Tạp chí Tin học và Điều khiển học, T.24, S.2 (2008), 168–178

KHAI PHÁ TẬP MỤC THƯỜNG XUYÊN CỔ PHẦN CAO TRONG CƠ SỞ DỮ LIỆU LỚN

VŨ ĐỨC THI[1], NGUYỄN HUY ĐỨC[2]

[1] *Viện Công nghệ thông tin, Viện Khoa học và Công nghệ Việt Nam*
[2] *Khoa Thông tin - Máy tính, Trường Cao đẳng Sư phạm Trung ương*

Abstract. Itemsets share has been proposed to evaluate the significance of itemsets for mining association rules in databases. The Fast Share Measure (FSM) algorithm is one of the best algorithms to discover all share-frequent itemsets efficiently. However, FSM is fast only when dealing with small datasets. In this paper, we shall propose a revised version of FSM, called the Advanced FSM (AFSM) algorithm. AFSM prune the candidates more efficiently than FSM and therefore can improve the performance significantly.

Tóm tắt. Khai phá tập mục thường xuyên cổ phần cao (Share-Frequent Itemset) là một mở rộng của bài toán khai phá tập mục thường xuyên, đã được các tác giả đề xuất với mục đích đánh giá ý nghĩa của các tập mục trong khai phá luật kết hợp. Thuật toán FSM (Fast Share Measure) là một trong các thuật toán tốt để khai phá hiệu quả các tập mục thường xuyên cổ phần cao. Trong báo cáo này, chúng tôi đề xuất một cải tiến của thuật toán FSM. Việc cải tiến được thực hiện thông qua một chiến lược tỉa hiệu quả hơn các tập mục ứng viên, nhờ đó giảm bớt được thời gian thực hiện thuật toán khai phá.

1. MỞ ĐẦU

Bài toán cơ bản (hay còn gọi là bài toán nhị phân) khai phá luật kết hợp do Agrawal, T.Imielinski và A. N. Swami đề xuất và nghiên cứu lần đầu tiên vào năm 1993 [4], mục tiêu của bài toán là phát hiện các tập mục thường xuyên, từ đó tạo các luật kết hợp. Trong mô hình của bài toán nhị phân này, giá trị của mỗi mục dữ liệu trong một giao tác là 0 hoặc 1. Bài toán cơ bản khai phá luật kết hợp có nhiều ứng dụng, tuy vậy do tập mục thường xuyên chỉ mang ngữ nghĩa thống kê nên nó chỉ đáp ứng được phần nào nhu cầu ứng dụng thực tiễn.

Nhằm khắc phục hạn chế của bài toán cơ bản khai phá luật kết hợp, nhiều nhà nghiên cứu đã mở rộng bài toán theo nhiều hướng khác nhau. Năm 1997, Hilderman và các cộng sự đã đề xuất bài toán khai phá tập mục thường xuyên cổ phần cao [7]. Trong mô hình khai phá tập mục thường xuyên cổ phần cao, giá trị của mục dữ liệu trong giao tác là một số, số đó có thể là số nguyên (như số lượng đã bán của mặt hàng) hoặc số thực (như số tiền lãi được khi bán mặt hàng đó). Cổ phần (hay đóng góp) của một tập mục là số đo tỷ lệ đóng góp của tập mục trong cơ sở dữ liệu. Khai phá tập mục cổ phần cao là khám phá tất cả các tập mục có cổ phần không nhỏ hơn ngưỡng quy định bởi người sử dụng.

Trong bài toán cơ bản, các thuật toán khám phá được xây dựng theo phương pháp tìm kiếm từng bước. Cơ sở của các thuật toán là tính chất Apriori của tập mục thường xuyên

(hay còn gọi là tính chất phản đơn điệu - Anti monotone). Trong mô hình khai phá tập mục cổ phần cao, tính chất này không còn đúng nữa. Vì vậy việc rút gọn không gian tìm kiếm không thể thực hiện được như đối với khai phá tập mục thường xuyên. Trong [5–11], các tác giả đã đề nghị một số thuật toán khám phá tập mục thường xuyên cổ phần cao như các thuật toán ZP, ZSP, SIP, FSM,.... Thuật toán FSM [6] là một thuật toán nhanh cho phép khám phá tất cả các tập mục thường xuyên cổ phần cao. Thuật tỉa mà thuật toán này áp dụng có khả năng thu gọn phần nào tập mục ứng viên, tuy vậy có những nhược điểm nên hiệu quả không cao.

Bài báo này trình bày một thuật toán hiệu quả khám phá tất cả các tập mục thường xuyên cổ phần cao trong cơ sở dữ liệu lớn: thuật toán AFSM. Thuật toán AFSM được xây dựng dựa trên ý tưởng của thuật toán FSM, nhưng việc tỉa các tập mục ứng viên được thực hiện thông qua một hàm tới hạn mới hiệu quả hơn. Thuật toán AFSM còn hiệu quả hơn thuật toán FSM nhờ cải tiến việc kết nối và tỉa các tập mục ứng viên. Nội dung tiếp theo của bài báo gồm: Phần 2 nêu một số định nghĩa, thuật ngữ và phát biểu bài toán khai phá tập mục thường xuyên cổ phần cao. Phần 3 tóm tắt nội dung và nêu những nhược điểm của thuật toán FSM. Phần 4 trình bày thuật toán mới AFSM. Phần 5 đánh giá thuật toán và kết luận dựa trên việc phân tích thuật toán và các thử nghiệm.

