TẠP CHÍ KHOA HỌC ĐẠI HỌC ĐÀ LẠT Tập 8, Số 2, 2018 109–123

MAXLEN-FI: THUẬT TOÁN KHAI THÁC NHANH TẬP PHỔ BIẾN CÓ CHIỀU DÀI TỐI ĐA TRÊN DỮ LIỆU GIAO DỊCH

Phan Thành Huấn[a*], Lê Hoài Bắc[b]

[a]Bộ môn Tin học, Trường Đại học Khoa học Xã hội và Nhân văn, Đại học Quốc gia TP. Hồ Chí Minh, TP. Hồ Chí Minh, Việt Nam
[b]Khoa Công nghệ Thông tin, Trường Đại học Khoa học Tự nhiên, Đại học Quốc gia TP. Hồ Chí Minh, TP. Hồ Chí Minh, Việt Nam
[*]Tác giả liên hệ: Email: huanphan@hcmussh.edu.vn

Lịch sử bài báo
Nhận ngày 19 tháng 01 năm 2018
Chỉnh sửa ngày 22 tháng 03 năm 2018 | Chấp nhận đăng ngày 14 tháng 04 năm 2018

Tóm tắt

Trong khai thác dữ liệu, kỹ thuật quan trọng và được nghiên cứu nhiều là khai thác luật kết hợp. Khai thác tập phổ biến là một trong những bước cơ bản và chiếm nhiều thời gian trong khai thác luật kết hợp. Tuy nhiên, trong một số ứng dụng thực tế chỉ cần khai thác tập con đại diện của tập phổ biến với chi phí thời gian thấp để sinh luật kết hợp - tập phổ biến có chiều dài tối đa. Đây là tập hữu ích trong nhiều lĩnh vực ứng dụng thực. Trong bài viết, chúng tôi đề xuất thuật toán MAXLEN-FI khai thác nhanh tập phổ biến có chiều dài tối đa trên dữ liệu giao dịch dựa trên cấu trúc mảng itemset đồng xuất hiện. Sau cùng, chúng tôi trình bày kết quả thực nghiệm trên bộ dữ liệu thực và giả lập, cho thấy thuật toán đề xuất hiệu quả hơn so với thuật toán hiện hành.

Từ khóa: Luật kết hợp; Tập phổ biến; Tập phổ biến có chiều dài tối đa.

Mã số định danh bài báo: http://tckh.dlu.edu.vn/index.php/tckhdhdl/article/view/407

Loại bài báo: Bài báo nghiên cứu gốc có bình duyệt

MAXLEN-FI: A FAST ALGORITHM FOR MINING MAXIMUM LENGTH FREQUENT ITEMSETS

Phan Thanh Huan[a*], Le Hoai Bac[b]

[a]The Information Technology Department, University of Social Sciences and Humanities, VNU Hochiminh City, Hochiminh City, Vietnam
[b]The Faculty of Information Technology, University of Science, VNU Hochiminh City, Hochiminh City, Vietnam
[*]Corresponding author: Email: huanphan@hcmussh.edu.vn

Article history
Received: January 19th, 2018
Received in revised form: March 22nd, 2018 | Accepted: April 14th, 2018

Abstract

Association rule mining, one of the most important and well-researched techniques of data mining. Mining frequent itemsets are one of the most fundamental and most time-consuming problems in association rule mining. However, real-world applications are often sufficient to mine a small representative subset of frequent itemsets with low computational cost in generating association rules maximum length frequent itemsets. Maximum length frequent itemsets can be useful in many application domains. In this paper, we proposed a fast algorithm called MAXLEN-FI for mining maximum length frequent itemsets fast using an array of co-occurrence items. Finally, we presented experimental results on both synthetic and real-life datasets, which showed that the proposed algorithm performed better than the existing algorithms.

Keywords: Association rules; Frequent itemsets; Maximum length frequent itemsets.

Article identifier: http://tckh.dlu.edu.vn/index.php/tckhdhdl/article/view/407

Article type: (peer-reviewed) Full-length research article

1. GIỚI THIỆU

Khai thác luật kết hợp là một kỹ thuật quan trọng trong lĩnh vực khai thác dữ liệu. Mục tiêu khai thác là phát hiện những mối liên hệ giữa các giá trị dữ liệu trong dữ liệu giao dịch. Mô hình đầu tiên của bài toán khai thác luật kết hợp là mô hình nhị phân hay còn gọi là mô hình cơ bản được Agrawal, Imilienski, và Swami (1993) đề xuất nhằm phân tích dữ liệu giao dịch, phát hiện các mối liên hệ giữa các tập mục hàng hoá đã bán được tại các siêu thị. Từ đó có kế hoạch bố trí, sắp xếp, kinh doanh hợp lý, đồng thời tổ chức sắp xếp các quầy gần nhau như thế nào để có doanh thu trong các phiên giao dịch là lớn nhất.

Để có thể khai thác luật kết hợp nhanh chóng, một số nhóm tác giả như Agrawal và ctg. (1993); và Han, Pei, Yin, và Mao (2004) đã đề xuất phương pháp tìm tập phổ biến FI (*Frequent Itemset*) và dựa trên tập phổ biến để sinh luật kết hợp. Trong đó, Agrawal và ctg. (1993) đã đề xuất thuật toán sinh tập phổ biến Apriori có độ phức tạp dạng hàm mũ. Để cải tiến về mặt thời gian trong khai thác tập phổ biến, Han và ctg. (2004) đã đề xuất thuật toán khai thác tập phổ biến không sinh ứng viên trên cấu trúc FP-Tree. Tuy nhiên, trong thực tế việc phát sinh tập phổ biến tốn nhiều thời gian và có số lượng tập mục rất lớn. Vì vậy, một số nhóm tác giả như Zaki và Hsiao (2002) và Wang, Han, và Pei (2003) đã đề xuất khai thác tập phổ biến đóng CFI (*Closed Frequent Itemset*) có số lượng ít hơn tập phổ biến như thuật toán Charm và Closet+. Bên cạnh đó, một số nhóm tác giả như Burdick, Calimlim, và Gehrke (2001); và Zaki và Hsiao (2005) cũng đề xuất khai thác tập phổ biến tối đại MFI (*Maximal Frequent Itemset*) như thuật toán Mafia và GenMax.

