Phương pháp song song khai phá tập lợi ích cao dựa trên chỉ số hình chiếu

Parallel Method for Mining High Utility Itemsets from Projection-Based Indexing

Đậu Hải Phong, Nguyễn Mạnh Hùng

Abstract: *High utility itemsets (HUIs) mining is one of popular problems in data mining. Several parallel and sequential algorithms have been proposed in the literature to solve this problem. All the parallel algorithms to try reduce synchronization cost and caculation global profit of itemsets. In this paper, we present a parallel method for mining HUIs from projection-based indexing to speed up performance and reduce memory requirements. The experimental results show that the performance and number candidate of our algorithm is better than some non parallel algorithms.*

Keywords: *Data Mining, Parallel Mining, Shared Memory, High Utility, Projection index, PPB-Miner algorithm.*

I. GIỚI THIỆU

Ngày nay, với sự phát triển nhanh chóng của các kỹ thuật về cơ sở dữ liệu đã tạo điều kiện cho việc lưu trữ và sử dụng dữ liệu lớn trong kinh doanh, y tế, giáo dục, các tổ chức khoa học, chính phủ,… Một trong những chủ đề quan trọng trong các nghiên cứu về khai phá dữ liệu gần đây là tìm kiếm những tập mục lợi ích cao từ cơ sở dữ liệu giao dịch. Mục tiêu là trích xuất các thông tin hữu ích từ dữ liệu có quan tâm đến lợi ích, số lượng, chi phí,… của từng phần tử. Đã có các nghiên cứu được đề xuất để khai phá tập lợi ích cao [1]–[6],… Tuy nhiên, các thuật toán chủ yếu đều thực hiện khai phá tuần tự. Vấn đề đặt ra là khi dữ liệu lớn, các thuật toán tuần tự sẽ khó đáp ứng về mặt thời gian thực hiện và không gian lưu trữ.

Trong khai phá tập lợi ích cao có một số thách thức sau: Thứ nhất, với khối lượng dữ liệu lớn thì không gian tìm kiếm lớn và vấn đề về sự hợp nhất. Thứ hai, tập lợi ích cao không có tính chất đóng [7]. 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 CSDL để kiểm tra các ứng viên như trong một số thuật toán [2], [8], [9] …hoặc tiêu tốn nhiều thời gian và không gian bộ nhớ để sinh ra các cây điều kiện [10], [11], [12],…Thứ ba, với khối lượng dữ liệu lớn thì giới hạn về thời gian tính toán và yêu cầu về bộ nhớ trên một máy tính là không đáp ứng được. Do đó, việc thiết kế các thuật toán dựa trên kiến trúc song song là cần thiết.

Trong bài báo này chúng tôi xây dựng thuật toán song song PPB-Miner để khai phá tập lợi ích cao với một số đóng góp sau:

- Dùng bảng chỉ số để tăng tốc độ thực hiện và giảm yêu cầu bộ nhớ. Từ bảng chỉ số của các tập phần tử, sinh các ứng viên, tìm tập lợi ích cao và tạo nhanh bảng chỉ số từ tập tiền tố của nó.

- Sử dụng cấu trúc danh sách lợi ích (utility-list) để loại nhanh các ứng viên và độc lập xử lý các phần tử trên từng bộ xử lý.

- Tối ưu lưu trữ giá trị để tính danh sách lợi ích.

- Xây dựng thuật toán song song khai phá tập lợi ích cao trên mô hình chia sẻ bộ nhớ.

Nội dung tiếp theo của bài báo được tổ chức như sau: phần II trình bày một số khái niệm và định nghĩa. Các vấn đề liên quan đến khai phá tập lợi ích cao được trình bày trong phần III. Phần IV đề xuất

thuật toán PPB-Miner. Phần V trình bày kết quả đạt được và so sánh với các thuật toán khác. Cuối cùng là kết luận.

II. KHÁI NIỆM VÀ ĐỊNH NGHĨA

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 Bảng 1. Cơ sở dữ liệu giao dịch và Bảng 2. Bảng lợi ích ngoài của các phần tử.

Bảng 1. Cơ sở dữ liệu giao dịch

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 ngoài của các phần tử

Item	A	B	C	D	E	F
Lợi ích	3	10	1	6	5	2

Định nghĩa 1 [2] - 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 [2] - 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 [2] - 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 [2] - 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 [2] - 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 [2]– Tập phần tử lợi ích cao: Tập X được gọi là tập phần tử lợi ích cao (HUI – High Utility Itemsets) nếu $AU(X) \geq$ minutil, ngược lại gọi X là tập phần tử lợi ích thấp. Trong đó minutil là ngưỡng lợi ích tối thiểu cho trước.