2. BÀI TOÁN KHAI PHÁ TẬP MỤC THƯỜNG XUYÊN CỔ PHẦN CAO

Trước hết ta nêu định nghĩa của một số thuật ngữ ([6]).

Cho tập các mục (item) $I = \{i_1, i_2, ..., i_n\}$. Một giao tác (transaction) T là một tập con của I, $T \subseteq I$. Cơ sở dữ liệu là một tập các giao tác $DB = \{T_1, T_2, ..., T_m\}$. Mỗi giao tác được gán một định danh TID. Một tập mục con $X \subseteq T$, gồm k mục phân biệt được gọi là một k - tập mục. Giao tác T gọi là chứa tập mục X nếu $X \subseteq T$.

Ta ký hiệu $mv(i_p, T_q)$ là giá trị của mục i_p trong giao tác T_q. Tổng giá trị các mục trong giao tác T_q gọi là giá trị của giao tác, ký hiệu là $tmv(T_q)$, tức là $tmv(T_q) = \Sigma_{i_p \in T_q} mv(i_p, T_q)$. Tổng các giá trị của các mục dữ liệu trong cơ sở dữ liệu DB ký hiệu là Tmv,

$$Tmv = \sum_{T_q \in DB} \sum_{i_p \in T_q} mv(i_p, T_q).$$

Tương tự, với cơ sở dữ liệu con $db \subseteq DB$, $Tmv(db) = \Sigma_{T_q \in db} \Sigma_{i_p \in T_q} mv(i_p, T_q)$.

Ví dụ, cho cơ sở dữ liệu Bảng 2.1, có $mv(D, T01) = 1$, $mv(C, T03) = 3$, $tmv(T01) = 6$, $tmv(T03) = 10$, $Tmv(DB) = 47$.

Bảng 2.1. Cơ sở dữ liệu ví dụ

TID	A	B	C	D	E	F	G	H	tmv
T01	1	1	1	1			1	1	6
T02	4		3		1	2			10
T03	4		3		3				10
T04		4	1	2		2			9
T05	3	1		2					6
T06		3	2	1					6
imv	12	9	10	6	4	4	1	1	47

Ký hiệu db_X là tập các giao tác chứa tập mục X, tức $db_X = \{T_q / X \subseteq T_q,\ T_q \in DB\}$.

Định nghĩa 2.1. Cho giao tác T_q chứa tập mục X. Giá trị của tập mục X tại T_q, ký hiệu $imv(X, T_q)$, là tổng giá trị của các mục i_p trong T_q thuộc X, $imv(X, T_q) = \sum_{X \in T_q,\ i_p \in X} mv(i_p, T_q)$.

Định nghĩa 2.2. Cho tập mục X, db_X là tập các giao tác chứa X. Giá trị của tập mục X, ký hiệu $lmv(X)$, là tổng giá trị của tập mục X tại các giao tác trong db_X, tức là

$$lmv(X) = \sum_{T_q \in db_X} imv(X, T_q) = \sum_{T_q \in db_X} \sum_{i_p \in X} mv(i_p, T_q).$$

Định nghĩa 2.3. Cổ phần (đóng góp) của tập mục X, ký hiệu là $SH(X)$ là tỉ số giữa giá trị của tập mục X và tổng giá trị của tất cả các mục trong cơ sở dữ liệu, tức là $SH(X) \frac{lmv(X)}{Tmv}$.

$SH(X)$ cho biết trong tổng giá trị của tất cả các mục trong cơ sở dữ liệu thì giá trị của tập X chiếm bao nhiêu phần trăm. Ví dụ, với CSDL các giao tác bán hàng, $SH(X) = 25\%$ tức là trong số lượng hàng đã bán được thì số lượng các mặt hàng trong X chiếm 25%.

Định nghĩa 2.4. Cho ngưỡng cổ phần *minShare* $s\%$ và tập mục X, X được gọi là tập mục thường xuyên cổ phần cao nếu $SH(X) \geq minShare$.

Ví dụ, xét cơ sở dữ liệu các giao tác cho trong Bảng 2.1 và $minShare = 30\%$. Bảng 2.2 cho giá trị của các mục và giá trị cổ phần của chúng.