Trong một số ứng dụng thực tế, việc sử dụng các tập FI, CFI và MFI tốn rất nhiều chi phí tính toán cũng như số lượng tập mục phổ biến rất lớn. Hu, Sung, Xiong, và Fi (2008) đề xuất khai thác tập phổ biến có chiều dài tối đa LFI (*Maximum Length Frequent Itemset*). Đây là tập con của tập phổ biến FI và chỉ chứa các tập mục phổ biến có chiều dài tối đa. Hu và ctg. (2008) đã nêu một số bài toán cần khai thác tập LFI như sau:

- Bằng cách nào để một công ty du lịch đưa ra một gói tour du lịch mới cho một số địa điểm tham quan? Công ty tiến hành khảo sát khách hàng để xác định sở thích của họ trong số các địa điểm tham quan. Giả sử, công ty muốn gói tour này đáp ứng các yêu cầu như sau: (i) Số lượng khách hàng tham gia tour du lịch này phải không ít hơn một số nhất định (ví dụ, không dưới 20 khách là *ngưỡng phổ biến tối thiểu*); và (ii) Lợi ích mang đến cho mỗi khách hàng được tối đa hóa (*đáp ứng được tối đa yêu cầu của khách hàng*). Ở đây, chúng tôi giả định rằng lợi ích trên mỗi khách hàng là tỉ lệ thuận với số lượng địa điểm trong gói tour. Ngoài ra, khách hàng được giả định là sẽ trả chi phí thấp, nghĩa là họ sẽ không trả tiền nếu gói tour có những nơi họ không muốn đến. Bài toán này có thể được giải quyết bằng cách xác định LFI từ dữ liệu điều tra có ngưỡng không dưới 20. Các địa điểm có chiều dài lớn nhất tạo thành một gói tour cần thiết.

- Một số bài toán tương tự: Công ty bảo hiểm muốn thiết kế gói bảo hiểm để thu hút số lượng khách hàng nhất định và tối đa hóa số lượng đối tượng được bảo hiểm; Một siêu thị muốn thiết kế một kế hoạch bán hàng ràng buộc (tối đa hoá số lượng hàng mua cùng với một số lượng khách hàng nhất định).

Sau đây là một số thuật toán điển hình khai thác tập phổ biến có chiều dài tối đa LFI:

- *Thuật toán LFIMiner_ALL*: Hu và ctg. (2008) đề xuất khai thác tập phổ biến có chiều dài tối đa. Thuật toán này dựa trên cấu trúc FP-Tree và thuật toán FP-Growth, đồng thời đề xuất kỹ thuật cắt tỉa để tối ưu không gian tìm kiếm;

- *Thuật toán MaxLFI*: Tran, Ngo, và Nguyen (2011) cải tiến dựa trên thuật toán LFIMiner_ALL bằng cách thêm chiến lược *cắt tỉa trước* cho các mẫu cơ sở và lượng giá chiều dài ban đầu để rút gọn không gian tìm kiếm tập mục phổ biến có chiều dài tối đa.

Trong ứng dụng thực tế, khi cần khai thác tập phổ biến có chiều dài tối đa thì người dùng có thể yêu cầu thực hiện khai thác tập phổ biến thỏa ngưỡng *minsup* trong nhiều chuỗi thao tác liên tiếp khác nhau. Các thuật toán trên chưa đáp ứng tốt yêu cầu này. Vì vậy, chúng tôi đề xuất thuật toán MAXLEN-FI khai thác nhanh tập phổ biến có chiều dài tối đa từ mảng chứa các *itemset* đồng xuất hiện và không đọc lại dữ liệu cho lần khai thác tiếp theo. Thuật toán đề xuất bao gồm các thuật toán con sau:

- Xây dựng mảng *Index_COOC* chứa *itemset* đồng xuất hiện và *itemset* xuất hiện ít nhất trong một giao dịch của từng *item* hạt nhân;

- Thuật toán MAXLEN-FI khai thác nhanh tập phổ biến có chiều dài tối đa dựa trên mảng *Index_COOC*.

Trong Mục 2, bài báo trình bày các khái niệm cơ bản về khai thác các tập phổ biến và tổ chức lưu trữ dữ liệu giao dịch. Mục 3 xây dựng thuật toán xác định mảng chứa *itemset* đồng xuất hiện và *itemset* xuất hiện *ít nhất trong một giao dịch* của từng *item* hạt nhân và thuật toán MAXLEN-FI khai thác nhanh tập phổ biến có chiều dài tối đa. Kết quả thực nghiệm được trình bày trong Mục 4 và kết luận ở Mục 5.

2. CÁC VẤN ĐỀ LIÊN QUAN

2.1. Một số khái niệm cơ bản

Cho $I = \{i_1, i_2, \ldots i_m\}$ là tập gồm m mục hàng riêng biệt, mỗi mục hàng gọi là *item*. Tập các mục $X = \{i_1, i_2, \ldots i_k\}$ $\forall i_j \in I\ (1 \le j \le k)$ gọi là *itemset*, tập mục có k *mục* gọi là *k-itemset*. $\mathcal{D}$ là dữ liệu giao dịch, gồm n bản ghi phân biệt gọi là tập các giao dịch $T = \{t_1, t_2, \ldots t_n\}$, mỗi giao dịch $t_j = \{i_{k_1}, i_{k_2}, \ldots i_{k_l}\}$, $\forall i_{k_l} \in I\ (1 \le k_l \le m)$.

Định nghĩa 1: Độ phổ biến (*support*) của *itemset* $X \subseteq I$, ký hiệu $sup(X)$, là số các giao dịch trong $\mathcal{D}$ có chứa X.

Định nghĩa 2: Cho $X \subseteq I$, X gọi là *itemset* phổ biến nếu $sup(X) \ge minsup$, trong đó *minsup* là ngưỡng phổ biến tối thiểu. Ký hiệu FI là tập hợp các tập mục phổ biến.

Tính chất 1: $\forall X \subseteq Y : sup(Y) \ge minsup \Longrightarrow sup(X) \ge minsup$;

Tính chất 2: $\forall X \subset Y : sup(X) < minsup \Longrightarrow sup(Y) < minsup$;

Để minh họa bằng các ví dụ, dữ liệu giao dịch $\mathcal{D}$ được cho như trong Bảng 1.

