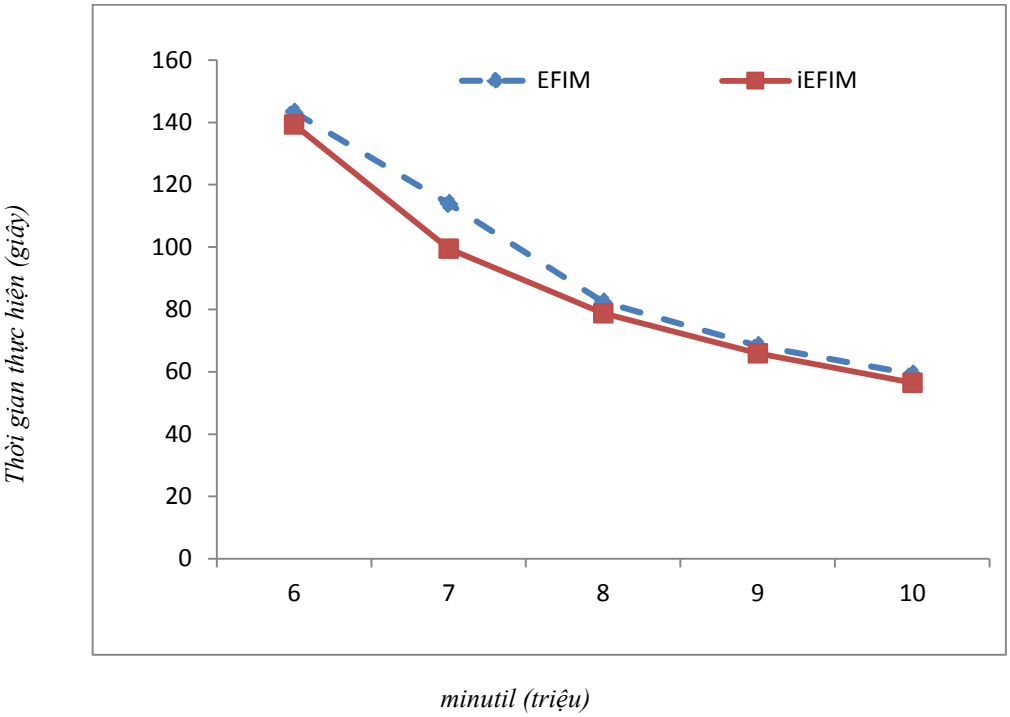


Hình 4.7. Đồ thị so sánh số lượng giao dịch CSDL T10I4D100K

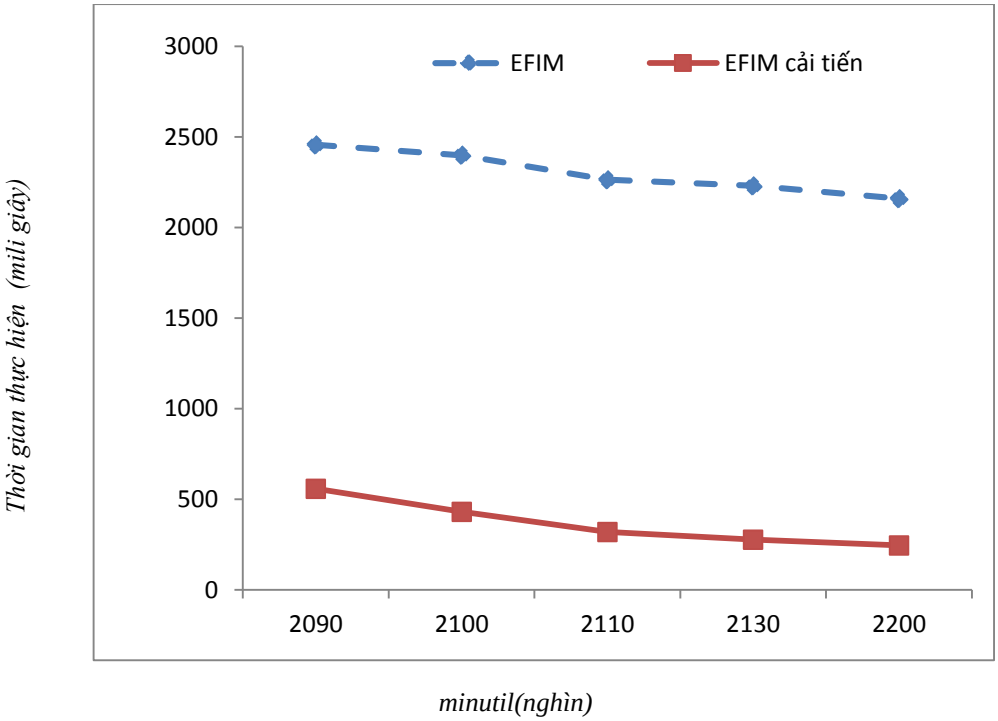