Bảng 2.2. Giá trị và cổ phần của các mục trong CSDL trên Bảng 2.1

Mục dữ liệu	$\{A\}$	$\{B\}$	$\{C\}$	$\{D\}$	$\{E\}$	$\{F\}$	$\{G\}$	$\{H\}$	Tổng số
$lmv(i_p)$	12	9	10	6	4	4	1	1	47
SH(ip)	25,5%	19,1%	21,3%	12,8%	8,5%	8,5%	2,1%	2,1%	100%

Xét $X = \{B, C, D\}$, giá trị của tập mục X là

$$lmv(X) = imv(X, T01) + imv(X, T04) + imv(X, T06) = 3 + 7 + 6 = 16.$$

$$SH(X) = lmv(X)/Tmv = 16/47 = 0.340 > 30\%.$$

Do đó, $X = \{B, C, D\}$ là tập mục thường xuyên cổ phần cao. Tập tất cả các tập mục thường xuyên cổ phần cao của cơ sở dữ liệu Bảng 2.1 được cho trong Bảng 2.3.

Bảng 2.3. Các tập mục thường xuyên cổ phần cao của CSDL Bảng 2.1

Tập mục thường xuyên cổ phần cao	$\{A, C\}$	$\{B, D\}$	$\{A, C, E\}$	$\{B, C, D\}$
lmv(X)	16	15	18	16
SH(X)	34,0%	31,9%	38,3%	34,0%

Định nghĩa 2.5. Cho CSDL giao tác DB và ràng buộc cổ phần *minShare*, bài toán khai phá tập mục thường xuyên cổ phần cao là việc tìm tập F tất cả các tập mục thường xuyên cổ phần cao, tức là tập $F = \{X / X \subseteq I,\ SH(X) \geq minShare\}$.

Có thể coi bài toán khai phá tập mục thường xuyên là trường hợp đặc biệt của bài toán khai phá tập mục thường xuyên cổ phần cao. Tính chất cơ bản được khai thác để xây

dựng các thuật toán khai phá tập mục thường xuyên là tính chất Apriori. Tính chất này có thể phát biểu như sau. Nếu một tập mục là thường xuyên thì mọi tập con khác rỗng của nó cũng là thường xuyên. Điều này có nghĩa các $(k+1)$-tập mục thường xuyên chỉ có thể được sinh ra từ các k-tập mục thường xuyên. Dễ thấy, tính chất này không còn đúng với ràng buộc cổ phần. Ví dụ, xét CDSL Bảng 2.1, $lmv(\{B,C,D\}) = 16$, $lmv(\{B,D\}) = 15$, $lmv(\{B,C\}) = 12, SH(\{B,C,D\}) = 16/47 = 34\%$, $SH(\{B,D\}) = 15/47 = 31,9\%$, $SH(\{B,C\}) = 12/47 = 25,5\%$, tức là $\{B,C,D\}$ là tập thường xuyên cổ phần cao, tập con của nó $\{B,D\}$ là tập thường xuyên cổ phần cao nhưng tập con khác $\{B,C\}$ lại không phải.

Do đó không thể áp dụng các thủ pháp lợi dụng tính chất Apriori vào việc khai phá tập mục thường xuyên cổ phần cao. Đã có nhiều công trình nghiên cứu các thuật toán tìm tập mục thường xuyên cổ phần cao. Các tác giả đã đề xuất một số thuật toán như thuật toán ZP, ZSP, SIP,... [5–11]. Yu-Chiang Li và cộng sự đã giới thiệu thuật toán FSM (Fast Share Measure) [6], đây là một thuật toán hiệu quả tìm tất cả các tập mục thường xuyên cổ phần cao. Thay vì khai thác tính chất Apriori, thuật toán FSM sử dụng một số tính chất của tập mục thường xuyên cổ phần cao để rút gọn số các tập mục ứng viên. Phần tiếp theo dưới đây trình bày nội dung cơ bản của thuật toán FSM này cùng một số nhận xét.

3. THUẬT TOÁN FSM

Cơ sở lý thuyết của thuật toán FSM là mệnh đề sau đây ([6]).

Ký hiệu:

$\min\_lmv = minShare \times Tmv$;

ML là độ dài cực đại của các giao tác trong cơ sở dữ liệu,

$ML = \max\{|T_q|/T_q \in DB\}$;

MV là giá trị cực đại của tất cả các mục trong cơ sở dữ liệu,

$MV = \max\{mv(i_p, T_q)/i_p \in T_q \wedge T_q \in DB\}$;

Mệnh đề 3.1. *Cho minShare và k-tập mục X không là tập thường xuyên cổ phần cao. Nếu*

$$lmv(X) + \frac{lmv(X)}{k} \times MV \times (ML - k) < \min\_lmv$$

thì tất cả các tập cha của X không là tập thường xuyên cổ phần cao.

Đặt

$$CF(X) = lmv(X) + \frac{lmv(X)}{k} \times MV \times (ML - k)$$

$CF(X)$ được gọi là hàm tới hạn. Mệnh đề trên đảm bảo rằng, nếu X không phải là tập thường xuyên cổ phần cao và $CF(X) < \min\_lmv$ thì không có tập cha nào của X là tập thường xuyên cổ phần cao. Thuật toán FSM dựa vào tính chất này để tỉa các tập mục ứng viên.

