RESEARCH ARTICLE

PHƯƠNG PHÁP KHAI THÁC TẬP HỮU ÍCH CAO TRÊN DỮ LIỆU GIAO DỊCH LUỒNG DỰA TRÊN CÂY HUSTREE

Trần Minh Thái , Trần Anh Duy, Phạm Đức Thành, Lê Thị Minh Nguyện, Nguyễn Thanh Trung

Khoa Công nghệ thông tin, Trường Đại học Ngoại ngữ -Tin học TP.HCM

thaitm@huflit.edu.vn, duyta@huflit.edu.vn, nguyenltm@huflit.edu.vn,

trungnt2@huflit.edu.vn, thanhpd@huflit.edu.vn

TÓM TẮT— Khai thác mẫu hữu ích cao trên luồng dữ liệu giao dịch là một bài toán quan trọng trong lĩnh vực khai thác dữ liệu. Việc khai thác này giúp phát hiện những tập sản phẩm có lợi nhuận cao trong cơ sở dữ liệu giao tác. Bên cạnh đó, khi số lượng hóa đơn được cập nhật liên tục sẽ tạo ra những mẫu hữu ích cao mới đồng thời thay đổi độ hữu ích của một số tập mẫu đã có trước đó. Việc cập nhật kịp thời thông tin của sự thay đổi này đóng vai trò quan trọng trong quá trình đưa ra các quyết định hiệu quả trong việc kinh doanh. Tuy nhiên, số lượng những phương pháp khai thác trên tập dữ liệu luồng giao dịch đang còn hạn chế nhất định. Nội dung bài báo này tập trung vào nghiên cứu và đề xuất một phương pháp khai thác dữ liệu luồng giao dịch mới dựa trên cấu trúc cây HUS-Tree đã có trước đây. Kết quả thực nghiệm chứng minh phương pháp khai thác mới có thời gian thực thi hiệu quả hơn giải pháp trước đó.

Từ khóa— Dữ liệu luồng giao dịch, khai thác dữ liệu, tập hữu ích cao, mẫu hữu ích cao.

I. GIỚI THIỆU

Khai thác dữ liệu luồng giao dịch [1], [2], [3], [4], [5] là một bài toán quan trọng trong lĩnh vực khai thác dữ liệu và là một phần trong chủ đề khai thác luật kết hợp [6]. Thông thường, khai thác luật kết hợp được chia ra thành hai bước. Đầu tiên là khai thác mẫu phổ biến hay còn gọi là tập sản phẩm phổ biến [7]. Kế đến là phát sinh luật từ các mẫu phổ biến này. Tuy nhiên, trong thực tế có những tập mẫu không phổ biến nhưng lại mang lại giá trị cao hơn các tập phổ biến. Ví dụ như việc người dùng mua ít sản phẩm nhưng giá trị của mỗi sản phẩm được mua lại lớn hơn những sản phẩm còn lại. Do đó, khai thác mẫu phổ biến có nhiều hạn chế khi chưa khảo sát đến số lượng sản phẩm (giá trị hữu ích nội) cũng như giá trị của các sản phẩm (giá trị hữu ích ngoại) được mua. Khai thác mẫu hữu ích cao được đề xuất giúp khắc phục những hạn chế của khai thác mẫu phổ biến, nghĩa là các giá trị hữu ích nội và hữu ích ngoại của từng sản phẩm sẽ được xem xét trong quá trình khai thác để xác định mẫu hữu ích cao có trong cơ sở dữ liệu (CSDL) giao dịch.

Do mỗi sản phẩm trong mỗi giao dịch đều có giá trị hữu ích khác nhau nên bài toán khai thác mẫu phổ biến không đáp ứng được nhu cầu khai thác trên các tập dữ liệu này. Từ đó khai thác tập hữu ích cao (High Utility Itemset - HUI) được áp dụng để tìm ra các mẫu hữu ích cao có trong CSDL. Các thuật toán khai thác mẫu hữu ích cao được đề xuất có thể kể tới là HMiner [8], EFIM [9], FHM [10] và D^{2}HUP [11].

Tuy nhiên, các thuật toán khai thác trên chỉ áp dụng cho các bộ dữ liệu tĩnh, nghĩa là không có sự thay đổi theo thời gian. Trong thực tế, một mẫu hay tập sản phẩm có thể là hữu ích cao trong khoảng thời gian này nhưng lại có thể trở thành mẫu hữu ích thấp trong khoảng thời gian khác. Chẳng hạn như những sản phẩm liên quan đến du lịch biển có thể là mẫu hữu ích cao vào mùa hè nhưng lại ít được mua vào mùa đông. Khai thác các mẫu hữu ích cao theo thời gian hay còn gọi là khai thác luồng dữ liệu giao dịch được đề xuất để giải quyết vấn đề này. Các phương pháp khai thác dữ liệu luồng giao dịch tiêu biểu có thể kể đến như MHUI-BIT [12], MHUI-TID [12], HUPMS [13].

Nội dung còn lại của bài báo được trình bày theo cấu trúc sau: Phần II trình bày các công trình nghiên cứu liên quan. Phần III trình bày các khái niệm và định nghĩa của bài toán. Phần mô tả xử lý và thuật toán đề xuất được trình bày trong phần IV. Cuối cùng, phần V, VI lần lượt trình bày thực nghiệm và kết luận.

II. CÁC NGHIÊN CỨU LIÊN QUAN

Trong những năm gần đây, nhiều lĩnh vực con trong khai thác dữ liệu như phân loại (classification) [14] và chuỗi thời gian (time-series) [14] cũng tiếp cận xử lý dữ liệu luồng vì hầu hết các phương pháp khai thác truyền thống không phân tích được nhiều về dữ liệu nhạy cảm theo thời gian. Dữ liệu luồng giao dịch là dữ liệu liên tục và có sự sắp xếp thứ tự của các sản phẩm theo thời gian. Do đặc tính liên tục cập nhật nên tính chất hữu ích của một hoặc một tập sản phẩm có thể bị thay đổi theo thời gian. Một tập sản phẩm có thể là hữu ích cao trong một khoảng thời gian này, nhưng cũng có thể trở nên không còn hữu ích trong một khoảng thời gian khác. Để giải quyết vấn đề khai thác trên tập dữ liệu thay đổi liên tục như vậy, bước đầu các thuật toán dựa trên cửa sổ trượt đã được đề xuất. Mặc dù phương pháp này đã giải quyết được yêu cầu của bài toán đặt ra. Tuy nhiên, hạn chế của chúng là phát sinh rất nhiều ứng viên gây tốn kém tài nguyên và xử lý.

