Kỷ yếu Hội nghị Khoa học Quốc gia lần thứ IX "Nghiên cứu cơ bản và ứng dụng Công nghệ thông tin (FAIR'9)"; Cần Thơ, ngày 4-5/8/2016
DOI: 10.15625/vap.2016.00090

THUẬT TOÁN HIỆU QUẢ KHAI THÁC TẬP PHỔ BIẾN TỐI ĐẠI TRÊN CƠ SỞ DỮ LIỆU GIAO DỊCH LỚN

Lê Hoài Bắc[1], Phan Thành Huấn[2]

[1] Khoa Công nghệ thông tin, Trường Đại học Khoa học Tự nhiên, Đại học Quốc gia Tp. Hồ Chí Minh
[2] Bộ môn Tin học, Trường Đại học Khoa học Xã hội và Nhân văn, Đại học Quốc gia Tp. Hồ Chí Minh
lhbac@fit.hcmus.edu.vn, huanphan@hcmussh.edu.vn

TÓM TẮT— *Khai thác luật kết hợp, một trong những kỹ thuật quan trọng nhất và được nghiên cứu nhiều nhất trong khai thác dữ liệu. Khai thác tập phổ biến tối đại là một trong những vấn đề cơ bản nhất trong khai thác luật kết hợp. Hầu hết các thuật toán tìm tập phổ biến tối thiểu trước, từ tập phổ biến tối thiểu suy ra tập phổ biến tối đại. Những phương pháp này tốn nhiều thời gian để tìm tập phổ biến tối đại. Để khắc phục vấn đề này, chúng tôi đề xuất một cách tiếp cận mới để tìm tập phổ biến tối đại trên cơ sở dữ liệu giao dịch lớn: kỹ thuật nén hiệu quả cơ sở dữ liệu giao dịch lớn, dùng cấu trúc dữ liệu lưu trữ dạng bit và tập chỉ mục chứa các item đồng xuất hiện để chiếu tính nhanh tập phổ biến tối đại. Sau cùng, chúng tôi trình bày kết quả thực nghiệm, cho thấy rằng thuật toán đề xuất tốt hơn so với các thuật toán hiện hành.*

Từ khóa— *Khai thác luật kết hợp, cơ sở dữ liệu giao dịch lớn, tập phổ biến tối đại, itemset đồng xuất hiện.*

I. 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 quan hệ giữa các giá trị dữ liệu trong cơ sở dữ liệu. 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 R. Agrawal, T. Imielinski và A. Swami đề xuất vào năm 1993 [1], phân tích cơ sở dữ liệu giao dịch, phát hiện các mối quan 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. Ngoài ra, có thể áp dụng tri thức này để dự đoán số lượng các mặt hàng được bán chạy nhất trong thời gian sắp tới. Tổng hợp các tri thức này để lên kế hoạch cho hoạt động, sản xuất, kinh doanh một cách thuận tiện hơn nhằm giảm bớt thời gian thống kê, tìm hiểu thị trường,...

Các thuật toán được đề xuất để khai thác luật kết hợp chia thành 2 giai đoạn [1, 2]:

Giai đoạn 1: Tìm tất cả các tập phổ biến (**FI**) từ CSDL nghĩa là tìm tất cả các tập mục X thoả mãn *sup* (X) ≥ *minsup*. Đây là giai đoạn tốn khá nhiều thời gian xử lý.

Giai đoạn 2: Sinh các luật tin cậy kết hợp từ các tập phổ biến tìm thấy ở giai đoạn thứ nhất. Giai đoạn này tương đối đơn giản và tốn kém ít thời gian hơn so với giai đoạn trên.

Trong thực tế, giai đoạn thứ nhất chiếm hầu hết thời gian cho toàn quá trình khai thác luật kết hợp [1, 2]. Nhằm cải tiến về mặt thời gian, đề xuất thay thế tập **FI** bằng tập nhỏ hơn, gọi là tập phổ biến đóng (**CFI**) [3, 4], tập **CFI** vẫn đầy đủ thông tin cho giai đoạn thứ hai. Ngoài ra, cũng có đề xuất cải tiến thay thế tập **CFI** bằng tập nhỏ hơn, gọi là tập phổ biến tối đại (**MFI**) [5, 6, 7], vẫn đảm bảo đầy đủ thông tin.

Một số thuật toán điển hình khai thác tập phổ biến tối đại **MFI [5, 6, 7]**:

Thuật toán Depth-Project [5]: Thuật toán tìm kiếm theo chiều sâu trên cây thứ tự từ điển của tập mục. Duyệt theo *chiều sâu* và *chiều rộng* trên dàn tập mục. Thuật toán tỉa cả tập con không phổ biến và tập cha phổ biến. Cơ sở dữ liệu được biểu diễn dạng vectơ bit, giảm đáng kể chi phí xác định độ phổ biến.