Thuật toán FSM

Thuật toán FSM duyệt nhiều lần CSDL, ở lần duyệt thứ k, C_k lưu tập các tập mục ứng viên, RC_k lưu các tập mục nhận được sau khi kiểm tra hàm tới hạn CF, F_k chứa tập các tập mục thường xuyên cổ phần cao nhận được. Giống như thuật toán Apriori, mỗi mục đơn

là một ứng viên. Trong lần duyệt thứ nhất, thuật toán duyệt CSDL tính giá trị của mỗi mục trong CSDL. Mỗi ứng viên 1- tập mục X sẽ bị tỉa nếu $CF(X) < \min_l mv$. Trong mỗi lần duyệt tiếp theo, thuật toán nối hai $(k-1)$-tập mục ứng viên trong RC_{k-1} nếu chúng có $(k-2)$ mục đầu giống nhau và nhận được k-tập mục (giả sử các mục của CSDL đã được sắp thứ tự). Tất cả k tập con với độ dài $(k-1)$ của mỗi k-tập mục trong C_k là phải thuộc RC_k, nếu không k-tập mục này sẽ bị tỉa. Sau khi C_k được sinh ra, xóa tập RC_{k-1}. Tiếp theo, thuật toán duyệt cơ sở dữ liệu để tìm tập mục thường xuyên cổ phần cao. Với mỗi tập mục X trong C_k, nếu tập mục này có $lmv(X)/Tmv$ lớn hơn $minShare$ thì X được thêm vào F_k. Ngược lại, nếu $CF(X) < \min\_lmv$ thì tập ứng viên X bị loại khỏi RC_k.

Quá trình trên cứ lặp cho đến khi không có tập mục ứng viên nào được sinh ra.

Nhận xét:

Thuật toán FSM sử dụng hàm tới hạn $CF(X)$ để thu gọn tập các tập mục ứng viên. Tuy vậy số các tập mục của tập RC_k vẫn còn lớn do giá trị hàm tới hạn $CF(X)$ còn cao. Hơn nữa thuật toán còn mất nhiều thời gian ở bước nối mỗi cặp $(k-1)$-tập mục trong RC_{k-1} để được một k-tập mục.

Nhằm khắc phục những hạn chế của thuật toán FSM, chúng tôi đề nghị một thuật toán mới khai phá tập mục thường xuyên cổ phần cao trên cơ sở cải tiến thuật toán FSM, gọi là thuật toán AFSM (Advanced FSM).

4. THUẬT TOÁN AFSM

4.1. Cơ sở lý thuyết

Định nghĩa 4.1. Cho cơ sở dữ liệu DB và k-tập mục X. Một tập mục cha cỡ $(k+1)$ của X trong giao tác T_q được ký hiệu bằng X^{k+1}, ở đó $X^{k+1} \subseteq T_q \in DB$.

Định nghĩa 4.2. Cho cơ sở dữ liệu DB và k-tập mục X. Tập tất cả các tập cha cỡ $(k+1)$ của X trong DB được ký hiệu bằng $S(X^{k+1})$, ở đó $X^{k+1} \in S(X^{k+1})$.

Định nghĩa 4.3. Cho cơ sở dữ liệu DB và k-tập mục X. Tập tất cả các giao tác chứa ít nhất một X^{k+1} được ký hiệu bằng $db_{S(X^{k+1})}$. Độ hỗ trợ của $S(X^{k+1})$ ký hiệu bằng $Sup(S(X^{k+1}))$ và $Sup(S(X^{k+1})) = |db_{S(X^{k+1})}|$.

Ví dụ, trong Bảng 2.1, lấy $X = \{B, C, D\}$ ta có:

X^{k+1} là $\{A, B, C, D\}$, $\{B, C, D, G\}$, $\{B, C, D, H\}$, và $\{B, C, D, F\}$.

$S(X^{k+1}) = \{\{A, B, C, D\}, \{B, C, D, G\}, \{B, C, D, H\}, \{B, C, D, F\}\}$.

$db_{S(X^{k+1})} = \{T01, T04\}$, $Sup(S(X^{k+1})) = 2$.

Với $X = \{B, C, D, F\}$ thì $db_{S(X^{k+1})} = \emptyset$, $Sup(S(X^{k+1})) = 0$.

Dễ dàng nhận thấy $\dfrac{lmv(X)}{k} \geq Sup(X) \geq Sup(S(X^{k+1})) \geq \max\limits_{X^{k+1} \in S(X^{k+1})} Sup(S(X^{k+1}))$.

