Một mô hình hiệu quả khai phá tập mục lợi ích cao

An Efficient Model for Mining High Utility Itemsets

Đậu Hải Phong, Nguyễn Mạnh Hùng

Abstract: *Today, high utility itemsets mining is an important research issue in data mining because it considers the profit and quantity of items in each transaction. Most high utility itemsets algorithms such as UP-Growth [9], Udepth [5], Two-Phase [3], PB (Projection-Based) [8], ect. use TWU model (Transaction Weight Utility) for pruning candidates. However, the number of candidate itemsets generated in these algorithms is enormous. In this paper, we propose a new candidate weight utility model (CWU) and HP (High Projection) algorithm base on CWU to reduce the number of candidate itemsets. The experimental results show that the performance and number candidate of our algorithm is better than Two-Phase [3], PB [8].*

Keywords: *Data Mining, Frequent Itemsets, High Utility, Candidate Weight Utility, HP algorithm.*

I. GIỚI THIỆU

Khai phá tập phổ biến từ cơ sở dữ liệu giao dịch là một trong các kỹ thuật khai phá dữ liệu và có rất nhiều ứng dụng trong kinh doanh, y tế, giáo dục,... Mục tiêu của khai phá tập phổ biến là tìm ra các tập phần tử có tần suất xuất hiện lớn hơn một ngưỡng hỗ trợ tối thiểu cho trước. Từ tập phổ biến sinh ra các luật dựa trên độ tin cậy tối thiểu. Trong bài toán khai phá tập phổ biến cơ bản thì mọi phần tử là tương đương nhau và chỉ quan tâm đến tần suất xuất hiện của phần tử trong các giao dịch. Tuy nhiên, trong thực tế, các phần tử trong cơ sở dữ liệu giao dịch thường có các thuộc tính định lượng như: số lượng, lợi ích... Năm 2003, Chan [1] đã đưa ra khái niệm lợi ích của tập phần tử để khắc phục những nhược điểm của tập phần tử phổ biến cơ bản. Lợi ích của một phần tử trong một giao dịch được tính toán dựa trên lợi ích trong giao dịch và lợi ích ngoài của phần tử đó. Ví dụ, lợi ích trong từng giao dịch như số lượng của mặt hàng trong mỗi lần mua hàng; lợi ích ngoài như giá trị lợi nhuận của từng mặt hàng đó đem lại. Một tập có lợi ích cao khi giá trị lợi ích của nó không nhỏ hơn một ngưỡng lợi ích tối thiểu cho trước.

Một vấn đề khó khăn trong khai phá tập lợi ích cao là các tập lợi ích không có tính chất đóng [2]. Tính chất này đảm bảo một tập là tập lợi ích cao thì các tập con của nó cũng là tập lợi ích cao. Do vậy, số lượng các ứng cử viên được sinh ra rất lớn và chi phí lớn về thời gian duyệt dữ liệu nhiều lần để kiểm tra các ứng viên như trong một số thuật toán [3-5]. Để tránh duyệt dữ liệu nhiều lần thì một số thuật toán dùng cấu trúc cây để tìm tập lợi ích cao như [6,9,10]. Nhưng các thuật toán này tiêu tốn nhiều thời gian và không gian bộ nhớ để sinh ra các cây điều kiện của từng tiền tố.

Trong [7] Liu đã trình bày mô hình TWU khai phá tập lợi ích cao để loại bớt tập ứng viên. Giá trị lợi ích của tất cả các phần tử trong giao dịch được tổng hợp như là lợi ích của giao dịch và lấy nó làm cận trên lợi ích của các tập phần tử trong giao dịch đó. Khi đó, lợi ích giao dịch có trọng số (viết tắt: TWU – Transactions Weight Utility) của một tập phần tử được định nghĩa là tổng các lợi ích của giao dịch có chứa tập đó. Trong thuật toán [7], Liu đã sử dụng mô hình TWU để loại đi các tập có TWU nhỏ hơn ngưỡng lợi ích tối thiểu cho trước để giảm số lượng tập ứng cử viên, sau đó duyệt lại cơ sở dữ liệu để xác định lợi ích thực tế của các tập và tiếp tục xác định khả năng các tập lớn hơn của nó có là tập lợi ích cao hay không để sinh tiếp các ứng cử viên.

Trong các thuật toán sử dụng mô hình TWU, thuật toán [5,8] thực hiện tìm kiếm tập ứng viên theo chiều sâu. Ví dụ, xét tập phần tử I={A, B, C, D, E}, từ phần

tử A có thể sinh ra các tập ứng viên {AB}, {ABC}, {ABCD}, {ABCDE},...; từ phần tử B sinh ra các tập ứng viên {BC}, {BCD}, {BCDE},... Ta thấy, các tập ứng viên sinh ra từ phần tử B không còn xuất hiện phần tử A, nhưng khi lấy TWU làm cận trên thì vẫn có giá trị lợi ích của phần tử A. Điều này làm tăng số lượng tập ứng viên và chi phí kiểm tra các tập ứng viên.

Trong bài báo này, chúng tôi đề xuất mô hình CWU (Candidate Weight Utility) và thuật toán khai phá tập lợi ích cao dựa trên mô hình này nhằm giảm số lượng tập ứng viên.

Nội dung tiếp theo của bài báo được tổ chức như sau: Phần II vấn đề liên quan đến khai phá tập lợi ích cao. Mô hình CWU được đề xuất trong Phần III. Phần IV đề xuất thuật toán khai phá tập lợi ích cao sử dụng mô hình CWU. Phần V, trình bày kết quả đạt được và so sánh với các thuật toán khác. Cuối cùng là kết luận.

II. TỔNG QUAN VẤN ĐỀ NGHIÊN CỨU

Cho một cơ sở dữ liệu gồm các giao dịch T_i là D = $\{T_1,T_2,T_3,\ldots T_n\}$, các giao dịch được xác định duy nhất bởi Tid, I=$\{i_1,i_2,i_3,\ldots i_n\}$ là các phần tử (item) xuất hiện trong các giao dịch, $X \subseteq I$ là tập các phần tử (itemsets). Một tập X được gọi là tập k-phần tử khi số lượng phần tử của X là k.

Để thuận lợi trong giải thích các khái niệm, chúng tôi đưa ra một cơ sở dữ liệu giao dịch được biểu diễn dưới dạng bảng như Bảng 1. Bảng lợi ích ngoài của các phần tử được cho trong Bảng 2.