Thuật toán Mafia [6]: Thuật toán tích hợp tìm kiếm theo chiều sâu trên dàn tập mục. Và dùng 3 cách để loại bỏ các tập không phổ biến tối đại. Cơ sở dữ liệu được biểu diễn dạng vectơ bit theo chiều dọc. Thuật toán đặc biệt hiệu quả khi các tập phổ biến trong CSDL là rất dài.

Thuật toán GenMax [7]: Để tìm tập phổ biến tối đại, phương pháp đơn giản nhất là phương pháp tìm kiếm đệ quy quay lui. Tuy nhiên phương pháp này có không gian tìm kiếm lớn, chi phí tính toán và chi phí duyệt CSDL lớn. Thuật toán **GenMax** trong tối ưu thuật toán tìm kiếm đệ quy quay lui, sử dụng một kỹ thuật để loại bỏ nhánh không cần thiết trong không gian tìm kiếm và tính độ phổ biến nhanh.

Hầu hết các thuật toán khai thác tập phổ biến tối đại đã được các tác giả trên thế giới đề xuất, đều có nhược điểm lớn và không thực tế: mỗi lần cần khai thác tập phổ biến tối đại với *minsup* khác thì thuật toán thực hiện tính toán lại độ phổ biến của các tập mục, phát sinh lại cây tìm kiếm hoặc dàn tập mục tương ứng và xác định tập phổ biến tối đại thỏa *minsup* mới. Trong thực tế, khi cần khai thác luật kết hợp thì người dùng có thể yêu cầu thực hiện khai thác luật kết hợp thỏa ngưỡng *minsup* và *minconf* trong nhiều chuỗi thao tác liên tiếp nhau. Vì vậy, các thuật toán đã được các tác giả trên thế giới chưa đáp ứng nhu cầu thực tế. Ngoài ra, các thuật toán trên chưa đáp ứng trên cơ sở dữ liệu giao dịch cỡ lớn và rất lớn trong kỷ nguyên bùng nổ dữ liệu như hiện nay.

Để đáp ứng tốt trong kỷ nguyên bùng nổ dữ liệu, cần xây dựng những hệ thống lưu trữ dữ liệu thực sự hiệu quả và thông minh hơn theo hướng tích hợp những tính năng như ảo hóa lưu trữ, nén dữ liệu, chống trùng lắp dữ liệu và tự động phân loại dữ liệu nhằm cân đối tốt nhất giữa chi phí lưu trữ dữ liệu, tốc độ và các phương thức truy xuất dữ liệu.

Nhằm đóng góp trong lĩnh vực khai thác dữ liệu thích ứng với cơ sở dữ liệu lớn, nhóm tác giả đề xuất thuật toán khai thác hiệu quả tập phổ biến tối đại **LA-MFI** trên cơ sở dữ liệu giao dịch cỡ lớn:

- Thuật toán nén cơ sở dữ liệu giao dịch cỡ lớn trên nền tảng kiến thức đại số Bool;
- Thuật toán xác định mảng **Index_COOC** gồm *itemset* đồng xuất hiện của từng *item hạt nhân*;
- Thuật toán khai thác hiệu quả tập phổ biến tối đại **LA-MFI** dựa trên mảng **Index_COOC**.

Trong phần II, bài báo trình bày các khái niệm cơ bản, tiêu chuẩn dữ liệu và đề xuất thuật toán nén cơ sở dữ liệu giao dịch cỡ lớn trên nền tảng đại số Bool. Phần III, xây dựng thuật toán xác định mảng chứa *itemset* đồng xuất hiện của từng *item* hạt nhân và thuật toán hiệu quả khai thác tập phổ biến tối đại. Kết quả thực nghiệm được trình bày trong phần IV và kết luận ở phần V.

II. CÁC VẤN ĐỀ LIÊN QUAN

A. Một số khái niệm cơ bản

Cho $I = \{I_1, I_2,..., I_m\}$ là tập gồm m thuộc tính riêng biệt, mỗi thuộc tính gọi là *item*. Tập mục $X \subseteq I$ gọi là *itemset*, tập mục có k mục gọi là *k-itemset*. $\mathcal{D}$ là cơ sở dữ liệu, gồm n bản ghi phân biệt gọi là tập các giao dịch $T = \{T_1, T_2,..., T_n\}$, mỗi giao dịch $T_i=\{I_{k1}, I_{k2},..., I_{kj}\}$, $I_{kj} \in I$ $(1 \le k_j \le m)$.

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 được gọi là tập mục phổ biến nếu $sup(X) \ge minsup$, trong đó *minsup* là độ phổ biến tối thiểu. Tập các tập mục phổ biến gọi là tập phổ biến, ký hiệu là **FI**.

Định nghĩa 3: Cho $X \subseteq I$, X được gọi là tập mục phổ biến tối đại nếu X là tập mục phổ biến và không có tập cha là tập mục phổ biến. Tập các tập mục phổ biến tối đại gọi là tập phổ biến tối đại, ký hiệu là **MFI**.

