KHAI THÁC LUẬT THIẾT YẾU NHẤT TỪ TẬP PHỔ BIẾN ĐÓNG

Lê Hoài Bắc, Võ Đình Bảy
Trường Đại học Khoa học Tự nhiên, ĐHQG – HCM
(Bài nhận ngày14 tháng 12 năm 2006, hoàn chỉnh sửa chữa ngày 16 tháng 12 năm 2007)

TÓM TẮT: *Theo cách khai thác luật kết hợp truyền thống, việc tìm tất cả các luật kết hợp từ CSDL thỏa minSup và minConf gặp nhiều bất lợi khi số tập phổ biến lớn. Do đó cần có một phương pháp thích hợp để khai thác với số luật ít hơn nhưng vẫn bảo đảm tích hợp đầy đủ tất cả các luật của phương pháp khai thác truyền thống. Bài báo đề xuất thuật toán sinh luật thiết yếu nhất từ tập phổ biến đóng: chỉ lưu lại các luật có tiền kiện nhỏ nhất và hậu kiện lớn nhất theo quan hệ tập con. Thực nghiệm chứng tỏ tập luật kết quả khá nhỏ so với tập luật truyền thống, thời gian khai thác luật cũng nhanh hơn so với truyền thống bởi vì khai thác luật thiết yếu nhất dựa vào tập phổ biến đóng* (**FCI** – *Frequent Closed Itemsets*) *trong khi khai thác luật truyền thống dựa vào tập phổ biến* (**FI** – *Frequent Itemsets*) *mà* $|FCI| \leq |FI|$.

Từ khóa: *Tập phổ biến, tập phổ biến đóng, Minimal generator, luật truyền thống, luật thiết yếu nhất.*

1. GIỚI THIỆU

Trong hầu hết các thuật toán khai thác luật, các tác giả đặc biệt chú ý đến vấn đề làm thế nào để tìm tập phổ biến nhanh nhất có thể. Chính vì vậy, có khá nhiều tác giả chỉ tập trung vào việc nghiên cứu nhằm tìm ra thuật toán hiệu quả nhất cho bài toán tìm tập phổ biến. Tuy nhiên, với các CSDL đặc (mật độ trùng lặp các item giữa các dòng dữ liệu cao) hoặc khi *minSup* nhỏ dẫn đến số lượng tập phổ biến khá lớn thì thời gian khai thác và khối lượng bộ nhớ yêu cầu để lưu trữ tập phổ biến và luật kết hợp khá lớn – Vì vậy, các tác giả M. Zaki [7] và Y. Bastide [4] đã đưa ra một cách tiếp cận mới nhằm làm giảm khối lượng lưu trữ và thời gian khai thác: đó chính là khai thác luật kết hợp dựa vào tập đóng. Cách tiếp cận này có ưu điểm là số luật kết hợp giảm đáng kể so với phương pháp truyền thống nhưng vẫn bảo đảm tích hợp đầy đủ các luật còn lại. Do muốn bảo toàn thông tin về độ phổ biến(support) và độ tin cậy(confidence) của luật nên cả hai đều chỉ rút gọn trên các tập luật có cùng độ phổ biến và độ tin cậy. Tuy nhiên, khi người dùng muốn khai thác tập các luật thỏa *minSup* và *minConf* (nhưng không cần biết thông tin về độ phổ biến và độ tin cậy của từng luật), làm thế nào để khai thác tập luật nhỏ nhất thỏa mãn yêu cầu người dùng?.

Gần đây, các tác giả T. Xia, Y. Du, J. Shan, D. Zhang trong [5] đề xuất phương pháp khai thác luật thiết yếu nhất dựa vào tập phổ biến tối đại nhằm giới hạn không gian lưu trữ và thời gian khai thác so với phương pháp của Aggarwal và Yu [3]. Nhưng do khai thác trực tiếp trên tập phổ biến tối đại nên việc tính độ phổ biến của các tập con mất nhiều thời gian do phải đọc nhiều lần CSDL.

2. KHAI THÁC TẬP PHỔ BIẾN ĐÓNG

Để khai thác tập phổ biến đóng, chúng tôi sử dụng thuật toán **CHARM** được trình bày trong [6]. **CHARM** có ưu điểm là không sinh ứng viên và dựa vào phương pháp chia để trị nhằm tìm kiếm các tập phổ biến đóng với chỉ một lần đọc CSDL.