Năm 2010, Ahmed [13] và cộng sự đã đề xuất cấu trúc cây gọi là HUS-Tree dùng để lưu trữ thông tin của luồng dữ liệu. Theo đó, mỗi khi cửa sổ trượt thay đổi thì sẽ cập nhật lại giá trị của các nút có trong cây. Thuật toán HUPMS (HUP Mining over Stream data) được đề xuất để khai thác mẫu hữu ích cao dựa trên cây HUS-Tree. Thuật toán HUPMS sẽ phát sinh cây tiền tố của các mẫu hữu ích có độ dài một (1-itemset). Sau đó, thực hiện kiểm tra các mẫu ứng viên để xóa đi một số nút trong cây tiền tố và tạo ra cây điều kiện. Sau đó, các mẫu hữu ích cao sẽ được phát sinh từ cây điều kiện này. Điểm hạn chế của thuật toán này là phải phát sinh rất nhiều cây tiền tố và cây điều kiện mỗi khi tiến hành khai thác mẫu dẫn đến tiêu tốn rất nhiều thời gian. Cấu trúc HUS-Tree tương tự cũng được sử dụng trong đề xuất của nhóm tác giả trong công trình [16] nhằm khai thác mẫu hữu ích cao trên dữ liệu có cập nhật ngưỡng hữu ích tối thiểu.

Từ hạn chế trên, nội dung bài báo đề xuất phương pháp khai thác mẫu hữu ích cao mà không cần phải xây dựng cây tiền tố cũng như cây điều kiện nhưng vẫn tìm ra được các mẫu hữu ích cao có trong CSDL luồng giao dịch.

III. ĐỊNH NGHĨA BÀI TOÁN

Đặt $I = \{i_1, i_2, \dots, i_n\}$ là tập các sản phẩm (hay các sự kiện) và $D = \{T_1, T_2, \dots, T_m\}$ là CSDL giao dịch với mỗi $T_j \in D$ là một tập con của tập *I*.

Định nghĩa 1 [1]: Độ hữu ích nội (internal utility) của mỗi sản phẩm ký hiệu là $iu(i,T_j)$ là số lần xuất hiện của sản phẩm *i* trong giao dịch T_j. Ví dụ trong Bảng 1, sự kiện *a* xuất hiện 2 lần trong giao dịch T_1, nên độ hữu ích nội của sự kiện *a* trong giao dịch T_1 ký hiệu là $iu(a,T_1)=2$. Độ hữu ích ngoại (external utility) của mỗi sản phẩm (item), ký hiệu là $eu(i)$, là giá trị của sản phẩm *i*. Ví dụ trong Bảng 2, độ hữu ích ngoại của sản phẩm *a* là $eu(a) = 5$. Độ hữu ích của sản phẩm *i* trong trong giao dịch T_j ký hiệu là $u(i,T_j)$ được tính bằng công thức (1). Ví dụ: $u(a, T_1) = 2 \times 5 = 10$ với dữ liệu từ Bảng 1 và Bảng 2.

$$u(i, T_j) = eu(i) \times iu(i, T_j) \tag{1}$$

Độ hữu ích của nhóm sản phẩm *X* (itemset *X*) trong giao dịch T_j, ký hiệu là $u(X,T_j)$ được tính bằng công thức (2). Với $X = \{i_1, i_2, \dots, i_k\}$ là tập gồm *k* sản phẩm, $X \subseteq T_j$ và $1 \le k \le n$. Ví dụ, nhóm 2-sản phẩm *ac* trong giao dịch T_1 có độ hữu ích là $u(ac, T_1) = 2 \times 5 + 3 \times 10 = 40$ với dữ liệu từ Bảng 1 và Bảng 2.

$$u(X, T_j) = \sum_{i \in X} u(i, T_j) \tag{2}$$

Độ hữu ích của nhóm sản phẩm *X* trong CSDL, ký hiệu là $u(X)$, được tính bằng công thức (3). Ví dụ, độ hữu ích của *ac* trong Bảng 1 là $u(ac) = u(X, T_1) + u(X, T_5) + u(X, T_6) + u(X, T_7) = 235$.

$$u(X) = \sum_{T_j \in D} u(X, T_j) \tag{3}$$

Độ hữu ích của giao dịch T_j, ký hiệu là $tu(T_j)$, là tổng giá trị hữu ích của các sản phẩm có trong giao dịch T_j và được tính bằng công thức (4). Ví dụ, độ hữu ích của giao dịch T_1 trong Bảng 1 là $tu(T_1) = u(a, T_1) + u(c, T_1) + u(d, T_1) = 2 \times 5 + 3 \times 10 + 4 \times 7 = 68$.

$$tu(T_j) = \sum_{i \in T_j} u(i, T_j) \tag{4}$$

Cho ngưỡng hữu ích tối thiểu δ là tỉ lệ phần trăm cho trước dựa trên tổng giá trị hữu ích của tất cả các giao dịch có trong CSDL. Xét ví dụ trong Bảng 1, tổng hữu ích của tất cả các giao dịch là 734. Nếu $\delta = 20\%$ thì giá trị hữu ích tối thiểu có thể được tính theo công thức (5). Giả sử $\delta = 20\%$ thì giá trị hữu ích tối tiểu là $minutil = 20\% \times 734 = 146.8$

$$minutil = \delta \times \sum_{T_j \in D} tu(T_j) \tag{5}$$

Một tập các sản phẩm *X* được gọi là mẫu hữu ích cao nếu $u(X) \ge minutil$. Bài toán tìm mẫu hữu ích cao được định nghĩa là tìm tất cả các tập *X* thỏa điều kiện $u(X) \ge minutil$.