Bảng 1. Cơ sở dữ liệu giao dịch minh họa

Tid	Giao dịch					
	A	B	C	D	E	F
1	1	0	2	1	1	1
2	0	1	25	0	0	0
3	0	0	0	0	2	1
4	0	1	12	0	0	0
5	2	0	8	0	2	0
6	0	0	4	1	0	1
7	0	0	2	1	0	0
8	3	2	0	0	2	3
9	2	0	0	1	0	0
10	0	0	4	0	2	0

Bảng 2. Bảng lợi ích của các phần tử

Item	A	B	C	D	E	F
Lợi ích	3	10	1	6	5	2

Định nghĩa 1 [8] - Lợi ích trong (internal utility) của mỗi phần tử là giá trị của mỗi phần tử trong từng giao dịch. Ký hiệu: $O(i_k,T_j)$ – là lợi ích trong của phần tử i_k trong giao dịch T_j.

Ví dụ, $O(A,T_1) = 1$; $O(C,T_1) = 2$ trong Bảng 1.

Định nghĩa 2 [8] - Lợi ích ngoài (external utility) của mỗi phần tử là giá trị lợi ích của mỗi phần tử trong bảng lợi ích. Ký hiệu: $S(\{i_k\})$ là lợi ích ngoài của phần tử i_k.

Ví dụ, $S(\{A\}) = 3$; $S(\{B\}) = 10$ trong Bảng 2.

Định nghĩa 3 [8] - Lợi ích của một phần tử trong giao dịch là tích của lợi ích trong và lợi ích ngoài của phần tử đó. Ký hiệu: $U(i_k,T_j) = S(\{i_k\}) * O(i_k,T_j)$ là lợi ích của phần tử i_k trong giao dịch T_j.

Ví dụ, $U(\{A\},T_1) = 3*1 = 3$; $U(\{C\},T_1) = 1*2 = 2,\ldots$

Định nghĩa 4 [8] - Lợi ích của một tập phần tử X trong một giao dịch T_j là tổng giá trị lợi ích tất cả phần tử của tập X trong giao dịch T_j. Ký hiệu: $U(X,T_j) = \sum_{i_k \in X \wedge X \subseteq T_j} U(i_k, T_j)$ – là lợi ích của tập phần tử X trong một giao dịch T_j.

Ví dụ, $U(\{AC\},T_1) = 3*1 + 1*2 = 5$.

Định nghĩa 5 [8] - Lợi ích của một tập phần tử X trong cơ sở dữ liệu là tổng lợi ích của tập phần tử X trong tất cả giao dịch chứa X. Ký hiệu: $AU(X) = \sum_{T_j \in D \wedge X \subseteq T_j} U(X, T_j)$.

Ví dụ, xét tập{AC}, ta thấy {AC}, xuất hiện trong các giao dịch: T_1, T_5 nên ta có: $AU(\{AC\}) = U(\{AC\},T_1) + U(\{AC\},T_5) = (3*1 + 1*2) + (3*2 + 1*8) = 19$.

Định nghĩa 6 [8] – Tập phần tử lợi ích cao: Tập X được gọi là tập phần tử lợi ích cao (HU – High Utility) nếu $AU(X) \geq \alpha$, ngược lại gọi X là tập phần tử lợi ích thấp. Trong đó α là ngưỡng lợi ích tối thiểu cho trước.

Ví dụ, lợi ích tối thiểu α = 12 thì {AC} là tập phần tử lợi ích cao.

Định nghĩa 7 [8] - Lợi ích của một giao dịch là tổng lợi ích của các phần tử trong giao dịch đó. Ký hiệu: $TU(T_j) = \sum_{i_k \in T_j} U(i_k, T_j)$ – là lợi ích của giao dịch T_j.

Ví dụ, $TU(T_1)$ = 1*3 + 2*1 + 1*6 + 1*5 + 1*2 = 18, $TU(T_2)$ = 1*10 + 25*1 = 35.

Định nghĩa 8 [8] - Lợi ích giao dịch có trọng số của một tập phần tử X là tổng lợi ích của các giao dịch có chứa tập phần tử X. Ký hiệu: $TWU(X) = \sum_{X \subseteq T_j \wedge T_j \in D} TU(T_j)$ là lợi ích giao dịch có trọng số của tập phần tử X.

Ví dụ: $TWU(\{AC\}) = TU(T_1) + TU(T_5)$ = 18 + 24 = 42.

Hiện nay, đã có nhiều thuật toán khai phá tập lợi ích cao như CTU-Mine [10], HUC-Prune [11], UP-Growth [9], UDepth [5], PB [8], Two – Phase [3]. Thuật toán Two – Phase sử dụng mô hình TWU để loại bớt tập ứng viên, sau đó duyệt lại cơ sở dữ liệu để xác định lợi ích thực tế của các tập. Với phương pháp sinh ứng viên và kiểm tra ứng viên thì phải duyệt dữ liệu nhiều lần để tìm tập lợi ích cao.

Để tiết kiệm chi phí trong việc sinh các tập ứng cử viên và giảm số lần duyệt cơ sở dữ liệu, một số thuật toán sử dụng cấu trúc cây đã được đề xuất như: CTU-Tree [10], HUC-Tree [11], UP-Growth [9] các thuật toán này gồm một số bước chính sau: xây dựng cây, sinh các ứng viên từ cây bằng thuật toán tăng trưởng mẫu, xác định các tập lợi ích cao từ các tập ứng viên. Với cách tiếp cận này thì thuật toán vẫn tốn nhiều bộ nhớ và thời gian duyệt cây.

Thuật toán UDepth [5] được Wei đưa ra thực hiện khai phá cơ sở dữ liệu theo chiều dọc. Thuật toán này gồm các bước: duyệt dữ liệu để xác định TWU của từng phần tử; loại bỏ các phần tử có TWU nhỏ hơn ngưỡng tối thiểu; sắp xếp lại các phần tử có TWU cao theo thứ tự giảm dần; từ mỗi phần tử i_k có TWU cao, tìm tất cả các tập có phần tử i_k là tiền tố và duyệt lại cơ sở dữ liệu một lần nữa để xác định các tập lợi ích cao.

Năm 2013, Gou và cộng sự đưa ra thuật toán PB [8] dựa trên các bảng chỉ số để tăng tốc độ thực hiện và giảm yêu cầu bộ nhớ. Thuật toán này sử dụng bảng chỉ số của các tập để sinh các ứng viên, tìm tập lợi ích cao và tạo nhanh bảng chỉ số từ tập tiền tố của nó.

Hầu hết các thuật toán khai phá tập lợi ích cao, như trong [3,5,8],... đều sử dụng mô hình TWU làm cơ sở để cắt tỉa các tập ứng viên. Với một phần tử a và một tập phần tử {X}, ta có $TWU(\{aX\}) = \sum_{aX \subseteq T_j \wedge T_j \in D} TU(T_j)$ là cận trên của AU({aX}). Tương tự, có TWU({X}) là cận trên của AU({X}). Ta thấy {X} ⊆{aX} nên số giao dịch chứa {X} sẽ lớn hơn hoặc bằng số giao dịch chứa {aX}. Vậy, TWU({X}) là tổng lợi ích của các giao dịch chứa {X} sẽ lớn hơn hoặc bằng TWU({aX}) là tổng lợi ích của các giao dịch chứa {aX}.