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

Định nghĩa 7 [2] - 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 [2] - 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: $TWU(\{AC\}) = TU(T_1) + TU(T_5) = 18 + 24 = 42$.

Định nghĩa 9 [2] – Cho một tập phần tử X và một giao dịch T, sao cho $X \subseteq T$ thì tập hợp tất cả các phần tử đứng sau X trong T kí hiệu là T\X.

Ví dụ, trong Bảng 1 thì T1\{AC} = {DEF}.

Định nghĩa 10 [2] – Lợi ích còn lại của tập phần tử X trong giao dịch T, kí hiệu : RU(X,T) là tổng lợi ích của các phần tử trong T\X trong giao dịch T, và RU(X,T) = = $\sum_{i \in (T\setminus X)} U(i, T)$.

Định nghĩa 11 – Tổng lợi ích còn lại của một tập phần tử X trong cơ sở dữ liệu là tổng lợi ích còn lại của tập phần tử X trong tất cả giao dịch chứa X. Ký hiệu: SRU(X) = $\sum_{T_j \in D \wedge X \subseteq T_j} RU(X, T_j)$.

Định nghĩa 12 [4] – Cấu trúc utility-list của tập phần tử: utility-list của tập phần tử X bao gồm 3 trường: tid, iutil, rutil. Trong đó:

- Tid là chỉ số của giao dịch chứa X;

- iutil là lợi ích của X trong Tid, tức là U(X, Tid);

- rutil là lợi ích còn lại của X trong Tid, tức là RU(X, Tid);

Định lý 1 [4] – Cho một utility-list của tập phần tử X, nếu tổng X.iutils và X.rutils nhỏ hơn ngưỡng lợi ích tối thiểu (minutil) thì lợi ích của các tập phần tử mở rộng từ tập phần tử X cũng nhỏ hơn lợi ích tối thiểu.

III. VẤN ĐỀ LIÊN QUAN

Trong phần này, chúng tôi trình bày một số nghiên cứu liên quan đến thuật toán khai phá tập lợi ích cao.

III.1. Thuật toán tuần tự khai phá tập lợi ích cao

Năm 2005, Ying Liu đã đưa ra thuật toán hai pha (two-phase) để khai phá nhanh tập lợi ích cao [3]. Pha một, tìm tất cả các tập ứng viên có TWU lớn hơn ngưỡng minutil. Pha hai, với mỗi tập ứng viên tính toán chính xác lợi ích của tập đó. Với thuật toán này đòi hỏi duyệt dữ liệu nhiều lần và sinh ra nhiều ứng viên.

Năm 2010, thuật toán sử dụng cấu trúc cây mẫu lợi ích [11] được Vincent và cộng sự giới thiệu. Thuật toán gồm 3 bước sau: bước 1, xây dựng cây mẫu lợi ích (UP-tree); bước 2, sinh các tập tiềm năng từ UP-tree bằng thuật toán UP-Growth; bước 3, xác định các tập lợi ích cao từ tập tiềm năng. Thuật toán này yêu cầu phức tạp trong xây dựng và duyệt cây nhiều lần.

Năm 2012, Mengchi Liu giới thiệu thuật toán HUI-Miner [4] khai phá tập lợi ích cao nhưng không sinh tập ứng viên. Thuật toán sử dụng cấu trúc utility-list để loại nhanh tập ứng viên và không cần duyệt dữ liệu nhiều lần. Nhưng nhược điểm của thuật toán này là chi phí kết hợp các tập lợi ích cao là tương đối lớn.

Thuật toán UDepth [9] 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 [2] 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. Nhược điểm của thuật toán này là vẫn sử dụng mô hình TWU làm ngưỡng trên để cắt tỉa các tập ứng viên và do mô hình này tạo ra ngưỡng cao dẫn đến số lượng ứng viên được sinh ra lớn làm tốn nhiều chi phí kiểm tra ứng viên.

Năm 2015, trong [13] các tác giả đã đề xuất mô hình CWU để loại bỏ tập ứng viên. Đây là mô hình tương đối hiệu quả trong các thuật toán khai phá tập lợi ích cao theo chiều sâu như [2], [9], [12], v.v..