Hình 4.8. Đồ thị so sánh số lượng giao dịch CSDL T40I10D100K

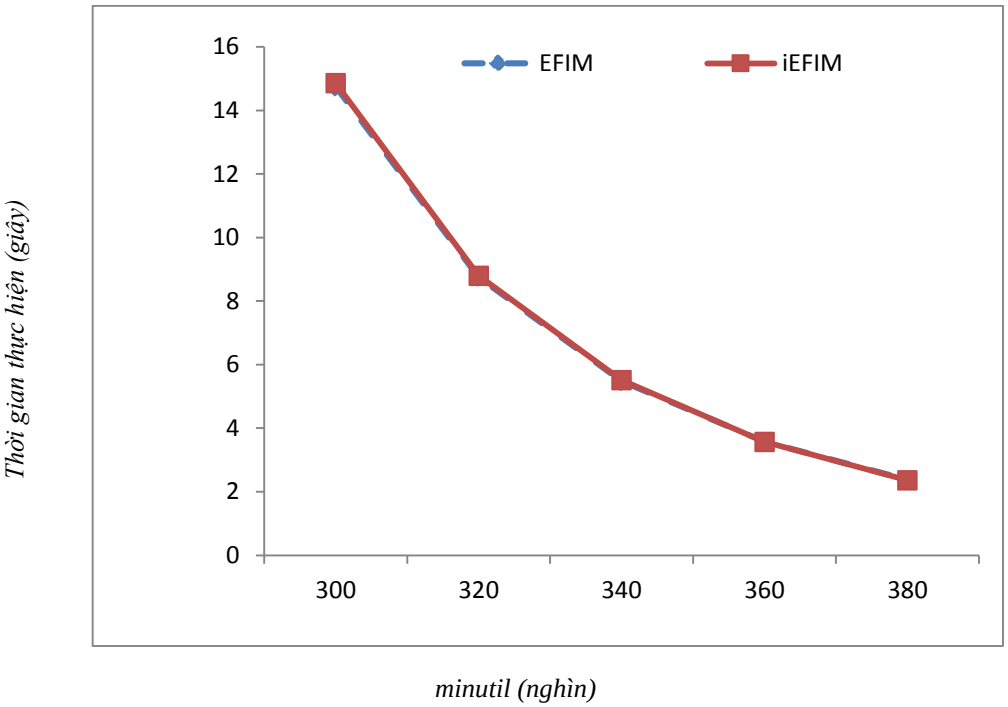
4.3. So sánh về thời gian



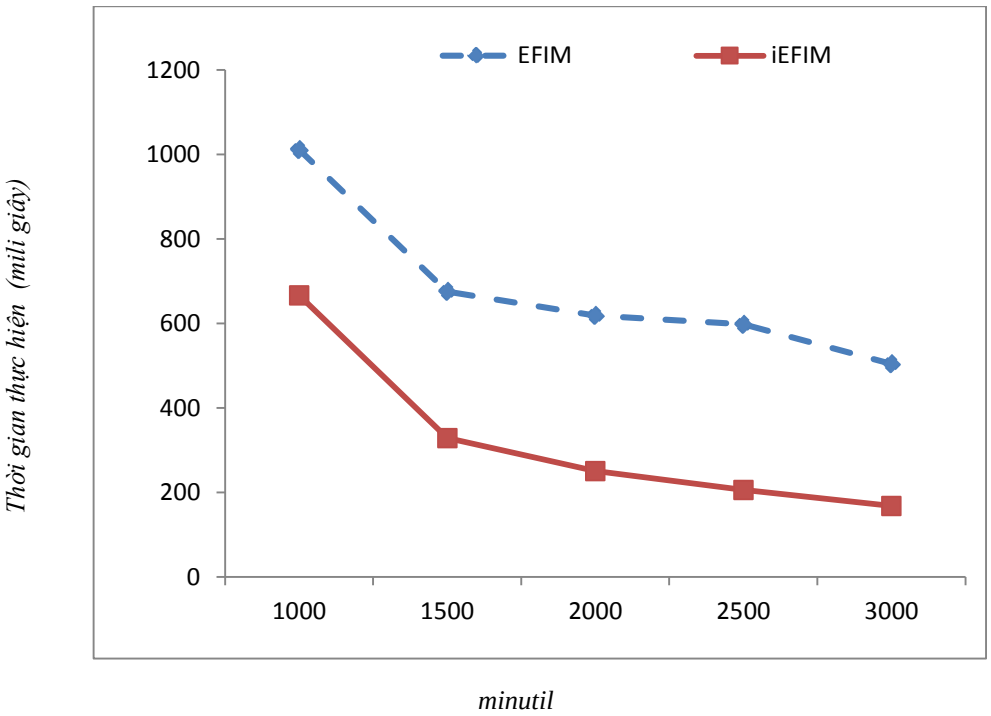