Trong bài toán khai thác mẫu phổ biến, tính chất Apriori được dùng để loại bỏ những mẫu không phổ biến. Tính chất này chỉ ra rằng, tất cả các tập mở rộng của một sự kiện không phổ biến đều không phổ biến. Tuy nhiên, tính chất này không thể áp dụng cho bài toán khai thác mẫu hữu ích cao. Ví dụ, nếu *minutil* là 146.8, thì mẫu *a* trong Bảng 1 không phải là mẫu hữu ích cao vì $u(a)=115$. Tuy nhiên, tập mở rộng của *a* là *ac* lại là mẫu hữu ích cao vì $u(ac)=235$. Do đó, một độ đo được gọi là trọng số hữu ích đã được đề xuất để giải bài toán khai thác mẫu hữu ích cao.

Định nghĩa 2 [1]: Trọng số hữu ích của một tập các sản phẩm *X*, ký hiệu là *twu(X)*, được tính bằng tổng giá trị hữu ích của các giao dịch có chứa *X*, được định nghĩa bằng công thức (6). Ví dụ, trong Bảng 1 trọng số hữu ích của sản phẩm *d* được tính là *twu(d)*=68+43=111. Do vậy, tính chất Apriori có thể áp dụng được với độ đo này. Ví dụ, với *minutil*=146.8 thì *twu(d)<minutil*. Khi đó, mẫu mở rộng của *d* là *dc* cũng có trọng số hữu ích nhỏ hơn *minutil* hay *twu(dc) < minutil*.

$$twu(X) = \sum_{X \subseteq T_j \in D} tu(T_j) \tag{6}$$

X được gọi là mẫu có trọng số hữu ích cao nếu *twu(X)>minutil*. Dữ liệu luồng giao dịch (transaction stream) là loại dữ liệu phổ biến trong thực tế. Dữ liệu dạng này có thể thấy trong dữ liệu bán hàng, dữ liệu về các thao tác trên trang web, dữ liệu thời tiết, … Trong đó, số lượng dữ liệu là không giới hạn và có thể tăng trưởng theo thời gian. Một lô giao dịch (batch) gồm nhiều giao dịch (transaction). Một cửa sổ (window) bao gồm nhiều lô giao dịch. Hình 1 là một ví dụ minh hoạ về dữ liệu luồng giao dịch. Trong đó, CSDL được chia làm 4 lô có kích thước bằng nhau và 2 cửa sổ. Mỗi lô bao gồm 2 giao dịch. Mỗi cửa sổ gồm 3 lô. Cửa sổ W_1 bao gồm lô B_1, B_2 và B_3. Cửa sổ W_2 bao gồm lô B_2, B_3 và B_4.

Bảng 1. *Cơ sở dữ liệu giao dịch*

TID	a	b	c	d	e	tu
T_1	2	0	3	4	0	68
T_2	4	3	0	0	0	44
T_3	0	5	7	0	0	110
T_4	0	2	0	1	5	43
T_5	6	7	2	0	0	106
T_6	3	0	5	0	1	69
T_7	8	8	4	0	2	152
T_8	0	9	5	0	5	142

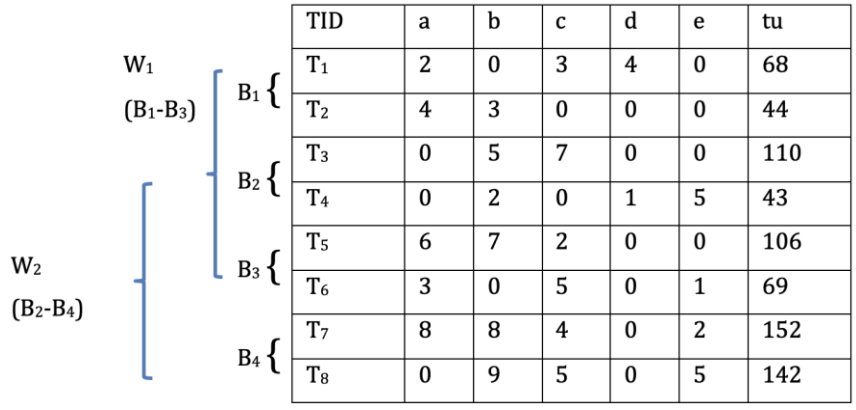
Bảng 2. *Độ hữu ích ngoại của sản phẩm*

item	eu(i)
a	5
b	8
c	10
d	7
e	4

Định nghĩa 3 [3]: Độ hữu ích của tập sản phẩm *X* trong một lô B_k được tính theo công thức (7).

$$u_{B_k}(X) = \sum_{T_j \in B_k} u(X, T_j) \tag{7}$$

Ví dụ, trong Hình 1 và dữ liệu Bảng 2, $u_{B_3}(ac) = u(ac, T_5) + u(ac, T_6) = 45 + 70 = 115$.



TID	a	b	c	d	e	tu
T_1	2	0	3	4	0	68
T_2	4	3	0	0	0	44
T_3	0	5	7	0	0	110
T_4	0	2	0	1	5	43
T_5	6	7	2	0	0	106
T_6	3	0	5	0	1	69
T_7	8	8	4	0	2	152
T_8	0	9	5	0	5	142

Hình 1. *Ví dụ về dữ liệu luồng giao dịch.*

Định nghĩa 4 [3]: Độ hữu ích của tập sản phẩm *X* trong một cửa sổ W_l được định nghĩa bằng công thức (8). Ví dụ, trong Hình 1 và dữ liệu Bảng 2, $u_{W_2}(ac) = u_{B_2}(ac) + u_{B_3}(ac) + u_{B_4}(ac) = 0 + 115 + 80 = 195$.