Định lý 4.1. *Cho cơ sở dữ liệu DB và k-tập mục X. Ta có:*

$$Tmv(db_{S(X^{k+1})}) \leq lmv(X) + |db_{S(X^{k+1})}| MV(ML - k).$$

Chứng minh:

Với mọi giao tác $T_q \in DB$ chứa X ta có:

$$tmv(T_q) = \sum_{t_p \in T_q} mv(i_p, T_q)$$
$$= \sum_{i_p \in X} mv(i_p, T_q) + \sum_{t_p \in T_q \setminus X} mv(i_p, T_q) \leq imv(X, T_q) + MV(ML - k).$$

Do đó:

$$Tmv(db_{S(X^{k+1})}) = \sum_{T_q \in db_{S(X^{k+1})}} tmv(T_q) \leq \sum_{T_q \in db_{S(X^{k+1})}} [imv(X, T_q) + MV(ML - k)$$
$$\leq \sum_{T_q \in db_X} imv(X, T_q) + |db_{S(X^{k+1})}|MV(ML - k) =$$
$$lmv(X) + |db_{S(X^{k+1})}|MV(ML - k).$$

ta có $\sum_{T_q \in db_{S(X^{k+1})}} imv(X, T_q) \leq \sum_{T_q \in db_X} imv(X, T_q)$ vì $db_{S(X^{k+1})} \subseteq db_X$

■

Định lý 4.2. *Cho ngưỡng minShare và k-tập mục X không là tập mục thường xuyên cổ phần cao. Nếu* $Tmv(db_{S(X^{k+1})}) < \min\_lmv$ *thì mọi tập cha của tập mục X đều không phải tập mục thường xuyên cổ phần cao.*

Chứng minh:

Giả sử X' là tập mục cha bất kỳ của X với độ dài $(k + i)$, trong đó $0 < i \leq (ML - k)$. Vì $db_{X'} \subseteq db_{S(X^{k+1})}$ nên $Tmv(db_{X'}) \leq Tmv(db_{S(X^{k+1})})$.

Do đó $lmv(X') \leq Tmv(db_{X'}) \leq Tmv(db_{S(X^{k+1})})$. Nếu $Tmv(db_{S(X^{k+1})}) < \min\_lmv$ thì $lmv(X') < \min\_lmv,\ SH(X') = \frac{lmv(X')}{Tmv} < minShare$.

Vậy X' không là tập mục thường xuyên cổ phần cao. ■

Nhận xét 4.1. Từ Định lý 4.2, có thể sử dụng $Tmv(db_{S(X^{k+1})})$ làm hàm tới hạn cho thuật toán mới AFSM để tỉa các tập mục ứng viên. Ký hiệu hàm tới hạn của thuật toán FSM và thuật toán mới AFSM tương ứng là $CF_{FSM}(X)$ và $CF_{AFSM}(X)$. Định lý sau so sánh giá trị của 2 hàm tới hạn này.

Định lý 4.3. *Cho cơ sở dữ liệu DB và k-tập mục X. Khi đó:*

$$CF_{AFSM}(X) \leq CF_{FSM}(X).$$

Chứng minh:

Ta có $CF_{AFSM}(X) = Tmv(db_{S(X^{k+1})}),\ CF_{FSM}(X) = lmv(X) + \frac{lmv(X)}{k} MV(ML - k)$.

Theo Định lý 4.1 ta có

$$Tmv(db_{S(X^{k+1})}) \leq lmv(X) + |db_{S(X^{k+1})}|MV(ML - k).$$

Vì
$$|db_{S(X^{k+1})}| = Sup(S(X^{k+1})) \leq Sup(X) \leq \frac{lmv(X)}{k}$$
nên ta có:
$$\begin{aligned} CF_{AFSM}(X) = Tmv(db_{S(X^{k+1})}) &\leq lmv(X) + |db_{S(X^{k+1})}|MV(ML-k) \\ &\leq lmv(X) + \frac{lmv(X)}{k}MV(ML-k) = CF_{FSM}(X). \end{aligned}$$
Vậy $CF_{AFSM}(X) \leq CF_{FSM}(X)$. ■

Nhận xét 4.2. Từ kết quả của Định lý 4.3, ta thấy rằng thuật toán AFSM sẽ tỉa các tập mục ứng viên nhiều hơn do giá trị của hàm tới hạn nhỏ hơn, do đó tập RC_k được sinh ra nhỏ hơn. Điều này làm cho thuật toán AFSM thực hiện hiệu quả hơn thuật toán FSM.

4.2. Thuật toán AFSM

Dựa trên ý tưởng của thuật toán FSM, chúng tôi xây dựng thuật toán mới AFSM như sau:

- Sử dụng hàm tới hạn mới $CF_{AFSM}(X) = Tmv(db_{S(X^{k+1})})$ cho thuật toán AFSM.
- Cải tiến bước sinh tập ứng viên: Trong thuật toán FSM, quá trình sinh tập C_k đòi hỏi phải kết nối các cặp tập mục trong RC_{k-1}. Thuật toán phải so sánh $(k-2)$ mục đầu của hai $(k-1)$-tập mục trong RC_{k-1}, do đó độ phức tạp thời gian để sinh C_k là $O(n^{2k-2})$, ở đó n là số các mục. Để rút gọn thời gian thực hiện ở bước sinh tập ứng viên, chúng tôi cải tiến như sau: thay cho việc nối hai $(k-1)$-tập mục bất kỳ trong RC_{k-1}, chúng ta kết nối mỗi tập $(k-1)$-tập mục trong RC_{k-1} với một mục đơn trong RC_1 để sinh ra tập C_k. Với cải tiến này, thời gian thực hiện ở bước sinh tập ứng viên giảm xuống là $O(n^k)$.