Hình 4.9. Đồ thị so sánh thời gian thực nghiệm CSDL Accident



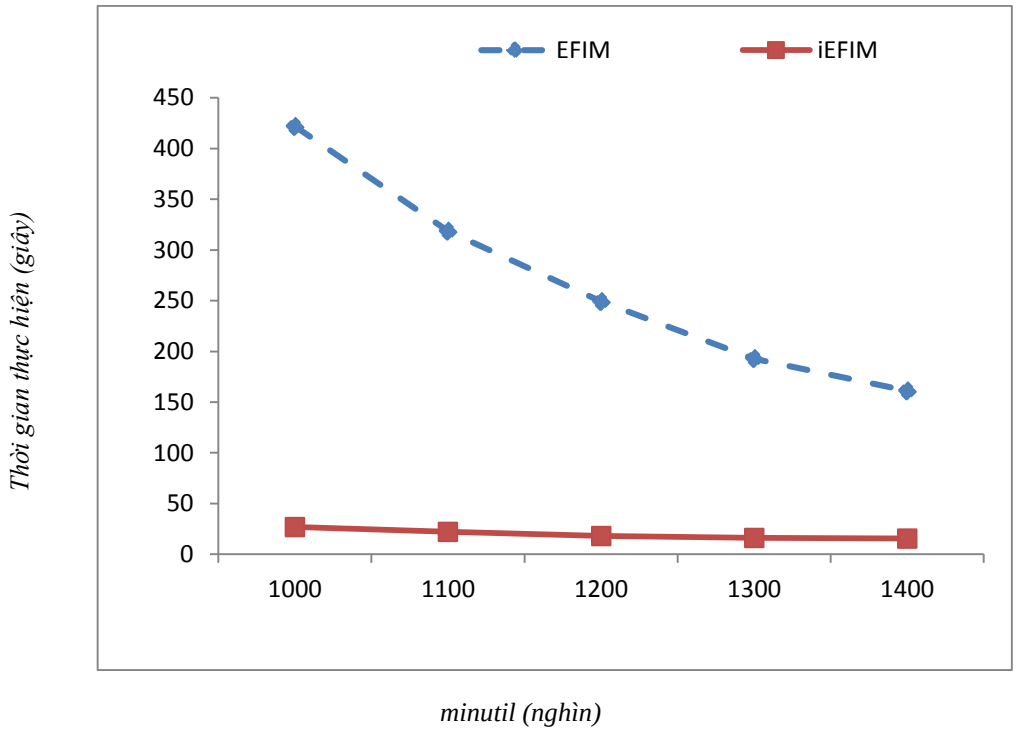
Hình 4.10. Đồ thị so sánh thời gian thực nghiệm CSDL BMS