Bảng 1. Dữ liệu giao dịch $\mathcal{D}$ cho các ví dụ

Mã giao dịch	Tập mục hàng								
t1	A		C		E	F			
t2	A		C				G		
t3					E				H
t4	A		C	D		F	G		
t5	A		C		E		G		
t6					E				
t7	A	B	C		E				
t8	A		C	D					
t9	A	B	C		E		G		
t10	A		C		E	F	G		

Ví dụ 1: Dữ liệu giao dịch $\mathcal{D}$ trong Bảng 1, có 8 *item* riêng biệt là $I = \{A, B, C, D, E, F, G, H\}$ và 10 giao dịch T = {*t1, t2, t3, t4, t5, t6, t7, t8, t9, t10*}. Với *minsup* = 2, ta có tập mục $X = \{A, C, E, G\}$ xuất hiện trong *t5, t9 và t10* và $sup(ACEG) = 3 \geq minsup$, ta nói rằng $X = \{A, C, E, G\}$ là phổ biến theo ngưỡng *minsup* = 2.

Theo Tính chất 1 thì các tập con của $X = \{A, C, E, G\}$ cũng phổ biến, nghĩa là tất cả tập con của *X* đều phổ biến: $sup(A) = 8$, $sup(C) = 8$, $sup(E) = 7$, $sup(G) = 5$, $sup(AC) = 8$, $sup(AE) = 5$, $sup(AG) = 5$, $sup(CE) = 5$, $sup(CG) = 5$, $sup(EG) = 3$, $sup(ACE) = 5$, $sup(ACG) = 5$, $sup(AEG) = 3$, $sup(CEG) = 3 \geq minsup$. Tương tự, với $Y = \{H\}$ thì $sup(H) = 1 < minsup$, ta nói rằng $Y = \{H\}$ *không phổ biến* theo ngưỡng *minsup* = 2. Theo Tính chất 2 thì các tập cha của $Y = \{H\}$ cũng *không phổ biến*, nghĩa là $Y = \{E, H\}$ cũng *không phổ biến*, với *sup*(EH) = 1 < *minsup* = 2.

Định nghĩa 3: Cho $X \subseteq I$, *X* được gọi là *itemset* phổ biến đóng nếu *X* là tập mục phổ biến và không có tập cha cùng độ phổ biến. Tập các *itemset* phổ biến đóng gọi là tập hợp các tập mục phổ biến đóng, ký hiệu là CFI.

Ví dụ 2: theo Ví dụ 1, ta có tập mục $X = \{A, C, E, G\}$ có $sup(ACEG) = 3$ và tập mục $Z = \{C, E, G\}$ có $sup(CEG) = 3$, theo định nghĩa trên thì $X = \{A, C, E, G\}$ là *itemset* phổ biến đóng, $Z = \{C, E, G\} \subset X = \{A, C, E, G\}$ nên $Z = \{C, E, G\}$ không là *itemset* phổ biến đóng.

Định nghĩa 4: Cho $X \subseteq I$, *X* được gọi là *itemset* phổ biến tối đại nếu *X* là tập mục phổ biến và không có tập cha là tập mục phổ biến. Tập các *itemset* phổ biến tối đại gọi là tập phổ biến tối đại, ký hiệu là MFI.

Ví dụ 3: Theo Ví dụ 1, ta có tập mục $X = \{A, C, E, G\}$ có $sup(ACEG) = 3$ và tập mục $Z = \{C, E, G\}$ có $sup(CEG) = 3$ là hai tập mục phổ biến, theo định nghĩa trên thì $X = \{A, C, E, G\}$ là *itemset* phổ biến tối đại, còn tập mục $Z = \{C, E, G\} \subset X = \{A, C, E, G\}$ nên $Z = \{C, E, G\}$ không là *itemset* phổ biến tối đại.

Định nghĩa 5: Cho $X \in FI$, *X* được gọi là *itemset* phổ biến có chiều dài tối đa nếu $\forall Y \in FI$ thì $|X| \geq |Y|$, tức là số lượng mục hàng của tập mục phổ biến *X* lớn hơn hoặc bằng số lượng mục hàng của bất kỳ tập mục phổ biến có trong FI. Tập các *itemset* phổ biến có chiều dài tối đa gọi là tập phổ biến có chiều dài tối đa, ký hiệu là LFI.

Tính chất 3: $\forall X \in LFI : X \in MFI;$

Trong Bảng 2, tập phổ biến FI, tập phổ biến đóng CFI, tập phổ biến tối đại MFI, và tập phổ biến có chiều dài tối đa LFI chứa *k-itemset* với *minsup* = 2. Số lượng tập mục phổ biến $|FI|=39$, số lượng tập mục phổ biến đóng $|CFI|=10$, số lượng tập mục phổ biến tối đại $|MFI|=5$ và số lượng tập mục phổ biến có chiều dài tối đa $|LFI|=4$. Tỷ suất $|CFI|/|FI| = \frac{10}{39} \times 100\% = 26\%$, $|MFI|/|FI| = \frac{5}{39} \times 100\% = 13\%$và $|LFI|/|FI| = \frac{4}{39} \times 100\% = 10\%$. Qua đó, ta dễ dàng thấy mối quan hệ giữa các tập phổ biến trên dữ liệu giao dịch như sau: LFI $\subseteq$ MFI $\subseteq$ CFI $\subseteq$ FI.

Bảng 2. Tập FI, CFI, MFI và LFI trên dữ liệu giao dịch D

k-itemset	Tập phổ biến FI	Tập đóng CFI	Tập tối đại MFI	Tập phổ biến có chiều dài tối đa LFI
1	B, D, F, G, E, A, C	E		
2	BE, BA, BC, DA, DC, FG, FE, FA, FC, GE, GA, GC, EA, EC, AC	AC		
3	BEA, BEC, BAC, DAC, FGA, FGC, FEA, FEC, FAC, GEA, GEC, GAC, EAC	DAC, FAC, EAC, GAC	DAC	
4	BEAC, FGAC, FEAC, GEAC	BEAC, FEAC, FGAC, GEAC	BEAC, FGAC, FEAC, GEAC	BEAC, FGAC, FEAC, GEAC

Ghi chú: *minsup = 2.*

2.2. Tổ chức lưu trữ dữ liệu giao dịch

Lưu trữ dữ liệu giao dịch dạng *bit* là cấu trúc dữ liệu hiệu quả trong khai thác tập phổ biến (Song & Yang, 2008). Chuyển đổi dữ liệu giao dịch thành ma trận nhị phân BiM, trong đó mỗi dòng tương ứng với một giao dịch và mỗi cột tương ứng với một *item*. Nếu *item* thứ *i* xuất hiện trong giao dịch *t* thì *bit* thứ *i* của dòng *t* trong BiM sẽ mang giá trị 1, ngược lại sẽ mang giá trị 0. Bảng 3 biểu diễn dạng bit của dữ liệu giao dịch Ɗ trong Bảng 1.