Giả sử các mục trong cơ sở dữ liệu đã được sắp thứ tự. Xét $(k-1)$-tập mục ứng viên $X^{k-1} = \{i_1, i_2, ..., i_{k-1}\}$ và 1-tập mục $X^1 = \{i_q\} \in RC_1$. Nếu $i_{k-1} < i_q$ thì tập $X^k = \{i_1, i_2, ..., i_{k-1}, i_q\}$ là một k-tập mục ứng viên thuộc C_k. Tiếp đến, ở bước tỉa, thủ tục sẽ xóa tất cả các tập mục $X^k \in C^k$ nếu X^k chứa ít nhất một $(k-1)$-tập mục con không thuộc RC_{k-1}. Thủ tục nối và tỉa như sau.

Function *Noi*(RC_{k-1}, RC_1)

1. $C_k := \emptyset$;
2. for each $X_p = \{i_1, i_2, ..., i_{k-1}\} \in RC_{k-1}$
3. for each $i_q \in RC_1$
4. if $i_{k-1} < i_q$ then {
5. $X = X_p \cup \{i_q\}$;
   $C_k := C_k \cup \{X\}$; }

Function *Tia*(C_k)

1. for each $X \in C_k$
2. if (có một tập con $k-1$ mục của X không thuộc RC_{k-1}) then
3. $C_k := C_k \backslash \{X\}$

Từ các cơ sở lý thuyết đã trình bày, chúng tôi đề nghị thuật toán AFSM như sau.

Thuật toán AFSM( )

Input: DB cơ sở dữ liệu giao tác, ngưỡng cổ phần $minShare(s\%)$.

Output: Tập F gồm các tập mục thường xuyên cổ phần cao.
Method:

1. $k := 1,\ F_1 := \phi,\ C_1 := I$
2. for each $T \in DB$ // duyệt cơ sở dữ liệu DB
3. tính giá trị các 1-tập mục trong T; // tính $lmv(i_p)$ và cho mỗi
4. for each $i_p \in C_1$
5. if $lmv(i_p) \geq \min\_lmv$ then
6. $F_1 := F_1 \cup \{i_p\}$
7. else if $CF_{AFSM}(i_p) < \min\_lmv$ then
8. $C_1 := C_1 \backslash \{i_p\}$
9. $RC_1 := C_1$
10. repeat
11. $k := k + 1$
12. $C_k := Noi(RC_{k-1}, RC_1)$
13. if $(k > 2)$ then $C_k := Tia(C_k)$
14. for each $T \in DB$ // duyệt cơ sở dữ liệu DB
15. tính giá trị mỗi tập ứng viên thuộc T
16. for each $X \in C_k$
17. if $lmv(X) \geq \min\_lmv$ then
18. $F_k := F_k \cup \{X\}$
19. else if $CF_{AFSM}(X) < \min\_lmv$ then
20. $C_k := C_k \backslash \{X\}$
21. $RC_k := C_k$
22. until $C_k = \phi$
23. return $F = \cup F_k$

Ví dụ, cho cơ sở dữ liệu giao tác là Bảng 2.1, $minShare = 30\%$. Thuật toán AFSM thực hiện việc phát hiện các tập mục thường xuyên cổ phần cao như sau:

Ta có: $Tmv = 47,\ MV = 4,\ ML = 6, \min\_lmv = minShare \times Tmv = 30\% \times 47 = 14,1$.

Bước k=1. Thực hiện dòng lệnh 1 cho kết quả $F_1 = \phi,\ C_1 = \{A, B, C, D, E, F, G, H\}$. Thực hiện dòng lệnh 2-3 ta tính được lmv của các mục dữ liệu. Bảng 4.1 sau biểu diễn các kết quả tính toán sau.

Bảng 4.1. Các giá trị lmv và hàm tới hạn với k=1

	A	B	C	D	E	F	G	H
lmv	12	9	10	6	4	4	1	1
CF_{AFSM}	32	27	41	27	20	19	6	6
CF_{FSM}	252	189	210	126	84	84	21	21

Vì các giá trị lmv đều nhỏ hơn $\min\_lmv = 14,1$ nên không có mục nào là cổ phần cao.

$$db_{S(\{A\}^2)} = \{T01, T02, T03, T05\},$$

$$CF_{AFSM}(\{A\}) = Tmv(db_{S(\{A\}^2)}) = tmv(T01) + tmv(T02) + tmv(T03) + tmv(T05)$$
$$= 6 + 10 + 10 + 6 = 32 > \min\_lmv,$$

do đó A không bị tỉa khỏi C_1. Làm tương tự ta có G và H bị tỉa. Thực hiện các lệnh từ 4-8 ta nhận được $F_1 = \phi,\ C_1 = \{A, B, C, D, E, F\}$. Dòng lệnh 9 cho kết quả $RC_1 = \{A, B, C, D, E, F\}$.