Trong các thuật toán khai phá tập lợi ích cao UDepth [5], PB [8],... thì tập lợi ích cao được khai phá theo chiều sâu. Giả sử, {aX} là tất cả các tập có tiền tố là phần tử a, {bX} là tất cả các tập có tiền tố là phần tử b. Khi khai phá các tập trong {bX} sẽ không còn chứa phần tử a. Nhưng khi tính TWU({bX}) có thể vẫn gồm giá trị lợi ích của phần tử a. Điều này làm TWU({bX}) là cận trên của AU({bX}) lớn hơn mức cần thiết và khi dùng TWU({bX}) để tỉa các tập ứng viên sẽ không hiệu quả.

III. ĐỀ XUẤT MÔ HÌNH CWU

Từ những nhận xét trong Phần II, chúng tôi đề xuất mô hình CWU (Candidate Weight Utility) để khắc phục nhược điểm của mô hình TWU.

Định nghĩa 9 - Tập tiền tố của một phần tử It_x là tập các phần tử trong I mà đứng trước phần tử It_x: $ListItemPrefix(It_x) = \{j \in I \mid j \prec It_x\}$.

Định nghĩa 10 - Tiền tố của một tập Y là tập các phần tử trong I mà đứng trước phần tử đầu tiên của tập Y: $ListItemPrefix(Y) = \{ j \in I \mid j \prec y_1\}$, trong đó: y_1 là phần tử đầu tiên trong tập Y.

Định nghĩa 11 - Lợi ích ứng viên có trọng số (CWU – Candidate Weight Utility) của tập phần tử Y, ký hiệu là CWU(Y) được xác định như sau:

Đặt X = ListItemPrefix(Y), thì

$$CWU(Y) = \sum_{Y \subseteq T_j} TU(T_j) - \sum_{Y \subseteq T_j} \sum_{i_k \in X \wedge i_k \in T_j} U(i_k, T_j)$$

Nếu ListItemPrefix(Y) = {∅} thì $\sum_{Y \subseteq T_j} \sum_{i_k \in X \wedge i_k \in T_j} U(i_k, T_j) = 0$.

Định nghĩa 12 - Nếu CWU(Y) ≥ α' với α' là ngưỡng tối thiểu lợi ích ứng viên cho trước thì Y gọi là tập ứng viên lợi ích trọng số cao (HCWU- High Candidate Weight Utility). Ngược lại, Y được gọi là tập ứng viên lợi ích trọng số thấp (LCWU – Low Candidate Weight Utility).

Định lý 1 - Cho 2 tập Y_k – là tập k-phần tử, Y_{k-1} – (k-1)-phần tử và là tập tiền tố của tập Y_k. Nếu Y_k là HCWU thì Y_{k-1} cũng là HCWU.

Chứng minh:

Ta có, $\{aY_k\}$ là tập các giao dịch chứa Y_k, $\{aY_{k-1}\}$ là tập các giao dịch chứa Y_{k-1}. Khi Y_{k-1} là tập tiền tố của Y_k thì $\{aY_k\} \subseteq \{aY_{k-1}\}$. Gọi X = ListItemPrefix(Y_{k-1}) = ListItemPrefix(Y_k), theo Định nghĩa 11 ta có:

$$CWU(Y_{k-1}) = \sum_{Y_{k-1} \subseteq T_q} TU(T_q) - \sum_{Y_{k-1} \subseteq T_q} \sum_{i_t \in X \wedge i_t \in T_q} U(i_t, T_q)$$

$$\geq \sum_{Y_k \subseteq T_p} TU(T_p) - \sum_{Y_k \subseteq T_p} \sum_{i_t \in X \wedge i_t \in T_p} U(i_t, T_p) = CWU(Y_k) \geq \alpha'$$

Như vậy, nếu tập Y_{k-1} là tập ứng viên lợi ích trọng số thấp thì tất cả các tập có tiền tố là Y_{k-1} cũng là tập ứng viên lợi ích trọng số thấp và không thể là tập lợi ích cao. Điều này cho phép loại các tập ứng viên.

Định lý 2 - Cho HCWUs – gồm các tập Y' có CWU(Y') ≥ α', HUs – gồm các tập Y có AU(Y) ≥ α với α là các ngưỡng lợi ích tối thiểu cho trước. Nếu α = α' thì HUs ⊆ HCWUs.

Chứng minh:

Gọi X = ListItemPrefix(Y), $Z_j = T_j \backslash XY$. Khi đó,

$$TU(T_j) = \sum_{i_t \in X \wedge i_t \in T_j} U(i_t, T_j) + \sum_{i_k \in Y \wedge i_k \in T_j} U(i_k, T_j) + U(Z_j, T_j).$$

$$\alpha' = \alpha \leq AU(Y) = \sum_{Y \subseteq T_q} U(Y, T_q)$$

$$= \sum_{Y \subseteq T_q} TU(T_q) - \sum_{Y \subseteq T_q} \sum_{i_t \in X \wedge i_t \in T_q} U(i_t, T_q) - \sum_{Y \subseteq T_q} \sum_{i_k \in Z_q \wedge i_k \in T_q} U(i_k, T_q) \leq \sum_{Y \subseteq T_q} TU(T_q) - \sum_{Y \subseteq T_q} \sum_{i_t \in X \wedge i_t \in T_q} U(i_t, T_q) = CWU(Y)$$

Như vậy, nếu dùng mô hình CWU để loại bỏ các tập ứng viên thì không bỏ sót các tập lợi ích cao.

Để minh chứng mô hình CWU có số ứng viên ít hơn mô hình TWU, chúng tôi đưa ra bổ đề sau.

Bổ đề 1 - Cho tập mục bất kỳ Y, ta luôn có CWU(Y)≤ TWU(Y).

Chứng minh:

Gọi X = ListItemPrefix(Y). Khi đó,

$$CWU(Y) = \sum_{Y \subseteq T_q} TU(T_q) - \sum_{Y \subseteq T_q} \sum_{i_t \in X \wedge i_t \in T_q} U(i_t, T_q) \leq \sum_{Y \subseteq T_q} TU(T_q) = TWU(Y)$$

Bổ đề 2 - Cho HCWUs – gồm các tập Y' có CWU(Y') ≥ α và HTWUs – gồm các tập Y có TWU(Y) ≥ α, với α là các ngưỡng lợi ích tối thiểu cho trước, thì HCWUs ⊆ HTWUs.