2.1 Một số định nghĩa [4],[6]

2.1.1 Kết nối Galois

Cho quan hệ hai ngôi $\delta \subseteq I \times T$ chứa CSDL cần khai thác, trong đó I là tập các danh mục còn T là tập các giao tác. Đặt $X \subseteq I$ và $Y \subseteq T$. Ta định nghĩa hai ánh xạ giữa $P(I)$ và $P(T)$ như sau:

a) $t : I \mapsto T,\ t(X) = \{y \in T \mid \forall x \in X, x\delta y\}$

b) $i : T \mapsto I,\ i(Y) = \{x \in I \mid \forall y \in Y, x\delta y\}$

2.1.2 Định nghĩa toán tử đóng

Cho $X \subseteq I$ và ánh xạ $c : P(I) \to P(I)$ với $c(X) = i(t(X))$. Ánh xạ c định nghĩa như trên được gọi là toán tử đóng (Closure Operator).

2.1.3 Định nghĩa tập đóng

Cho $X \subseteq I$. X được gọi là tập đóng khi và chỉ khi $c(X) = X$. X được gọi là tập phổ biến đóng nếu X phổ biến và X là tập đóng.

Ví dụ: xét CSDL được cho trong bảng 1 ta có

Do c(AW) = $i(t$(AW)) = i(1345) = ACW $\Rightarrow$ AW không phải là tập *đóng*. Do c(ACW) = $i(t$(ACW)) = i(1345) = ACW $\Rightarrow$ ACW là tập *đóng*.

2.2 Thuật toán 1 – Thuật toán CHARM

Đầu vào: CSDL D và ngưỡng phổ biến ***minSup***.

Kết quả: tập **FCI** gồm tất cả các tập phổ biến đóng của CSDL.

Phương pháp thực hiện:

CHARM(D, *minSup*)
  [∅]= $\{l_i \times t(l_i) : l_i \in I \wedge Sup(l_i) \geq minSup\}$
  CHARM-EXTEND([∅], **C** =∅)
  return **C**
CHARM-EXTEND([*P*], **C**)
  for each $l_i \times t(l_i)$ *in* [*P*] do
    $P_i = P \cup l_i$ *and* $[P_i] = \varnothing$
    for each $l_j \times t(l_j)$ *in* P *with* $j > i$ do
      $X = l_j$ *and* $Y = t(l_i) \cap t(l_j)$
CHARM-PROPERTY(*X*×*Y*, l_i, l_j, [P_i], [*P*]) **SUBSUMPTION-CHECK**(**C**, P_i)
CHARM-EXTEND([P_i], **C**)
    delete ([P_i])

CHARM-PROPERTY(*X*×*Y*, l_i,l_j,[P_i],[*P*])
  if $Sup(X) \geq minSup$ then
    if $t(l_i) = t(l_j)$ then
      Remove l_j from [*P*]

$P_i = P_i \cup l_j$

elseif $t(l_i) \subset t(l_j)$ then

$P_i = P_i \cup l_j$

elseif $t(l_i) \supset t(l_j)$ then

Remove l_j from [P]

Add $X \times Y$ to $[P_i]$

else

Add $X \times Y$ to $[P_i]$

Hình 1 - Thuật toán tìm tập phổ biến đóng thỏa ngưỡng minSup.

2.3 Minh họa

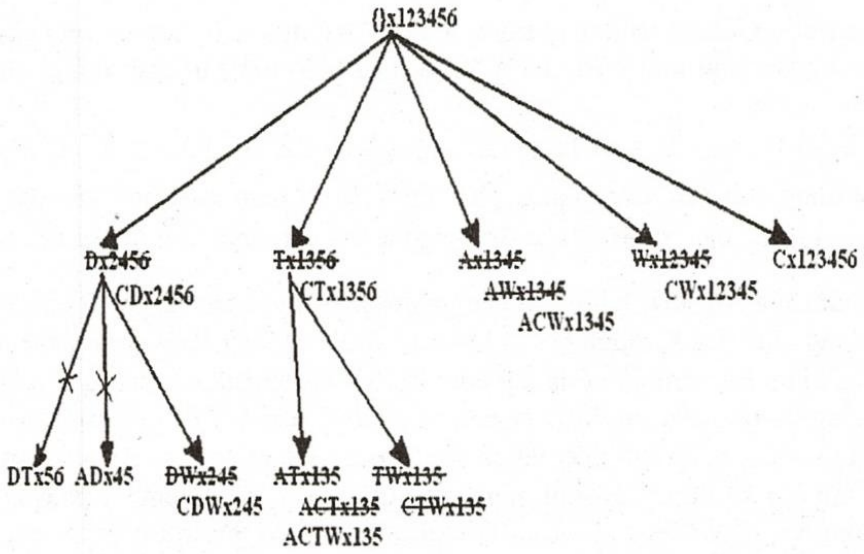
Xét CSDL sau [6]:

Bảng 1: CSDL mẫu ⇒ **Định dạng dữ liệu dọc**

Mã giao dịch	Nội dung giao dịch
1	A, C, T, W
2	C, D, W
3	A, C, T, W
4	A, C, D, W
5	A, C, D, T, W
6	C, D, T