III.2. Thuật toán song song khai phá tập lợi ích cao

Năm 2008, A. Erwin [14] đã đề xuất thuật toán sử dụng mô hình TWU với tăng trưởng mẫu dựa trên cấu trúc dữ liệu cây mẫu lợi ích nén (CTU-tree). Thuật toán đã song song lược đồ chiếu (projection scheme) để lưu trữ trên đĩa khi bộ nhớ chính không đủ do dữ liệu quá lớn. Kết quả thực nghiệm chỉ ra thuật toán thực hiện hiệu quả với dữ liệu lớn, dày và có tập mẫu lớn.

Năm 2009, B. Vo [15] và nhóm tác giả đã đề xuất một phương pháp song song khai phá tập lợi ích cao với dữ liệu được phân chia theo chiều dọc, sử dụng

cấu trúc cây WIT để lưu trữ dữ liệu cục bộ trên mỗi bộ xử lý. Các phần tử trên mỗi SlaverSite chỉ gửi về MasterSite nếu TWU của nó lớn hơn ngưỡng tối thiểu và MasterSite chỉ kết hợp để khai phá tập lợi ích cao nếu các tập đó xuất hiện ở hai SlaverSite khác nhau.

Năm 2013, Kannimuthu [5] và cộng sự đã trình bày thuật toán FUI khai phá tập lợi ích cao, các công việc được phân chia theo cách tiếp cận một master và nhiều slave. Các giá trị lợi ích được trích xuất song song trên các slave. Tổng lợi ích sẽ được tính ở master. Kết quả thực nghiệm cho thấy, thời gian thực thi nhanh hơn so với các thuật toán trước.

IV. ĐỀ XUẤT THUẬT TOÁN

Trong phần này, chúng tôi trình bày thuật toán song song PPB-Miner khai phá tập lợi ích cao dựa trên bảng chỉ số (IT) và bảng ứng viên (TC) nhằm tính nhanh giá trị AU và SRU của các tập phần tử trong quá trình khai phá. Áp dụng định lý 1, sử dụng tổng iutils và rutils tương ứng với AU và SRU để tỉa ứng viên.

Để tiết kiệm bộ nhớ nhưng vẫn tính được danh sách lợi ích, với mỗi phần tử i_k trong giao dịch T_j, chúng tôi lưu trữ thêm đại lượng UR. Trong đó, $UR(i_k,T_j) = U(i_k,T_j) + RU(i_k,T_j)$.

Với cách tổ chức dữ liệu như trên có thể tính $U(i_k,T_j) = UR(i_k,T_j) - UR(i_{k+1},T_j)$ và $RU(i_k,T_j) = UR(i_{k+1},T_j)$. Trong đó, i_{k+1} là phần tử ngay phía sau i_k. Bằng cách lưu trữ này, ta có thể vừa tiết kiệm bộ nhớ không cần lưu cả iutil và rutil.

Ví dụ, từ cơ sở dữ liệu minh họa ở Bảng 1 với minutil = 56. Với lần duyệt dữ liệu lần đầu ta tính được AU và TWU của từng phần tử được kết quả như trong Bảng 3. Từ Bảng 3 loại D (vì TWU(D) = 50 < 56) và sắp xếp giảm dần theo AU được tập $HTWU_1$.

Bảng 3. Kết quả tính TWU và AU

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

Ta loại D khỏi các giao dịch. Sau đó tiến hành sắp xếp các phần tử trên mỗi giao dịch giảm dần theo AU có thứ tự lần lượt là C:57, E:45, B:40, A:24, F:12 ta được Bảng 4.

Bảng 4. Lợi ích UR của các phần tử trong từng giao dịch

Tid	Giao dịch
1	(C,12), (E,10), (A,5), (F,2)
2	(C,35), (B,10),
3	(E,12), (F,2)
4	(C,22), (B,10)
5	(C,24), (E,16), (A,6)
6	(C,6), (F,2)
7	(C,2)
8	(E,45), (B,35), (A,15), (F,6)
9	(A,6)
10	(C,14), (E,10)

Một số cấu trúc được sử dụng trong thuật toán PPB-Miner gồm:

- Bảng tập ứng viên TC_k có k-phần tử với tiền tố là tập X, mỗi tập phần tử chứa: lợi ích thực tế AU(X) và tổng lợi ích còn lại SRU(X) tương ứng.