Chứng minh:

Nếu X là tập mục bất kỳ thuộc HCWUs thì CWU (X) ≥ α.

Theo Bổ đề 1, ta có TWU(X) ≥ CWU(X) ≥ α, do đó X thuộc HTWUs.

Vậy, HCWUs ⊆ HTWUs.

Theo Bổ đề 2, ta có mô hình CWU sinh ra số lượng tập ứng viên ít hơn so với mô hình TWU.

IV. ĐỀ XUẤT THUẬT TOÁN

Trong phần này, chúng tôi trình bày thuật toán HP được cải tiến từ thuật toán PB [8] sử dụng kết hợp hai mô hình TWU và CWU.

Thuật toán PB dựa trên bảng tập ứng viên tạm thời (TC) để lưu trữ và lấy nhanh các giá trị TWU và AU trong quá trình khai phá. Sau khi phân tích và triển khai chúng tôi thấy thuật toán PB có một số bước có thể cải tiến như sau:

- Để xây dựng bảng TC cho tập gồm 1 phần tử thì thuật toán đã phải tiêu tốn hai lần duyệt dữ liệu. Lần thứ nhất tính TU cho các giao dịch, lần thứ hai tính TWU và AU cho từng phần tử.

- Mặt khác, tập ứng viên X' được sinh trực tiếp trên từng giao dịch T_j chứa tập tiền tố X kết hợp với các phần tử i_p xuất hiện phía sau tập tiền tố X dựa trên bảng chỉ số (IT). Khi đó, $AU(\{X'\}) = \sum_{X' \subseteq T_j} U(X', T_j) = \sum_{X \subseteq T_j} U(X, T_j) + \sum_{X \subseteq T_j \wedge i_p \in T_j} U(i_p, T_j)$. Như vậy, thuật toán lại phải tính lại giá trị U của tập tiền tố X trong các giao dịch.

- Một tồn tại nữa trong thuật toán PB là sau khi tính được TWU của từng phần tử đã không loại bỏ các phần tử có TWU nhỏ hơn ngưỡng tối thiểu cho trước dựa vào tính chất đóng của TWU. Điều này làm tăng thời gian kiểm tra khả năng kết hợp của tập tiền tố với phần tử đó để sinh ra tập ứng viên.

- Trong thuật toán PB các phần tử được xử lý theo thứ tự từ điển và sinh các tập ứng viên bằng cách kết hợp tập tiền tố với phần tử phía sau trong từng giao dịch. Nên khi các phần tử có tần suất xuất hiện cao (thường là các phần tử có TWU hoặc AU cao) được xử lý sau thì khả năng tập tiền tố cần kết hợp với các phần tử phía sau càng cao. Điều này làm tăng số lượng ứng viên.

Để khắc phục những tồn tại trên của thuật toán PB, chúng tôi có đưa ra thuật toán HP sử dụng kết hợp cả hai mô hình: TWU và CWU để làm giảm số lượng tập ứng viên. Trong thuật toán có sử dụng một số cấu trúc sau:

- Bảng ứng viên TC_k gồm: các tập k-phần tử, lợi ích ứng viên có trọng số - CWU và lợi ích thực tế của tập ứng viên - AU. Các giá trị CWU, AU trong bảng TC_1 gồm các tập 1-phần tử được tính toán trong cùng 1 lần duyệt dữ liệu và lần đầu CWU được tính như TWU. Sau mỗi lần tìm tất cả các tập lợi ích cao với một tiền tố X thì giá trị $CWU(Y) = \sum_{Y \subseteq T_j} TU(T_j) - \sum_{Y \subseteq T_j} \sum_{i_k \in X \wedge i_k \in T_j} U(i_k, T_j)$ với X = ListItemPrefix(Y). Từ bảng TC_k có thể xác định ngay tập lợi ích cao và loại bỏ các tập không có khả năng sinh ra các tập có lợi ích cao dựa vào tính chất đóng của CWU. Ví dụ, bảng ứng viên 1 phần tử trong Bảng 3.

Bảng 3. Bảng TC_1 với tập gồm 1 phần tử

Itemsets	A	B	C	D	E	F
CWU	99	102	133	50	113	87
AU	24	40	57	24	45	12

- Bảng chỉ số IT_X của tập X gồm: các giao dịch T_j chứa tập X, vị trí p của phần tử cuối cùng của tập X xuất hiện trong giao dịch T_j và $U(X,T_j)$. Ví dụ, bảng chỉ số IT_A của tập {A} trong Bảng 4. Với một thứ tự đã được sắp xếp trong giao dịch thì từ bảng IT_A có thể xác định nhanh tập các ứng viên 2-phần tử bằng cách kết hợp A với từng phần tử sau A trong từng giao dịch. Ví dụ, với giao dịch 5 và sau vị trí 1 sinh được tập các ứng viên {AC}, {AE}. Trong giao dịch 8, sau vị trí 1 sinh được tập các ứng viên {AB}, {AE}, {AF}. Từ Bảng 4, có thể tính nhanh $U(\{AB\},T_8) = U(\{A\},T_8) + U(\{B\},T_8) = 9 + 10*2 = 29$. Trong đó giá trị $U(A,T_8) = 9$ đã được tính trong quá trình xây dựng bảng IT_A.

Bảng 4. Bảng chỉ số IT_A của tập {A}

Tid	Vị trí cuối	U({A},Tj)
1	1	3
5	1	6
8	1	9
9	1	6

- Bảng giao dịch lợi ích - UT_i chứa giá trị lợi ích của phần tử i trong từng giao dịch gồm: giao dịch T_j chứa i và $U(i,T_j)$. Sau khi tìm tất cả tập lợi ích cao với tiền tố là phần tử i thì dựa vào bảng UT_i sẽ tính được

CWU(Y) với phần tử i = ListItemPrefix(Y). Ví dụ, bảng UT_A của phần tử A trong Bảng 5 và A = ListItemPrefix(B) thì CWU(B) = $TU(T_2) + TU(T_4) + TU(T_8) – U(A,T_8) = 35 + 22 + 45 – (3*3) = 93$. Nhưng TWU(B) = $TU(T_2) + TU(T_4) + TU(T_8) = 35 + 22 + 45 = 102$.

Bảng 5. Bảng UT_A của phần tử A

Tid	U(A,Tj)
1	3
5	6
8	9
9	6

IV.1. Mô tả thuật toán HP

INPUT: cơ sở dữ liệu giao dịch, lợi ích mỗi phần tử, α - ngưỡng lợi ích tối thiểu.