Mã danh mục	Các giao dịch có chứa danh mục
A	1, 3, 4, 5
C	1, 2, 3, 4, 5, 6
D	2, 4, 5, 6
T	1, 3, 5, 6
W	1, 2, 3, 4, 5

Hình 2 minh họa quá trình tìm kiếm tập FCI thỏa ngưỡng phổ biến *minSup* = 50% trên cây IT-tree. Đầu tiên, tập $I = \{A, C, D, T, W\}$ được sắp xếp theo chiều tăng dần của độ phổ biến thành $l = \{D, T, A, W, C\}$. Khi $l_i = D$, nó kết hợp với các $l_j \in \{T, A, W, C\}$ thành các nút con $\{DT, DA, DW, DC\}$, do $Sup(DT) = Sup(DA) = 2 < minSup$ nên cả 2 không được sinh ra ở mức kế. Mặt khác, do $t(D) \cap t(C) = 2456 = t(D)$ nên D không thể là tập đóng và ta thay tập D bằng $D \cup C$.



Hình 2 - Cây IT-tree tìm tập phổ biến đóng thỏa ngưỡng minSup

3. LUẬT KẾT HỢP THIẾT YẾU NHẤT

3.1 Luật kết hợp [7]

3.1.1 Định nghĩa

Luật kết hợp là phép kéo theo có dạng $Y \xrightarrow{q,p} X - Y$ *(*X, Y *là các tập phổ biến) trong đó* $Y \subset X, Y \neq \varnothing$ *và* p = $\frac{Sup(X)}{Sup(Y)} \geq$ minConf *gọi là độ tin cậy của luật còn* q = Sup(X) *gọi là độ phổ biến của luật. Do* X *là phổ biến nên luật sinh ra là phổ biến.*

3.1.2 Tính chất

1. Nếu $X \to Y$ là luật kết hợp thì $X \cup A \to Y \setminus A$ cũng là luật kết hợp $\forall A \subset Y$.

2. Nếu $X \to Y$ là luật kết hợp thì $X \to Y \setminus A$ cũng là luật kết hợp $\forall A \subset Y$.

3. Nếu $X \to Y$ không là luật kết hợp thì $X \setminus A \to Y \cup A$ cũng không là luật kết hợp $\forall A \subset X$.

3.2 Minimal Generator (mG) [7], [9]

3.2.1 Định nghĩa

Cho X là tập đóng. Ta nói itemset X' là một ***generator*** của X khi và chỉ khi:

1. $X' \subseteq X$

2. $Sup(X') = Sup(X)$

Gọi $G(X)$ là tập các ***generator*** của X. Ta nói rằng $X' \in G(X)$ là một ***mG*** nếu nó không có tập con trong $G(X)$. Đặt $G^{min}(X)$ là tập tất cả các ***mG*** của X. Theo định nghĩa, $G^{min}(X) \neq \varnothing$ vì nếu nó không có ***generator*** hoàn toàn thì chính X là ***mG***. Chẳng hạn: xét tập đóng $ACTW$, các ***generator*** là $\{AT, TW, ACT, ATW, CTW\}$ và $G^{min}(ACTW) = \{AT, TW\}$.

Thuật toán tìm ***mG*** được trình bày trong hình 3. Nó dựa vào thực tế: ***mG*** của một itemset đóng X là các itemset con nhỏ nhất của X (thỏa điều kiện trên) nhưng không chứa bất kì tập đóng con nào của X. Đặt $S = \{X_i \mid (c(X_i) = X_i) \wedge (X_i \subset X) \wedge (\neg \exists X_j : (c(X_j) = X_j) \wedge (X_i \subset X_j \subset X))\}$ là tập tất cả các itemset đóng con trực tiếp của X. Thứ nhất, bất kì item xuất hiện lần đầu tiên trong X thuộc $I = X - \bigcup_{X_i \in S} X_i$ đều là ***mG*** theo định nghĩa. Từ các item còn lại, ta tìm tất cả các ***mG*** sử dụng một hàm tựa *Apriori*. Khởi tạo các ***generator*** ứng viên là tất cả các item đơn xuất hiện trong các tập con của X, nghĩa là $G_1(X)=\{i \mid i \in X - I\}$. Với mỗi ***generator*** ứng viên hiện hành $G \in G_k$ ta kiểm tra xem G có là tập con của bất kì itemset nào trong S hay không? Nếu đúng thì G không là ***mG***, nếu sai thì G là ***mG***, ta thêm G vào $G^{min}(X)$ và xóa khỏi G_k. Sau khi ta đã xét tất cả $G \in G_k$, ta đã tìm thấy tất cả các ***mG*** có kích thước k. Bước kế tiếp là sinh các ứng viên cho lần lặp kế tiếp. Với mỗi ***generator*** $G' \in G_{k+1}$, tất cả chúng phải là tập con trực tiếp có mặt trong G_k. Gọi $G' = i_1 i_2 \ldots i_k i_{k+1}$ là một ứng viên có thể trong G_{k+1}, việc kiểm tra tập con được thực hiện bằng cách kiểm tra xem liệu tập con G_j với kích thước k có đạt được bằng cách bỏ item i_j từ G' hiện diện trong G_k. Vì chúng ta xóa từ G_k ***minimal generator*** G bất kì, nên những tập không phải tập cha của G thậm chí cũng có thể trở thành các ***generator*** ứng viên. Kế tiếp, chúng ta lặp lại toàn bộ quá trình xử lý với G_{k+1} là tập ứng viên hiện hành. Xử lý kết thúc khi không còn ứng viên nào được sinh ra.