Bảng 1. Cơ sở dữ liệu giao dịch $\mathcal{D}$

Mã giao dịch	Tập mục
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: Cho CSDL $\mathcal{D}$ như trong *Bảng 1*, có 8 *item* riêng biệt $I = \{A, B, C, D, E, F, G, H\}$ và 10 giao dịch $T = \{T1, T2, T3, T4, T5, T6, T7, T8, T9, T10\}$ phân biệt. Trong *Bảng 2*, cho thấy tập phổ biến **FI** và tập phổ biến tối đại **MFI** chứa *k-itemset* với *minsup* = 2 (2 giao dịch) và *minsup* = 5 (5 giao dịch). Trường hợp *minsup* = 2, $|FI| = 39$ và $|MFI| = 5$, tỷ suất $|MFI|/|FI| = 5/39 \times 100\% = 13\%$; *minsup* = 5, tỷ suất $|MFI|/|FI| = 2/11 \times 100\% = 18\%$. Qua đó, chúng ta thấy tập phổ biến tối đại **MFI** nhỏ hơn rất nhiều so với tập phổ biến **FI**.

Bảng 2. Tập FI, MFI với minsup = 2 và minsup = 5 trên CSDL $\mathcal{D}$

k-itemset	Tập phổ biến **FI** (*minsup*=2)	Tập phổ biến tối đại **MFI** (*minsup*=2)	Tập phổ biến **FI** (*minsup*=5)	Tập phổ biến tối đại **MFI** (*minsup*=5)
1	D, B, F, G, E, C, A		G, A, C, E	
2	AB, AD, AF, BC, BE, CD, CF, EF, EG, AG, CG, AE, CE, AC, FG		AG, CG, AE, CE, AC	
3	ABC, ABE, ACD, ACF, AEF, BCE, CEF, AEG, CEG, ACG, ACE, AFG, CFG	***ACD***	ACG, ACE	***ACG, ACE***
4	ABCE, ACFG, ACEF, ACEG	***ABCE, ACFG, ACEF, ACEG***		

B. Tiêu chuẩn dữ liệu

Trong lĩnh vực khai thác dữ liệu, kỹ thuật khai thác luật kết hợp rất quan trọng. Nhằm rút trích các thông tin cần thiết hữu ích từ lượng dữ liệu lớn hàng Zbyte, Ybyte. Rất cần thiết khi định ra tiêu chuẩn độ lớn dữ liệu. Tác giả Youcef Djenouri và cộng sự [8] có đề cập đến tiêu chuẩn độ lớn dữ liệu như sau:

Cơ sở dữ liệu giao dịch cỡ trung (***Medium***): số giao dịch từ *6.000* đến *90.000*, số item từ *500* đến *16.000*, số item trung bình trên giao dịch từ *2* đến *500*.

Cơ sở dữ liệu giao dịch cỡ lớn (***Large***): số giao dịch từ *100.000* đến *500.000*, số item từ *1.000* đến *1.600*, số item trung bình trên giao dịch từ *2* đến *10*.

Cơ sở dữ liệu giao dịch cỡ rất lớn (***Big***): số giao dịch trên *1.600.000*, số item trên *500.000*.

Bảng 3. Mô tả tiêu chuẩn dữ liệu

Phân loại CSDL	Tên CSDL	Dung lượng	Số giao dịch	Số item	Số item nhỏ nhất/ giao dịch	Số item lớn nhất/ giao dịch	Số item trung bình/ giao dịch
Cỡ trung	**Pumsb**	16,30 Mb	49.046	2.113	74	74	74
	Pumsb_star	11,03 Mb	49.046	2.088	49	63	50
	Retail	4,07 Mb	88.162	16.470	1	76	10
Cỡ lớn	**T40I10D100K**	15,10 Mb	100.000	1.000	4	77	40
	T40I1KD100K	15,30 Mb	100.000	1.000	16	70	40
	T40I1KD200K	30,50 Mb	200.000	1.000	14	67	38

C. Kỹ thuật nén dữ liệu

Để đáp ứng tốt trong kỷ nguyên bùng nổ dữ liệu, đặc biệt cần xây dựng những hệ thống lưu trữ dữ liệu thực sự hiệu quả và thông minh hơn theo hướng tích hợp những tính năng như ảo hóa lưu trữ, nén dữ liệu, chống trùng lắp dữ liệu và tự động phân loại dữ liệu nhằm cân đối tốt nhất giữa chi phí lưu trữ dữ liệu, tốc độ và các phương thức truy xuất dữ liệu. Phần này, nhóm tác giả tập trung nghiên cứu kỹ thuật nén dữ liệu thích hợp để lưu trữ CSDL giao dịch cỡ lớn.

Nhóm tác giả đề xuất kỹ thuật nén dữ liệu cho CSDL giao dịch lớn, dựa trên nền tảng đại số Bool như sau:

- Mỗi *item* tương ứng với *một biến* Bool;
- Mỗi giao dịch T_i tương ứng một *đơn thức tối tiểu* (minterm) - mt_i;
- Lưu trữ *đơn thức tối tiểu* tương ứng dưới dạng bit.

