KHAI THÁC TẬP HỮU ÍCH CAO TƯƠNG QUAN TRÊN CƠ SỞ DỮ LIỆU GIAO DỊCH CÓ LỢI NHUẬN ÂM

Nguyễn Văn Lễ[*], Nguyễn Thị Thanh Thủy[+], Mạnh Thiên Lý[+]

[*]Khoa Công nghệ thông tin, Trường Đại học Công nghiệp Thực phẩm TPHCM

[+]Khoa Công nghệ thông tin, Trường Đại học Công nghiệp Thực phẩm TPHCM

Tóm tắt: Việc khai thác tập hữu ích cao tương quan (Correlated Hight Utility Itemset) trong cơ sở dữ liệu giao dịch đã được nghiên cứu rộng rãi để khám phá hành vi mua hàng của người dùng. Từ đó, các nhà quản lý doanh nghiệp có thể điều chỉnh chiến lược bán hàng một cách phù hợp để tăng lợi nhuận. Các cách tiếp cận khai thác tập hữu ích cao tương quan trước đây chỉ thực hiện trên cơ sở dữ liệu có lợi nhuận dương mà ít khi quan tâm đến các giá trị lợi nhuận âm. Trên thực tế, các doanh nghiệp có thể giảm lợi nhuận của các mặt hàng tồn kho lâu ngày để kích thích người mua, thậm chí có thể giảm lợi nhuận đến mức âm để tiêu thụ hết lượng hàng tồn kho. Để khai thác tập hữu ích cao tương quan hiệu quả trên cơ sở dữ liệu giao dịch có lợi nhuận âm, chúng tôi đề xuất thuật toán CHN (Correlated High utility itemset with Negative profit). Đánh giá thử nghiệm trên cả năm cơ sở dữ liệu Chess, Mushroom, Pumsb, Retail và Kosarak và cho thấy thuật toán CHN có hiệu suất thực thi một cách hiệu quả.

Từ khóa: Tập hữu ích cao, tính tương quan, khai thác dữ liệu, cơ sở dữ liệu giao dịch, tập hữu ích cao tương quan.

I. GIỚI THIỆU

Xã hội ngày càng phát triển thì nhu cầu mua sắm của khách hàng ngày càng tăng. Các tổ chức kinh doanh cũng đã được thành lập ở khắp mọi nơi với các hình thức bán hàng đa dạng. Khi khách hàng càng có nhiều sự lựa chọn thì doanh nghiệp càng phải nghiên cứu thói quen mua hàng của họ. Ban đầu, các doanh nghiệp quan tâm đến việc tìm hiểu các mặt hàng thường được khách hàng mua cùng nhau. Nhiều thuật toán đã được đề xuất để khai thác các luật kết hợp để giải quyết hiệu quả nhu cầu này [1,2]. Tuy nhiên, luật kết hợp có sự giới hạn là không đề cập đến lợi nhuận của sản phẩm. Các thuật toán khai thác tập hữu ích cao (HUIM) đã được nghiên cứu để tìm ra tập các sản phẩm có lợi nhuận cao (High Utility Itemset – HUI) như: Thuật toán Two-Phase [3], UP-Growth [4], HUI-Miner [5], FHM [6], EFIM [7], D2HUP [8]. Năm 2016, Lin và cộng sự đề xuất thuật toán FHN [9] khai thác HUI trên cơ sở dữ liệu có lợi nhuận âm một cách hiệu quả. Mặc dù các thuật toán này khá hữu ích trong việc khám phá các tập hữu ích cao, tuy nhiên số lượng HUI được tìm thấy có thể rất lớn vì chúng chỉ tìm thấy các HUI dựa trên ngưỡng tối thiểu và bỏ qua mối tương quan giữa các mục bên trong mẫu. Nhiều HUI chứa các mặt hàng có tương quan yếu và thực tế không mang lại ý nghĩa. Để giải quyết vấn đề này, J. C. W. Lin và cộng sự đã đề xuất thuật toán FDHUP [10] để khai thác HUI có ràng buộc tần suất cao. Sau đó, vào năm 2017, W. Gan và cộng sự đề xuất thuật toán CoHUIM [11] xem xét cả tính tương quan và lợi nhuận giữa các sản phẩm trong giao dịch. Tuy nhiên, thuật toán này tạo ra một tập hợp lớn các ứng cử viên và việc quét lại cơ sở dữ liệu gốc nhiều lần làm tăng đáng kể thời gian thực hiện và bộ nhớ lưu trữ. Năm 2019, nhóm tác giả W. Gan và cộng sự đề xuất thuật toán CoUPM [12] cải tiến từ thuật CoHUIM bằng việc sử dụng cấu trúc lưu trữ Ulility-List tăng hiệu suất khai thác tập hữu ích cao tương quan. Tuy nhiên, thuật toán này vẫn kém hiệu quả khi thực thi trên cơ sở dữ liệu có lợi nhuận âm. Để giải quyết vấn đề này, chúng tôi đề xuất một thuật toán mới có tên là CHN để khai thác tập hữu ích cao tương quan một cách hiệu quả trên cơ sở dữ liệu giao dịch có lợi nhuận âm.

Những đóng góp chính của bài báo:

a) Áp dụng cấu trúc dữ liệu PNU – List để lưu trữ dữ liệu trong quá trình khai thác tập hữu ích cao tương quan CHUIs trên cơ sở dữ liệu giao dịch có lợi nhuận âm.

b) Áp dụng nhiều chiến lược tỉa để giảm không gian tìm kiếm như: U-Prune, LA-Prune, Kulc-Prune, EUCS.

c) Kết quả thực nghiệm so sánh với thuật toán CoUPM cho thấy thuật toán CHN có thời gian thực hiện nhanh hơn thuật toán CoUPM trên 5 cơ sở dữ liệu là Chess, Mushroom, Pumsb, Retail và Kosarak.

Cấu trúc bài báo được chia làm 6 phần. Phần 1 trình bày giới thiệu; Phần 2 trình bày các công trình liên quan; Phần 3 trình bày các định nghĩa và ký hiệu; Phần 4 trình bày thuật toán đề xuất CHN; Phần 5 trình bày kết quả thực nghiệm và đánh giá; Phần 6 trình bày kết luận và hướng phát triển.

II. CÁC CÔNG TRÌNH LIÊN QUAN

Khai thác tập hữu ích cao (HUI) đã được nghiên cứu rộng rãi và có nhiều ứng dụng trong thực tế để hỗ trợ việc kinh doanh hiệu quả hơn. Các tiếp cận ban đầu khai thác HUI chủ yếu dựa trên hai pha, trong đó pha một sinh ra các ứng viên có khả năng là tập hữu ích cao, pha hai thực hiện quét lại cơ sở dữ liệu để xác định ứng viên nào thực sự là HUI. Một số thuật toán được đề xuất trong giai đoạn này như: Two-Phase [3], CTU-Mine [13], TWU-Mining

Tác giả liên hệ: Nguyễn Văn Lễ,
Email: lenv@hufi.edu.vn
Đến tòa soạn: 8/2020, chỉnh sửa: 9/2020, chấp nhận đăng: 10/2020.