3.2.2 Thuật toán 2

Đầu vào: Tập phổ biến đóng X và S là các tập đóng con trực tiếp của X

Kết quả: $G^{min}(X)$ – tập tất cả các ***minimal generator*** của X.

Phương pháp thực hiện:

```
MINIMAL_GENERATOR( X, S)
    G^min(X) = ∅
    I = X − ∪_{X_i ∈ S} X_i
    for all i ∈ I do
        G^min(X) = G^min(X) ∪ {i}
    G_1 = {i | i ∈ X − I}
    k = 1
    while G_k ≠ ∅ do
        for all G ∈ G_k do
            if G ⊆ X_j ∀X_j ∈ S then
                G^min(X) = G^min(X) ∪ {G}
                G_k = G_k − G
G_{k+1} ={G'=i_1 i_2 ... i_k i_{k+1} | ∀1≤j≤k+1, ∃ G_j ∈ G_k, G_j = i_1 i_2 ... i_{kj-}
        _1 i_{j+1} ... i_k i_{k+1}}
        k = k+1
```

Hình 3. Thuật toán tìm **Minimal Generator** của tập đóng X

Ví dụ: xét $X = ACTW$, Ta có $S = \{CT, ACW\} \Rightarrow I = \varnothing$ và $G_1 = \{A, C, T, W\}$. Ta thấy rằng các item này là tập con của các tập trong S nên chúng không là ***generator*** đơn. Với lần lặp kế, ta có $G_2 = \{AC, AT, AW, CT, CW, TW\}$. Từ G_2, do AT, TW không phải là tập con của các tập trong S nên ta thêm chúng vào $G^{min}(X)$ và xóa chúng khỏi $G_2 \Rightarrow G^{min}(X)=\{AT, TW\}$ và $G_2 = \{AC, AW, CT, CW\}$. Bây giờ, với lần lặp thứ 3 ta có $G_3 = \{ACW\}$. Do đây là tập con của một

itemset trong S nên nó không thể là *generator*. Cuối cùng ta có $G_4 = \varnothing \Rightarrow$ dừng. Vậy $G^{min}(ACTW) = \{AT, TW\}$.

3.3 Khai thác luật thiết yếu nhất

3.3.1 Định nghĩa 1 – Luật tổng quát

Cho R_i chỉ luật $X_i \xrightarrow{q_i, p_i} Y_i$; Ta nói luật R_1 là tổng quát hơn luật R_2, kí hiệu $R_1 \prec R_2$, nếu $X_1 \subseteq X_2$, $Y_1 \supseteq Y_2$ và $X_1 \cup Y_1 \subset X_2 \cup Y_2$, nghĩa là R_2 có thể được sinh ra từ R_1 bằng cách thêm các item vào vế trái hoặc giảm bớt các item ở vế phải của R_1 đồng thời nó phải là luật chứa trong R_1.

3.3.2 Định nghĩa 2 – Tập luật thiết yếu nhất

Cho $R = \{R_1, ..., R_n\} (R_i = X_i \xrightarrow{q_i, p_i} Y_i)$ là tập tất cả các luật kết hợp truyền thống. Đặt $R_E = \{R_i \in R: (\neg \exists R_j \in R: R_j \prec R_i)\}$, R_E được gọi là tập luật thiết yếu nhất của R.

3.3.3 Nhận xét

1. Các luật thiết yếu nhất được suy từ ***Minimal Generator*** của tập đóng X sang tập đóng Y trong đó $X \subseteq Y$.

2. Luật thiết yếu nhất có độ tin cậy 100% được suy từ ***Minimal Generator*** của tập đóng X sang chính X.

3. Nếu $X \subset Y$ và $X \rightarrow Y - X\ (R_1)$ là luật thiết yếu nhất thì

(a) $X \rightarrow Z - X\ (R_2)$ và

(b) $Z \rightarrow Y - Z\ (R_3)$ không là luật thiết yếu nhất $\forall Z: X \subset Z \subset Y$.