Cho $I = \{I_1, I_2,..., I_m\}$ là tập gồm m thuộc tính riêng biệt, mỗi thuộc tính gọi là *item*. $\mathcal{D}$ là cơ sở dữ liệu, gồm n bản ghi phân biệt gọi là tập các giao dịch $T = \{T_1, T_2,..., T_n\}$, mỗi giao dịch $T_i=\{I_{k1}, I_{k2},..., I_{kj}\}$, $I_{kj} \in I$ $(1 \le k_j \le m)$.

Cơ sở dữ liệu $\mathcal{D}$ được nén như sau:

- Có m *item* tương ứng với m biến Bool;
- CSDL $\mathcal{D}$ có tối đa $(2^m - 1)$ giao dịch phân biệt tương ứng $(2^m - 1)$ *đơn thức tối tiểu* – mt_i $(1 \le i < 2^m)$;
- Cấu trúc dữ liệu lưu trữ *đơn thức tối tiểu*, gồm: <**pos**: vị trí nhóm, **minterm**: byte lưu trữ đơn thức> – mỗi bộ <**pos**, **minterm**> lưu trữ được nhóm 8 *đơn thức tối tiểu*.
- Đơn thức tối tiểu mt_i được lưu vào bộ <**pos** = (i *div* 8)+1, **minterm**: bit thứ (i *mod* 8) được bật>

Bảng 4. Các giao dịch trên CSDL $\mathcal{D}$ biểu diễn dạng thập phân

Mã giao dịch	A	B	C	D	E	F	G	H	Giá trị thập phân
T1	*1*	0	*1*	0	*1*	*1*	0	0	172
T2	*1*	0	*1*	0	0	0	*1*	0	162
T3	0	0	0	0	*1*	0	0	*1*	9
T4	*1*	0	*1*	*1*	0	*1*	*1*	0	182
T5	*1*	0	*1*	0	*1*	0	*1*	0	170
T6	0	0	0	0	*1*	0	0	0	8
T7	*1*	*1*	*1*	0	*1*	0	0	0	232
T8	*1*	0	*1*	*1*	0	0	0	0	176
T9	*1*	*1*	*1*	0	*1*	0	*1*	0	234
T10	*1*	0	*1*	0	*1*	*1*	*1*	0	174

Bảng 3, biểu diễn các giao dịch trên CSDL $\mathcal{D}$ dưới dạng thập phân tương ứng với thứ tự của các *đơn thức tối tiểu* lần lượt là $\{mt_8, mt_9, mt_{162}, mt_{170}, mt_{172}, mt_{174}, mt_{176}, mt_{182}, mt_{232}, mt_{234}\}$. Áp dụng kỹ thuật nén dữ liệu tương ứng 10 đơn thức tối tiểu sẽ trở thành:

pos	**2**	**21**	**22**	**23**	**30**
minterm	00000***11***	00000***1***00	0***1***0***1***0***1***00	0***1***000000***1***	00000***1***0***1***

Hình 1. Nén CSDL $\mathcal{D}$

Hình 1, Kết quả nén CSDL $\mathcal{D}$ gồm 10 giao dịch tương ứng thành 10 *đơn thức tối tiểu* {mt_8, mt_9, mt_{162}, mt_{170}, mt_{172}, mt_{174}, mt_{176}, mt_{182}, mt_{232}, mt_{234}} vào trong danh sách gồm 5 bộ {<**2**, 3>; <**21**, 4>; <**22**, 84>, <**23**, 65>, <**30**, 5>}.

III. CÁC THUẬT TOÁN

A. Thuật toán sinh các item đồng xuất hiện

Định nghĩa 4: Tập chiếu của *item* I_k trên CSDL $\mathcal{D}$: $\pi(I_k)=\{t\in \mathcal{D} \mid I_k \subseteq t\}$ là tập các giao dịch có chứa *item* I_k.

Định nghĩa 5: Cho $I_k \in I$, ta gọi I_k là *item hạt nhân*. Tập mục $X_{cooc} \subseteq I$ gọi *đồng xuất hiện cùng độ phổ biến* với I_k: X_{cooc} là *itemset* xuất hiện cùng *item hạt nhân* I_k và $\pi(I_k) = \pi(X_{cooc})$. Ký hiệu, $cooc(I_k) = X_{cooc}$.

Ví dụ 2: Cho CSDL $\mathcal{D}$ như trong *Bảng 1*. 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 $sup(B) = sup(ACE) = 2$.

Dưới đây là thuật toán sinh các *item* đồng xuất hiện với từng *item* trong CSDL giao dịch và lưu trữ vào mảng **Index_COOC**. Mỗi phần tử trong **Index_COOC** gồm 3 thành phần sau:

- **Index_COOC[j].***item*: lưu trữ *item hạt nhân thứ j*;
- **Index_COOC[j].***sup*: lưu trữ *độ phổ biến* của *item hạt nhân thứ j*;
- **Index_COOC[j].***cooc*: lưu trữ *itemset* đồng xuất hiện cùng *item hạt nhân thứ j* dưới dạng bit;

Ngoài ra, thuật toán 1 còn thực hiện nén CSDL vào mảng **Dataset**.