OUTPUT: Tất cả các tập lợi ích cao

1: For $T_j \in D$ {

1.1:Tính $TU(T_j)$;

1.2: Xây dựng bảng ứng viên một phần tử TC_1(Itemsets, CWU, AU);

}

2: For $i \in TC_1$ {

2.1: If CWU(i) ≥ α, đưa i vào tập $HCWU_1$ theo thứ tự giảm dần theo AU;

2.2: If AU(i) ≥ α, đưa i vào tập HUs;

}

3: For $T_j \in D$ {

3.1: If $CWU(i_j) < \alpha$, {

3.1.1. Loại $i_j \in T_j$;

3.1.2. $TU(T_j) = TU(T_j) – U(i_j,T_j)$;

}

3.2: Sắp xếp $i_j \in T_j$ giảm dần theo AU;

3.3. Cập nhật lại CWU cho bảng TC_1;

}

4: For $i \in HCWU_1$ {

4.1: If CWU(i) ≥ α,

4.1.1: Xây dựng bảng UT_i ;

4.1.2: Xây dựng bảng IT_i ;

4.1.3: Đặt k = 1; //k là số lượng phần tử trong tập phần tử

4.1.4: **Search-HU({i}, k, IT_i)**; //Tìm tất cả tập lợi ích cao theo chiều sâu với tiền tố i

}

5: For $(j,U) \in UT_i$ {

5.1: $TU(T_j) = TU(T_j) – U(i,T_j)$;

}

6: For X ∈ HUs {

6.1: Hiển thị X;

}

Hàm Search-HU(X,k,$IT_{\{x\}}$)

INPUT*:* X – tập tiền tố; k – số phần tử trong tập; IT_X – bảng chỉ số của tập X.

OUTPUT*:* danh sách các tập có lợi ích cao.

//Xây dựng bảng TC_{k+1} với X là tiền tố dựa trên bảng $IT_{\{X\}}$

1: TC_{k+1}={};

2: For $(j,p) \in IT_{\{X\}}${

2.1: For $i_{p+1} \in T_j$ {

//p là phần tử cuối cùng của tập X

2.1.1: If $i_{p+1} \in HCWU_k$ then, X’ = X ∪ i_{p+1};

2.1.2: If X’ ∉ TC_{k+1}, đưa (X’, $TU(T_j)$, $U(X,T_j) + U(i_{p+1},T_j)$) vào bảng TC_{k+1} (Itemsets, CWU, AU);

2.1.3: If X’ ∈ TC_{k+1} then:

- CWU(X’) = CWU(X’) + $TU(T_j)$;
- AU(X’) = AU(X’) + $U(X,T_j) + U(i_{p+1},T_j)$;

2.1.4: Nếu k = 1, $CWU(i_{p+1}) = CWU(i_{p+1}) – U(i_p,T_j)$;

}

}

3: For X’ ∈ TC_{k+1} {

3.1: If CWU(X’) ≥ α, đưa X’ vào tập $HCWU_{k+1}$;

3.2: If AU(X’) ≥ α, đưa X’ vào tập HUs;

}

4: For X’ ∈ $HCWU_{k+1}$ {

4.1: Xây dựng $IT_{\{X'\}}$ từ $IT_{\{X\}}$;

4.2: k = k +1;

4.3: **Search-HU ({X’}, k, $IT_{X'}$);** //để tìm tập lợi ích cao theo chiều sâu với tiền tố là tập {X’}

}

5: Return HUs;

IV.2. Ví dụ minh họa

Trong phần này chúng tôi sẽ minh họa các bước của thuật toán thông qua cơ sở dữ liệu giao dịch trong Bảng 1 và bảng lợi ích tương ứng trong Bảng 2 với ngưỡng lợi ích tối thiểu α=56.

Bước 1, duyệt từng $T_j \in D$:

1.1. Tính $TU(T_j)$ ={18, 35, 12, 22, 24, 12, 8, 45, 12, 14} tương ứng với TU của các giao dịch từ 1 đến 10.

1.2. Xây dựng bảng TC_1 gồm: CWU, AU của từng phần tử. Ta có bảng TC_1 có kết quả như Bảng 6.

Bảng 6. Bảng TC_1 với tập gồm 1 phần tử

Itemsets	A	B	C	D	E	F
CWU	99	102	133	50	113	87
AU	24	40	57	24	45	12

Bước 2, duyệt các phần tử i trong Bảng 6.

2.1. Nếu phần tử có CWU(i) ≥ 56 thì đưa vào tập $HCWU_1$. Ta được, tập $HCWU_1$ sau khi sắp xếp theo AU gồm: {C, E, B, A, F}.

2.2. Nếu AU(i) ≥ 56 thì đưa vào tập HUs. Ta được tập HUs ={C:57}.

Bước 3, duyệt lại từng T_j:

3.1. Nếu CWU(i) < α thì:

3.1.1. Loại i khỏi giao dịch T_j, ta thấy có CWU(D) = 50< 56 nên loại D khỏi các giao dịch 1, 6, 7, 9.

3.1.2. Cập nhật lại TU của các giao dịch 1, 6, 7, 9 vì D đã bị loại khỏi các giao dịch trên. Vậy ta được TU ={12, 35, 12, 22, 24, 6, 2, 45, 6, 14} tương ứng với các giao dịch từ 1 đến 10.

3.2. Sau khi loại D khỏi các giao dịch và sắp xếp các giao dịch giảm dần theo AU được kết quả như Bảng 7.

3.3. Cập nhật lại CWU cho bảng TC_1 sau khi loại D, được kết quả như Bảng 8;

Bước 4, với mỗi phần tử $i \in HCWU_1$:

4.1. Nếu CWU(i) > α thì thực hiện: (giả sử với phần tử C có CWU(C) = 115 > α).

4.1.1. Xây dựng bảng UT_C như Bảng 9;

4.1.2. Xây dựng bảng IT_C như Bảng 10;

4.1.3. Đặt k =1;

4.1.4. Tìm tất cả tập lợi ích cao theo chiều sâu với tiền tố C bằng cách gọi hàm ***Search-HU({C},k,$IT_{\{C\}}$)***.

Bảng 7. Bảng lợi ích các giao dịch

Tid	C	E	B	A	F
1	2	1	0	1	1
2	25	0	1	0	0
3	0	2	0	0	1
4	12	0	1	0	0
5	8	2	0	2	0
6	4	0	0	0	1
7	2	0	0	0	0
8	0	2	2	3	3
9	0	0	0	2	0
10	4	2	0	0	0

Bảng 8. Bảng TC_1 sau khi cập nhật lại CWU

Itemsets	A	B	C	D	E	F
CWU	87	102	115	50	107	75
AU	24	40	57	24	45	12

Bảng 9. Bảng UT_C của phần tử C

Tid	U(C,Tj)
1	1*2 = 2
2	1*25 = 25
4	1*12 = 12
5	1*8 = 8
6	1*4 = 4
7	1*2 = 2
10	1*4 = 4

Bảng 10. Bảng chỉ số IT_C của phần tử C

Tid	Vị trí cuối	U(C,Tj)
1	1	1*2 = 2
2	1	1*25 = 25
4	1	1*12 = 12
5	1	1*8 = 8
6	1	1*4 = 4
7	1	1*2 = 2
10	1	1*4 = 4

Bước 5, duyệt từng bộ (j,U) ∈ UT_i.