[14], UP-Growth [4], UP-Growth+ [15]. Những thuật toán này tốn nhiều thời gian và bộ nhớ vì phải thực hiện trên hai pha và quét cơ sở dữ liệu nhiều lần. Năm 2012, Liu và cộng sự đề xuất thuật toán HUI-Miner [5] cùng với cấu trúc dữ liệu mới là Utility-List để khai thác HUI trên một pha cùng với chiến lược tỉa TWU đã làm giảm đáng không gian tìm kiếm và tăng hiệu suất thực thi của thuật toán. Năm 2014, Fournier và cộng sự bổ sung chiến lược tỉa EUCP vào thuật toán HUI-Miner và đề xuất thuật toán FHM [6] có thời gian khai thác HUI nhanh vượt trội so với thuật toán HUI-Miner. Năm 2017, Zida cùng cộng sự đề xuất thuật toán EFIM [7] sử dụng cơ chế chiếu và gộp trên cơ dữ liệu có hiệu quả cao khi khai thác trên cơ sở dữ liệu thưa. Năm 2018, Krishnamoorthy đề xuất thuật toán HMiner [16] với cấu trúc dữ liệu mới là CUL và nhiều chiến lược tỉa hiệu quả cho hiệu suất thực thi vượt trội cả về thời gian và bộ nhớ sử dụng so với thuật toán EFIM. Tuy nhiên, các thuật toán này vẫn kém hiệu quả và tìm ra số lượng HUI không đầy đủ khi khai thác trên cơ sở dữ liệu có lợi nhuận âm. Để giải quyết vấn đề này, Lin và cộng sự đã đề xuất cải tiến cấu trúc Utility-list để lưu trữ tách biệt lợi nhuận dương và âm và đề xuất thuật toán FHN [9] để khai thác đầy đủ tập HUI trên cơ sở dữ liệu giao dịch có lợi nhuận âm. Ngoài ra, năm 2019, Singh và cộng sự đề xuất thuật toán EHNL [17] để khai thác HUI hiệu quả với ràng buộc chiều dài tập mục trên cơ sở dữ liệu có lợi nhuận âm.

Tập hữu ích cao được khai thác với các thuật toán trình bày ở trên đã có nhiều ứng dụng hữu ích trong thực tế. Tuy nhiên, các thuật toán này chỉ quan tâm đến độ hữu ích của tập mục kết quả mà không quan tâm đến sự tương quan giữa các mục bên trong. Do đó, kết quả khai thác được còn chứa nhiều tập mục có mối tương quan thấp giữa các mục bên trong và không có ý nghĩa trong thực tế. Ví dụ, trong cơ sở dữ liệu phát sinh một giao dịch mua hàng gồm một sản phẩm (A) có trị giá lợi nhuận rất cao kèm với một sản phẩm (B) có lợi nhuận thấp và chỉ có một giao dịch này trong cơ sở dữ liệu.

Khi khai thác HUI, tập gồm hai sản phẩm này sẽ được tìm thấy vì lợi nhuận cao vượt ngưỡng mức lợi nhuận tối thiểu. Tuy nhiên, sự tương quan giữa hai sản phẩm này là rất kém vì chỉ có một giao dịch trong cơ sở dữ liệu nên khi kết luận rằng khi khách hàng mua sản phẩm A thì sẽ mua kèm sản phẩm B là không chính xác. Để giải quyết vấn đề này, nhiều nghiên cứu đã được đề xuất để tính toán mối tương quan giữa các mục trong một tập mục được khai thác. Khái niệm về "độ đo" giữa các mục được Omiecinski [18] đưa ra từ rất sớm để khai thác luật kết hợp với ba độ đo là any-confidence, all-confidence và bond. Tuy nhiên, ba độ đo này vẫn chưa đánh giá được mối tương quan giữa các mục trong tập mục được khai thác. Năm 2018, Fournier-Viger và cộng sự [19] đã trình bày khái niệm "Correlation" dựa trên ba độ đo trên và đề xuất thuật toán CHIs để khai thác tập hữu ích cao tương quan trên cơ sở dữ liệu giao dịch. Bên cạnh đó, Lin và cộng sự đề xuất thuật toán CoHUIM [11] khai thác tập hữu ích cao tương quan (CoHUI) dựa trên độ đo Kulc và cơ chế chiếu trên cơ sở dữ liệu trong quá trình khai thác. Năm 2019, Gan và cộng sự [12] đề xuất thuật toán CoUPM khai thác CoHUI dựa trên cấu trúc dữ liệu Utility-list với nhiều chiến lược tỉa khác nhau đã cho kết quả khai thác CoHUI hiệu quả hơn thuật toán CoHUIM.

III. CÁC ĐỊNH NGHĨA VÀ KÝ HIỆU

Cho tập $I = \{i_1, i_2, \dots, i_m\}$ gồm m mục phân biệt trong cơ sở dữ liệu giao dịch $D = \{T_1, T_2, \dots, T_n\}$, với n là số lượng giao dịch trong D. $\forall T_j \in D$, $T_j = \{x_l | l = 1, 2, \dots, N_j, x_l \in I\}$, với N_j là số các mục trong giao dịch T_j. Bảng 1 trình bày ví dụ về cơ sở dữ liệu giao dịch D và Bảng 2 trình bày lợi nhuận các mục.

Mỗi mục x_i trong tập mục I có một giá trị lợi nhuận xác định là $P(x_i)$ và có một số lượng mua $Q(x_i, T_j)$ trong giao dịch T_j. Độ hữu ích của mục x_i trong giao dịch T_j ký hiệu là $U(x_i, T_j)$ với $U(x_i, T_j) = P(x_i) * Q(x_i, T_j)$.

Độ hữu ích của một tập mục $X = \{x_1, x_2, \dots, x_k\} \subseteq T_j$ định nghĩa là $U(X, T_j) = \sum_{x_i \in X} U(x_i, T_j)$. Độ hữu ích dương của tập mục X trong giao dịch T_j được định nghĩa là $PU(X, T_j) = \sum_{x_i \in X \wedge U(x_i, T_j) > 0} U(x_i, T_j)$. Độ hữu ích âm của tập X trong giao dịch T_j được định nghĩa là $NU(X, T_j) = \sum_{x_i \in X \wedge U(x_i, T_j) < 0} U(x_i, T_j)$. Ta có $U(X, T_j) = PU(X, T_j) + NU(X, T_j)$.

Độ hữu ích dương của tập mục X trong cơ sở dữ liệu D được định nghĩa $PU(X) = \sum_{X \subseteq T_j \wedge T_j \in D} PU(X, T_j)$. Độ hữu ích âm của X trong D được định nghĩa $NU(X) = \sum_{X \subseteq T_j \wedge T_j \in D} NU(X, T_j)$. Độ hữu ích của tập mục X trong cơ sở dữ liệu D được định nghĩa là $U(X) = PU(X) + NU(X)$. Độ hữu ích dương của một giao dịch $T_j \in D$ được định nghĩa là $PTU(T_j) = \sum_{x_i \in T_j \wedge \mathrm{U}(x_i) > 0} U(x_i, T_j)$ [9].

Bảng I. Cơ sở dữ liệu giao dịch có lợi nhuận âm

TID	Giao dịch (T)	Số lượng (Q)	Độ hữu ích (U)	Độ hữu ích dương (PTU)
T_1	a, b, c, d	3, 1, 4, 2	9, -1, 16, -4	25
T_2	a, b, c	3, 3, 2	9, -3, 8	17
T_3	a, c, d, e, f	4, 1, 2, 2, 1	12, 4, -4, 6, 3	25
T_4	b, c, e	2, 5, 4	-2, 20, 12	32
T_5	b, c, d, e	6, 5, 4, 6	-6, 20, -8, 18	38
T_6	a, b, c, d, e, f	2, 5, 1, 3, 1, 1	6, -5, 4, -6, 3, 3	16
T_7	c, e	2, 3	8, 9	17
T_8	c, d, e	5, 2, 4	20, -4, 12	32

Bảng II. Lợi nhuận của các mục

Các mục (Items)	1	2	3	4	5	6
Lợi nhuận (P)	3	-1	4	-2	3	3

Định nghĩa 1. Tập mục X được gọi là một tập mục hữu ích cao (HUI) nếu độ hữu ích của X lớn hơn hoặc bằng giá trị ngưỡng độ hữu ích tối thiểu (minUtil). Trong đó giá trị minUtil được cung cấp bởi người dùng [16]. Ta có: $HUIs = \{X \mid U(X) >= minUtil\}$

Định nghĩa 2. Giá trị *Positive Transaction Weighted Utility* ($PTWU$) của tập mục X *trong D ký hiệu là* $PTWU(X) = \sum_{X \subseteq T_j \in D} PTU(T_j)$ [9]. Ví dụ, trong Bảng 1, *ta có* $PTWU(ab) = PTU(T_1) + PTU(T_2) + PTU(T_6) = 25 + 17 + 26 = 68$. Bảng 3 trình bày các giá trị $PTWU$ của các tập mục gồm một phần tử trong D.