Mã giả thuật toán 1. Xây dựng bảng Index_COOC và nén dữ liệu

Đầu vào*:* CSDL $\mathcal{D}$

Đầu ra*:* Mảng **Index_COOC**, mảng **Dataset**

1. Với mỗi phần tử *j* của mảng **Index_COOC** thực hiện:
2. ***Index_COOC*** [j].*item* = Ij
3. ***Index_COOC*** [j].*sup* = 0
4. ***Index_COOC*** [j].*cooc*= 2^m - 1
5. Với mỗi giao dịch T_i thực hiện:
6. Nén giao dịch T_i vào mảng **Dataset**
7. Với mỗi item *j* có trong giao dịch T_i thực hiện:
8. ***Index_COOC*** [j].*cooc* = ***Index_COOC*** [j].*cooc* & *vectorbit*(T_i)
9. ***Index_COOC*** [j].*sup* = ***Index_COOC*** [j].*sup* + 1
10. Sắp xếp mảng **Index_COOC** tăng dần theo *sup*
11. Trả về mảng **Index_COOC,** mảng **Dataset**

Từ dòng 1 đến dòng 4 là các bước khởi tạo cho mảng **Index_COOC**. Dòng 5 duyệt CSDL, ứng với từng giao dịch ta xem xét có chứa *item* thứ *j* thì thực hiện phép toán **AND** trên *bit* để xác định các phần tử cùng xuất hiện với *item j*. Sau cùng, dòng 10 và 11 là sắp xếp mảng **Index_COOC** tăng dần theo độ phổ biến của *item* và trả về.

Khởi tạo mảng **Index_COOC**: (*thành phần cooc được biểu diễn dạng bit*) số item là m = 8

item	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**
sup	0	0	0	0	0	0	0	0
cooc	*11111111*	*11111111*	*11111111*	*11111111*	*11111111*	*11111111*	*11111111*	*11111111*

Đọc giao dịch T1: {A, C, E, F} có biểu diễn dạng bit là **10101100**

item	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**
sup	**1**	0	**1**	0	**1**	**1**	0	0
cooc	**10101100**	11111111	**10101100**	11111111	**10101100**	**10101100**	11111111	11111111

Đọc giao dịch T2: {A, C, G} có biểu diễn dạng bit là **10100010**

item	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**
sup	**2**	0	**2**	0	1	1	**1**	0
cooc	**10100000**	11111111	**10100000**	11111111	10101100	10101100	**10100010**	11111111

Đọc giao dịch T3: {E, H} có biểu diễn dạng bit là **00001001**

item	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**
sup	2	0	2	0	**2**	1	1	**1**
cooc	10100000	11111111	10100000	11111111	**00001000**	10101100	10100010	**00001001**

Đọc giao dịch T4: {A, C, D, F, G} có biểu diễn dạng bit là **10110110**

item	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**
sup	**3**	0	**3**	**1**	2	**2**	**2**	1
cooc	**10100000**	11111111	**10100000**	**10110110**	00001000	**10100100**	**10100010**	00001001

Đọc giao dịch T5: {A, C, E, G} có biểu diễn dạng bit là **10101010**

item	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**
sup	**4**	0	**4**	1	**3**	2	**3**	1
cooc	**10100000**	11111111	**10100000**	10110110	**00001000**	10100100	**10100010**	00001001

Đọc giao dịch T6: {E} có biểu diễn dạng bit là **00001000**

item	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**
sup	4	0	4	1	**4**	2	3	1
cooc	10100000	11111111	10100000	10110110	**00001000**	10100100	10100010	00001001

Đọc giao dịch T7: {A, B, C, E} có biểu diễn dạng bit là **11101000**

item	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**
sup	**5**	**1**	**5**	1	**5**	2	3	1
cooc	**10100000**	**11101000**	**10100000**	10110110	**00001000**	10100100	10100010	00001001

Đọc giao dịch T8: {A, C, D} có biểu diễn dạng bit là **10110000**

item	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**
sup	**6**	1	**6**	**2**	5	2	3	1
cooc	**10100000**	11101000	**10100000**	**10110000**	00001000	10100100	10100010	00001001

Đọc giao dịch T9: {A, B, C, E, G} có biểu diễn dạng bit là **11101010**

item	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**
sup	**7**	**2**	**7**	2	**6**	2	**4**	1
cooc	**10100000**	**11101000**	**10100000**	10110000	**00001000**	10100100	**10100010**	00001001

Đọc giao dịch T10: {A, C, E, F, G} có biểu diễn dạng bit là **10101110**

item	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**
sup	**8**	2	**8**	2	**7**	**3**	**5**	1
cooc	**10100000**	11101000	**10100000**	10110000	**00001000**	**10100100**	**10100010**	00001001

Thuật toán 1, trả về mảng **Index_COOC** tương ứng là *cooc*(A) = {C}, *cooc*(B) = {A, C, E}, *cooc*(C) = {A}, *cooc*(D) = {A, C}, *cooc*(E) = {}, *cooc*(F) = {A, C}, *cooc*(G) = {A, C} và *cooc*(H) = {E}.