Bước k=2. Dòng lệnh 12 nối RC_1 với RC_1 ta được 15 tập mục độ dài $k = 2$. Dòng lệnh 13 thực hiện thủ tục tỉa nhưng không tỉa được tập mục nào.

Thực hiện các lệnh 14-20 cho kết quả là Bảng 4.2 sau.

Bảng 4.2. Các giá trị lmv và hàm tới hạn với k=2

	AB	AC	AD	AE	AF	BC	BD	BE	BF	CD	CE	CF	DE	DF	EF
lmv	6	16	7	12	6	12	15	0	6	8	9	8	0	4	3
CFAFSM	12	26	12	20	10	21	15	0	9	21	20	19	0	10	10
CFFSM	54	144	63	108	54	108	135	0	54	72	81	72	0	36	27

Từ bảng trên ta có: $F_2 = \{AC, BD\},\ C_2 = \{AC, AE, BC, BD, CD, CE, CF\}$.

Dòng lệnh 21 cho ta $RC_2 = \{AC, AE, BC, BD, CD, CE, CF\}$.

Lặp tiếp với $k = 3$, dòng lệnh 12 thực hiện nối RC_2 với RC_1 ta được:

$$C_3 = \{ACD, ACE, ACF, AEF, BCD, BCE, BCF, BDE, BDF, CDE, CDF, CEF\}.$$

Dòng lệnh 13 thực hiện tỉa: ACD bị tỉa vì có tập con AD không thuộc RC_2. Xét tương tự có 10 tập mục bị tỉa khỏi C_3 và ta được $C_3 := tia(C_3) = \{ACE, BCD\}$.

Thực hiện duyệt cơ sở dữ liệu và tính toán ta được kết quả là Bảng 4.3 sau.

Bảng 4.3. Các giá trị lmv và hàm tới hạn với k=3

	ACE	BCD
lmv	18	16
CFAFSM	10	15
CFFSM	90	80

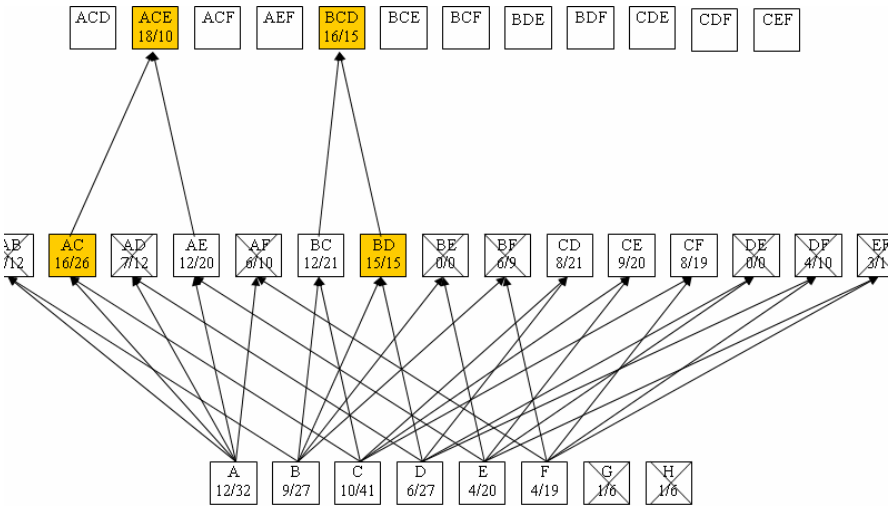
$$F_3 = \{ACE, BCD\},\ C_3 = \{ACE, BCD\},\ RC_3 = \{ACE, BCD\}.$$

Lặp tiếp với $k = 4$: nối RC_3 với RC_1 ta được $C_4 = \{ACEF, BCDE, BCDF\}$. Các tập mục này đều bị tỉa vì có tập con không thuộc RC_3, do đó $C_4 = tia(C_4) = \phi$, thuật toán dừng.

Kết quả thuật toán tìm được tập các tập mục thường xuyên cổ phần cao $F = F_1 \cup F_2 \cup F_3 = \{AC, BD, ACE, BCD\}$.

Chú ý: các Bảng 4.1, 4.2 và 4.3 tính cả giá trị hàm tới hạn của thuật toán FSM để so sánh. Ta thấy giá trị của CF_{FSM} lớn hơn giá trị của CF_{AFSM} rất nhiều, khi $k = 1$ thuật toán FSM không tỉa G và H, lặp với $k = 2$, thuật toán FSM chỉ tỉa 2 tập mục BE và DE, trong khi đó thuật toán A_{FSM} tỉa 8 tập mục.

Kết quả thực hiện của thuật toán được minh họa ở Hình 4.1 sau.



Hình 4.1. Hình minh họa không gian tìm kiếm tập mục thường xuyên cổ phần cao theo thuật toán AFSM

5. ĐÁNH GIÁ THUẬT TOÁN VÀ KẾT LUẬN

Chúng tôi đã tiến hành thử nghiệm thuật toán $AFSM$ trên một số cơ sở dữ liệu giao tác được tạo bằng phương pháp tạo số ngẫu nhiên. Căn cứ vào các kết quả thử nghiệm và phân tích thuật toán, chúng tôi nhận thấy $AFSM$ có những ưu điểm sau so với FSM.