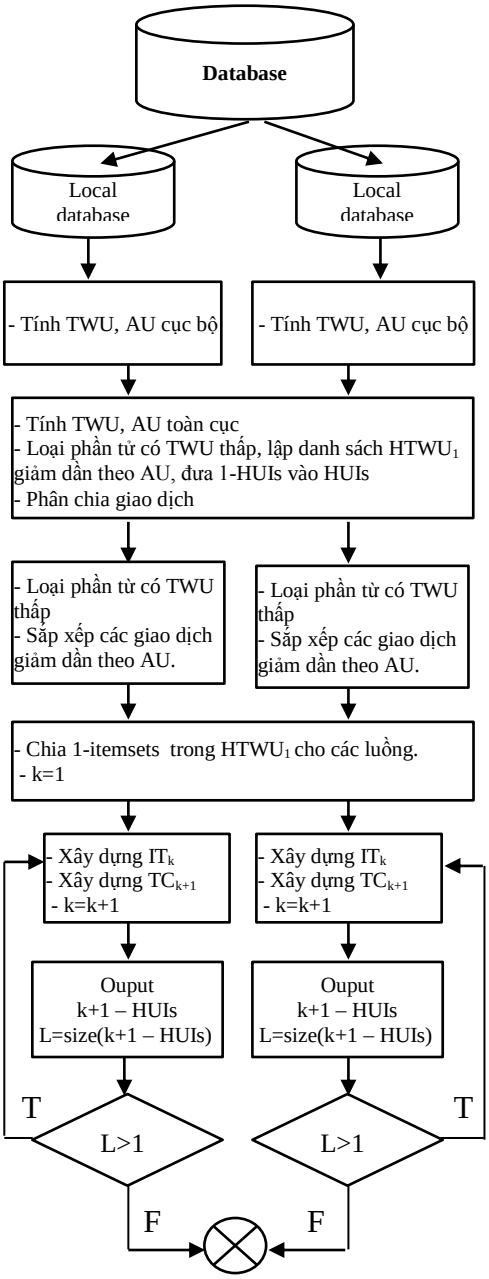
Ví dụ, trong Bảng 5 gồm các tập ứng viên có 2 phần tử với tiền tố {C}.

Bảng 5. Tập ứng viên TC_2 với tiền tố {C}

Itemsets	AU	SRU
CE	39	11
CA	19	2
CF	10	0
CB	57	0

- 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; $U(X,T_j)$ – giá trị lợi ích của tập X trong giao dịch T_j; $RU(X,T_j)$ – giá trị lợi ích các phần tử còn lại sau tập X trong giao dịch T_j. Ví dụ, từ Bảng 4 ta xây dựng được bảng chỉ số IT_C của tập {C} trong Bảng 6 như sau: $U(\{C\},T_1) = UR(\{C\},T_1) - UR(\{E\},T_1) = 12 - 10 = 2$; $RU(\{C\},T_1) = UR(\{E\},T_1)$; tương tự với các giao dịch 2, 4, 5, 6, 7, 10. Với một thứ tự đã được sắp xếp trong giao dịch thì từ bảng IT_C có thể xác định nhanh tập các ứng viên 2-phần tử bằng cách kết hợp C với từng phần tử sau C trong từng giao dịch. Ví

dụ, với giao dịch 5 và sau vị trí 1 sinh được tập các ứng viên {CE}, {CA} và có thể tính nhanh $U(\{CE\},T_5) = U(\{C\},T_5) + (UR(\{E\},T_5) - UR(\{A\},T_5)) = 8 + (16-6) = 18$ và $RU(\{CE\},T_5) = UR(\{A\},T_5) = 6$. Tương tự, $U(\{CA\},T_5) = U(\{C\},T_5) + (UR(\{A\},T_5) - 0) = 8 + (6 - 0) = 14$ và $RU(\{CA\},T_5) = 0$ vì A là phần tử cuối cùng trong giao dịch 5.



Hình 1. Thuật toán PPB-Miner

Giả sử với hai luồng xử lý, thuật toán PPB-Miner được mô tả trong Hình 1.