Chứng minh:

(a) Do $X \subseteq X$ và $Y - X \supset Z - X$ (vì $Z \subset Y$) nên theo định nghĩa 1, $R_1 \prec R_2 \Rightarrow R_2$ không là luật thiết yếu nhất.

(b) Do $X \subset Z$ và $Y - X \supset Y - Z$ (vì $X \subset Z$) nên theo định nghĩa 1, $R_1 \prec R_3 \Rightarrow R_3$ không là luật thiết yếu nhất.

3.3.4 Thuật toán sinh tập luật thiết yếu nhất – thuật toán 3

Đầu vào: tập **FCI** chứa các tập phổ biến đóng thỏa ngưỡng phổ biến ***minSup*** và ngưỡng tin cậy ***minConf***.

Kết quả: tập **AR** gồm tất cả các luật thiết yếu nhất thỏa ***minConf***.

Phương pháp thực hiện:

```
MINING_ESSENTIAL_AR()
  AR = ∅
  SORT (FCI) // SX tập FCI tăng theo k-itemset
  for i = 1 to |FCI|-1 do
    X = FCI_i
    Superset = ∅
    f = true
    for j = |FCI| downto i+1 do
      Y = FCI_j
      if Sup(Y) / Sup(X) ≥ minConf then
        if X ⊂ Y and {¬∃Y' ∈ Superset | Y ⊂ Y'} then
          ENUMERATE_RULE(X, Y)
```

```
            Superset = Superset ∪ Y
            f = false
      if  f = true  then
          ENUMERATE_RULE( Y, Y)

ENUMERATE_RULE(X, Y, conf)
  for all Z ∈ mG(X)  do
     if  ESSENTIAL_RULE(Z, Y \ Z)  then
         AR = AR ∪ {Z → Y \ Z (Sup(Y),conf)}

ESSENTIAL_RULE(X, Y)
  for all  X1→Y1, ∈ AR  do
     if  X ⊇ X1 and  Y ⊆ Y1 and X ∪ Y  ⊂ X1 ∪ Y1 then
        return false
  return true
```

Hình 4 - Thuật toán sinh tập luật thiết yếu nhất từ tập phổ biến đóng

3.3.5 Định lý 1 – Thuật toán tìm luật thiết yếu nhất ở trên là đúng đắn.

Chứng minh: để chứng minh định lý, ta cần chứng minh hai vấn đề sau:

1. Luật thiết yếu nhất được sinh ra giữa tập đóng X và tập đóng Y ($X \subset Y$) nếu có chỉ từ $mG(X)$ sang Y.

Chứng minh: xét tất cả các X' và Y': $mG(X) \subset X' \subset X$, $mG(Y) \subset Y' \subset Y$. Do $Sup(mG(X)) = Sup(X) = Sup(X')$, $Sup(mG(Y) = Sup(Y) = Sup(Y')$ nên luật $mG(X) \rightarrow Y$ (R_1) và luật $X' \rightarrow Y'$ (R_2) có cùng độ phổ biến và độ tin cậy. Có hai khả năng xảy ra như sau:

+ Nếu luật R_1 không thỏa *minConf* thì luật R_2 cũng không thỏa *minConf*.

+Nếu luật R_1 là thiết yếu nhất thì rõ ràng R_2 không là luật thiết yếu nhất vì ta có $X' \cup Y' \subset X \cup Y \Rightarrow R_1 \prec R_2$.

2. Nếu có luật thiết yếu nhất được sinh ra giữa tập đóng X và tập đóng Y thì mọi luật sinh ra giữa X và Z, giữa Z và Y đều không là luật thiết yếu nhất trong đó $X \subset Z \subset Y$.

Chứng minh: hiển nhiên do nhận xét 3 (phần 3.3.3).

Do thuật toán sắp xếp tập **FCI** tăng dần theo k-itemset nên việc xét luật sinh ra giữa tập đóng X và tập đóng Y (trong đó X xét từ đầu về cuối, Y xét từ cuối về đầu) là đầy đủ về luật. Mặt khác, do thuật toán có kiểm tra một luật có thiết yếu hay không bằng hàm **ESSENTIAL_RULE** nên nó bảo đảm tính không dư thừa luật. (đpcm).

3.3.6 Minh họa thuật toán

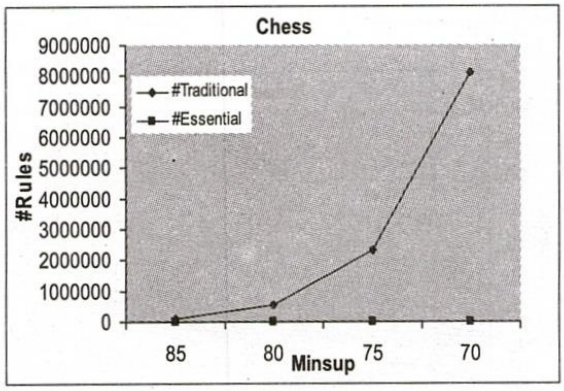
Xét CSDL trong bảng 1 với *minSup* = 50%, *minConf* = 0%: kết quả có 7 luật thiết yếu nhất trong khi số luật truyền thống là 60. Tỉ lệ tích hợp luật = 7/60*100% = 11.67%.