5.1. Cập nhật $TU(T_j) = TU(T_j) – U(i,T_j)$. Giả sử với UT_C thì giá trị TU của các giao dịch 1, 2, 4, 5, 6, 7, 10 sẽ giảm đi tương ứng là {2, 25, 12, 8, 4, 2, 4}. Kết quả TU lần lượt của từng giao dịch từ 1 đến 10 là {10, 10, 12, 10, 16, 2, 0, 45, 6, 10}.

Bước 6, hiển thị danh sách tập lợi ích cao trong HUs.

Hàm Search-HU({C},k,IT_C)

//Xây dựng bảng TC_2 với C là tiền tố dựa trên bảng IT_C

Bước 1, TC_2={};

Bước 2, Với mỗi bộ (j,p) trong IT_C thực hiện. Giả sử với bộ (1,1) – trong giao dịch 1 và vị trí 1.

2.1. Với mỗi phần tử i_{p+1} đứng sau vị trí p trong giao dịch T_j thực hiện. Giả sử với phần tử E.

2.1.1. Ta có, E ∈ $HCWU_1$ nên tạo tập {CE} = C ∪ E;

2.1.2. Ta có, {CE} ∉ TC_2 nên đưa: {CE}; $TU(T_1)$=12; $U(\{C\},T_1) + U(E,T_1) = 2 + 5*1 =7$ vào TC_2(Itemsets, CWU, AU).

Lặp lại *Bước 2.1*, với hai phần tử A, F sau vị trí 1 trong giao dịch 1 được $HCWU_2$ = ({CE}, {CA}, {CF}) và Bảng TC_2 như Bảng 11.

Lặp lại *Bước 2*, với bộ (2,1) – trong giao dịch 2, sau vị trí 1 có phần tử B ∈ $HCWU_1$ nên tạo tập {CB} = {C} ∪ B.

Ta thấy, {CB} ∉ TC_2 nên đưa: {CB}; $TU(T_2) = 35$; $U(\{C\},T_2) + U(B,T_2) = 25 + 10*1 = 35$ vào TC_2(Itemsets, CWU, AU). Kết quả như Bảng 12.

Lặp lại *Bước 2*, với bộ (4,1) - trong giao dịch 4, sau vị trí 1 có phần tử B ∈ $HCWU_1$ nên tạo tập {CB} = {C} ∪ B.

2.1.3. Vì {CB} ∈ TC_2 nên chỉ cập nhật giá trị CWU và AU của {CB} trong Bảng 12 như sau:

- $CWU(\{CB\}) = CWU(\{CB\}) + TU(T_4) = 35 + 22 = 57$;
- $AU(\{CB\}) = AU(\{CB\}) + U(\{C\},T_4) + U(B,T_4) = 35 + 22 = 57$.

Tương tự, lặp lại *Bước 2* với các bộ (5,1), (6,1), (7,1), (10,1) ta được kết quả bảng TC_2 với tiền tố C kết quả như Bảng 13.

Bảng 11. Bảng TC_2 với tiền tố C

Itemsets	CWU	AU
CE	12	7
CA	12	5
CF	12	4

Bảng 12. Bảng TC_2 với tiền tố C

Itemsets	CWU	AU
CE	12	7
CA	12	5
CF	12	4
CB	35	35

Bảng 13. Bảng TC_2 với tiền tố C

Itemsets	CWU	AU
CE	50	39
CA	36	19
CF	18	10
CB	57	57

Bảng 14. Bảng TC_1 sau khi cập nhật lại CWU

Itemsets	A	B	C	D	E	F
CWU	77	65	115	50	93	69
AU	24	40	57	24	45	12

Bảng 15. Bảng chỉ số $IT_{\{CB\}}$ của tập {CB}

Tid	Vị trí cuối	U({CB},Tj)
2	2	35
4	2	22

2.1.4. Nếu k = 1 thì $CWU(i_{p+1}) = CWU(i_{p+1}) – U(i_p,T_j)$. Ví dụ, phía sau phần tử C trong giao dịch 1 có các phần tử E, A, F và CWU của các phần tử này đều giảm đi $U(\{C\},T_1) = 2$. Vậy sau khi duyệt bộ (1,1) thì giá trị CWU tương ứng của E, A, F là 105, 85, 73. Tương tự, sau khi duyệt các bộ: (2,1), (4,1), (5,1), (6,1), (7,1), (10,1) thì giá trị CWU trong bảng TC_1 có kết quả như Bảng 14.

Bước 3, duyệt từng tập X'∈ TC_2.

3.1. Chỉ có CWU({CB}) = 57> 56 nên $HCWU_2$ = ({CB}).

3.2. Chỉ có AU({CB}) = 57 > 56 nên HUs = {C:57, CB:57}.

Bước 4, với mỗi X' ∈ $HCWU_2$

4.1. Xây dựng bảng $IT_{\{CB\}}$ từ bảng IT_C được kết quả như Bảng 15.

4.2. k = k + 1; //k = 1 +1 = 2

4.3. Gọi hàm ***Search-HU({CB}, k, $IT_{\{CB\}}$)*** để tìm tất cả tập lợi ích cao với tiền tố {CB}.

Khi gọi hàm ***Search-HU({CB},k,$IT_{\{CB\}}$)*** để tìm tập lợi ích cao gồm 3 phần tử bằng cách duyệt 2 bộ (2,2) và (4,2) trong bảng $IT_{\{CB\}}$ để sinh ứng viên gồm 3 phần tử. Nhưng vị trí 2 trong giao dịch 2 và 4 là vị trí cuối nên không có tập ứng viên gồm 3 phần tử nào được sinh ra. Vậy, sau khi tìm kiếm tập lợi ích cao với tiền tố C được HUs = {C, CB}. Và kết thúc hàm ***Search-HU({C},1,$IT_{\{C\}}$)***.

Lặp lại bước 4 trong chương trình chính để tìm tập lợi ích cao với tiền tố là các phần tử E, B, A, F trong $HCWU_1$ bằng cách gọi hàm ***Search-HU*** giống như phần tử C ở trên.