Bảng 6. Chỉ số IT_C của tập {C}

Tid	Vị trí cuối	$U(\{C\},T_j)$	$RU(\{C\},T_j)$
1	1	2	10
2	1	25	10
4	1	12	10
5	1	8	16
6	1	4	2
7	1	2	0
10	1	4	10

IV.1. Mô tả thuật toán PPB-Miner

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

OUTPUT: Tất cả các tập lợi ích cao

Công việc của Master:

1. Phân chia các giao dịch cho các luồng theo phương pháp động sử dụng thư viện OpenMP

2. Đợi các luồng tính TWU, AU cục bộ xong thì thực hiện:

2.1. Tính TWU, AU toàn cục

2.2. Từ tập I, loại các phần tử có TWU nhỏ hơn minutil, lập danh sách $HTWU_1$ với các phần tử giảm dần theo AU; đưa 1-HUIs vào tập HUIs;

2.3. Phân chia các giao dịch cho các luồng và đợi các luồng thực hiện xong việc loại các phần tử có TWU nhỏ hơn minutil và sắp xếp các phần tử trong giao dịch giảm dần theo AU;

4. Phân chia từng phần tử trong $HTWU_1$ cho các luồng để khai phá HUIs.

Công việc của từng luồng (Threads):

1. Nhận dữ liệu từ Master để thực hiện tính TWU và AU cục bộ;

2. Nhận giao dịch từ Master và thực hiện loại các phần tử có TWU nhỏ hơn minutil và sắp xếp các phần tử trong giao dịch giảm dần theo AU

3. Nhận phần tử i trong $HTWU_1$ từ Master thực hiện:

3.1. k=1;

3.2. X = i;

3.3. Xây dựng bảng IT_1 của những phần tử i;

3.4. Gọi hàm ***PB-Miner(X,k,IT$_X$)*** để khai phá tập các HUIs;

//Hàm PB-Miner

Hàm PB-Miner(X,k,IT$_{\{x\}}$)

INPUT*:* X – tập phần tử tiền tố; k – số phần tử trong tập; IT_X – bảng chỉ số của tập X; $HTWU_1$.

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) ∈ $IT_{\{X\}}${

//p là phần tử cuối cùng của tập X

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

2.1.1: If ($i_{p+1} \in HTWU_k$) {X' = X ∪ i_{p+1}};

2.1.2: If (X' ∉ TC_{k+1}) {

- Chèn (X', U(X,T_j) + (UR(i_{p+1},T_j) - UR(i_{p+2},T_j), UR(i_{p+2},T_j)) vào bảng TC_{k+1} (Itemsets, AU, SRU).

//Chú ý, nếu i_{p+2} là cuối thì UR(i_{p+2},T_j) = 0;

}

2.1.3: If (X' ∈ TC_{k+1}) {

- SRU(X') = SRU(X') + UR(i_{p+2},T_j);
- AU(X') = AU(X') + U(X,T_j) + (UR(i_{p+1},T_j) - UR(i_{p+2},T_j);

}

}

3: For X' ∈ TC_{k+1} {

3.1: If (AU(X') + SRU(X') ≥ minutil) {

Chèn X' vào tập $HTWU_{k+1}$;

}

3.2: If (AU(X') ≥ minutil){

Chèn X' vào tập HUIs;

}

4: For (X' ∈ $HTWU_{k+1}$){

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

4.2: k = k +1;

4.3: **PB-Miner (X', k, IT$_{X'}$);**

//tìm tập lợi ích cao theo chiều sâu với tiền tố là tập {X'}

}

5: Return HUIs;

IV.2. Ví dụ minh họa

Trong phần này chúng tôi sẽ minh họa các bước của thuật toán với hai luồng xử lý. Cơ sở dữ liệu giao dịch, bảng lợi ích ngoài tương ứng ở Bảng 1 và Bảng 2, ngưỡng lợi ích tối thiểu minutil = 56.

Công việc của Master:

Bước 1, Master thực hiện phân chia các giao dịch cho các luồng;

Bước 2, Sau khi nhận TWU, AU cục bộ từ các luồng và thực hiện:

Bước 2.1, Tính TWU, AU toàn cục được kết quả như Bảng 7.

Bảng 7. Kết quả TWU và AU toàn cục

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

Bảng 8. Bảng $HTWU_1$

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

Bước 2.2, Từ Bảng 7, loại D vì TWU(D) = 50 < 56 và sắp xếp giảm dần theo AU được $HTWU_1$. Kết quả như Bảng 8.

Từ Bảng 7 ta có AU(C) = 57 > 56 nên đưa C và HUIs ta được HUIs = {C :57};

Bước 2.3, Phân chia các giao dịch cho các luồng, giả sử luồng 1 phụ trách từ giao dịch 1 đến giao dịch 5; luồng 2: phụ trách từ giao dịch 6 đến giao dịch 10. Đợi luồng 1 và luồng 2 loại các phần tử có TWU nhỏ hơn 56 và sắp xếp các phần tử trong giao dịch giảm dần theo AU xong.