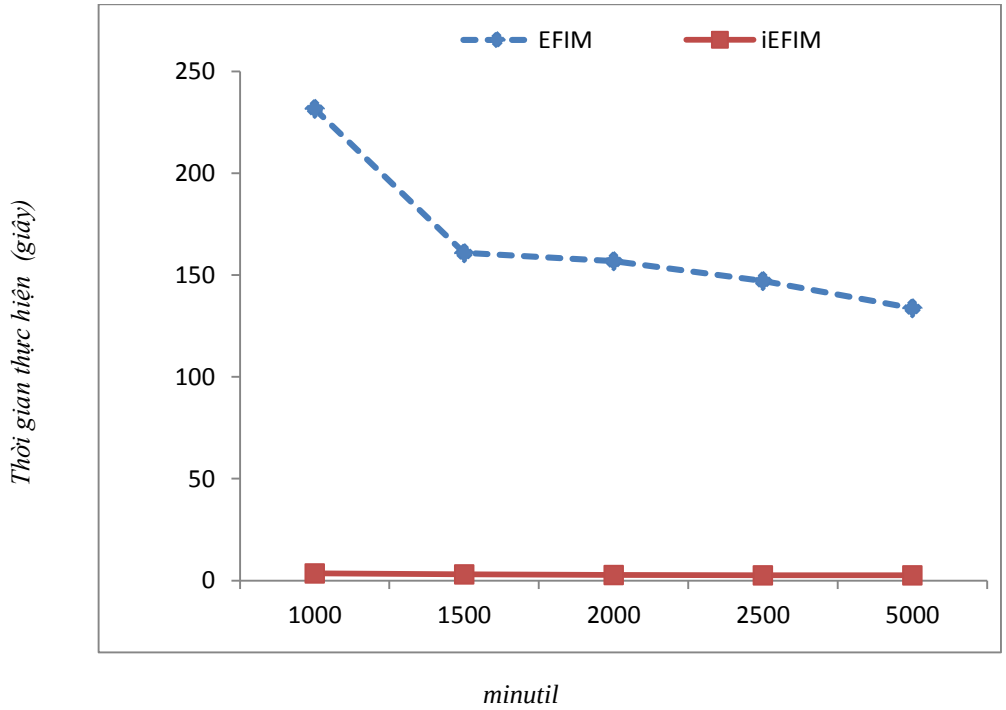
Hình 4.11. Đồ thị so sánh thời gian thực nghiệm CSDL Chess



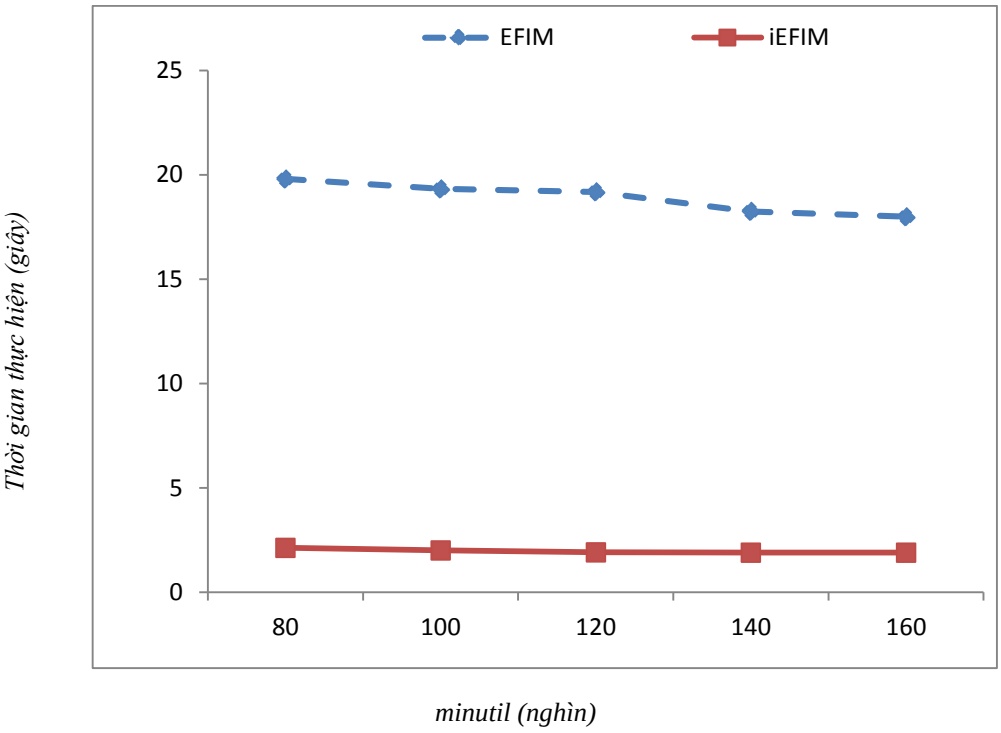
Hình 4.12. Đồ thị so sánh thời gian thực nghiệm CSDL Foodmart