$$u_{W_l}(X) = \sum_{B_k \in W_l} u_{B_k}(X) \tag{8}$$

Ngưỡng hữu ích tối thiểu trong một cửa sổ W_l ký hiệu là δ_{W_l} là giá trị phần trăm cho trước dựa trên tổng giá trị hữu ích của tất cả các giao dịch có trong cửa sổ W_l. Trong Hình 1, tổng hữu ích của tất cả các giao dịch trong cửa sổ W_2 là 622. Nếu $\delta_{W_l} = 20\%$ thì độ hữu ích tối thiểu trong cửa sổ W_l có thể được tính theo công thức (9). Ví dụ, với $\delta_{W_2} = 20\%$ thì giá trị hữu ích tối tiểu là $minutil_{W_2} = 20\% \times 622 = 124.4$

$$minutil_{W_l} = \delta_{W_l} \times \sum_{T_j \in W_l} tu(T_j) \quad (9)$$

Một tập các sản phẩm X được gọi là mẫu hữu ích cao trong cửa sổ W_l nếu $u_{W_l}(X) \geq minutil_{W_l}$. Bài toán tìm mẫu hữu ích cao được định nghĩa là tìm tất cả các tập X thỏa điều kiện $u_{W_l}(X) \geq minutil_{W_l}$. Ví dụ, với $\delta_{W_2} = 20\%$ thì mẫu ac là mẫu hữu ích cao trong cửa sổ W_2 vì $u_{W_2}(ac) = 195$. Trọng số hữu ích của một tập các sản phẩm X trong cửa sổ W_l, ký hiệu là $twu_{W_l}(X)$, được tính bằng tổng giá trị hữu ích của các giao dịch trong cửa sổ W_l có chứa X. Ví dụ, $twu_{W_2}(ac) = twu_{B_2}(ac) + twu_{B_3}(ac) + twu_{B_4}(ac) = 106 + 69 + 152 = 327$. Mẫu X được gọi là mẫu có trọng số hữu ích cao trong cửa sổ W_l nếu $twu_{W_l}(X) \geq minutil_{W_l}$. Ví dụ, với $minutil_{W_2} = 124.4$ thì mẫu ac là mẫu có trọng số hữu ích cao vì $twu_{W_2}(ac) = 327$.

IV. THUẬT TOÁN KHAI THÁC MẪU TRÊN CÂY HUS-TREE

A. CẤU TRÚC CÂY HUS-TREE

Trong phần này, bài báo trình bày và mô tả về cấu trúc cây, gọi là HUS-Tree (High Utility Stream Tree), cho phép lưu trữ thông tin về các giao dịch trong CSDL. Cấu trúc dữ liệu này được đề xuất trong bài báo của tác giả Ahmed [13] năm 2010.

Bên trong cấu trúc cây HUS-Tree, mỗi nút sẽ lưu trữ thông tin định danh của một sản phẩm (hoặc sự kiện) kèm theo một danh sách các giá trị của các lô giao dịch, gồm nhiều giao dịch liên tiếp nhau, chứa sản phẩm đó. Bên cạnh đó, một liên kết được tạo ra giữa các nút con và nút cha để cho phép truy xuất dữ liệu giữa các nút khác nhau trong quá trình khai thác.

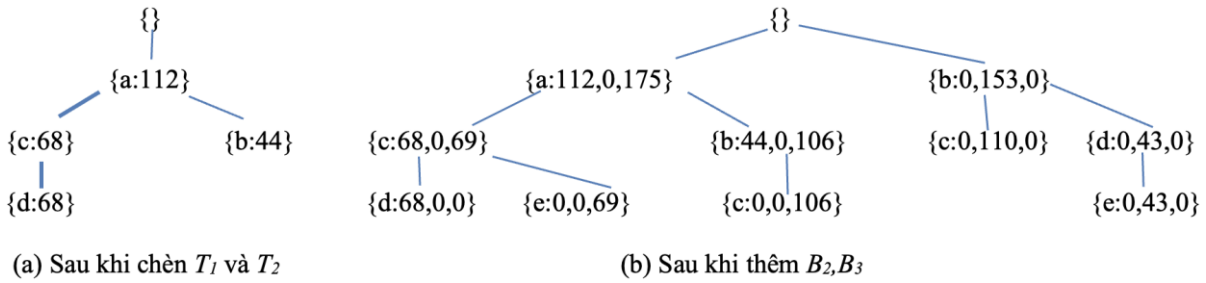
Dựa trên CSDL mẫu trong Bảng 1, chúng ta duyệt qua từng sản phẩm trong từng giao dịch theo thứ tự sắp xếp thứ tự để đưa từng sản phẩm vào cấu trúc cây. Đầu tiên, thao tác duyệt sẽ được thực hiện trên giao dịch T_1. Giao dịch T_1 có giá trị *tu* là 68 và chứa các sản phẩm $\{a, c, d\}$. Sản phẩm *a* sẽ được thêm vào cây bằng cách tạo ra nút chứa item-id là *a* và giá trị *twu* là 68. Ta có nút {a:68}. Sau đó, sản phẩm *c* được thêm vào như nút con của *a* với giá trị *twu* là 68. Ta có nút {c:68} là nút con của {a:68}. Sau đó, sản phẩm *d* được thêm vào như nút con của *c* với giá trị *twu* là 68. Ta có nút {d:68} là nút con của nút {c:68}. Tiếp theo giao dịch T_2 được duyệt để thêm các sản phẩm vào cây. Giao dịch T_2 có giá trị *tu* là 44 và gồm có 2 sản phẩm là {a,b}. Sản phẩm *a* trong giao dịch T_2 được thêm vào cây với giá trị *twu* là 44. Tuy nhiên, do trong cây đã có nút *a* rồi và giao dịch T_1và T_2 thuộc cùng một lô B_1 như Hình 1. Nên giá trị *twu* của sản phẩm *a* trong giao dịch T_2 sẽ được cộng dồn vào nút {a:68} thành {a:112}. Sản phẩm *b* được thêm vào như nút con của {a:112} với giá trị *twu* là 44. Hình 2(a) minh họa cây HUS-Tree sau khi đã duyệt qua giao dịch T_1 và T_2. Tương tự như vậy, các lô B_2 và B_3 được thêm vào cây HUS-Tree như Hình 2(b).