Bảng 16. So sánh giá trị CWU và TWU

Tiền tố	CWU	TWU
C	{C}: 115, {CB}: 57, {CF}: 18, {CA}: 36, {CE}:50	{C}: 115, {CB}: 57, {CF}: 18, {CA}: 36, {CE}: 50
E	{E}:93, {EB}:45, {EF}: 67, {EA}:71, {EAF}: 55	{E}: 107, {EB}: 45, {EF}: 69, {EA}: 81, {EAF}: 57
B		{B}: 102, {BF}: 45, {BA}: 45
A		{A}: 87, {AF}: 57
F		{F}: 75

Kết thúc quá trình khai phá với ngưỡng lợi ích tối thiểu α = 56 ta thu được tập lợi ích cao HUs= {C:57, CB:57}.

Bảng 16 cho ta thấy sử dụng mô hình CWU hiệu quả hơn mô hình TWU trong việc cắt tỉa các ứng viên. Mô hình CWU sinh ra 10 ứng viên còn mô hình TWU sinh ra 16 ứng viên.

V. KẾT QUẢ ĐÁNH GIÁ

Trong phần này, chúng tôi so sánh kết quả thực hiện thuật toán HP của chúng tôi với thuật toán TP [3], PB [8]. Đầu tiên chúng tôi giới thiệu về dữ liệu dùng để thử nghiệm. Tiếp theo là kết quả thực hiện và số lượng các tập ứng viên.

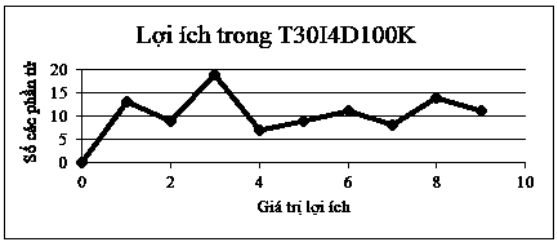
V.1. Môi trường và dữ liệu

Thuật toán được thực hiện trên máy tính IBM core 2 due 2.4GHz với 2 GB bộ nhớ, chạy trên Windows 7. Chương trình chúng tôi viết bằng Visual C++ 2010. Dữ liệu thử nghiệm gồm: Mushroom [13] và T30I4D100K được sinh từ bộ sinh dữ liệu của IBM [12]. Đặc điểm của bộ dữ liệu được mô tả phía dưới:

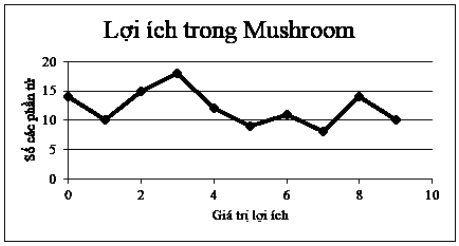
Database	T	D	N
T30I4D100K	30	100.000	100
Mushroom	23	8.124	119

Trong đó: T – là số phần tử trung bình trong một giao dịch; N – là số phần tử khác nhau; D – số giao dịch.

Các bộ dữ liệu này đều chưa có giá trị lợi ích ngoài cho từng phần tử và trong các giao dịch chỉ cho biết phần tử xuất hiện. Do vậy, chúng tôi sinh ngẫu nhiên số lượng cho mỗi phần tử trong mỗi giao dịch với giá trị thuộc từ 1 đến 5 và lợi ích ngoài của mỗi phần tử từ 0.1 đến 10. Hình 1a cho biết việc phân bổ lợi ích ngoài của các phần tử trong T30I4D100K. Hình 1b cho biết việc phân bổ lợi ích ngoài của các phần tử trong Mushroom.



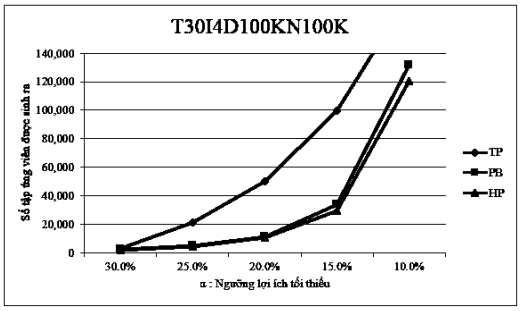
Hình 1a. Biểu đồ phân bố lợi ích ngoài của các phần tử trong T30I4D100K



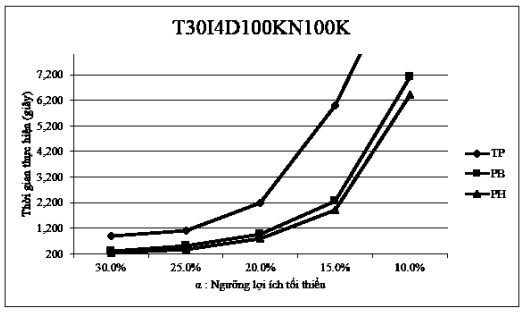
Hình 1b. Biểu đồ phân bố lợi ích ngoài của các phần tử trong Mushroom

V.2. Thời gian thực hiện và số ứng viên

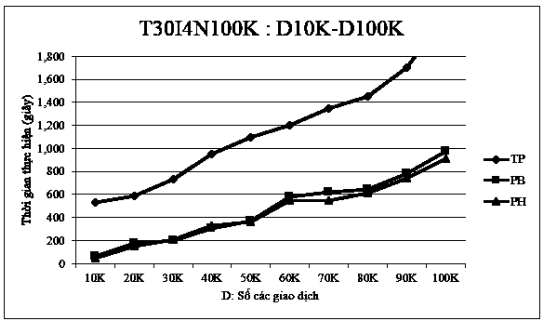
Kết quả thử nghiệm, so sánh giữa thuật toán HP với các thuật toán TP, PB trên bộ dữ liệu T30I4D100K thể hiện trong Hình 2a, 2b, 2c. Cụ thể, Hình 2a so sánh số lượng ứng viên tương ứng với các ngưỡng lợi ích tối thiểu khác nhau, Hình 2b so sánh thời gian thực hiện khai phá khi thay đổi ngưỡng lợi ích tối thiểu, Hình 2c so sánh thời gian thực hiện khai phá khi cố định ngưỡng lợi ích tối thiểu là 20% và thay đổi số lượng giao dịch. Hình 3a, 3b so sánh số ứng viên sinh ra và thời gian thực hiện khai phá giữa các thuật toán tương ứng với các ngưỡng lợi ích tối thiểu khác nhau trên bộ dữ liệu Mushroom.