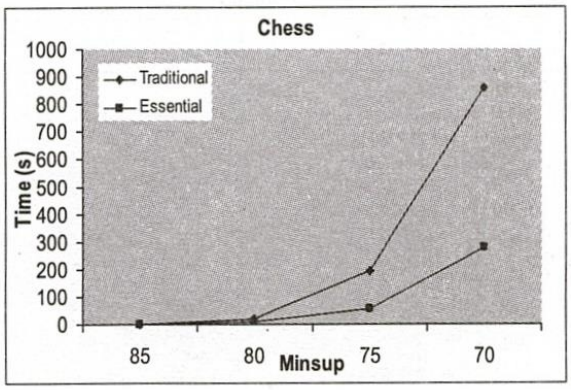
Bảng 2: Tập tất cả các thiết yếu nhất với *minSup* = 50% và *minConf* = 0%

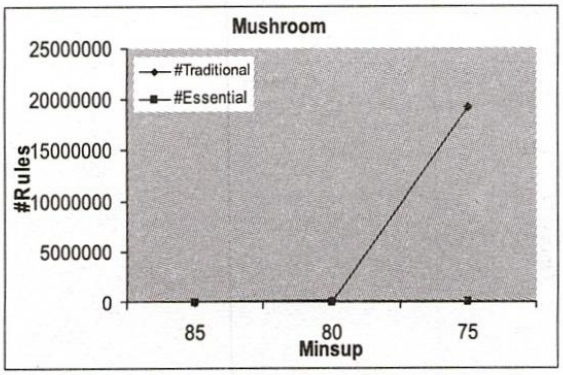
STT	Tập đóng	*Sup*	***mG***	Superset	Các luật thỏa *minConf*
1	*C*	6	*C*	*ACTW, CDW*	$C \xrightarrow{3,3/6} ATW$, $C \xrightarrow{3,3/6} DW$
2	*CD*	4	*D*	*CDW*	$D \xrightarrow{3,3/4} CW$
3	*CT*	4	*T*	*ACTW*	$T \xrightarrow{3,3/4} ACW$

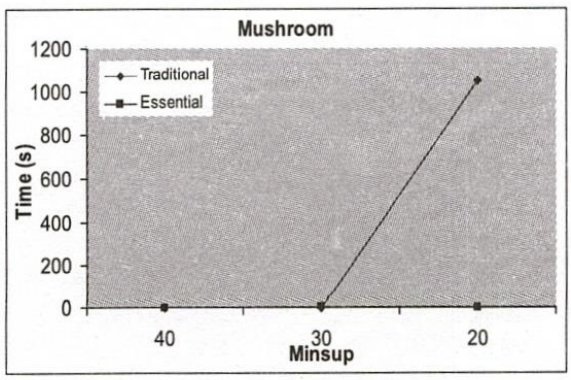
4	CW	5	W	$ACTW, CDW$	$W \xrightarrow{3,3/5} ACT$, $W \xrightarrow{3,3/5} CD$
5	ACW	4	A	$ACTW$	$A \xrightarrow{3,3/4} CTW$
6	CDW	3	DW		
7	$ACTW$	3	AT, TW		