Bước 3, Phân công: luồng 1 phụ trách khai phá HUIs với tiền tố: C, B, F; luồng 2: phụ trách khai phá HUIs với tiền tố: E, A.

Công việc của các luồng (Threads):

Bước 1, Tính TWU và AU cục bộ;

Bước 2, Đợi Master tính xong TWU, AU toàn cục và nhận phân chia giao dịch và thực hiện loại bỏ D trong các giao dịch và sắp xếp lại các phần tử trong từng giao dịch giảm dần theo AU. Kết quả như Bảng 4.

Bước 3, Giả sử phân công như sau: luồng 1 phụ trách khai phá HUIs với tiền tố: C, B, F; luồng 2: phụ trách khai phá HUIs với tiền tố: E, A.

Giả sử thực hiện trên luồng 1 với phần tử C:

Bước 3.1, k=1;

Bước 3.2, X={C}

Bước 3.3, Xây dựng được bảng IT_C như sau: trong giao dịch 1 ta có $U(C,T_1)$ =$UR(C,T_1)$ – $UR(E,T_1)$ = 12 – 10 = 2; $RU(C,T_1)$ = $UR(E,T_1)$ = 10. Tương tự cho các giao dịch 2, 3, 5, 6, 7, 10. Kết quả như Bảng 9.

Bảng 9. Bảng chỉ số IT_C của tiền tố C

Tid	Vị trí cuối	$U(C,T_j)$	$RU(C,T_j)$
1	1	12 - 10 = 2	10
2	1	35 - 10 = 25	10
4	1	22 - 10 = 12	10
5	1	24 - 16 = 8	16
6	1	6 - 2 = 4	2
7	1	2 - 0 = 2	0
10	1	14 - 10 = 4	10

Bước 3.4, Luồng 1 gọi hàm ***PB-Miner({C},1,IT_C)*** để khai phá các tập HUIs

Hàm PB-Miner({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, $E \in HCWU_1$ nên tạo tập {CE} = C ∪ E;

2.1.2. Ta có, {CE} ∉ TC_2 nên đưa tương ứng: Itemsets = {CE}; $U(\{CE\},T_1)$ = $U(\{C\},T_1)$ + ($UR(E,T_1)$ - $UR(A,T_1)$) = 2 + (10 - 5) = 7; $RU(\{CE\},T_1)$ = $UR(A,T_1)$ = 5 vào TC_2(Itemsets, AU, SRU).

Lặp lại *Bước 2.1*, với hai phần tử A, F sau vị trí 1 trong giao dịch 1 Bảng TC_2 như Bảng 10.

Bảng 10. Bảng TC_2 với tiền tố C

Itemsets	AU	RU
CE	7	5
CA	5	2
CF	4	0

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 tương ứng: Itemsets = {CB}; $U(\{CB\},T_2)$ = $U(\{C\},T_1)$ + ($UR(B,T_2)$ - $UR(\emptyset,T_2)$) = 25 + (10 - 0) =35; $RU(\{CB\},T_2)$ = $UR(\emptyset,T_2)$ = 0 vào TC_2(Itemsets, AU, SRU). Kết quả như Bảng 11.

Bảng 11. Bảng TC_2 với tiền tố C

Itemsets	AU	SRU
CE	7	5
CA	5	2
CF	4	0
CB	35	0

Lặp lại *Bước 2*, với bộ (4, 1) - trong giao dịch 4, sau vị trí 1 có phần tử B ∈ $HTWU_1$ nên tạo tập {CB} = {C} ∪ B.

2.1.3. Vì {CB} ∈ TC_2 nên chỉ cập nhật giá trị AU và SRU của {CB} trong Bảng 9 như sau:

- $AU(\{CB\})$ = $AU(\{CB\})$ + $U(\{C\},T_4)$ + ($UR(B,T_4)$ - $UR(\emptyset,T_4)$) = 35 + 12 + (10 - 0) = 57 ;
- $SRU(\{CB\})$ = $SRU(\{CB\})$ + $RU(\emptyset,T_4)$ = 0 + 0 = 0.

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 12.

Bảng 12. Bảng TC_2 với tiền tố C

Itemsets	AU	SRU
CE	39	11
CA	19	2
CF	10	0
CB	57	0

Bước 3, duyệt từng tập X’∈ TC_2.