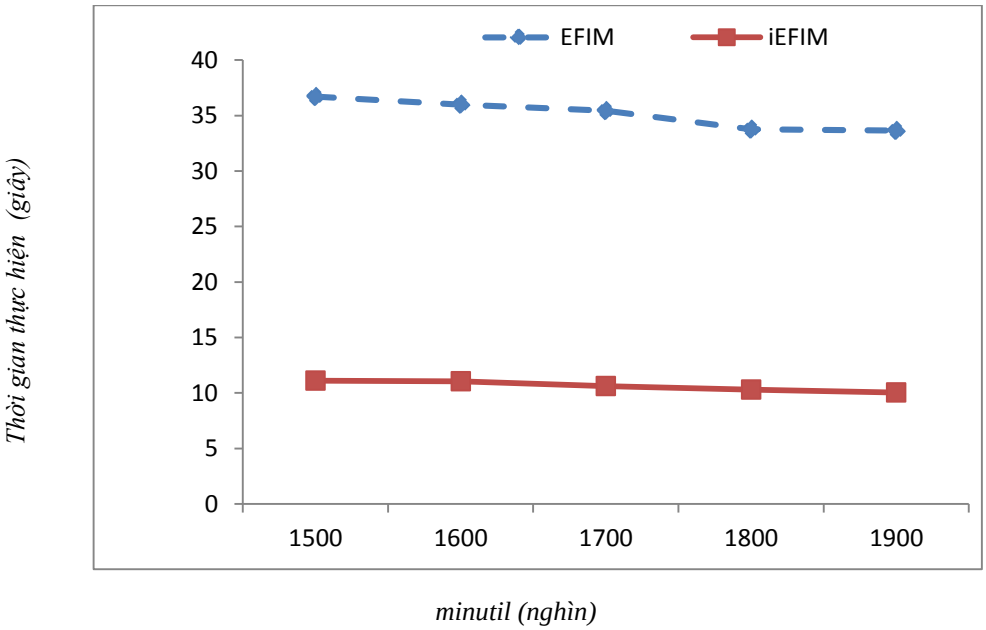
Hình 4.13. Đồ thị so sánh thời gian thực nghiệm CSDL Kosarak



Hình 4.14. Đồ thị so sánh thời gian thực nghiệm CSDL Retail



Hình 4.15. Đồ thị so sánh thời gian thực nghiệm CSDL T10I4D100K



Hình 4.16. Đồ thị so sánh thời gian thực nghiệm CSDL T40I10D100K

Về thời gian thực hiện được thể hiện tại Bảng 4.2 và các đồ thị so sánh từ 4.9 đến 4.16 tương ứng với 8 CSDL chuẩn được mô tả tại bảng 4.1, thuật toán *i*EFIM nhanh hơn hẳn EFIM trên CSDL thưa, giảm thời gian thực hiện từ 2 (Foodmart, hình 4.12) đến 60 lần (Retail, hình 4.14). Đối với CSDL đặc/rất đặc như Accident, Chess thì thời gian cải thiện không đáng kể (hình 4.9 và 4.11).