Bảng 3. Biểu diễn dạng bit của dữ liệu giao dịch Ɗ

Mã giao dịch	A	B	C	D	E	F	G	H
t1	1	0	1	0	1	1	0	0
t2	1	0	1	0	0	0	1	0
t3	0	0	0	0	1	0	0	1
t4	1	0	1	1	0	1	1	0
t5	1	0	1	0	1	0	1	0
t6	0	0	0	0	1	0	0	0
t7	1	1	1	0	1	0	0	0
t8	1	0	1	1	0	0	0	0
t9	1	1	1	0	1	0	1	0
t10	1	0	1	0	1	1	1	0

3. CÁC THUẬT TOÁN

3.1. Tập chiếu và *itemset* đồng xuất hiện

Tập chiếu của mục hàng i_k trên dữ liệu giao dịch D: $\pi(i_k) = \{t \in D \mid i_k \in t\}$ là tập các giao dịch có chứa mục hàng i_k (π-*đơn điệu giảm*).

$$sup(i_k) = |\pi(i_k)| \quad (1)$$

Tập chiếu của tập $X = \{i_1, i_2, \ldots i_k\}$, $\forall i_{j=\overline{1,k}} \in I$, $\pi(X) = \pi(i_1) \cap \pi(i_2) \ldots \cap \pi(i_k)$.

$$sup(X) = |\pi(X)| \quad (2)$$

Ví dụ 4: Theo Bảng 1, có $\pi(A) = \{t1, t2, t4, t5, t7, t8, t9, t10\}$ và $\pi(B) = \{t7, t9\}$. Khi đó, $\pi(AB) = \pi(A) \cap \pi(B) = \{t1, t2, t4, t5, t\underline{7}, t8, t\underline{9}, t10\} \cap \{t\underline{7}, t\underline{9}\} = \{t\underline{7}, t\underline{9}\}$, $\pi(B) \subseteq \pi(A)$ và $\pi(AB) \subseteq \pi(A)$.

Định nghĩa 6: Cho $i_k \in I$, ta gọi i_k là *item* hạt nhân. Tập $X_{cooc} \subseteq I$ gọi đồng xuất hiện với i_k: X_{cooc} là tập chứa các *item* đồng xuất hiện cùng i_k thì $\pi(i_k) \equiv \pi(i_k \cup X_{cooc})$. Ký hiệu, $cooc(i_k) = Xcooc$.

Ví dụ 5: Xem *item B* là *item* hạt nhân, ta xác định được *itemset* đồng xuất hiện cùng độ phổ biến với *item B* là $cooc(B) = \{A, C, E\}$ và $sup(\underline{B}) = sup(\underline{B}ACE) = 2$.

Định nghĩa 7: Cho $i_k \in I$, ta gọi i_k là *item* hạt nhân. Tập $Y_{looc} \subseteq I$ chứa các *item* xuất hiện cùng với i_k ít nhất trong một giao dịch, nhưng không đồng xuất hiện: $1 \leq |\pi(i_k \cup i_{looc})| < |\pi(i_k)|, \forall i_{looc} \in Y_{looc}$. Ký hiệu, $looc(i_k) = Y_{looc}$.

Ví dụ 6: Xem *item G* là *item* hạt nhân, ta xác định được các *item* xuất hiện cùng với *item G* ít nhất trong một giao dịch là $looc(G) = \{B, D, E, F\}$ có $\pi(G) = \{2, 4, 5, 9, 10\}$ và $\pi(\underline{G}B) = \{9\}$, $\pi(\underline{G}E) = \{5, 9, 10\}$.

3.2. Thuật toán sinh *itemset* đồng xuất hiện có thứ tự

Chúng tôi đã trình bày thuật toán sinh *itemset* đồng xuất hiện (Lê & Phan, 2016). Dưới đây là thuật toán cải tiến (bổ sung theo Định nghĩa 7) sinh các *item* đồng xuất hiện và *item* xuất hiện ít nhất trong cùng một giao dịch với từng *item* trong dữ liệu giao dịch và lưu trữ vào mảng *Index_COOC*. Mỗi phần tử trong mảng *Index_COOC* gồm bốn thành phần: *Index_COOC[k].item*: là *item* hạt nhân thứ *k*; *Index_COOC[k].sup* là độ phổ biến của *item* hạt nhân thứ *k*; *Index_COOC[k].cooc* là các *item* đồng xuất hiện cùng *item* hạt nhân thứ *k*; và *Index_COOC[k].looc* là các *item* xuất hiện cùng *item* hạt nhân thứ *k* ít nhất trong một giao dịch. Mã giả thuật toán 1 - xây dựng bảng *Index_COOC* như sau:

- *Đầu vào*: Dữ liệu giao dịch *D*.
- *Đầu ra*: Mảng *Index_COOC*; Ma trận *dataset BiM*.
- *Chi tiết thuật toán*:

1 Với mỗi phần tử *k* của mảng *Index_COOC* thực hiện:
2 *Index_COOC[k].item* = i_k
3 *Index_COOC[k].sup* = 0
4 *Index_COOC[k].cooc*= 2^m - 1
5 *Index_COOC[k].looc*= 0
6 Với mỗi giao dịch t_i thực hiện:
7 Lưu giao dịch t_i vào ma trận *BiM*
8 Với mỗi item *k* có trong giao dịch t_i thực hiện:
9 *Index_COOC[k].cooc* = *Index_COOC[k].cooc* AND *vectorbit*(t_i)
10 *Index_COOC[k].looc* = *Index_COOC[k].looc* OR *vectorbit*(t_i)
11 *Index_COOC[k].sup* = *Index_COOC[k].sup* + 1
12 Sắp xếp mảng *Index_COOC* tăng dần theo *sup*
13 Trả về mảng *Index_COOC*, ma trận *BiM*

Từ Dòng 1 đến Dòng 5 là các bước khởi tạo cho mảng *Index_COOC*. Dòng 6 duyệt dữ liệu giao dịch, ứng với từng giao dịch ta xem xét có chứa *item* thứ *k* thì thực hiện phép toán AND trên *bit* để xác định các *item* đồng xuất hiện với *item k* (Dòng 9) và thực hiện phép toán OR trên bit để xác định các *item* xuất hiện với *item k* ít nhất trong một giao dịch, nhưng không là đồng xuất hiện (Dòng 10). Thuật toán được minh họa trên dữ liệu giao dịch D được mô tả theo các bước như trong Bảng 4.