Bảng 5. Trả về mảng Index_COOC sắp 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

Bảng 5, mảng **Index_COOC** được sắp theo độ phổ biến của từng item (minh họa thành phần *cooc* theo *item*).

B. Thuật toán khai thác tập phổ biến tối đại LA-MFI

Theo khảo sát, hầu hết các thuật toán khai thác tập phổ biến tối đại đã được các tác giả trên thế giới đề xuất [5, 6, 7] đều có nhược điểm lớn và không thực tế: mỗi lần cần khai thác tập phổ biến tối đại với *minsup* khác thì thuật toán thực hiện tính toán lại độ phổ biến của các tập mục, phát sinh lại cây tìm kiếm hoặc dàn tập mục tương ứng và xác định tập phổ biến tối đại thỏa *minsup* mới. Trong thực tế, khi cần khai thác luật kết hợp thì người dùng có thể yêu cầu thực hiện khai thác luật kết hợp thỏa ngưỡng *minsup* và *minconf* trong nhiều chuỗi thao tác liên tiếp nhau. Các thuật toán đã được các tác giả trên thế giới chưa đáp ứng nhu cầu thực tế.

Nhóm tác giả đã xây dựng thuật toán **LA-MFI** khai thác tập phổ biến tối đại dựa trên mảng **Index_COOC** chứa tất cả các *itemset* đồng xuất hiện của tất cả *item* trong CSDL có thể thực hiện chuỗi khai thác nhanh tập **MFI**.

Mã giả thuật toán 2. LA-MFI khai thác tập phổ biến tối đại

Đầu vào*:* mảng **Index_COOC**, *minsup, maxsup (độ phổ biến lớn nhất của các item)*

Đầu ra*:* Tập phổ biến tối đại **MFI**

1. Với mỗi *item* thỏa *minsup*, xem xét:

2. **Nếu** *sup*(*item*) = *minsup* **thì**
3. **MFI** = **MFI** ∪ {*item* ∪ *cooc* (*item*)}
4. **Ngược lại**
5. **Nếu** *(cooc*(*item*) = Ø *và sup* (*item*) = *maxsup)* **thì**
6. **MFI**= **MFI** ∪{*item*}
7. **Ngược lại**
8. Tìm tập *item* J_k thỏa:
9. $$\begin{cases} sup(J_k) \geq sup(item) \\ J_k \in I \setminus \{item \cup cooc(item)\} \end{cases}$$
10. Tập $\mathcal{F}^*$ = {sinh các tập phổ biến tối đại từ J_k có *sup* (*k_item*) ≥ *minsup*}
11. *mfi* ={*item* ∪ *cooc*(*item*)}∪{∀ f ∈ $\mathcal{F}^*$}
12. **MFI** = **MFI** ∪ *mfi*
13. Trả về tập phổ biến tối đại **MFI**

Ví dụ 3: Cho CSDL $\mathcal{D}$ như trong *Bảng 1* và minsup = 2

Item B, D có sup(B) = sup(D) = minsup = 2 (điều kiện dòng 2):

Xét *item* B, có *cooc* (B) = {A, C, E}, sinh tập mục phổ biến tối đại là (**B**ACE, 2). Lúc này, **MFI** ={(**B**ACE, 2)}. Xét *item* D – có *cooc* (D) = {A, C}, sinh tập mục phổ biến tối đại là (**D**AC,2), **MFI** ={(**B**ACE, 2), (**D**AC, 2)}.

Xét *item* F (*ngược lại - dòng 7*), lúc này *item* F – *cooc*(F) = {A, C} còn *item* G, E, A và C có *sup* > *minsup* = 2. Tập J_k ={G, E}, sinh tập phổ biến $\mathcal{F}^*$={G, E, GE}. Tập mục phổ biến tối đại sinh ra trong bước này là *mfi* = {(**F**ACE, 2), (**F**ACG, 2)}. Kết thúc bước này, ta có **MFI**={(**B**ACE, 2), (**D**AC, 2), (**F**ACE, 2), (**F**ACG, 2)}.

Xét *item* G (*ngược lại - dòng 7*), lúc này *item* G – *cooc*(G) = {A, C} chỉ còn *item* E có *sup*(E) = 7 > *minsup* = 2. Tập J_k ={E}, sinh tập phổ biến $\mathcal{F}^*$={E}. Tập mục phổ biến tối đại sinh ra trong bước này là *mfi* = {(**G**ACE, 3)}. Kết thúc bước này, có **MFI**={(**B**ACE, 2), (**D**AC, 2), (**F**ACE, 2), (**F**ACG, 2), (**G**ACE, 3)}.