Với cây HUS-Tree, khi cửa sổ dịch chuyển từ W_1 sang W_2 thì dữ liệu trong lô giao dịch B_1sẽ bị xóa bỏ. Sau đó, dữ liệu trong lô giao dịch B_4 sẽ được thêm vào cây. Để xóa dữ liệu của lô giao dịch B_1cây HUS-Tree được duyệt lại một lần nữa. Mỗi lần duyệt qua một nút của cây, ta xóa đi dữ liệu *twu* đầu tiền trong mảng giá trị *twu*. Vì thứ tự các phần tử trong mảng lưu trữ *twu* ở mỗi nút sẽ tương ứng với các lô B_1, B_2và B_3. Sau đó, những nút có tất cả giá trị trong mảng lưu trữ *twu* bằng 0 sẽ bị xóa khỏi cây. Sau cùng, các sản phẩm của lô giao dịch B_4 sẽ được thêm vào cây. Hình 3 minh họa thao tác chuyển dịch sang cửa sổ W_2.

B. PHƯƠNG PHÁP KHAI THÁC CÂY HUS-TREE

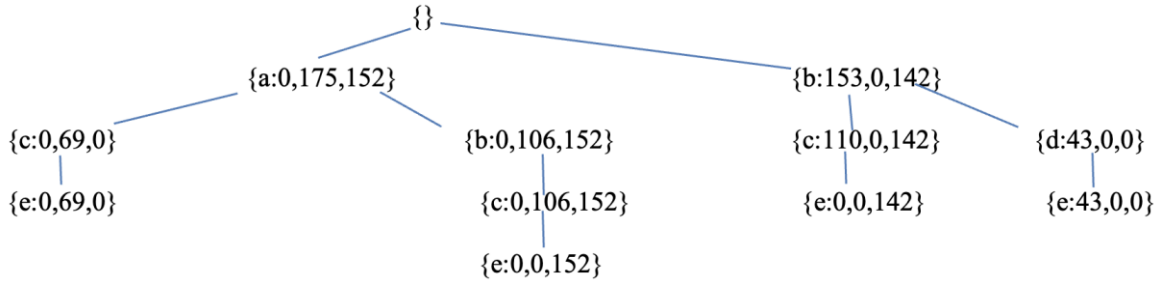
Từ cấu trúc cây HUS-Tree trong Hình 3, chúng ta có thể nhận ra các đặc điểm như sau:

Đặc tính 1: Những sản phẩm thuộc cùng một nhánh là các sản phẩm đã xuất hiện trong cùng một giao dịch. Ví dụ từ Hình 3 thấy rằng các nút {a:0,175,152} và nút {c:0,69,0} và nút {e:0,69,0} cùng một nhánh. Đồng thời các sản phẩm {a,c,e} xuất hiện trong cùng giao dịch T_6 và T_7.



(a) Sau khi chèn T_1 và T_2 (b) Sau khi thêm B_2,B_3

Hình 2. *Cây HUS-Tree.*



Hình 3. *Cây HUS-Tree sau khi chuyển dịch sang W_2.*

Đặc tính 2: Mẫu hữu ích cao có thể được phát sinh bằng cách ghép sản phẩm trong cùng một nhánh và giá trị hữu ích của mẫu là danh sách giá trị của lô giao dịch của nút thấp nhất trong nhánh. Ví dụ từ Hình 3 thấy rằng với nhánh gồm các nút {a:0,175,152} và nút {c:0,69,0} và nút {e:0,69,0} có thể phát sinh mẫu ec bằng cách ghép định danh của nút {c:0,69,0} và nút {e:0,69,0}. Khi đó, độ hữu ích của mẫu ec sẽ là giá trị lô giao dịch của nút thấp nhất trong nhánh là nút {e:0,69,0}. Giá trị hữu ích của mẫu ec là {ec:0,69,0}. Sau đó, mẫu eca có thể được phát sinh bằng cách ghép định danh của nút {ec:0,69,0} và nút {a:0,175,152}. Và do trong nhánh, nút {e:0,69,0} là nút thấp nhất nên độ hữu ích của mẫu eca là giá trị lô giao dịch của nút {e:0,69,0}. Giá trị hữu ích của mẫu eca là {eca:0,69,0}.

Từ hai đặc điểm trên, quá trình khai thác mẫu trên cây HUS-Tree có thể được phát biểu như sau. Từ cây HUS-Tree đã có, thực hiện thao tác duyệt từ nút lá đến nút gốc. Với mỗi một nhánh, thực hiện ghép định danh sản phẩm của mỗi nút trong nhánh lại với nhau để tạo mẫu ứng viên. Sau đó, kiểm tra độ hữu ích của mẫu ứng viên bằng các cộng các giá trị hữu ích trong lô giao dịch của mẫu ứng viên đó lại và so sánh với giá trị minutil. Những mẫu ứng viên có độ hữu ích lớn hơn ngưỡng minutil sẽ được giữ lại và lặp lại thao tác ghép tạo mẫu ứng viên mới. Sau quá trình duyệt và ghép như vậy ta sẽ thu được tất cả các mẫu hữu ích cao.

Để thuận tiện cho quá trình khai thác, một bảng băm dùng để lưu địa chỉ của các nút được tạo ra.

a	→	{a:0,175,152}
b	→	{b:0,106,152} →{b:153,0,142}
c	→	{c:0,69,0}→{c:0,106,152}→{c:110,0,142}
d	→	{d:43,0,0}
e	→	{e:0,69,0}→{e:0,0,152}→{e:0,0,142}→{e:43,0,0}

Hình 4. *Bảng băm lưu địa chỉ các nút.*

C. THUẬT TOÁN ĐỀ XUẤT

Trên cơ sở phân tích các đặc điểm của cấu trúc cây HUS-Tree và nhận xét phương pháp khai thác hiệu quả trên cấu trúc này, chúng tôi đề xuất thuật toán với tên gọi HUSPM-REVERSE được thể hiện trong Thuật toán 1 và một thủ tục con được trình bày trong Thuật toán 2.