Tính chất 1. Nếu $PTWU(X) < minUtil$ thì mọi tập mục mở rộng từ X đều không là HUI [16].

Định nghĩa 3. Độ hỗ trợ của tập mục X trong cơ sở dữ liệu D ký hiệu $SUP(X)$ là số lượng các giao dịch chứa X trong D [16]. Ví dụ $SUP(ab) = 3$ vì có 3 giao dịch chứa

{ab} là T_1, T_2 và T_6. Độ hỗ trợ của mỗi mục trong cơ sở dữ liệu D thể hiện ở Bảng 3.

Bảng III. Độ hỗ trợ của các mục

Items	f	a	b	d	e	c
PTWU	41	83	128	136	160	202
SUPPORT	2	4	5	5	6	8

Định nghĩa 4. (Thứ tự các mục) Thứ tự toàn phần của các mục trong cơ sở dữ liệu D được sắp xếp tăng dần theo độ hỗ trợ [11]. Xét 2 mục x và y, Ta có $x \prec$ y nếu $SUP(x) < SUP(y)$. Trường hợp $SUP(x) = SUP(y)$ thì thứ tự dựa vào thứ tự Alphabet của x và y. Với cơ sở dữ liệu D (Bảng 1) và độ hỗ trợ (Bảng 3) thì thứ tự toàn phần được xác định là f $\prec$ a $\prec$ b $\prec$ d $\prec$ e $\prec$ c

Với giá trị *minUtil = 42*, mục f bị loại bỏ khỏi cơ sở dữ liệu D (Tính chất 1) và toàn bộ cơ sở dữ liệu D được sắp xếp lại theo thứ tự toàn phần được trình bày trong Bảng 4.

Định nghĩa 5. Sự tương quan giữa các phần tử trong một tập mục $X = \{x_1, x_2, \dots, x_k\}$ được định nghĩa là $kulc(X) = \frac{1}{k}\left(\sum_{x_i \in X} \frac{Sup(X)}{\sup(x_i)}\right)$. Với $0 \leq kulc(X) \leq 1$ [11].

Giá trị $kulc(X)$ càng gần 1 thì tính tương quan giữa các mục trong X càng cao. Ngược lại, $kulc(X)$ càng gần 0 thì tính tương quan giữa các mục trong X càng thấp.

Ví dụ: $kulc(ab) = \frac{1}{2}\left(\frac{\sup(ab)}{\sup(a)} + \frac{\sup(ab)}{\sup(b)}\right) = \frac{1}{2}\left(\frac{3}{4} + \frac{3}{5}\right) = 0.675$

Tính chất 2. Xét thứ tự toàn phần $x_1 \prec x_1 \prec \cdots \prec x_k \prec x_{k+1}$ của các mục trong cơ sở dữ liệu D, khi đó, giá trị kulc thỏa tính chất *Downward closure* là $kulc(x_1, \dots, x_{k+1}) \leq kulc(x_1, \dots, x_k)$ [11].

Bảng IV. Cơ sở dữ liệu giao dịch sau khi loại bỏ f và sắp xếp theo thứ tự toàn phần

TID	**Giao dịch (T)**	**Số lượng (Q)**	**Độ hữu ích (U)**	**Độ hữu ích dương (PTU)**
T_1	a, b, d, c	3, 1, 2, 4	9, -1, -4, 16	25
T_2	a, b, c	3, 3, 2	9, -3, 8	17
T_3	a, d, e, c	4, 2, 2, 1	12, -4, 6, 4	22
T_4	b, e, c	2, 4, 5	-2, 12, 20	32
T_5	b, d, e, c	6, 4, 6, 5	-6, -8, 18, 20	38
T_6	a, b, d, e, c	2, 5, 3, 1, 1	6, -5, -6, 3, 4	13
T_7	e, c	3, 2	9, 8	17
T_8	d, e, c	2, 4, 5	-4, 12, 20	32

Định nghĩa 6. Tập mục X được gọi là tập hữu ích cao tương quan (*Correlated High Utility Itemset - CHUI*) nếu $kulc(X) \geq minCor$ và $U(X) \geq minUtil$. Trong đó minUtil là giá trị ngưỡng độ hữu ích tối thiểu và minCor là giá trị ngưỡng tương quan tối thiểu [11].

$$CHUIs = \{X \subseteq I \mid U(X) \geq minUtil \wedge kulc(X) \geq minCor\}$$

Ví dụ, trong Bảng 4, $U(abc) = 43, kulc(abc) = 0.575$. Với $minUtil < 43$ và $minCor < 0.575$ thì tập mục $\{abc\}$ là một CHUI. Ngược lại, tập mục $\{abc\}$ không phải CHUI.

Định nghĩa 7. Xét tập mục $X \subseteq T_j$, tập tất cả các mục sau X trong T_j được định nghĩa là $Re(X, T_j) = \{x_i \in T_j \mid \forall y \in X, y \prec x_i\}$.

Ví dụ: Với dữ liệu trong Bảng 4, $Re(ad, T_1) = \{c\}$, $Re(ad, T_3) = Re(ad, T_6) = \{e, c\}$

Định nghĩa 8. Độ hữu ích dương còn lại sau tập mục X trong giao dịch T_j, ký hiệu là $PRU(X, T_j)$, là tổng độ hữu ích của tất cả các mục có độ hữu ích dương trong tập $Re(X, T_j)$ [9]:

$$PRU(X, T_j) = \sum_{x_i \in Re(X,T_j) \wedge U(x_i,T_j) > 0} U(x_i, T_j)$$

Ví dụ: Với dữ liệu trong Bảng 4, $PRU(ab, T_6) = U(e, T_6) + U(c, T_6) = 3 + 4 = 7$. Trong đó $U(d, T_6)$ sẽ không tính vào PRU vì có độ hữu ích âm.

IV. THUẬT TOÁN CHN

A. Cấu trúc PNU-List

PNU-List được cấu trúc lại từ cấu trúc dữ liệu Utility-List [5,9]. Mỗi tập mục được biểu diễn bởi một PNU-List tương ứng. Cấu trúc *PNU-List* của một tập mục X lưu trữ tập các bộ dữ liệu, mỗi bộ chứa bốn thành phần là $(Tid, Putil, Nutil, Prutil)$. Trong đó, Tid là số thứ tự của giao dịch có chứa X, $Putil$ và $Nutil$ là độ hữu ích dương và độ hữu ích âm của tập mục X trong giao dịch Tid, $Prutil$ là tổng độ hữu ích dương của các mục sau X trong giao dịch Tid.

PNU-List của tập mục một phần tử gồm năm mục được sắp xếp theo thứ tự toàn phần a $\prec$ b $\prec$ d $\prec$ e $\prec$ c. Trong đó, mục f đã bị loại do $PTWU(f) < minUtil$ (Tính chất 1). PNU-List của tập mục {a} gồm 4 bộ với Tid lần lượt có giá trị là 1, 2, 3, 6 nghĩa là tập mục {a} xuất hiện trong các giao dịch tương ứng T_1, T_2, T_3, T_6 trong cơ sở dữ liệu D (Bảng 4). Xét bộ thứ nhất với Tid=1, ta có độ hữu ích dương $Putil = U(a, T_1) = 9$, độ hữu ích âm $Nutil = 0$ (vì tập mục $\{a\}$ chỉ có 1 phần tử là a có giá trị độ hữu ích dương là 9, không có phần tử nào khác có giá trị độ hữu ích âm); độ hữu ích dương của các mục sau $\{a\}$ là $Prutil = 16$ (vì các mục sau $\{a\}$ trong giao dịch T_1 gồm {b, d, c} nhưng chỉ c có độ hữu ích dương là 16 còn b và d có độ hữu ích âm). Tương tự cho các bộ còn lại và các PNU-List khác.