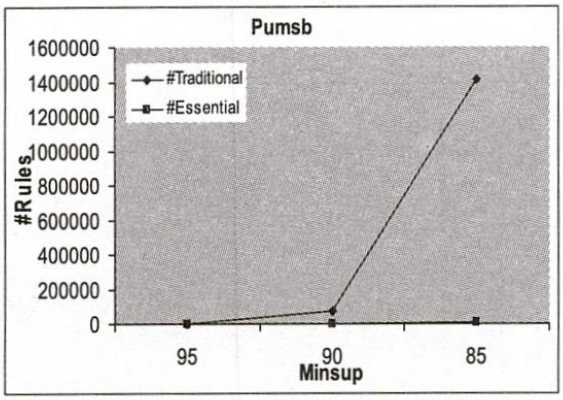
3.4 Kết quả thực nghiệm

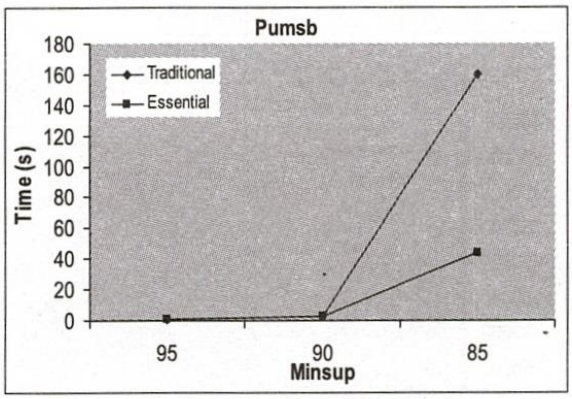


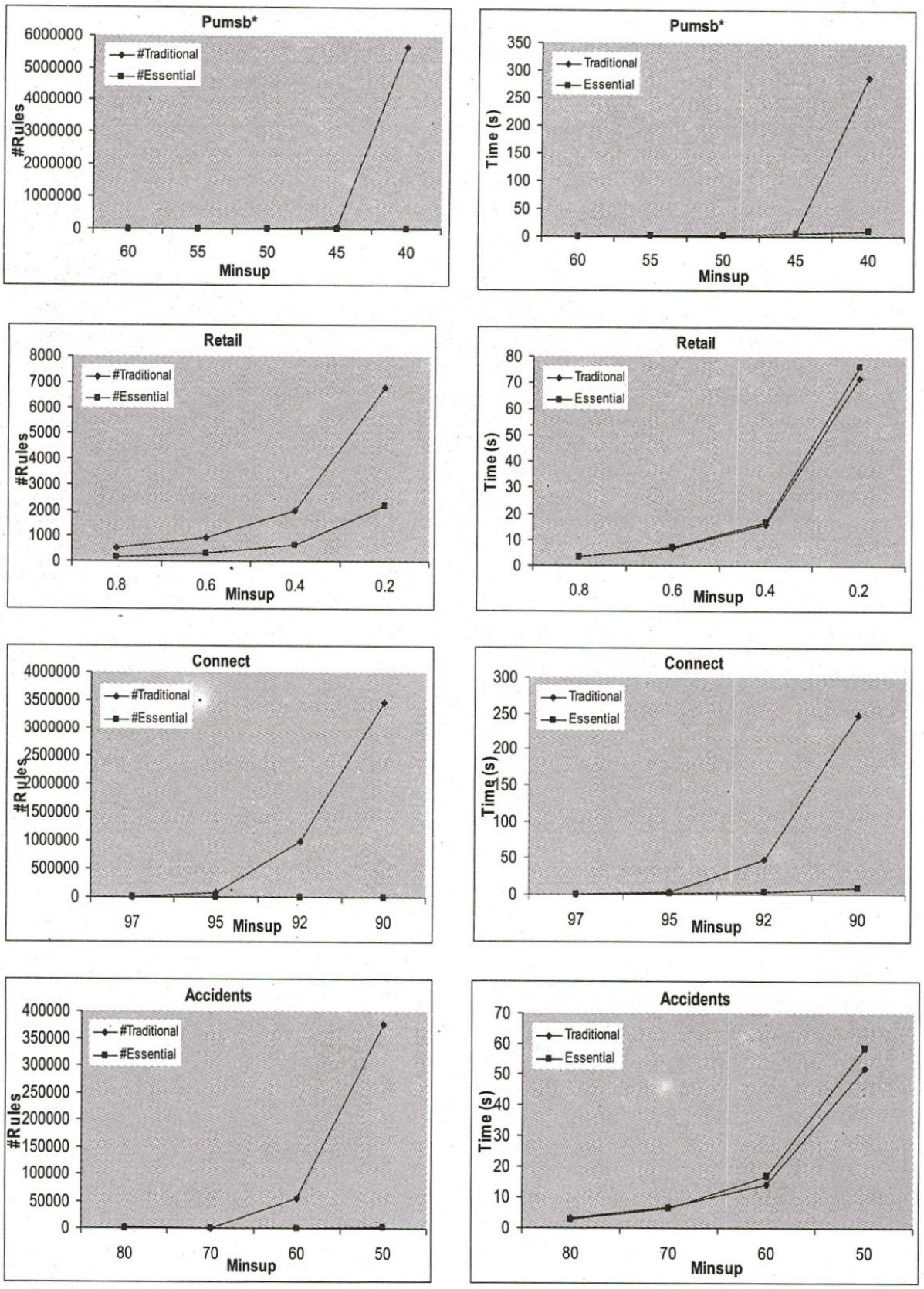












Hình 5 - Kết quả thực nghiệm trên các CSDL với *minConf* = 0%

Các CSDL chuẩn được lấy từ http://fimi.cs.helsinki.fi/data có đặc điểm như sau:

Tên CSDL	Số giao dịch	Số danh mục	Độ dài trung bình	Độ dài tối đa
Chess	3196	75	37	37
Mushroom	8124	120	23	23
Pumsb*	49046	7117	50	62
Pumsb	49046	7117	73.6	74
Connect	67557	130	43	43
Retail	88162	16469	10.3	76
Accidents	340183	468	33.8	51

Từ kết quả của hình 5, có thể thấy số luật thiết yếu nhất ít hơn nhiều so với số luật truyền thống. Bên cạnh đó, thời gian khai thác luật thường cũng nhỏ hơn so với khai thác tập luật theo phương pháp truyền thống. Số luật càng lớn, độ lệch thời gian giữa khai luật truyền thống và luật thiết yếu nhất càng lớn.