Bảng 4. Minh họa thuật toán xây dựng mảng *Index_COOC* trên dữ liệu *D*

item	A	B	C	D	E	F	G	H
Khởi tạo mảng *Index_COOC* với *cooc* và *looc* biểu diễn dạng bit, số *item* là *m = 8*								
sup	0	0	0	0	0	0	0	0
cooc	11111111	11111111	11111111	11111111	11111111	11111111	11111111	11111111
looc	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
Đọc giao dịch t_1: *{A, C, E, F}* có biểu diễn dạng bit là 10101100								
sup	1	0	1	0	1	1	0	0
cooc	10101100	11111111	10101100	11111111	10101100	10101100	11111111	11111111
looc	10101100	00000000	10101100	00000000	10101100	10101100	00000000	00000000
Đọc giao dịch t_2: *{A, C, G}* có biểu diễn dạng bit là 10100010								
sup	2	0	2	0	1	1	1	0
cooc	10100000	11111111	10100000	11111111	10101100	10101100	10100010	11111111
looc	10101110	00000000	10101110	00000000	10101100	10101100	10100010	00000000
Đọc giao dịch t_3: *{E, H}* có biểu diễn dạng bit là 00001001								
sup	2	0	2	0	2	1	1	1
cooc	10100000	11111111	10100000	11111111	00001000	10101100	10100010	00001001
looc	10101110	00000000	10101110	00000000	10101101	10101100	10100010	00001001
Đọc giao dịch t_4: *{A, C, D, F, G}* có biểu diễn dạng bit là 10110110								
sup	3	0	3	1	2	2	2	1
cooc	10100000	11111111	10100000	10110110	00001000	10100100	10100010	00001001
looc	10111110	00000000	10111110	10110110	10101101	10111110	10110110	00001001

Bảng 4. Minh họa thuật toán xây dựng mảng *Index_COOC* trên dữ liệu *D* (tiếp theo)

item	A	B	C	D	E	F	G	H
Đọc giao dịch t_5: *{A, C, E, G}* có biểu diễn dạng bit là 10101010								
sup	4	0	4	1	3	2	3	1
cooc	10100000	11111111	10100000	10110110	00001000	10100100	10100010	00001001
looc	10111110	00000000	10111110	10110110	10101111	10111110	10111110	00001001
Đọc giao dịch t_6: *{E}* có biểu diễn dạng bit là 00001000								
sup	4	0	4	1	4	2	3	1
cooc	10100000	11111111	10100000	10110110	00001000	10100100	10100010	00001001
looc	10111110	00000000	10111110	10110110	10101111	10111110	10111110	00001001
Đọc giao dịch t_7: *{A, B, C, E}* có biểu diễn dạng bit là 11101000								
sup	5	1	5	1	5	2	3	1
cooc	10100000	11101000	10100000	10110110	00001000	10100100	10100010	00001001
looc	11111110	11101000	11111110	10110110	11101111	10111110	10111110	00001001
Đọc giao dịch t_8: *{A, C, D}* có biểu diễn dạng bit là 10110000								
sup	6	1	6	2	5	2	3	1
cooc	10100000	11101000	10100000	10110000	00001000	10100100	10100010	00001001
looc	11111110	11101000	11111110	10110110	11101111	10111110	10111110	00001001
Đọc giao dịch t_9: *{A, B, C, E, G}* có biểu diễn dạng bit là 11101010								
sup	7	2	7	2	6	2	4	1
cooc	10100000	11101000	10100000	10110000	00001000	10100100	10100010	00001001
looc	11111110	11101010	11111110	10110110	11101111	10111110	11111110	00001001
Đọc giao dịch t_{10}: *{A, C, E, F, G}* có biểu diễn dạng bit là 10101110								
sup	8	2	8	2	7	3	5	1
cooc	10100000	11101000	10100000	10110000	00001000	10100100	10100010	00001001
looc	11111110	11101010	11111110	10110110	11101111	10111110	11111110	00001001

Thuật toán 1 trả về *Index_COOC* sắp tăng theo độ phổ biến của *item* theo Bảng 5.

Bảng 5. Mảng *Index_COOC* có thứ tự tăng theo độ phổ biến của *item*

item	H	B	D	F	G	E	A	C
sup	1	2	2	3	5	7	8	8
cooc	E	A, C, E	A, C	A, C	A, C	Ø	C	A
looc	Ø	G	F, G	D, E, G	B, D, E, F	A, B, C, F, G, H	B, D, E, F, G	B, D, E, F, G

Định nghĩa 8: Cho $i_k \in I(i_1 \prec i_2 \prec ... \prec i_m)$ thứ tự theo độ phổ biến, ta gọi i_k là *item hạt nhân*. Tập $X_{lexcooc} \subseteq I$ gọi *đồng xuất hiện có thứ tự* với *item* i_k: $X_{lexcooc}$ là tập các *item* xuất hiện cùng i_k và $\pi(i_k) \equiv \pi(i_k \cup i_j)$, $i_k \prec i_j$, $\forall i_j \in X_{lexcooc}$. Ký hiệu, $lexcooc(i_k) = X_{lexcooc}$.

Định nghĩa 9: Cho $i_k \in I(i_1 \prec i_2 \prec ... \prec i_m)$ thứ tự theo độ phổ biến, ta gọi i_k là *item hạt nhân*. Tập $Y_{lexlooc} \subseteq I$ chứa các *item* xuất hiện có thứ tự cùng với i_k *ít nhất trong một giao dịch*, nhưng không đồng xuất hiện: $1 \le |\pi(i_k \cup i_{lexlooc})| < |\pi(i_k)|, \forall i_{lexlooc} \in Y_{lexlooc}$. Ký hiệu, $lexlooc(i_k)=Y_{lexlooc}$. Bổ sung các dòng lệnh 14, 15 và 16 vào Thuật toán 1, ta có như sau:

14 Với mỗi phần tử *j* của mảng *Index_COOC*:
15 *Index_COOC*[k].*cooc*= *lexcooc*(i_k)
16 *Index_COOC*[k].*looc*= *lexlooc*(i_k)

Chỉ có *itemset* đồng xuất hiện của *item C* cần hiệu chỉnh. Ta có, *cooc(C) = {A}* và $A \pi C$, nên *lexcooc(C) = {∅}*. Tương tự, ta có *looc(G) = {B, D, E, F}* và $B, D \pi F \pi G \pi E$, nên *lexlooc(G) = {E}*. Sau khi thực hiện dòng 14, 15 và 16, ta có kết quả như trong Bảng 6.