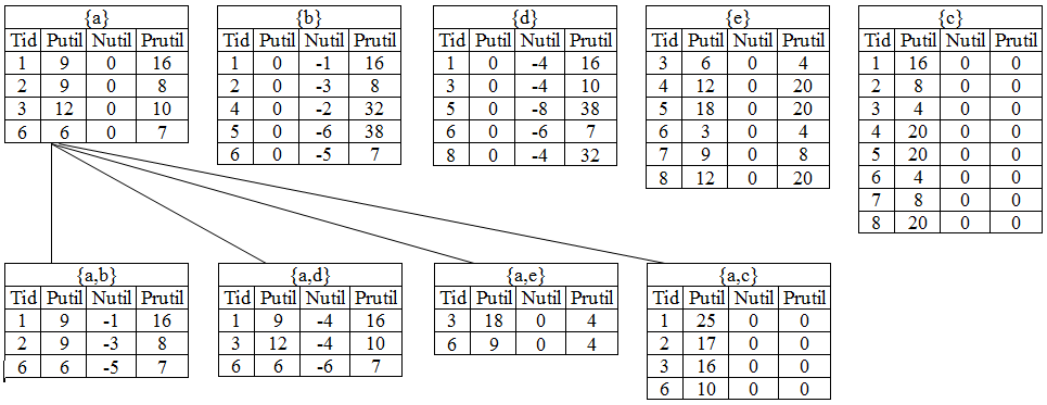
B. Các chiến lược tỉa

Chiến lược tỉa được áp dụng trong quá trình khai thác tập $CHUIs$ nhằm thu hẹp không gian tìm kiếm đồng thời tăng hiệu suất thực thi của thuật toán, đặc biệt là thời gian thực hiện và dung lượng bộ nhớ sử dụng. Các chiến lược tỉa áp dụng trong bài báo này gồm: U-Prune, Kulc-Prune, LA-Prune và EUCS-Prune [5,6,10,11].

Chiến lược 1 (U-Prune): Nếu tổng độ hữu ích dương của tập mục X và tổng độ hữu ích dương của các mục sau X nhỏ hơn $minUtil$ thì mọi tập mở rộng từ X đều không phải là $CHUI$. Nghĩa là nếu $PU(X) + PRU(X) < minUtil$ thì $\forall Y \supseteq X, Y \notin CHUI$. Khi đó ngừng mở rộng với tập mục X.

Ví dụ: Xét tập mục $\{a, e\}$ với *PNU-List* tương ứng trong Hình 1, $U(ae) + RU(ae) = \sum(Putil + Prutil) = 27 + 8 = 35 < minUtil = 42$. Khi đó ngừng mở rộng với tập $\{a, e\}$.

Chiến lược 2 (LA-Prune): Xét 2 tập X và Y, nếu $PU(X) + PRU(X) - \sum_{X \subseteq T_j \in D \wedge Y \not\subset T_j} PU(X, T_j) + PRU(X, T_j) < minUtil$ thì tập mục XY không phải là $CHUI$ và mọi tập mở rộng từ XY cũng không phải là $CHUI$ (nghĩa là $\forall Z \supseteq XY$ thì Z không phải là $CHUI$).



Hình 1. PNU-List của các tập mục 1 phần tử và 2 phần tử

Chiến lược 3 (EUCS-Prune): Xét X, Y là hai tập mục 1 phần tử, Nếu $PTWU(X,Y) < minUtil$ thì tập mục XY không phải $CHUI$ và mọi tập mở rộng từ XY cũng không phải $CHUI$. Khi đó dừng mở rộng với itemset XY.

Chiến lược 4 (Kulc-Prune): Nếu $kulc(X) < minCor$ thì mọi tập mở rộng từ X không phải lài một $CHUI$. Giả sử rằng y là phần tử mở rộng từ tập X để có tập Xy, theo tính chất 2, ta có $kulc(Xy) < kulc(X) < minCor$. Do đó mọi tập mở rộng từ X không phải là $CHUI$.

C. Thuật toán CHN

Thuật toán chính CHN có dữ liệu đầu vào là cơ sở dữ liệu giao dịch D có lợi nhuận âm, kết quả thực hiện thuật toán là tập hữu cao có tương quan (CHUIs). Đầu tiên quét cơ sở dữ liệu D để tính $SUP(i), RTWU(i)$ và $U(i)$ cho mỗi mục $i \in I$ trình bày ở dòng 1. Nếu $RTWU(i) \geq minUtil$ thì đưa i vào tập I^* và loại bỏ các mục $i \notin I^*$ khỏi cơ sở dữ liệu D trình bày ở dòng 2. Dòng 3 sắp xếp các mục trong tập I^* tăng dần theo độ hỗ trợ SUP đồng thời sắp xếp các mục trong cơ sở dữ liệu D theo thứ tự của I^*. Dòng 4 quét cơ sở dữ liệu D để xây dựng danh sách *PNU-List* cho từng phần tử $i \notin I^*$, ký hiệu là ULs. Dòng 5 khởi tạo cấu trúc $EUCS$ và dòng 6 gọi thực hiện thuật toán 2 là $SearchCHUI$.

Thuật toán 1: (Thuật toán chính - CHN)

Vào: D: Cơ sở dữ liệu giao dịch có lợi nhuận âm, $minUtil$: Ngưỡng độ hữu ích tối thiểu, $minCor$: Ngưỡng tương quan tối thiểu.

Ra: Các tập mục độ hữu ích cao có tương quan ($CHUIs$)

1. Quét cơ sở dữ liệu D để tính $SUP(i), RTWU(i)$ và $U(i)$ cho mỗi mục i có trong I.
2. Tính $I^* = \{i \in I \mid RTWU(i) \geq minUtil\}$ và loại bỏ các mục $i \notin I^*$ khỏi cơ sở dữ liệu D.
3. Sắp xếp I^* tăng theo SUP, sắp xếp các mục trong D theo thứ tự của I^*.
4. Quét cơ sở dữ liệu D để tính *PNU-List* cho mỗi phần tử $i \notin I^*$ là ULs
5. Khởi tạo cấu trúc $EUCS$
6. $SearchCHUI(\emptyset, ULs, minUtil, minCor, EUCS)$

Thuật toán 2 ($SearchCHUI$) thực hiện đệ quy mở rộng không gian tìm kiếm để xác định một tập mục có phải là $CHUI$ hay không. Thuật toán có đầu vào gồm P: một *PNU-List* ở mức trên đóng vai trò là tiền tố, danh sách các *PNU-List* có tiền tố là P; ngưỡng độ hữu ích tối thiểu minUtil; ngưỡng tương quan tối thiểu minCor và cấu trúc $EUCS$. Đầu ra là các tập mục hữu ích cao có tương quan ($CHUIs$). Dòng 1 duyệt qua từng *PNU-List* X có trong danh sách *PNU-List* ULs. Dòng 2 kiểm tra điều kiện nếu $U(X) \geq minUtil$ và $kulc(X) \geq minCor$ X là một tập hữu ích cao tương quan $CHUI$. Dòng 5 kiểm tra điều kiện nếu $PU(X) + PRU(X) \geq minUtil$ (*chiến lược tỉa U-Prune*) và điều kiện $kulc(X) \geq minCor$ (*chiến lược tỉa Kulc-prune*) thì tạo danh sách các *PNU-List* mở rộng ($exULs$) từ *PNU-List* X từ dòng 6 đến 11. ngược lại ngừng mở rộng với X. Dòng 8 áp dụng chiến lược tỉa *EUCS-Prune*, nếu $EUCS(X,Y) \geq minUtil$ thì thực hiện thủ tục $PNU\text{Construct}(P,X,Y)$ để tạo *PNU-List* XY từ 2 *PNU-List* X và Y và thêm vào danh sách $exULs$, ngược lại không tạo *PNU-List* XY. Dòng 12 gọi đệ quy thủ tục $SearchCHUI$ để tiếp tục mở rộng không gian tìm kiếm.