Thuật toán 1: HUSPM-REVERSE (T, minutil)
Input: Cây HUS-Tree T, độ hữu ích tối thiểu minutil
Output: Mẫu hữu ích cao
Begin
Duyệt T phát sinh bảng băm b.
foreach(id_item_list in b)

```
        Tạo danh sách lưu mẫu L
        foreach(nút n in id_item_list)
            Duyệt T và ghép nút n với những nút cha của n để tạo mẫu nx
            Lưu nx vào L
        foreach(mẫu m in L)
            if độ hữu ích của mẫu m >= minutil
                MiningSubPattern(T, m, minsup)
                Xuất mẫu m
End
```

Thuật toán 2: MiningSubPattern (T, m, minutil)
Input: Cây HUS-Tree T, độ hữu ích tối tiểu minutil, mẫu hữu ích m
Output: Mẫu hữu ích cao dạng mx

```
Begin
    Tạo danh sách lưu mẫu L.
    Duyệt T và ghép mẫu m với những nút cha của m để tạo mẫu mx
    Lưu nx vào L
    foreach(mẫu mx in L)
        if độ hữu ích của mẫu mx >= minutil
            MiningSubPattern(T, mx, minsup)
            Xuất mẫu mx
End
```

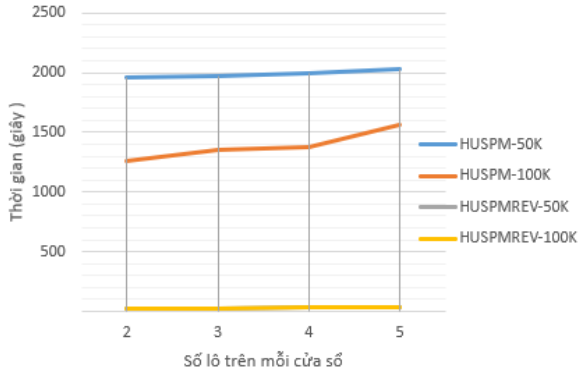
D. VÍ DỤ MINH HOẠ

Giả sử giá trị *minutil* bằng 260. Từ bảng băm trong Hình 4, sản phẩm *c* có trọng số hữu ích bằng tổng giá trị hữu ích trong các lô giao dịch bằng 69+106+152+110+142=579. Sau đó, lấy ra danh sách id_item_list các nút chứa sản phẩm *c* trong bảng băm gồm {c:0,69,0},{c:0,106,152},{c:110,0,142}. Từ nút {c:0,69,0}, thực hiện duyệt ngược về nút gốc trên cây HUS-Tree để phát sinh mẫu *ca* với độ hữu ích là {ca:0,69,0}. Từ nút {c:0,106,152}, thực hiện duyệt ngược về nút gốc trên cây HUS-Tree để phát sinh mẫu {cb:0,106,152} và {ca:0,106,152}. Từ nút {c:110,0,142}, thực hiện duyệt ngược về nút gốc trên cây HUS-Tree để phát sinh mẫu {cb:110,0,142}. Sau đó thực hiện thao tác tính trọng số hữu ích của các mẫu bằng cách cộng các giá trị hữu ích lại. Với mẫu *ca* ta có {ca:0,69,0} và {ca:0,106,152} nên độ hữu ích của mẫu *ca* là 69+106+152=327. Với mẫu *cb* ta có {cb:0,106,152} và {cb:110,0,142} nên trọng số hữu ích của mẫu *cb* là 106+152+110+142=510. Cả mẫu *ca* và *cb* đều là mẫu hữu ích cao khi trọng số hữu ích của mẫu đều lớn hơn *minutil*. Sau đó tiếp tục thực hiện phát sinh mẫu với *ca* và *cb*. Với mẫu *ca* ta thấy rằng nút chứa *a* là nút sát nút gốc nên không thể phát sinh thêm. Với mẫu *cb* ta có 2 nút là {cb:0,106,152} và {cb:110,0,142}. Trong đó nút {cb:110,0,142} có nút *b* sát nút gốc nên không thể phát sinh mẫu được nữa. Còn nút {cb:0,106,152} có nút cha là nút {a:0,175,152} nên có thể phát sinh mẫu {cba:0,106,152}. Mẫu *cba* có trọng số hữu ích là 106+152=258. Do trọng số hữu ích của mẫu *cba* nhỏ hơn *minutil* nên mẫu *cba* không phải là mẫu hữu ích cao. Quá trình khai thác cứ lặp lại như vậy cho đến khi duyệt hết tất cả các nút trong bảng băm Hình 4.

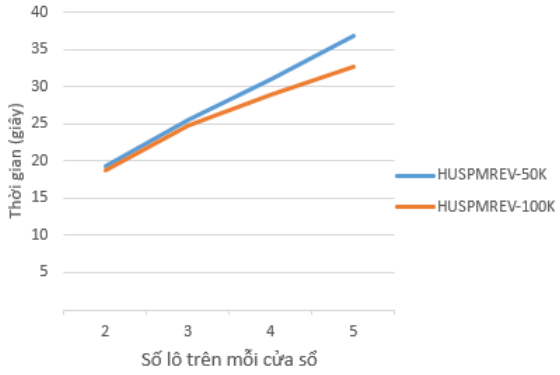
V. KẾT QUẢ THỰC NGHIỆM

Trong phần này, bài báo trình bày kết quả thực nghiệm để so sánh hiệu quả của thuật toán HUPMS và thuật toán đề xuất HUPMS-REV. Dữ liệu thực nghiệm là dữ liệu giao dịch của khách hàng từ một chuỗi cửa hàng bán lẻ ở California, Hoa Kỳ. Tên bộ dữ liệu là Chain-store [14] bao gồm 1,112,949 giao dịch và 46,086 sự kiện khác nhau. Chương trình được viết bằng ngôn ngữ Java chạy trên hệ điều hành Windows 10 Intel® Core™ i7-9700, 16384 MB RAM.