Bảng 6. Mảng *Index_COOC* có thứ tự

item	H	B	D	F	G	E	A	C
sup	1	2	2	3	5	7	8	8
cooc	E	E, A, C	A, C	A, C	A, C	Ø	C	Ø
looc	Ø	G	F, G	G, E	E	A, C	Ø	Ø

Bổ đề 1: $\forall i_k \in I$, nếu $sup(i_k) \ge minsup$ và $lexcooc(i_k)=X_{lexcooc}$ thì $sup(i_k \cup X_{lexcooc}) \ge minsup$.

Chứng minh: Theo Định nghĩa 8, khi đó, $\pi(i_k) \equiv \pi(i_k \cup X_{lexcooc})$ mà $sup(i_k) \ge minsup$, suy ra $sup(i_k \cup X_{lexcooc}) \ge minsup$.

Ví dụ 7: Xem *item B* là *item* hạt nhân với $sup(B) = 2 \ge minsup = 2$, ta xác định được các *item* cùng xuất hiện với *B - lexcooc(B) = {E, A, C}* và $sup(BEAC) = 2 \ge minsup$.

Bổ đề 2: $\forall i_k \prec i_j$ với $i_j \in Y_{lexlooc}$, nếu $sup(i_k \cup i_j) \ge minsup$ và $lexcooc(i_k)=X_{lexcooc}$ thì $sup\left(i_k \cup i_j \cup X_{lexcooc}\right) \ge minsup$.

Chứng minh: Theo Định nghĩa 8 và 9. Khi đó, $|\pi(i_k \cup i_j)| < |\pi(i_k)| = |\pi(i_k \cup X_{lexcooc})|, \forall i_j \in Y_{lexlooc}$ mà $sup(i_k \cup i_j) \ge minsup$, suy ra $sup(i_k \cup i_j \cup X_{lexcooc}) \ge minsup$.

Ví dụ 8: Xem *item F* là *item* hạt nhân với $sup(B) = 2 \ge minsup = 2$, ta xác định được các *item* cùng xuất hiện với *B - lexcooc(B) = {E, A, C}* và $sup(BEAC) = 2 \ge minsup$.

3.3. Thuật toán MAXLEN-FI

Thuật toán MAXLEN-FI khai thác nhanh tập phổ biến có chiều dài tối đa dựa trên mảng *Index_COOC* chứa các *itemset* đồng xuất hiện và *itemset* xuất hiện ít nhất trong cùng một giao dịch với từng *item hạt nhân* (*có thứ tự*).

Định nghĩa 10: $\forall i_k \in I$, $sup(i_k) \ge minsup$, gọi *MAXLEN* là chiều dài tối đa của tập mục phổ biến tiềm năng, bao gồm *item* hạt nhân i_k và các *item* đồng xuất hiện thỏa ngưỡng *minsup*. Nghĩa là:

$$MAXLEN = max(|lexcooc(i_k)|+1),\ \forall i_k \in I: sup(i_k) \geq minsup \quad (3)$$

Bổ đề 3: $\forall i_k \in I$, nếu $sup(i_k) \geq minsup$ và $|lexcooc(i_k)+lexlooc(i_k)+1| < MAXLEN$ thì i_k không sinh ra *itemsets* phổ biến có chiều dài tối đa.

Chứng minh: theo Bổ đề 1 và 2, ta có: $sup(i_k \cup i_j \cup X_{lexcooc}) \geq minsup$, $\forall i_j \in Y_{lexlooc}$ mà $sup(i_k \cup i_j \cup X_{lexcooc}) < MAXLEN$ nghĩa là $\notin LFI$.

Ví dụ 9: Dữ liệu giao dịch $\mathcal{D}$ trong Bảng 1 và *minsup = 2*, ta có *MAXLEN = 4* (*lexcooc(B) = {E, A, C}*). Theo Bổ đề 3, ta không xem xét các *item* hạt nhân *E, A,* và *C*. Mã giả Thuật toán 2, khai thác tập phổ biến có chiều dài tối đa MAXLEN-FI như sau:

- *Đầu vào*: Mảng *Dataset, Index_COOC*
- *Đầu ra*: Tập phổ biến có chiều dài tối đa *LFI*
- *Chi tiết thuật toán*:

```
1   Với mỗi Index_COOC[k].sup ≥ minsup
2   MAXLEN = max(|Index_COOC[k].cooc|)+1
3   LFI_temp ={∅}
4   Với mỗi Index_COOC[k].sup ≥ minsup
5           Co = Index_COOC[k].cooc
6           Lo = Index_COOC[k].looc
7           Nếu (|Co|+1) ≥ MAXLEN) thì
8                   LFI_temp = LFI_temp ∪{ik ∪Co}
9           Nếu (|Co|+|Lo|+1) ≥ MAXLEN) thì
10                  Lsub ← các tập con của Lo(sắp giảm theo số lượng item)
11                  Với mỗi lsub ∈ Lsub
12                          Nếu (sup(Co ∪ lsub) ≥ minsup) thì
13                                  Nếu (|Co ∪ lsub |= MAXLEN) thì
14                                          LFI_temp = LFI_temp ∪{ Co ∪ lsub }
15                                  Nếu (|Co ∪ lsub |> MAXLEN) thì
16                                          MAXLEN = |Co ∪ lsub |
17                                          LFI_temp = {Co ∪ lsub }
18  LFI = LFI_temp
19  Trả về tập phổ biến LFI
```

Ví dụ 10: Cho dữ liệu giao dịch $\mathcal{D}$ trong Bảng 1 và *minsup* = 2. Sau khi thực hiện Thuật toán 1, ta có mảng chứa các *itemset* đồng xuất hiện như Bảng 6. Các *item B, D, F, G, E, A,* và *C* thỏa ngưỡng *minsup = 2*. Dòng 2 cho kết quả của *MAXLEN* = 4;

Xét *item B – sup(B) = 2 ≥ minsup, Co[B] = {E, A, C}, LFI_temp = {(BEAC, 2)}* và *Lo[B] = {G}*, ta có *sup(BEACG) = 1 < minsup*, không thêm vào *LFI_temp*; Xét *item D – sup(D) = 2 ≥ minsup, Co[D] = {A, C}, Lo[D] = {F, G}* và *Lsub[D] = {FG, F, G}*. Ta có *sup(DACFG) = sup(DACF) = sup(DACG) = 1 < minsup*, không thêm vào *LFI_temp*; Xét *item F – sup(F) = 2 ≥ minsup, Co[F] = {A, C}, Lo[F] = {G, E}* và *Lsub[F] = {GE, G, E}*. Ta có *sup(FACGE) = 1 < minsup*, không thêm vào *LFI_temp*; *sup(FACG) = sup(FACE) = 2 ≥ minsup* và *|FACG| = |FACE| = MAXLEN*, lúc này *LFI_temp = {(BEAC, 2), (FACG, 2), (FACE, 2)}*; Xét *item G – sup(G) = 5 ≥ minsup, Co[G] = {A, C}, Lo[G] = {E}* và *Lsub[G] = {E}*. Ta có *sup(GACE) = 3 ≥ minsup* và *|GACE| = MAXLEN = 4* nên *LFI_temp = {(BEAC, 2), (FACG, 2), (FACE, 2), (GACE, 3)}*; Còn lại *item E, A,* và *C* không xét theo

dòng 9 (Bổ đề 3); Dữ liệu giao dịch *D* và *minsup = 2*, ta có tập mục phổ biến có chiều dài tối đa là *LFI = {(BEAC, 2), (FACG, 2), (FACE, 2), (GACE, 3)}*.