Nguyên nhân: hiệu quả của thuật toán *i*EFIM so với EFIM ở chỗ giảm được số giao dịch cần tham gia xử lý để giảm thời gian thực hiện. Tuy nhiên, *i*EFIM lại phát sinh chi phí khác đó là thời gian xây dựng *P-set* và *Pex-set*. Tóm lại, hiệu quả của *i*EFIM phải hài hòa giữa việc giảm thời gian xử lý giao dịch thừa và thời gian tạo *P-set* và *Pex-set*. Vì thế, khi dữ liệu thưa, chi phí tạo *P-set* nói chung thấp trong khi số lượng giao dịch giảm rất lớn nên thời gian thực nghiệm giảm và ngược lại, số lượng giao dịch giảm ít, thời gian tạo *P-set* tăng như Accident, Chess , thì ít nhiều *i*EFIM chậm hơn EFIM.

CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1. Kết luận

Luận văn đã trình bày chi tiết các bài toán khai thác tập hữu ích cao, khái quát các công trình nghiên cứu liên quan và các thuật toán khai thác tập hữu ích cao như Two-Phase, TWU-Mining, EFIM.

Trên cơ sở hiệu quả thuật toán TWU-Mining với Two-Phase và EFIM, luận văn đã đề xuất giải pháp chiếu ngược *P-set* và thuật toán cải tiến gọi là *i*EFIM (*improved EFIM*) thông qua giải pháp *P-set* đã giảm được đáng kể số lượng giao dịch tham gia quá trình khai thác tập hữu ích cao nhờ đó giảm bớt được thời gian thực hiện thuật toán khai thác, đặc biệt là trên các CSDL thưa.

Thuật toán cải tiến đã được cài đặt và thử nghiệm thành công trên một số CSDL chuẩn lớn được cộng đồng nghiên cứu về HUI sử dụng, như CSDL Accidents, BMS-POS, Chess, Foodmart, Kosarak, Retail, T10I4D100K, T40I10D100K. Tương ứng với mỗi CSDL là sự so sánh số giao dịch tham gia thuật toán và thời gian thực hiện của thuật toán gốc EFIM và *i*EFIM.

Với đề xuất giải pháp *P-set* và thuật toán cải tiến trên, luận văn đã có đóng góp nhất định về mặt khoa học trong lĩnh vực khai thác tập hữu ích cao, góp phần giảm thiểu thời gian với CSDL thưa.

5.2. Hướng phát triển

Trong luận văn này, với giải pháp chiếu ngược *P-set* để tăng tốc độ khai thác tập hữu ích cao bằng cách hạn chế quét các số giao dịch thừa. Bằng thực nghiệm đã chứng minh được hiệu quả của *P-set* với dữ liệu thưa và cũng phù hợp với các môi trường dữ liệu kinh doanh trong thực tế được thể hiện như CSDL Foodmart.

Với hiệu quả này, giải pháp trên sẽ mở hướng nghiên cứu, vận dụng vào các hướng khai thác khác tập hữu ích cao như khai phá HUI đóng, khai phá Top-k HUI,

khai thác tập HUI với dạng dữ liệu không chắc chắn…. hay lai ghép nhiều kỹ thuật khác nhau để tăng tốc độ, giảm không gian tìm kiếm và không gian bộ nhớ.

TÀI LIỆU THAM KHẢO

[1] R. Agrawal, T. Imielinski and A. N. Swami , "Mining association rules between sets of items in large databases," in *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, Washington D.C., pp. 207 – 216, 1993.

[2] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules in large databases," in *Proc. Int'l Conf. Very Large Data Bases*, pp. 487-499, 1994.

[3] M. Liu and J. Qu, "High utility itemsets without candidate generation," in *21st ACM International Conference on Information and Knowledge Management*, pp. 55-64, 2012.

[4] H. Yao, H. J. Hamilton and C. J. Butz, "A foundational approach to mining itemset utilities from databases," in *In Proc. SIAM Int'l Conf. Data Mining*, 2004.

[5] H. Yao and H. J. Hamilton, "Mining Itemset Utilitied from Transaction Databases," *Data and Knowledge Engeneering,* vol. 59, no. 3, p. 603–626, 2006.

[6] Y. Liu, W. K. Liao and A. N. Choudhary, "A two-phase algorithm for fast discovery of high utility itemsets," in *Proc. Pacific-Asia Conf. Knowledge Discovery and Data Mining*, pp.689-695, 2005.

[7] C. Ahmed, S. K. Tanbeer, B. -S. Jeong and Y.-K. Lee, "Efficient tree structures for high utility pattern mining in incremental databases," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 12, p. 1708–1721, 2009.