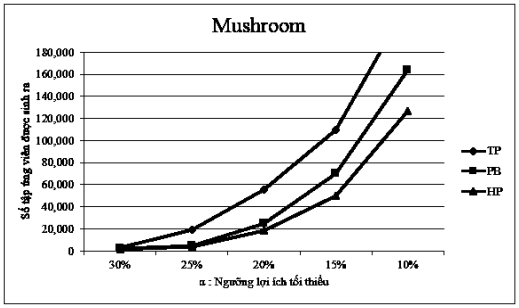
Hình 2a. So sánh số lượng ứng viên được sinh ra với ngưỡng lợi ích khác nhau



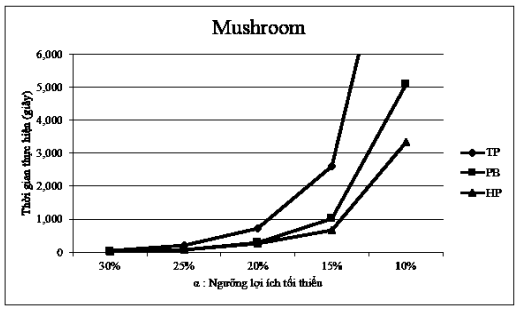
Hình 2b. So sánh thời gian thực hiện với ngưỡng lợi ích khác nhau



Hình 2c. So sánh thời gian thực hiện theo số lượng giao dịch khác nhau với ngưỡng α=20%



Hình 3a. So sánh số lượng ứng viên được sinh ra với ngưỡng lợi ích khác nhau



Hình 3b. So sánh thời gian thực hiện với ngưỡng lợi ích khác nhau

VI. KẾT LUẬN

Trong bài báo này chúng tôi đã phân tích ưu, nhược điểm của một số thuật toán khai phá tập lợi ích cao được đề xuất trong những năm gần đây, phân tích ưu, nhược điểm của mô hình TWU nhằm loại bớt tập ứng viên. Trên cơ sở đó, chúng tôi đã đề xuất mô hình lợi ích ứng viên có trọng số – CWU (Candidate Weight Utility). Kết quả thử nghiệm cho thấy mô hình CWU sinh ra số lượng ứng viên ít hơn mô hình TWU.

Chúng tôi đã đề xuất thuật toán HP được cải tiến từ thuật toán PB [8] ở một số điểm: thực hiện khai phá theo thứ tự giảm dần lợi ích của phần tử, thay đổi cấu trúc của bảng chỉ số IT giúp tính toán nhanh các giá trị CWU và U, sử dụng mô hình CWU để giảm số lượng tập ứng viên. Thử nghiệm, so sánh thuật toán đề xuất với hai thuật toán TP và PB trên các bộ dữ liệu chuẩn.

Thời gian tiếp theo, chúng tôi sẽ thử nghiệm mô hình CWU trên một số thuật toán khác nhằm tiếp tục khẳng định ưu điểm của mô hình này.

TÀI LIỆU THAM KHẢO

[1] R. CHAN, Q. YANG, AND Y. SHEN, "*Mining high utility itemsets*". In Proc. of Third IEEE Int'l Conf. on Data Mining, pp. 19-26, 2003.

[2] R.AGRAWAL, AND R. SRI KANT, "*Fast algorithms for mining association rules in large databases*", Proc. 20th IntI. Conf. Very Large Data Bases (VLDB'94), pp. 487-499, September 1994.

[3] Y. LIU, W. LIAO, AND A. N. CHOUDHARY, "*A two-phase algorithm for fast discovery of high utility itemsets*", Proc. 9th Pacific-Asia Conf. Knowledge Discovery and Data Mining (PAKDD'05), pp. 689-695, May 2005.

[4] H. YAO, H.J. HAMILTON, AND C. BUTZ, "*A foundational approach to mining itemset utilities from databases*", Proc. 4th SIAM IntI. Conf. Data Mining (SDM'04), pp. 482-486, April 2004.

[5] WEI SONG, YU LIU, JINHONG LI, "*Vertical Mining for High Utility Itemsets*", IEEE International Conference on Granular Computing, 2012.

[6] C. F. AHMED, S. K. TANBEER, B.-S. JEONG, AND Y.-K. LEE, "*HUC-Prune: an efficient candidate pruning technique to mine high utility patterns*," Appl. Intell., vol. 34, pp.181-198, April 2011.

[7] LIU Y, LIAO W, CHOUDHARY A "*A fast high utility itemsets mining algorithm*". In: Proceeding of the utility-based data mining workshop, pp 90–99, 2005.

[8] GUO-CHENG LAN, TZUNG-PEI HONG, VINCENT S. TSENG, "*An efficient projection-based indexing approach for mining high utility itemsets*", Knowl Inf Syst (2014) 38:85–107, Springer-Verlag London 2013.

[9] VINCENT S. TSENG, CHENG-WEI WU, BAI-EN SHIE, PHILIP S.YU, "*UP-Growth: An Efficient Algorithm for High Utility Itemset Mining*", KDD'10, July 25–28, Washington, DC, USA, 2010.

[10] A.ERWIN, R. P. GOPALAN, AND N. R. ACHUTHAN, "*CTU-Mine: an efficient high utility item set mining algorithm using the pattern growth approach*," Proc. 7th IEEE IntI. Conf. Computer and Information Technology (CIT'07), pp. 71-76, October 2007.

[11] C. F. AHMED, S. K. TANBEER, B.-S. JEONG, AND Y.-K. LEE, "*HUC-Prune: an efficient candidate pruning technique to mine high utility patterns*". Appl. Intell., vol. 34, pp. 181-198, April 2011.

[12] IBM Quest Data Mining Project, Quest Synthetic Data Generation Code. Available at (http:// www.almaden.ibm.com/cs/quest/syndata.html)

[13] https://archive.ics.uci.edu/ml/datasets

Nhận bài ngày: 1/10/2014

SƠ LƯỢC VỀ TÁC GIẢ

ĐẬU HẢI PHONG

Sinh năm: 1977

Tốt nghiệp đại học về Hệ thống thông tin quản lý năm 2000 tại Đại học Thăng Long, CNTT năm 2006 tại HVKTQS. Nhận bằng thạc sỹ CNTT 2008 tại Học viện Kỹ thuật Quân sự. Hiện nay đang công tác tại Khoa Toán và Tin học – trường Đại học Thăng Long.

Lĩnh vực nghiên cứu: Khai phá dữ liệu, Tính toán song song, Cơ sở dữ liệu phân tán.

Email: phong4u@gmail.com; ĐT: 0912.441.435

NGUYỄN MẠNH HÙNG

Sinh năm 1974.

Tốt nghiệp đại học về CNTT năm 1998 tại Học viện Kỹ thuật Quân sự. Bảo vệ luận án Tiến sỹ năm 2004 tại Trung tâm tính toán – Viện hàn lâm khoa học Liên bang Nga.

Hiện nay đang công tác tại Phòng Sau đại học – Học viện Kỹ thuật Quân sự.

Lĩnh vực nghiên cứu: cấu trúc dữ liệu hiện đại, khai phá dữ liệu, phân loại gói tin hiệu năng cao.

Email: manhhungk12@mta.edu.vn; ĐT: 0989.146.397