Xét item E, có *cooc*(E) = {∅} còn *item* A và C có *sup*(A) = *sup*(C) = 8 > *minsup*. Tập J_k ={A, C}, sinh tập phổ biến $\mathcal{F}^*$={A, C, AC}. Tập mục phổ biến tối đại sinh ra trong bước này là *mfi* = {(**A**CE, 5)} mà **A**CE ⊂ **B**ACE trong **MFI**, nên không đưa *mfi* ={(**A**CE, 5)} vào **MFI**.

Sau cùng, chỉ còn *item* A và C: sinh tập mục phổ biến tối đại (**A**C, 8), mà **A**C ⊂ **B**ACE, nên không thêm vào **MFI**.

Với CSDL $\mathcal{D}$ ở Bảng 1 và *minsup* = 2, ta có: **MFI**={(**B**ACE, 2), (**D**AC, 2), (**F**ACE, 2), (**F**ACG, 2) (**G**ACE, 4)}.

IV. KẾT QUẢ THỰC NGHIỆM

Thực nghiệm trên máy tính *Panasonic CF-74*, Core Duo 2.0 GHz, 2GB RAM, thuật toán cài đặt trên C#, Microsoft Visual Studio 2010.

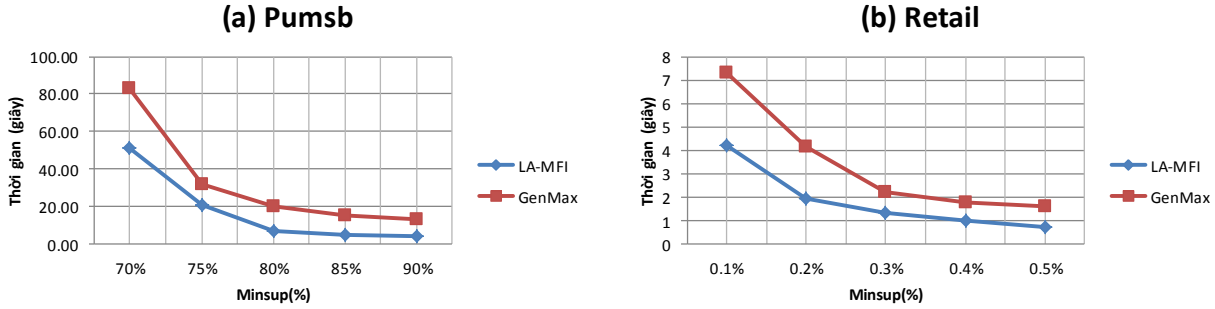
Nghiên cứu thực nghiệm trên *hai nhóm dữ liệu*:

Nhóm CSDL thực cỡ trung: sử dụng CSDL thực từ kho dữ liệu về học máy của Trường Đại học California (Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science) gồm 2 tập **Pumsb** và **Retail**.

Nhóm CSDL giả lập cỡ lớn: sử dụng phần mềm phát sinh dữ liệu giả lập của trung tâm nghiên cứu IBM Almaden (IBM Almaden Research Center, San Joe, California 95120, U.S.A [http://www.almaden.ibm.com]) gồm 2 tập **T40I10D100K** và **T40I10D200K**.

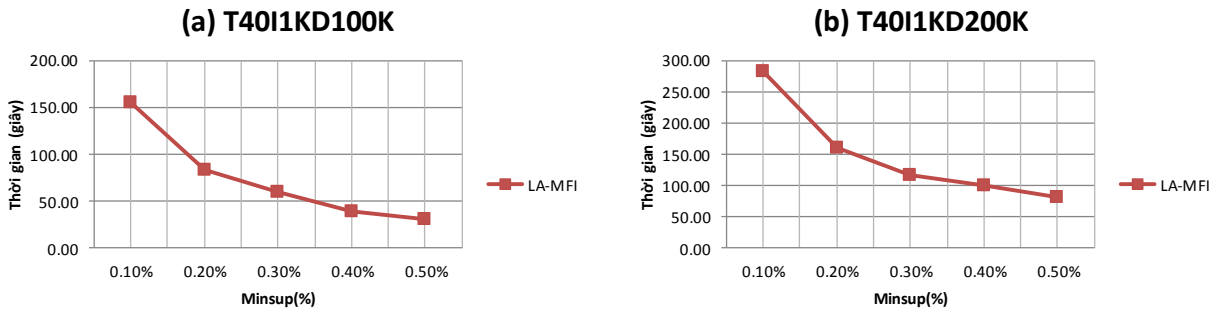
Bảng 6. Kích thước và tỷ lệ nén trên các tập CSDL thực nghiệm

Tên CSDL	Dung lượng	Số giao dịch	Số item	Kích thước dữ liệu sau nén	Tỷ lệ nén dữ liệu
Pumsb	16,30Mb	49.046	2.113	28,08Kb	99,8%
Retail	4,07Mb	88.162	16.470	66,61Kb	98,4%
T40I1KD100K	15,30Mb	100.000	1.000	83,33Kb	99,5%
T40I1KD200K	30,50Mb	200.000	1.000	197,27Kb	99,4%