4.KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Khai thác luật kết hợp truyền thống sinh ra khá nhiều luật dư thừa thường gây khó khăn cho người sử dụng. Vì vậy, cần có phương pháp khai thác hiệu quả hơn tập luật kết quả. Một trong những phương pháp đó là khai thác luật tập thiết yếu nhất dựa vào tập phổ biến đóng. Tập luật thiết yếu nhất thường khá nhỏ so với tập luật truyền thống nhưng tích hợp đầy đủ các luật còn lại. Chính vì vậy, có thể nói đây là một trong những phương pháp hiệu quả để khai thác luật kết hợp hiểu theo nghĩa: tính tiện dụng trong khai thác, giảm không gian và thời gian khai thác luật.

Tuy nhiên, với một số CSDL(chẳng hạn Retail, Accidents) thì thời gian vẫn chưa hiệu quả hơn phương pháp truyền thống do thuật toán 3 xét luật $X \rightarrow Y$ với tất cả các luật hiện có để kiểm tra nó có thiết yếu nhất hay không? Vì vậy, cần có phương pháp kiểm tra hiệu quả hơn với mong muốn làm giảm thời gian khai thác luật.

Xét về khía cạnh khai thác luật, đôi khi người dùng không cần biết tất cả các luật kết quả mà chỉ muốn biết các luật theo mong muốn. Do đó, có thể hướng đến phương pháp tìm luật theo yêu cầu người dùng (chẳng hạn truy vấn trên tập luật, …).

MINING ESSENTIAL RULES USING FREQUENT CLOSED ITEMSETS

Le Hoai Bac, Vo Dinh Bay
University of Natural Sciences, VNU - HCM

ABSTRACT: *According to the traditional association rules mining, finding all association rules satisfied minSup and minConf will face to many disadvantages in cases of the large frequent itemsets. Thus, it is necessary to have a suitable method for mining in number of fewer rules but make sure fully integrating rules of traditional methods. In this paper, we present an algorithm of generating essential rules from frequent closed itemsets: only stores rules having smallest antecedent and largest consequent based on parents-child relationship. Experiment shows that the resulted rule set is rarely as small as traditional set,*

the time for rule mining is also faster than the time for traditional because the mining essential rules are based on frequent closed itemsets (FCI) whereas mining traditional rules based on frequent itemsets (FI) that satisfies $|FCI| \leq |FI|$.

***Keywords**: Frequent Itemset, Frequent Closed Itemset, Minimal generator, Traditional Association Rules, Essential Association Rules.*

TÀI LIỆU THAM KHẢO

[1]. R. Agrawal and R. Srikant, *Fast algorithms for mining association rules*, In VLDB'94.

[2]. R. Agrawal, T. Imielinski, and A. Swami, *Mining association rules between sets of items in large databases*, In SIGMOD'93.

[3]. C. C. Aggarwal and P. S. Yu, *Online Generation of Association Rules*, In Proceedings of International Conference on Data Engineering (ICDE), pages 402–411, (1998).

[4]. Y. Bastide, N. Pasquier, R. Taouil, G. Stumme, L. Lakhal, *Mining Minimal Non-Redundant Association Rules using Closed Frequent Itemssets*, In 1st International Conference on Computational Logic, (2000).

[5]. T. Xia, Y. Du, J. Shan and D. Zhang, *ERMiner:A Novel Method to Mine Essential Rules*, ACM SIGKDD'04 August 2225, Seattle, USA,(2004).

[6]. M. J. Zaki, C.J. Hsiao, *Efficient Algorithms for Mining Closed Itemsets and Their Lattice Structure*, IEEE Transactions on Knowledge and Data Engineering, (2005).

[7]. M. J. Zaki, *Mining Non-Redundant Association Rules*, Data Mining and Knowledge Discovery, 9, 223–248, Kluwer Academic Publishers. Manufactured in The Netherlands, (2004).

[8]. M. J. Zaki and K. Gouda, *Fast Vertical Mining Using Diffsets*, Proc. Ninth ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining, Aug. (2003).

[9]. M. J. Zaki, *Generating Non-Redundant Association Rules*, In 6th ACM SIGKDD Intl Conf. Knowledge Discovery and Data Mining, (2001).