4. KẾT QUẢ THỰC NGHIỆM

Kết quả được thực nghiệm trên máy tính CF-74, Core Duo 2.0 GHz, 4GB RAM, thuật toán cài đặt trên C#, Microsoft Visual Studio 2010 với hai nhóm dữ liệu:

- Nhóm dữ liệu thực có *mật độ dày*: Sử dụng dữ liệu thực từ kho dữ liệu về học máy của Đại học California (Lichman, 2013) bao gồm hai tập *Chess* và *Mushroom*;
- Nhóm dữ liệu giả lập có *mật độ thưa*: Sử dụng phần mềm phát sinh dữ liệu giả lập của Trung tâm Nghiên cứu IBM Almaden (IBM Almaden Research Center, 2004) gồm hai tập *T10I4D100K* và *T40I10D100K*.

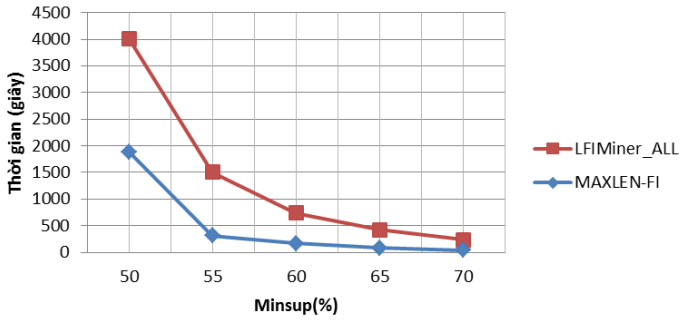
Trong phần thực nghiệm, chúng tôi sử dụng bốn tập dữ liệu được mô tả ở Bảng 7 và so sánh thuật toán đề xuất MAXLEN-FI với thuật toán LFIMiner_ALL do Hu và ctg. (2008) đề xuất. Để so sánh về thời gian thực hiện giữa thuật toán đề xuất và thuật toán LFIMiner_ALL, chúng tôi chỉ tiến hành so sánh theo từng ngưỡng *minsup* và cả hai thuật toán đều cho cùng kết quả số lượng *itemsets* phổ biến có chiều dài tối đa.

Chúng tôi sử dụng hai tập *Chess* và *Mushroom* để so sánh hiệu suất của thuật toán đề xuất MAXLEN-FI với thuật toán LFIMiner_ALL. Hình 1a và Hình 1b cho thấy thời gian thực hiện thuật toán MAXLEN-FI khai thác tập phổ biến có chiều dài tối đa theo các ngưỡng *minsup* khác nhau trên hai tập dữ liệu *Chess* và *Mushroom* nhanh hơn thuật toán LFIMiner_ALL. Tuy nhiên, khi quan sát trên dữ liệu *Mushroom* (mật độ 19.3%) ở Hình 1b thì hiệu suất của thuật toán MAXLEN-FI cao hơn rất nhiều so với thuật toán LFIMiner_ALL trên dữ liệu *Chess* (mật độ 49.3%) ở Hình 1a.

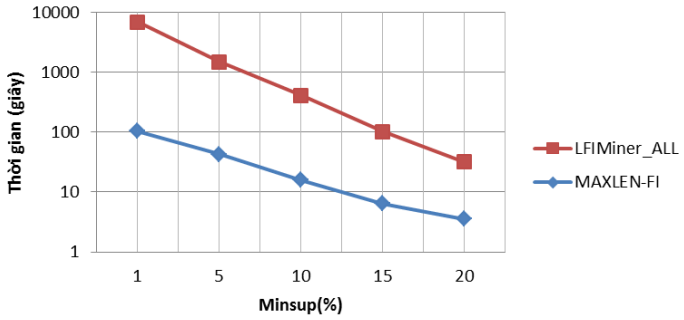
Bảng 7. Dữ liệu thực nghiệm

Tên dữ liệu	Số mục hàng	Số giao dịch	Số mục hàng trung bình/ giao dịch	Mật độ (%)
Chess	75	3,196	37	49.3%
Mushroom	119	8,142	23	19.3%
T10I4D100K	870	100,000	10	1.1%
T40I10D100K	942	100,000	40	4.2%

Hình 2a và 2b cho thấy thuật toán LFIMiner_ALL không thích hợp với dữ liệu thưa khi thay đổi *minsup*. Thuật toán MAXLEN-FI ổn định và nhanh hơn thuật toán LFIMiner_ALL. Đồng thời hiệu suất của thuật toán MAXLEN-FI rất cao so với LFIMiner_ALL trên dữ liệu thưa. Kết quả trên cũng cho thấy thuật toán khai thác tập phổ biến MAXLEN-FI tốt hơn thuật toán LFIMiner_ALL. Thuật toán MAXLEN-FI cần được so sánh thêm với thuật toán khác. Ngoài ra, thuật toán cũng cần thực nghiệm thêm trên nhiều tập dữ liệu có mật độ khác nhau, cũng như trên nhiều dữ liệu cỡ lớn. Bảng 8 thống kê *số lượng itemset* phổ biến có chiều dài tối đa và chiều dài tối đa của *itemset* phổ biến trong LFI trên hai nhóm dữ liệu thực nghiệm với các ngưỡng *minsup* khác nhau.