Đầu tiên, bài báo tiến hành so sánh hiệu quả về mặt thời gian thực hiện của thuật toán đề xuất (HUSPMREV) và thuật toán HUPSM trên bộ dữ liệu Chain-store với kích thước cửa sổ trượt khác nhau bằng cách thay đổi số lô trong một cửa sổ và số giao dịch trong một lô. Hình 5 cho thấy sự khác nhau về kích thước của cửa sổ với số lượng lô từ 2 đến 5. Mỗi thuật toán sẽ được thực hiện với kích thước mỗi lô là 50,000 (50K) và 100,000 (100K) giao dịch. Giá trị ngưỡng được chọn là 0.25%. Kết quả thực nghiệm cho thấy rằng thuật toán HUPSM-REV hiệu quả hơn rất nhiều so với thuật toán HUPSM do không tốn thời gian phát sinh cây tiền tố và cây điều kiện. Hoạt động phát sinh mẫu được thực hiện trực tiếp trên cây HUS-Tree nên không cần tốn thời gian duyệt cây điều kiện. Kết quả thực nghiệm cũng cho thấy rằng, khi số lượng giao dịch trong một lô là 50,000 sẽ tốn nhiều thời gian khai thác hơn do phải thực hiện hoạt động khai thác trước khi trượt sang cửa sổ khác (Hình 6).

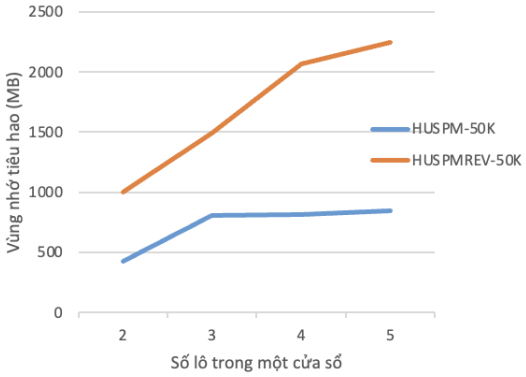


Hình 5. *So sánh thời gian thực hiện thuật toán HUPSM và HUPSM-REV.*

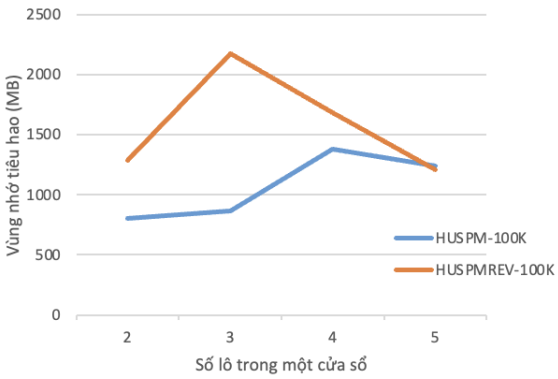


Hình 6. *So sánh thời gian thực hiện HUPSM-REV với số lượng giao dịch mỗi lô khác nhau.*

Tiếp theo, bài báo thực nghiệm so sánh về bộ nhớ của HUSPMREV và HUPSM. Kết quả trong Hình 7 thể hiện HUSPMREV tiêu tốn nhiều bộ nhớ hơn khi số lượng giao dịch trong một lô thấp. Trong khi đó, khi tăng số lượng giao dịch trong mỗi lô và số lô trong mỗi cửa sổ trượt từ 3 trở lên thì thuật toán HUSPMREV có mức độ sử dụng bộ nhớ ít hơn.



Hình 7. *So sánh bộ nhớ sử dụng của HUPSM và HUPSM-REV với kích thước mỗi lô là 50,000 giao dịch.*



Hình 8. *So sánh bộ nhớ sử dụng của HUPSM và HUPSM-REV với kích thước mỗi lô là 100,000 giao dịch.*

VI. KẾT LUẬN

Bài báo đã khảo sát và trình bày vấn đề khai thác tập hữu ích cao trong CSDL giao tác có sự thay đổi theo thời gian. Trong đó, nội dung bài báo tập trung vào nghiên cứu thuật toán khai thác mẫu hữu ích cao dựa trên cấu trúc cây HUS-Tree. Kết quả thực nghiệm cho thấy thuật toán đề xuất cho kết quả tốt hơn về mặt thời gian thực hiện so với thuật toán gốc HUPMS. Tuy nhiên, bộ nhớ của thuật toán được đề xuất có phần hạn chế hơn đối với những trường hợp số lượng giao dịch trong mỗi lô và số lượng lô trong một cửa số thấp.

Hướng phát triển trong tương lai có thể là cải tiến thêm cấu trúc cây HUS-Tree để có thể lưu trữ hiệu quả hơn, sử dụng danh sách liên kết để thay thế cấu trúc cây trong quá trình khai thác, áp dụng các giải pháp xử lý song song để tăng tốc độ xử lý, v.v.. Đồng thời nghiên cứu thực nghiệm các giải pháp khác hiệu quả hơn trong việc khai thác mẫu hữu ích cao trên dữ liệu luồng.

VII. LỜI CẢM ƠN

Nghiên cứu được tài trợ bởi Trường Đại học Ngoại ngữ – Tin học Thành phố Hồ Chí Minh trong khuôn khổ Đề tài mã số H2022-05

VIII. TÀI LIỆU THAM KHẢO

[1] H. Yao, H. J. Hamilton and C. J. Butz, "A foundational approach to mining itemset utilities from databases," *in the 2004 SIAM International Conference on Data Mining*, 2004.

[2] Y. Liu, W.-k. Liao and A. Choudhary, "A two-phase algorithm for fast discovery of high utility itemsets," *in Advances in Knowledge Discovery and Data Mining: 9th Pacific-Asia Conference*, 2005.