Thuật toán 2: $SearchCHUI$

Vào: P: *PNU-List* với vai trò là tiền tố; ULs: Danh sách các *PNU-List* có tiền tô là *PNU-List* P, $minUtil$: Ngưỡng độ hữu ích tối thiểu, $minCor$: Ngưỡng tương quan tối thiểu, $EUCS$: Cấu trúc EUCS

Ra: Các tập mục độ hữu ích cao có tương quan ($CHUIs$).

1. **for** each $X \in ULs$ **do**
2. **if** $U(X) \geq minUtil \wedge kulc(X) \geq minCor$ **then**
3. $CHUIs \leftarrow X$
4. **end if**
5. **if** $(PU(X) + PRU(X) \geq minUtil \wedge kulc(X) \geq minCor)$ //U-Prune và Kulc-Prune
6. $exULs = \emptyset$ //Khởi tạo danh sách *PNU-List* mở rộng từ X
7. **for** each Y after X in ULs **do**
8. **if** $EUCS(X,Y) \geq minUtil$ **then** //Áp dụng chiến lược tỉa EUCP
9. $exULs \leftarrow PNU\text{Construct}(P,X,Y)$;
10. **end if**
11. **end for**
12. $SearchCHUI(X, exULs, minUtil, minCor, EUCS)$; //gọi đệ quy thuật toán.
13. **end if**
14. **end for**

Thuật toán 3 ($PNUConstruct$) thực hiện kết hợp 2 *PNU-List* Px và Py thành *PNU-List* Pxy. Dòng 1 khởi tạo giá trị ban đầu cho ULA là tổng độ hữu ích dương $PU(P)$ và độ hữu ích còn lại sau P là $PRU(P)$. Dòng 2, 3 duyệt qua từng phần tử $ex \in Px$ và tìm phần tử $ey \in Py$ sao cho $ex.tid = ey.tid$. Nếu tìm thấy thì tạo phần tử exy kết hợp từ ex và ey trong các trường hợp xem xét tại dòng 4. Nếu *PNU-List* tiền tố $P \neq \emptyset$ (trường hợp 1 dòng 4), nghĩa là *PNU-List* Pxy đang xây dựng có từ 3 mục trở lên, ngược lại (trường hợp 2 dòng 7) thì Pxy là *PNU-List*

đang xây dựng chỉ có 2 mục. Dòng 6 tạo phần tử exy ứng với trường hợp 1, dòng 8 tạo phần tử exy ứng với trường hợp 2, dòng 10 thêm exy vào *PNU-List* Pxy. Dòng 12 xét điều kiện nếu không tồn tại $ey \in Py$ mà $ex.tid = ey.tid$ thì áp dụng chiến lược tỉa *LA-Prune* từ dòng 13 đến 15 để loại bỏ sớm những tập mục không phải là $CHUI$. Dòng 18 trả về kết quả là *PNU-List* Pxy.

Thuật toán 3: ($PNUConstruct$)

Vào: P: *PNU-List* với vai trò là tiền tố; Px, Py: Hai *PNU-List* cần kết hợp; $minUtil$: Ngưỡng độ hữu ích tối thiểu.

Ra: Pxy: *PNU-List* sau khi kết hợp Px và Py.

1. $Set\ ULA = PU(P) + PRU(P)$;
2. **for** each element $ex \in Px$ **then**
3. **if** $\exists\ ey \in Py \wedge ex.tid = ey.tid$ **then**
4. **if** $P \neq \emptyset$ **then**
5. Tìm $e \in P$ sao cho $e.tid = ex.tid$;
6. $exy =< ex.tid,\ ex.Putil + ey.Putil - e.Putil, ex.Nutil + ey.Nutil - e.Nutil\ ,\ ey.Prutil >$;
7. **else**
8. $exy =< ex.tid,\ ex.Putil + ey.Putil, ex.Nutil + ey.Nutil,\ ey.Prutil >$;
9. **end if**
10. $Pxy \leftarrow exy$;
11. **else**
12. $ULA = ULA - ex.Putil - ex.Prutil$;
13. **if** $ULA < minUtil$ **then** // áp dụng chiến lược tỉa LA-Prune
14. return null;
15. **end if**
16. **end if**
17. **end for**
18. return Pxy;

V. THỰC NGHIỆM

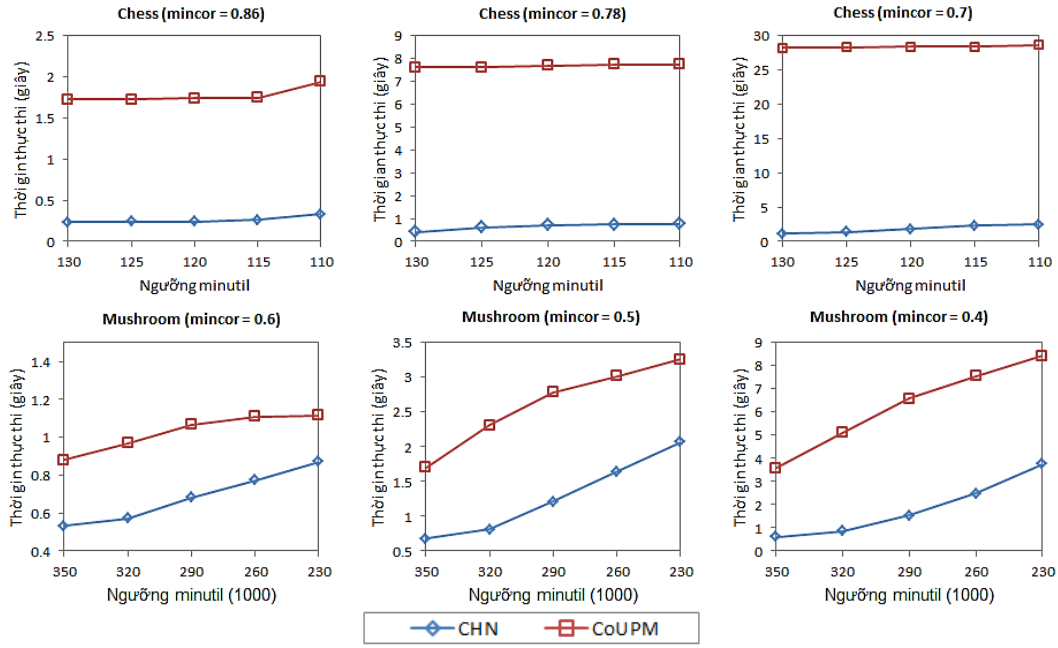
Thuật toán *CHN* được cài đặt bằng ngôn ngữ lập trình Java, chạy thử nghiệm trên máy tính Dell Precision Tower 3620, Intel Core i7-7800X CPU @3.5GHz, bộ nhớ RAM 32GB trên hệ điều hành Windows 10. Các cơ sở dữ liệu thử nghiệm được tải từ thư viện SPMF [20] là các cơ sở dữ liệu giao dịch có lợi nhuận âm gồm Chess, Mushroom, Pumsb, Retail và Kosarak. Chi tiết các cơ sở dữ liệu trình bày trong Bảng 5. Thực nghiệm của thuật toán *CHN* được so sánh với thuật toán mới nhất cùng khai thác tập *CHUI* là *CoUPM* [12]. Kết quả thực nghiệm được đánh giá dựa trên thời gian thực thi và dung lượng bộ nhớ sử dụng.