1. Thu gọn đáng kể không gian tìm kiếm nhờ hàm tới hạn có giá trị nhỏ hơn nhiều giá trị hàm tới hạn của thuật toán FSM. Điều này làm cho thuật toán thu gọn ngay không gian tìm kiếm từ lần lặp đầu, từ đó ảnh hưởng đến các lần lặp sau làm cho tập RC_k có kích thước nhỏ. Ví dụ, đối với cơ sở dữ liệu trong Bảng 2.1, ngay từ đầu xét các mục đơn đã có G và H bị loại, trong khi đó theo thuật tỉa của FSM thì các mục này không bị tỉa. Với cơ sở dữ liệu lớn có thể hy vọng $AFSM$ sẽ thu gọn không gian tìm kiếm nhiều hơn.

2. Thuật toán $AFSM$ thay đổi cách nối và tỉa khi sinh tập ứng viên C_k bằng cách nối một tập mục độ dài $(k-1)$ với một mục đơn đã giảm thời gian ở bước nối từ $O(n^{2k-2})$ xuống còn $O(n^k)$.

3. Số lần duyệt cơ sở dữ liệu của thuật toán $AFSM$ so với thuật toán FSM là như nhau.

Với những ưu điểm trên đây, chúng tôi cho rằng thuật toán $AFSM$ hiệu quả hơn so với thuật toán FSM trong [6].

TÀI LIỆU THAM KHẢO

[1] Vũ Đức Thi, *Cơ sở dữ liệu - Kiến thức và thực hành*, Nhà xuất bản Thống kê, năm 1997.

[2] Nguyễn Thanh Tùng, Khai phá tập mục lợi ích cao trong cơ sở dữ liệu, *Tạp chí Tin học và Điều khiển học* **23** (4) (2007) 364–373.

[3] Nguyễn Huy Đức, Khai phá luật kết hợp trong cơ sở dữ liệu lớn, *Kỷ yếu Hội thảo khoa học Quốc gia lần thứ nhất về nghiên cứu cơ bản và ứng dụng CNTT*, Hà Nội, 10/2003.

[4] R. Agrawal and R. Srikant, Fast algorithms for mining association rules, *Proceedings of 20th International Conference on Very Large Databases*, Santiago, Chile, 1994.

[5] B. Barber and H. J. Hamilton, Extracting share frequent itemsets with infrequent subsets, *Data Mining and Knowledge Discovery* **7** (2) (2003).

[6] Y. C. Li, J. S. Yeh, and C. C. Chang, A fast algorithm for mining share-frequent itemsets, "Lecture Notes in Computer Science", Springer-Verlag, Germany, 3399 (2005) 417–428.

[7] R. J. Hilderman, C. L. Carter, H. J. Hamilton, and N. Cercone, Mining association rules from market basket data using share measures and characterized itemsets, *Journal of Artificial Intelligence Tools* **7** (1998) 189–220.

[8] C. L. Carter, H. J. Hamilton, and N. Cercone, Share based measures for itemsets, *Lecture Notes in Computer Science*, Springer-Verlag, Germany, 1263 (1997) 14–24.

[9] B. Barber and H. J. Hamilton, Parametric algorithm for mining share frequent itemsets, *Journal of Intelligent Information Systems* **16** (2001) 277–293.

[10] B. Barber and H. J. Hamilton, Algorithms for mining share frequent itemsets containing infrequent subsets, "Lecture Notes in Computer Sciences", Springer-Verlag, Germany, 1910 (2000) 316–324.

[11] T. Y. Lin, Y. Y. Yao, and E. Louie, Value added association rules, *Advances in Knowledge Discovery and Data Mining, 6th Pacific-Asia Conference*, Taipei, 2002.

[12] Y. D. Shen, Q. Yang, and Z. Zhang, Objective-oriented utility-based association mining, *Proceedings of the 13th ACM SIGMOD International Conference on Knowledge Discovery and Data mining*, San Jose, California, USA, 2007.

[13] G. I. Webb, Discovering association rules with numeric variables, *Proc. of the 7th KDD*, 2001. NOI DAU?

[14] H. Yao, H. J. Hamilton, Mining itemsets utilities from transaction databases, *Data and Knowledge Engeneering* **59** (3) (2006).

[15] H. Yao, H. J. Hamilton, and C. J. Butz, A foundational approach to mining itemset utilities from databases, *Proceedings of the 4th SIAM International Conference on Data Mining*, Florida, USA, 2004.

[16] IBM Synthetic data, http://www.almaden.ibm.com/software/quest/Resources/index.shtml, 2004.

[17] H. Yao, H. J. Hamilton, and L. Geng, A unified framework for utility based measures for mining itemsets, *UBDM'06* Philadelphia, Pennsylvania, USA, August 2006.

Nhận bài ngày 7 - 7 - 2008