Hình 2. Thời gian thực hiện **LA-MFI** và **GenMax** trên CSDL cỡ trung **Pumsb**, **Retail** với các *minsup* khác nhau

Nhóm tác giả sử dụng 2 tập **Pumsb** và **Retail** để so sánh hiệu suất của thuật toán **LA-MFI** với thuật toán **GenMax**. Hình 2, cho thấy thuật toán **LA-MFI** thời gian thực hiện khai thác tập phổ biến tối đại theo các ngưỡng *minsup* khác nhau trên 2 tập dữ liệu **Pumsb** và **Retail** nhanh hơn thuật toán **GenMax**.



Hình 3. Thời gian thực hiện **LA-MFI** trên CSDL cỡ lớn **T40I10D100K** và **T40I10D200K** với các *minsup* khác nhau

Thuật toán **GenMax** không thực hiện được trên 2 tập dữ liệu **T40I10D100K** và **T40I10D200K** cỡ lớn. Hình 3, cho thấy thời gian thực hiện của thuật toán **LA-MFI** khai thác tập phổ biến tối đại theo các ngưỡng *minsup* khác nhau.

V. KẾT LUẬN

Bài báo đã đề xuất thuật toán nén cơ sở dữ liệu giao dịch cỡ lớn, thuật toán tính nhanh mảng **Index_COOC** chứa các *itemset* đồng xuất hiện và thuật toán **LA-MFI** khai thác hiệu quả tập phổ biến tối đại dựa trên mảng **Index_COOC**. Tuy nhiên, thuật toán **LA-MFI** khai thác tập phổ biến tối đại cần được nghiên cứu và cải tiến để đạt tốc độ thực hiện hiệu quả hơn.

Với kết quả đạt được từ thuật toán **LA-MFI**. Trong tương lai, nhóm tác giả sẽ cải tiến thuật toán trên để có thể khai thác tập phổ biến tối đại trên cơ sở dữ liệu có *trọng số*, đây là hướng nghiên cứu đang được quan tâm vì khả năng ứng dụng vào nhiều lĩnh vực, đặc biệt là trong kinh doanh.

VI. 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. HCM.

TÀI LIỆU THAM KHẢO

[1] R. Agrawal, T. Imilienski, A. Swami, “Mining association rules between sets of large databases”. Proceedings of the ACM SIGMOD International Conference on Management of Data, Washington, DC, pp. 207-216, 1993.

[2] R. Agrawal, R. Srikant, “Fast algorithms for mining association rules”. Proceedings of International Conference on Very Large Data Base, Santiago, Chile, pp. 478-499, 1994.

[3] M. J. Zaki, C. Hsiao, “CHARM: An efficient algorithm for closed association rule mining”. In 2nd SIAM International Conference on Data Mining, pages 457–473, April 2002.

[4] J. Pei, J. Han, R. Mao, “CLOSET: An efficient algorithm for mining frequent closed itemsets”. Proc. of ACM SIGMOD DMKD Workshop, Dallas, TX, May, 2000.

[5] R. Agarwal, V. Prasad, - “Depth first generation of long patterns,” In Proc. of the 6th ACM SIGKDD international conference on knowledge discovery and data mining, pp.108–118, 2000.

[6] D. Burdick, M. Calimlim, J. Gehrke, “MAFIA: a maximal frequent itemset algorithm for transactional databases”. In IEEE Intl. Conf. on Data Engineering, pp. 443–452, 2001.

[7] K. Gouda, M. J. Zaki, “GenMax: An Efficient Algorithm for Mining Maximal Frequent Itemsets”. In IEEE International Conference on Data Mining and Knowledge Discovery, Volume 11, pp. 1–20, 2005.

[8] Y. Djenouri, A. Bendjoudi, D. Djenouri, Z. Habbas, "Parallel Processing and Applied Mathematics", ISBN 978-3-319-32148-6, pp. 258-268, 2016.

AN EFFICIENT MINING ALGORITHM FOR MAXIMAL FREQUENT ITEMSETS IN A LARGE TRANSACTIONAL DATABASES

Le Hoai Bac, Phan Thanh Huan

ABSTRACT*— Association rule mining, one of the most important and well-researched techniques of Data Mining. Mining maximal frequent itemsets is one of the most fundamental problems in association rule mining. According to the survey, most of the algorithms in literature used to find minimal frequent item first, then with the help of minimal frequent itemsets derive the maximal frequent itemsets. These methods consume more time to find maximal frequent itemsets. To overcome this problem, we propose a new approach to fast dectect maximal frequent itemsets using: efficient compression algorithm a large transactional databases and array co-occurrence items of kernel item. Finally, we present result, which shows the proposed algorithm has better than the existing algorithms.*

Keywords— *Association rule mining, co-occurrence items, large transactional databases, maximal frequent itemsets.*