Bảng V. Đặc điểm các cơ sở dữ liệu thực nghiệm

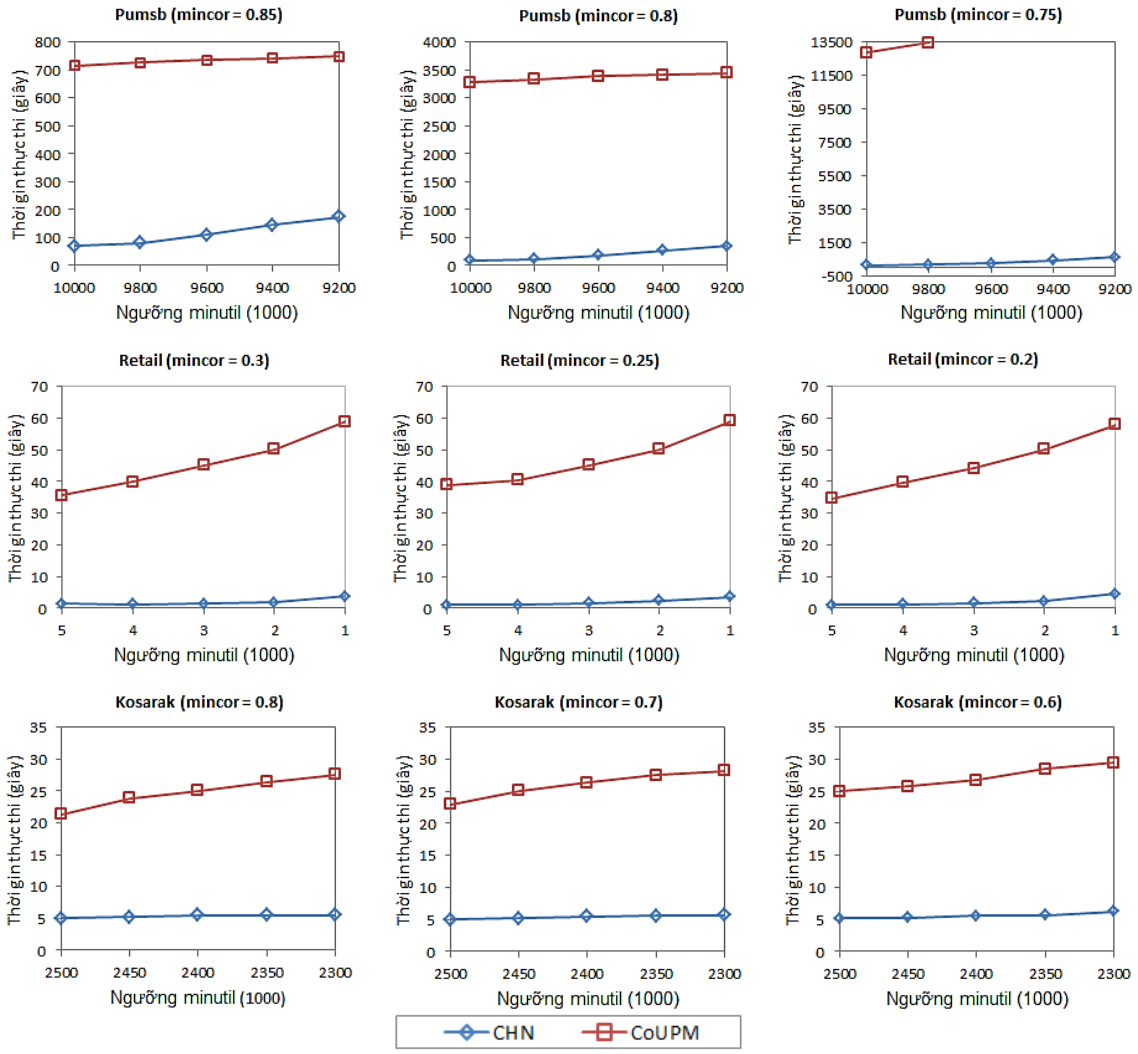
Cơ sở dữ liệu	Số lượng giao dịch	Số lượng item (I)	Độ dài trung bình (A)	Độ dày (A/I) %
Chess	3,196	75	37	49.3333
Mushroom	8,124	119	23	19.3277
Pumsb	49,046	2113	74	3.5021
Retail	88,162	16,470	10.3	0.0625
Kosarak	990,002	41,270	8.1	0.0196

Hình 2, 3 trình bày kết quả thực nghiệm so sánh giữa thuật toán CHN và thuật toán CoUPM về thời gian thực thi và bộ nhớ sử dụng. Kết quả thực nghiệm cho thấy thuật toán CHN có thời gian thực thi hiệu quả hơn thuật toán CoUPM trên tất cả các cơ sở dữ liệu từ rất dày như Chess đến cơ sở dữ liệu rất thưa như kosarak. Tuy nhiên bộ nhớ sử dụng của thuật toán CHN chỉ có hiệu quả hơn thuật toán CoUPM ở cơ sở dữ liệu Pumsb trên tất cả các ngưỡng và các ngưỡng tương quan lớn.

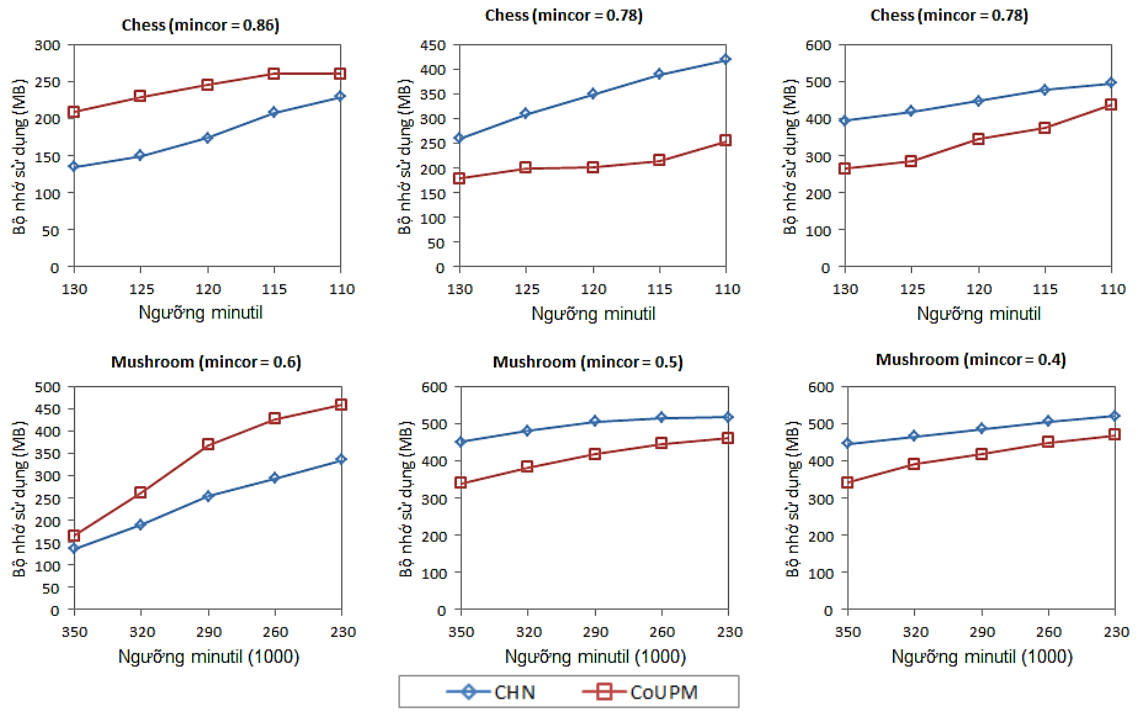
Hình 2 trình bày kết quả thực nghiệm của hai thuật toán CHN và CoUPM trên cơ sở dữ liệu rất dày Chess và dày trung bình Mushroom. Với cơ sở dữ liệu Chess, tại ngưỡng minCor=0.86 thời gian thực hiện của thuật toán CHN nhanh hơn thuật toán CoUPM từ 5 đến 7 lần. Đặc biệt, khi giảm ngưỡng minCor còn 0.7 thì thời gian thực hiện của CHN càng hiệu quả và có thời gian thực hiện ít hơn thuật toán CoUPM lên đến 25 lần tại ngưỡng minUtil=130,000. Với cơ sở dữ liệu Mushroom, thời gian thực hiện của thuật toán CHN tốt hơn thuật toán CoUPM ở tất cả các ngưỡng minCor và minUtil. Kết quả này chứng tỏ rằng các chiến lược tỉa và cấu trúc dữ liệu sử dụng trong thuật toán CHN là phù hợp khi khai thác tập CHUI trên cơ sở dữ liệu có lợi nhuận âm.