3.1. Chỉ có $AU(\{CB\})$ + $RU(\{CB\})$ = 57 + 0 = 57 > 56 nên $HTWU_2$ = ({CB}).

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

Bước 4, với mỗi X’ ∈ $HTWU_2$

4.1. Xây dựng bảng $IT_{\{CB\}}$ từ bảng IT_C được kết quả như Bảng 13.

Bảng 13. Bảng chỉ số $IT_{\{CB\}}$ của tập {CB}

Tid	Vị trí cuối	$U(\{CB\},Tj)$	$RU(\{CB\},Tj)$
2	2	35	0
4	2	22	0

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

4.3. Gọi hàm ***PB-Miner({CB}, k, $IT_{\{CB\}}$)*** để tìm tất cả tập lợi ích cao với tiền tố {CB} 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 ***PB-Miner({C},1,$IT_{\{C\}}$)***.

Bảng 14. So sánh số lượng ứng viên trên hai mô hình Utility-list và TWU

Luồng	Tiền tố	Utility-list	TWU
Luồng 1	C	{C}: 115, {CB}: 57, {CF}: 4, {CA}: 21, {CE}:50	{C}: 115, {CB}: 57, {CF}: 18, {CA}: 36, {CE}: 50
	B		{B}: 102, {BF}: 45, {BA}: 45
	F		{F}: 75
Luồng 2	E	{E}:93, {EB}:45, {EF}: 35, {EA}:51, {EAF}: 35	{E}: 107, {EB}: 45, {EF}: 69, {EA}: 81, {EAF}: 57
	A		{A}: 87, {AF}: 57

Luồng 1 lặp lại bước 4 tìm tập lợi ích cao với tiền tố là B, F giống như phần tử C ở trên. Tương tự, luồng 2 thực hiện tìm tập lợi ích cao với tiền tố E, A.

Kết thúc quá trình khai phá với ngưỡng lợi ích tối thiểu minutil = 56 trên 2 luồng ta thu được tập lợi ích cao HUIs = {C:57, CB:57}.

Bảng 14 sẽ cho ta thấy sử dụng mô hình utility-list hiệu quả hơn mô hình TWU trong việc cắt tỉa các ứng viên. Mô hình utility-list sinh ra 10 ứng viên còn mô hình TWU sinh ra 16 ứng viên.

V. KẾT QUẢ ĐÁNH GIÁ

Trong phần này, chúng tôi chỉ so sánh kết quả thực hiện thuật toán PPB-Miner với thuật toán HP [13] vì thuật toán HP đã được so sánh, đánh giá tối ưu hơn với thuật toán TP [8], PB [2]. 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.

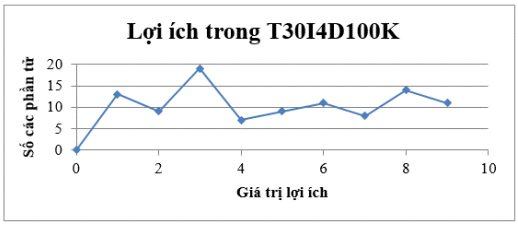
V.1. Môi trường và dữ liệu

Thuật toán được thực hiện trên máy tính HP core 7 due 2.4 GHz với 4 GB bộ nhớ, chạy trên Windows 7. Chương trình chúng tôi viết bằng Visual C++ 2010 với thư viện lập trình đa luồng OPENMP, số luồng thử nghiệm là 2. Dữ liệu thử nghiệm gồm: Mushroom [17] và T30I4D100K được sinh từ bộ sinh dữ liệu của IBM [16]. Đặc điểm của bộ dữ liệu được mô tả phía dưới:

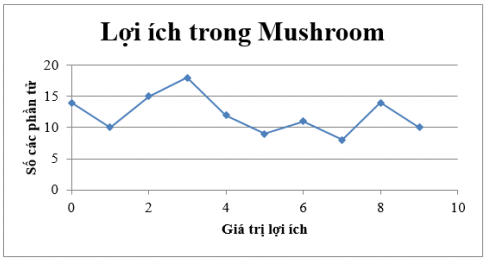
Database	T	D	N
T30I4D100K	30	100.000	100
Mushroom	23	8.124	119

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 2a cho biết việc phân bố lợi ích ngoài của các phần tử trong T30I4D100K. Hình 2b cho biết việc phân bố lợi ích ngoài của các phần tử trong Mushroom.