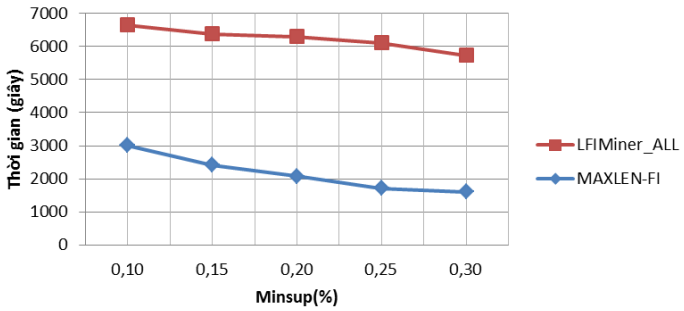
(a)



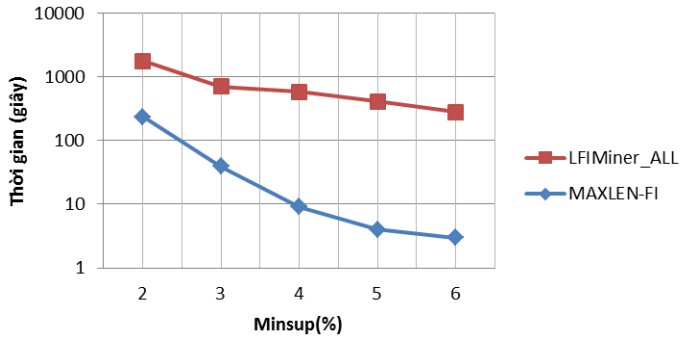
(b)

Hình 1. Thời gian thực hiện MAXLEN-FI và LFIMiner_ALL trên dữ liệu *Chess* và *Mushroom* với các ngưỡng *minsup* khác nhau

Ghi chú: (a) *Chess*; và (b) *Mushroom*.



(a)



(b)

Hình 2. Thời gian thực hiện MAXLEN-FI và LFIMiner_ALL trên dữ liệu thực *T10I4D100K* và *T40I10D100K* với các ngưỡng *minsup* khác nhau

Ghi chú: (a) *T10I4D100K*; và (b) *T40I10D100K*.

Bảng 8. LFI trên dữ liệu thực nghiệm theo *minsup*

Tên dữ liệu	*Minsup* (%)	Số lượng *itemset* phổ biến có chiều dài tối đa	Chiều dài tối đa của *itemset* phổ biến trong LFI
Chess	50	4	16
	55	6	15
	60	8	14
	65	22	13
	70	1	13
Mushroom	1	20	19
	5	24	17
	10	8	15
	15	2	15
	20	1	15
T10I4D100K	0,10	2	10
	0,15	2	10
	0,20	1	9
	0,25	10	8
	0,30	6	7
T40I10D100K	2	6	3
	3	302	2
	4	64	2
	5	15	2
	6	5	2

5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Trong bài viết này, chúng tôi đã đề xuất kiến trúc khai thác tập phổ biến có chiều dài tối đa gồm hai giai đoạn. Giai đoạn một là tính nhanh mảng *Index_COOC* chứa các *itemset* đồng xuất hiện và xuất hiện với *item hạt nhân* ít nhất trong một giao dịch. Đây là thuật toán cải tiến từ thuật toán của chính nhóm tác giả. Giai đoạn hai là đề xuất thuật toán MAXLEN-FI khai thác hiệu quả tập phổ biến có chiều dài tối đa dựa trên mảng *Index_COOC*. Với kiến trúc như trên, khi người dùng khai thác tập phổ biến với giá trị ngưỡng *minsup* khác thì thuật toán đề xuất chỉ thực hiện khai thác tập phổ biến trên mảng *Index_COOC* đã tính ở lần khai thác trước làm giảm đáng kể thời gian xử lý. Với những kết quả đạt được từ thuật toán đề xuất ở trên, trong tương lai, chúng tôi sẽ mở rộng thuật toán để có thể khai thác tập phổ biến có chiều dài tối đa trên dữ liệu giao dịch có *trọng số*, cũng như song song hóa thuật toán trên để có thể khai thác nhanh tập phổ biến có chiều dài tối đa trên các thiết bị cầm tay, điện thoại thông minh có bộ xử lý đa nhân, các hệ thống tính toán phân tán như Hadoop và Spark.

LỜI CẢM ƠN

Nhóm tác giả cảm ơn sự hỗ trợ từ Trường Đại học Khoa học Xã hội và Nhân văn, Đại học Quốc gia TP. Hồ Chí Minh.

TÀI LIỆU THAM KHẢO

Agrawal, R., Imilienski, T., & Swami, A. (1993). *Mining association rules between sets of large databases*. Paper presented at The ACM SIGMOD International Conference on Management of Data, USA.

Burdick, D., Calimlim, M., & Gehrke, J. (2001). *MAFIA: A maximal frequent itemset algorithm for transactional databases*. Paper presented at The 17th International Conference on Data Engineering, Germany.

Gouda, K., & Zaki, M. J. (2005). *GenMax: An efficient algorithm for mining maximal frequent itemsets*. Paper presented at The IEEE International Conference on Data Mining and Knowledge Discovery, China.

Han, J., Pei, J., Yin, Y., & Mao, R. (2004). Mining frequent patterns without candidate generation: A frequent pattern tree approach. *Data Mining and Knowledge Discovery, 8*(1), 53-87.

Hu, T., Sung, S. Y., Xiong, H., & Fi, Q. (2008). Discovery of maximum length frequent itemsets. *Information Sciences: An International Journal, 178*(1), 69-87.

IBM Almaden Research Center. (2004). *Almaden*. Retrieved from http://www.almaden.ibm.com.

Lê, H. B., & Phan, T. H. (2016). *DYN-FI: Thuật toán hiệu quả khai thác tập phổ biến trên dữ liệu giao dịch với ngưỡng phổ biến tối thiểu động*. Bài báo được trình bày tại Hội thảo Một số vấn đề chọn lọc về Công nghệ Thông tin và Truyền thông lần thứ 19, Việt Nam.

Lichman, M. (2013). *UCI machine learning repository*. Retrieved from http://archive.ics.uci.edu/ml.

Song, W., & Yang, B. (2008). Index-BitTableFI: An improved algorithm for mining frequent itemsets. *Knowledge-Based Systems, 21,* 507-513.

Tran, A. T., Ngo, T. P., & Nguyen, K. A. (2011). *An efficient algorithm for discovering maximal frequent item sets*. Paper presented at The IEEE International Conference on Knowledge Systems Engineering, Malaysia.

Wang, J., Han, J., & Pei, J. (2003). *CLOSET+: Searching for the best strategies for mining frequent closed itemsets*. Paper presented at The 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, USA.

Zaki, M. J., & Hsiao, C. (2002). *CHARM: An efficient algorithm for closed association rule mining.* Paper presented at The 2nd SIAM International Conference on Data Mining, USA.