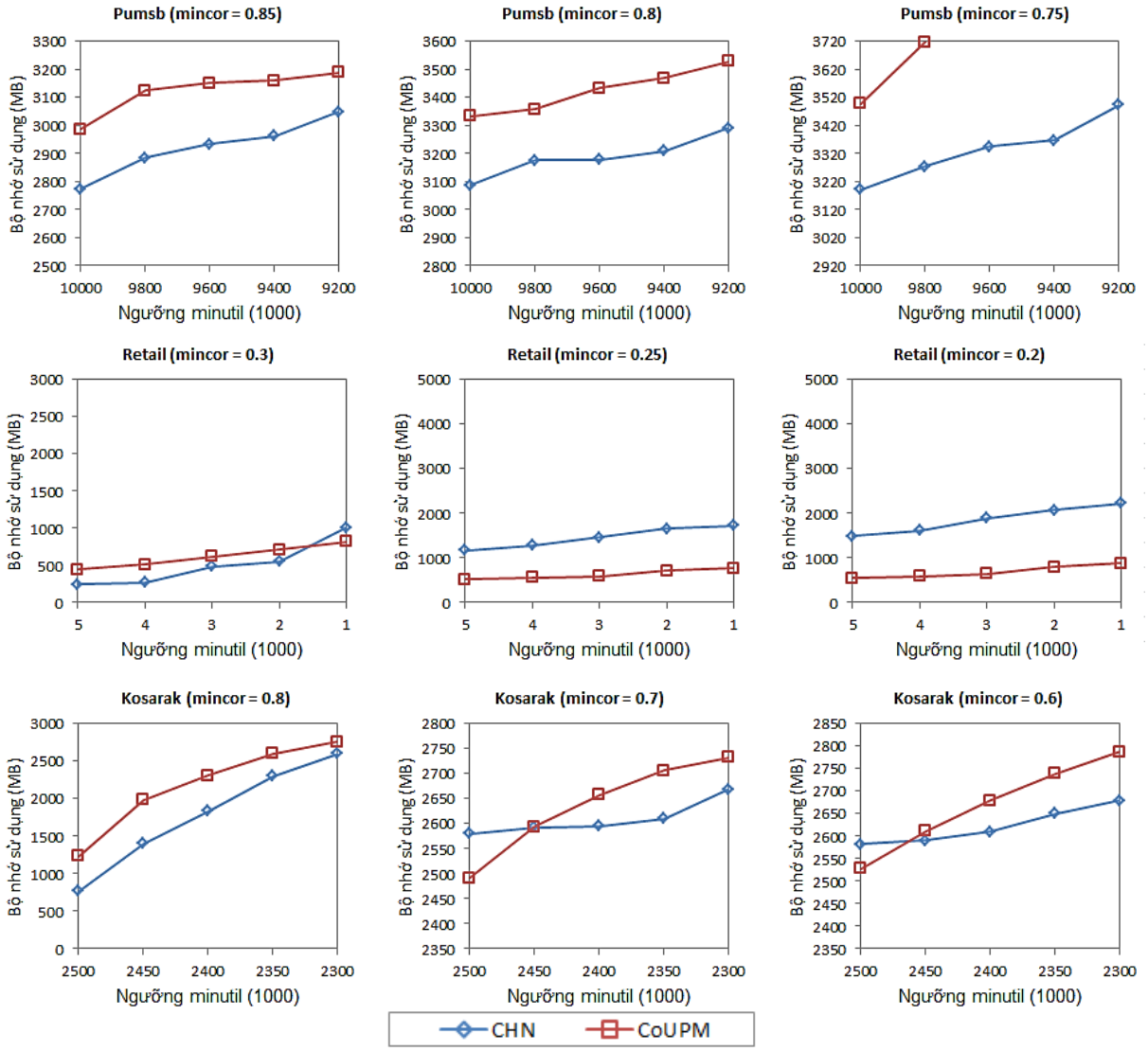
Hình 2. So sánh thời gian thực thi trên cơ sở dữ liệu dày



Hình 3. So sánh thời gian thực thi trên cơ sở dữ liệu thưa



Hình 4. So sánh bộ nhớ sử dụng trên cơ sở dữ liệu dày



Hình 5. So sánh bộ nhớ sử dụng trên cơ sở dữ liệu thưa

Hình 3 trình bày kết quả thực nghiệm trên cơ sở dữ liệu từ thưa như Pumsb cho đến rất thưa như Kosarak. Kết quả thực nghiệm cho thấy thuật toán CHN có thời gian thực hiện thấp hơn thuật toán CoUPM trên cả 3 cơ sở dữ liệu thử nghiệm. Với cơ sở dữ liệu lớn, có độ dài trung bình của giao dịch cao nhất như cơ sở dữ liệu Pumsb thì thời gian thực hiện của thuật toán CHN nhanh hơn thuật toán CoUPM từ 5 đến 10 lần tại ngưỡng minCor=0.85, từ 10 đến 30 lần tại ngưỡng minCor=0.8. Đặc biệt với ngưỡng minCor=0.75 thì thuật toán CHN cho ra kết quả ở tất cả các ngưỡng minUtil còn thuật toán CoUPM chỉ có kết quả ở 2 ngưỡng minUtil là 10,000,000 và 9,800,000, các ngưỡng còn lại không cho kết quả.

Với cơ sở dữ liệu Retail, biểu đồ thực nghiệm (Hình 3) cho thấy thuật toán CHN có tốc độ xử lý nhanh hơn hơn thuật toán CoUPM khoảng 25 lần tại ngưỡng minCor=0.3 và minUtil=5000. Tại các ngưỡng khác, thuật toán CHN vẫn chứng tỏ được tính hiệu quả hơn thuật toán CoUPM. Với cơ sở dữ liệu rất thưa như Kosarak, thuật toán CHN có thời gian thực thi ổn định trung bình khoảng 5 giây ở tất cả các ngưỡng minCor và minUtil. Trong khi thuật toán CoUPM có thời gian thực hiện cao hơn, trung bình khoảng 25 giây cho tất cả các ngưỡng.

Hình 4,5 so sánh bộ nhớ sử dụng của hai thuật toán CHN và CoUPM trên hai nhóm cơ sở dữ liệu dày và thưa. Kết quả thực nghiệm cho thấy thuật toán CHN có dung lượng bộ nhớ sử dụng thấp hơn thuật toán CoUPM ở hầu hết các cơ sở dữ liệu thực nghiệm tại các ngưỡng minCor cao. Khi ngưỡng minCor giảm xuống thấp hơn thì thuật toán CHN sử dụng bộ nhớ nhiều hơn thuật toán CoUPM ở hầu hết các cơ sở dữ liệu, ngoại trừ cơ sở dữ liệu lớn Pumsb. Kết quả này cho thấy rằng việc sử dụng các chiến lược tỉa và cấu trúc dữ liệu lưu trữ có ảnh hưởng đến bộ nhớ sử dụng của thuật toán CHN. Thật vậy, Với cấu trúc dữ liệu *PNU-List* sử dụng trong thuật toán CHN, mỗi bộ của một *PNU-List* gồm 4 giá trị là <Tid, Putil, Nutil, Prutil> trong khi cấu trúc dữ liệu Utility-List trong thuật toán CoUPM thì mỗi bộ chỉ gồm 3 giá trị <Tid, Iutil, Rutil>. Ngoài ra, thuật toán CHN áp dụng chiến lược tỉa EUCS để lưu ma trận các giá trị RTWU của 2 phần tử dẫn đến tốn kém bộ nhớ trong quá trình thực thi.