[3] H. Yao and H. J. Hamilton, "Mining itemset utilities from transaction databases," *Journal of Data & Knowledge Engineering*, vol. 59, no. 3, pp. 603-626, 2006.
[4] Y.-C. Li, J.-S. Yeh and C.-C. Chang, "Isolated items discarding strategy for discovering high utility itemsets," *Journal of Data & Knowledge Engineering*, vol. 64, no. 1, pp. 198-217, 2008.
[5] Q.-H. Duong, P. Fournier-Viger, H. Ramampiaro, K. Nørvåg and T.-L. Dam, "Efficient high utility itemset mining using buffered utility-lists," *Journal of Applied Intelligence*, vol. 48, pp. 1859-1877, 2018.
[6] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," *in 20th international conference very large databases*, 1994.
[7] P. Fournier-Viger, J. C.-W. Lin, T. Truong-Chi and R. Nkambou, "A survey of high utility itemset mining," *Journal of High-utility pattern mining: Theory, algorithms and applications*, pp. 1-45, 2019.
[8] S. Krishnamoorthy, "HMiner: Efficiently mining high utility itemsets," *Journal of Expert Systems with Applications*, vol. 90, pp. 168-183, 2017.
[9] 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," *Journal of Knowledge and Information Systems*, vol. 51, no. 2, pp. 595-625, 2017.
[10] P. Fournier-Viger, C.-W. Wu, S. Zida and V. S. Tseng, "FHM: Faster high-utility itemset mining using estimated utility co-occurrence pruning," *Foundations of Intelligent Systems: 21st International Symposium*, pp. 83-92, 2014.
[11] J. Liu, K. Wang and B. C. Fung, "Direct discovery of high utility itemsets without candidate generation," *in 12th IEEE international conference on data mining*, 2012.
[12] H.-F. Li, H.-Y. Huang, Y.-C. Chen, Y.-J. Liu and S.-Y. Lee, "Fast and memory efficient mining of high utility itemsets in data streams," *in 2008 eighth IEEE international conference on data mining*, 2008.
[13] C. F. Ahmed, S. K. Tanbeer and B.-S. Jeong, "Efficient mining of high utility patterns over data streams with a sliding window method," *Journal of Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*, pp. 99-113, 2010.
[14] A. Ghazikhani, R. Monsefi and H. Sadoghi Yazdi, "Online neural network model for non-stationary and imbalanced data stream classification," *International Journal of Machine Learning and Cybernetics*, vol. 5, no. 1, pp. 51-62, 2014.
[15] H. Li, "On-line and dynamic time warping for time series data mining," *International Journal of Machine Learning and Cybernetics*, vol. 6, no. 1, pp. 145-153, 2015.
[16] C. F. Ahmed, S. K. Tanbeer, B.-S. Jeong and H.-J. Choi, "Interactive mining of high utility patterns over data streams," *Expert Systems with Applications*, vol. 39, no. 15, pp. 11979-11991, 2012.
[17] J. Pisharath, Y. Liu, W.-k. Liao, A. Choudhary, G. Memik and J. Parhi, "NU-MineBench 2.0," Technical report, *Northwestern University*, 2005.

THE METHOD FOR HIGH UTILITY PATTERN MINING OVER TRANSACTIONAL DATA STREAM BASED ON HUSTREE

Tran Minh Thai, Tran Anh Duy, Pham Duc Thanh, Le Thi Minh Nguyen, Nguyen Thanh Trung

ABSTRACT— High utility pattern mining over transactional data streams is an important research issue in the field of data mining. The mining approach is used to discover high-utility itemsets in transactional databases. Furthermore, the constantly changing number of transactions in data streams generates new high-utility patterns and modifies the utility of previously discovered patterns. Timely updating of this changing information is crucial for making effective business decisions. However, the number of available mining methods for transactional data streams is still limited. In this paper, we propose a new method for mining transactional data streams using a HUS tree. The experimental results show that our new method is more efficient in terms of execution time than previous solutions.

Keywords: Transactional data stream, Data mining, High utility itemset, High-utility pattern.

TS. Trần Minh Thái tốt nghiệp cử nhân CNTT năm 2001 và thạc sỹ Tin học năm 2006 ĐH Khoa học Tự nhiên – ĐH Quốc gia TPHCM, nhận bằng tiến sỹ CNTT năm 2017 do ĐH Quốc gia TPHCM cấp. Ông từng là giảng viên và quản lý khoa CNTT trường CĐ CNTT TPHCM từ 2002 - 2015. Từ 2015 đến nay, ông là giảng viên và là trưởng bộ môn HTTT thuộc khoa CNTT Trường ĐH Ngoại ngữ - Tin học TPHCM. Lĩnh vực nghiên cứu chính của ông liên quan đến vấn đề khai thác dữ liệu, ẩn dữ liệu, xử lý dữ liệu lớn và nhận dạng.

ThS. Lê Thị Minh Nguyện tốt nghiệp thạc sĩ Khoa học máy tính năm 2007 tại trường Đại Học Công Nghệ Thông Tin Tp.HCM. Từng là giảng viên tại trường Cao đẳng Công nghệ Thông Tin từ 2003-2015. Từ năm 2015 đến nay là giảng viên thuộc khoa CNTT, trường Đại học Ngoại ngữ-Tin học TPHCM. Lĩnh vực nghiên cứu quan tâm là Khai thác dữ liệu.

ThS. Phạm Đức Thành nhận học vị Thạc sĩ năm 2006 tại Đại học Quốc gia Thành phố Hồ Chí Minh; hiện là Giảng viên khoa CNTT, trường Đại học Ngoại ngữ-Tin học TPHCM; lĩnh vực nghiên cứu quan tâm là Khai thác dữ liệu.

ThS. Trần Anh Duy nhận học vị thạc sĩ Khoa học máy tính trường Đại học Khoa học Tự nhiên năm 2017. Hiện là giảng viên khoa CNTT, trường Đại học Ngoại ngữ-Tin học TPHCM; lĩnh vực nghiên cứu quan tâm là Khai thác dữ liệu.

ThS. Nguyễn Thanh Trung tốt nghiệp thạc sĩ Khoa học Máy tính năm 2006 tại trường ĐH Khoa học Tự nhiên - ĐH Quốc gia TPHCM. Hiện là giảng viên khoa CNTT, trường ĐH Ngoại ngữ - Tin học TPHCM. Lĩnh vực nghiên cứu chính là Khai thác dữ liệu.