[8] B. Le, H. Nguyen, T. A. Cao and B. Vo, "A Novel Algorithm for Mining High Utility Itemsets," in *In Proceedings of 1st Asian Conference on Intelligent Information and Database Systems*, Quang Binh, Vietnam (IEEE press), 2009.

[9] V. S. Tseng, C. W. Wu, B. E. Shie and P. S. Yu, "Upgrowth: Anefficientalgorithm for high utility itemset mining," in *Proc. ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining*, pp. 253-262, 2010.

[10] B. Le, H. Nguyen and B. Vo, "An efficient strategy for mining high utility itemsets," *International Journal of Intelligent Information and Database Systems*, vol. 5, no. 2, pp. 164-176, 2011.

[11] S. Zida, P. Fournier-Viger, J. C.-W. Lin, C.-W. Wu and V. S. Tseng, "EFIM: A Highly Efficient Algorithm for High-Utility Itemset Mining," in *Advances in Artificial Intelligence and Soft Computing, Springer.*, pp. 530-546, 2015.

[12] C.-W. Wu, P. Fournier-Viger, P. S. Yu and V. S. Tseng, "Efficient Mining of a Concise and Lossless Representation of High Utility Itemsets," in *IEEE 11th International Conference on Data Mining*, pp. 824 - 833 , 2011.

[13] V. T. Tseng, C. W. Wu, P. Fournier-Viger and P. S. Yu, "Efficient Algorithms for Mining the Concise and Lossless Representation of High Utility Itemsets," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 3, pp. 726 - 739, 2015.

[14] C. W. Wu, B.-E. Shie, V. T. Tseng and P. S. Yu, "Mining top-K high utility itemsets," in *KDD '12 Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining* , pp. 78-86, 2012 .

[15] V. T. Tseng, C. W. Wu, P. Fournier-Viger and P. S. Yu, "Efficient Algorithms for Mining Top-K High Utility Itemsets," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 1, pp. 54 - 67, 2016.

[16] C. -J. Chu, V. S. Tseng and T. Liang, "An efficient algorithm for mining temporal high utility itemsets from data streams," *Journal of Systems and Software*, vol. 81, no. 7, p. 1105–1117, 2008.

[17] Bai-En Shie, Philip S. Yu and V. S. Tseng, "Efficient algorithms for mining maximal high utility itemsets from data streams with different models," *Expert Systems with Applications*, vol. 39, no. 17, p. 12947–12960, 2012.

[18] J. C.-W. Lin, W. Gan, P. Fournier-Viger, T. P. Hong and V. T. Tseng, "Efficient algorithms for mining high-utility itemsets in uncertain databases," *Knowledge-Based Systems*, vol. 96, p. 171–187, 2016.

[19] M. Zaki, "Scalable algorithms for association mining," *IEEE Transactions on Knowledge and Data Engineering*, vol. 12, no. 3, pp. 372 - 390, 2000.

[20] J. Han, J. Pei, Y. Yin and R. Mao, "Mining frequent patterns without candidate generation: A frequent pattern tree approach," *Data Mining and Knowledge Discovery*, vol. 8, no. 1, pp. 53-87, 2004.

[21] V. S. Tseng, B. E. Shie, C. W. Wu and P. S. Yu, "Efficient algorithms for mining high utility itemsets from transactional databases," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 8, pp. 1772-1786, 2013.

[22] P. Fournier-Viger, C. W. Wu, S. Zida and V. T. Tseng, "FHM: Faster High-Utility Itemset Mining using Estimated Utility Co-occurrence Pruning," in *Proc. 21st International Symposium on Methodologies for Intelligent Systems (ISMIS 2014), Springer*, pp. 83-92, 2014.

[23] S. Krishnamoorthy, "Pruning strategies for mining high utility itemsets," *Expert Systems with Applications*, vol. 42, no. 5, pp. 2371- 2381, 2015.

[24] P. Fournier-Viger, A. Gomariz, T. Gueniche, A. Soltani, C.-W. Wu and V. Tseng, "SPMF: a java open-source pattern mining library," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3389-3393, 2014.