VI. KẾT LUẬN

Bài báo này đề xuất thuật toán *CHN* khai thác tập hữu ích cao tương quan (*CHUIs*) trên cơ sở dữ liệu có lợi nhuận âm. Thuật toán sử dụng cấu trúc *PNU-List* biểu diễn tách biệt các giá trị lợi nhuận âm và dương để khai

thác đầy đủ tập $CHUIs$ trên các cơ sở dữ liệu có lợi nhuận âm. Ngoài ra, thuật toán còn áp áp dụng nhiều chiến lược tỉa hiệu quả để cắt giảm không gian tìm kiếm như: *U-Prune, Kulc-Prune, LA-Prune* và *EUCS-Prune*. Kết quả thực nghiệm cho thấy thuật toán CHN có thời gian thực hiện nhanh hơn thuật toán CoUPM trên tất cả các cơ sở dữ liệu có lợi nhuận âm được thử nghiệm. Bộ nhớ sử dụng của thuật toán CHN tốt hơn thuật toán CoUPM ở các ngưỡng minCor cao, tuy nhiên, với các ngưỡng minCor thì thuật toán CHN có dung lượng bộ nhớ sử dụng nhiều hơn.

Hướng phát triển tiếp theo là cải tiến cấu trúc dữ liệu lưu trữ và chiến lược tỉa để tăng hiệu suất thực thi của thuật toán trên các cơ sở dữ liệu có lợi nhuận âm, khai thác $CHUIs$ trên các dạng cơ sở dữ liệu khác như cơ sở dữ liệu động và cơ sở dữ liệu tăng trưởng.

TÀI LIỆU THAM KHẢO

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

[2] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," *In Proc. 20th Int. Conf. Very Large Data Bases (VLDB)*, pp. 487-499, 1994.

[3] Y. Liu, W. K. Liao, and A. Choudhary, "A two-phase algorithm for fast discovery of high utility itemsets," *In Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 689-695, 2005.

[4] V. S. Tseng, C. W. Wu, B. E. Shie, and P. S. Yu, "UP-Growth: an efficient algorithm for high utility itemset mining," *In Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 253-262, 2010.

[5] M. Liu and J. Qu, "Mining high utility itemsets without candidate generation," *In Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, pp. 55-64, 2012.

[6] P. Fournier-Viger, C. W. Wu, S. Zida, and V. S. Tseng, "FHM: Faster high-utility itemset mining using estimated utility co-occurrence pruning," *International Symposium on Methodologies for Intelligent Systems*, vol. 8502, pp. 83-92, 2014.

[7] S. Zida, P. Fournier-Viger, J. C. W. Lin, C. W. Wu, and V. S. Tseng, "EFIM: a fast and memory efficient algorithm for high-utility itemset mining," *Knowledge and Information Systems*, vol. 51, no. 2, pp. 595-625, 2017.

[8] J. Liu, K. Wang, and B. C. Fung, "Direct discovery of high utility itemsets without candidate generation," *IEEE 12th international conference on data mining*, pp. 984-989, 2012.

[9] J. C. W. Lin, P. Fournier-Viger, and W. Gan, "FHN: An efficient algorithm for mining high-utility itemsets with negative unit profits," *Knowledge-Based Systems*, vol. 111, pp. 283-298, 2016.

[10] J. C. W. Lin, W. Gan, P. Fournier-Viger, T. P. Hong, and H. C. Chao, "FDHUP: Fast algorithm for mining discriminative high utility patterns," *Knowledge and Information Systems*, vol. 51, pp. 873-909, 2017.

[11] W. Gan, J. C. W. Lin, P. Fournier-Viger, H. C. Chao, and H. Fujita, "Extracting non-redundant correlated purchase behaviors by utility measure," *Knowledge-Based Systems*, vol. 143, pp. 30-41, 2018.

[12] W. Gan, J. C. W. Lin, H. C. Chao, H. Fujita, and S. Y. Philip, "Correlated utility-based pattern mining," *Information Sciences*, vol. 504, pp. 470-486, 2019.

[13] A. Erwin, R. P. Gopalan, and N. R. Achuthan, "CTU-Mine: An efficient high utility itemset mining algorithm using the pattern growth approach," *IEEE International Conference on Computer and Information Technology*, pp. 71-76, 2007.

[14] B. Le, H. Nguyen, T. A. Cao, and B. Vo, "A novel algorithm for mining high utility itemsets," *First Asian Conference on Intelligent Information and Database Systems*, pp. 13-17, 2009.

[15] V. S. Tseng, B. E. Shie, C. W. Wu, and S. Y. Philip, "Efficient algorithms for mining high utility itemsets from transactional databases," *IEEE transactions on knowledge and data engineering*, vol. 25, pp. 1772-1786, 2012.

[16] S. Krishnamoorthy, "HMiner: Efficiently mining high utility itemsets," *Expert Systems with Applications*, vol. 90, pp. 168-183, 2017.

[17] K. Singh, A. Kumar, S. S. Singh, H. K. Shakya, and B. Biswas, "EHNL: An efficient algorithm for mining high utility itemsets with negative utility value and length constraints," *Information Sciences*, vol. 484, pp. 44-70, 2019.

[18] E. R. Omiecinski, "Alternative interest measures for mining associations in databases," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, pp. 57-69, 2003.

[19] P. Fournier-Viger, Y. Zhang, J. C. W. Lin, D. T. Dinh, and H. B. Le, "Mining correlated high-utility itemsets using various measures," *Logic Journal of the IGPL*, vol. 28, no. 1, pp. 19-32, 2020.

[20] P. Fournier-Viger, A. Gomariz, A. Soltani, and H. Lam. (2014) An Open-Source Data Mining Library. [Online]. http://www.philippe-fournier-viger.com

MINING CORRELATED HIGH UTILITY ITEMSETS ON TRANSACTION DATABASE WITH NEGATIVE PROFITS

Abstract: Correlated High Utility Itemset mining on transaction databases have been extensively studied in order to discover user buying behavior. As a result, business managers can adjust sales strategies accordingly to increase profits. The previous correlated high utility itemset mining approaches are mostly performed on transaction databases that have positive profit with little regard for negative profit. In fact, businesses often reduce profit margin of perennial inventories to stimulate purchasers. Profit can even be reduced to negative numbers so that all inventory can be consumed. In this paper, we propose the CHN algorithm to mine effectively correlated high utility itemset on the transaction database with negative profits. The experimental results show that the CHN algorithm has effective performance on all five databases as Chess, Mushroom, Pumsb, Retail, and Kosarak.

Keywords: high utility itemset, correlation, data mining, transaction database, correlated high utility itemset.

Nguyen Văn Lễ, nhận học vị Thạc sỹ năm 2011, chuyên ngành Truyền dữ liệu & Mạng máy tính tại Học viện Công nghệ Bưu chính Viễn Thông TPHCM. Hiện công tác tại khoa Công nghệ thông tin, trường Đại học Công nghiệp Thực phẩm TPHCM. Hướng nghiên cứu: Khai thác luật kết hợp, tập hữu ích cao trên cơ sở dữ liệu giao dịch, phân lớp văn bản.

Email: lenv@hufi.edu.vn

Nguyễn Thị Thanh Thủy, tốt nghiệp khoa Công nghệ thông tin, Trường Đại học Khoa học tự nhiên - Đại học quốc gia TP.HCM năm 2003 và nhận bằng thạc sỹ ngành Quản trị kinh doanh, Trường Đại học Kinh tế TP.HCM năm 2013. Hiện tại, đang là giảng viên khoa Công nghệ thông tin, Trường Đại học Công nghiệp Thực phẩm TP.HCM. Các hướng nghiên cứu gồm: khai thác luật kết hợp, khai thác tập hữu ích cao trong khai phá dữ liệu.

Email: thuyntt@hufi.edu.vn

Mạnh Thiên Lý, tốt nghiệp đại học ngành Công nghệ Thông tin tại Đại học Vinh năm 2006, nhận bằng Thạc sỹ Hệ thống Thông tin tại Học viện Kỹ thuật Quân sự Hà Nội năm 2010. Hiện tại là giảng viên trường Đại học Công nghiệp Thực phẩm TP.HCM. Một số hướng nghiên cứu chính: khai thác luật kết hợp, khai thác tập hữu ích cao trong khai phá dữ liệu.

Email: lymt@hufi.edu.vn