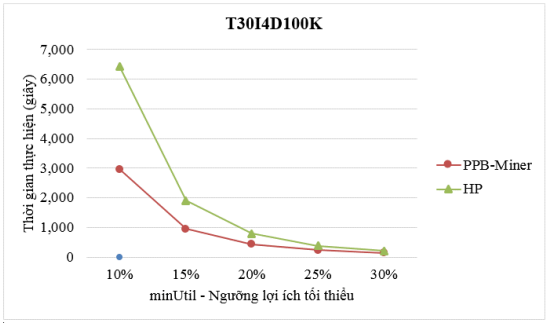
Hình 2a. Biểu đồ phân bố lợi ích ngoài của các phần tử trong T30I4D100K



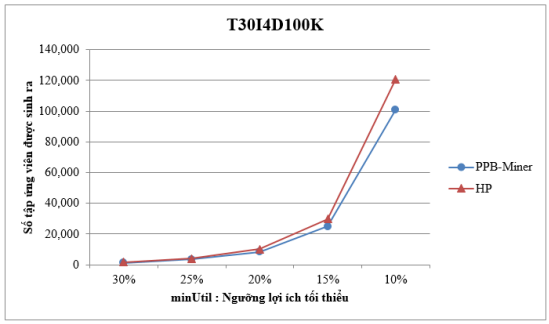
Hình 2b. Biểu đồ phân bố lợi ích ngoài của các phần tử trong Mushroom

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

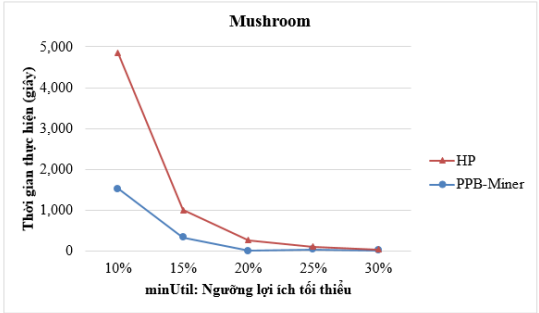
Kết quả thử nghiệm, so sánh giữa thuật toán PPB-Miner với thuật toán HP [13] trên bộ dữ liệu T30I4D100K và Mushroom. Hình 3a so sánh thời gian thực hiện khai phá tập lợi ích cao khi thay đổi ngưỡng lợi ích tối thiểu, Hình 3b so sánh số lượng ứng viên được sinh ra tương ứng với các ngưỡng lợi ích tối thiểu khác nhau. Hình 4a, 4b so sánh thời gian thực hiện khai phá tập lợi ích cao và số ứng viên sinh ra giữa hai thuật toán tương ứng với các ngưỡng lợi ích tối thiểu khác nhau trên bộ dữ liệu Mushroom.



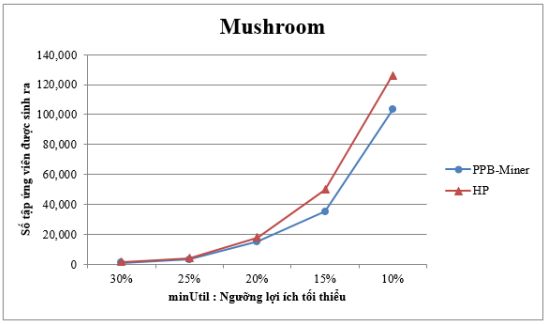
Hình 3a. So sánh thời gian thực hiện với ngưỡng lợi ích khác nhau



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



Hình 4a. So sánh thời gian thực hiện với ngưỡng lợi ích khác nhau



Hình 4b. So sánh số lượng ứng viên được sinh ra 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 đã được đề xuất trong những năm gần đây, phân tích ưu, nhược điểm của mô hình TWU, Utility-list nhằm loại bớt tập ứng viên. Trên cơ sở đó, chúng tôi đã đề xuất một cấu trúc lưu trữ tối ưu về bộ nhớ, kết hợp danh sách lợi ích và chỉ số hình chiếu trong thuật toán song song trên mô hình chia sẻ bộ nhớ dựa trên chỉ số hình chiếu. Kết quả thử nghiệm cho thấy thời gian thực thi và số lượng ứng viên sinh ra ít hơn một số thuật toán khác.

Thời gian tiếp theo, chúng tôi sẽ thử nghiệm thuật toán song song trên nền tảng Hapdoop, kiến trúc MapRedue để có thể thực hiện với các dữ liệu lớn.

TÀI LIỆU THAM KHẢO

[1] A. C.F. AND T. S.K, *Efficient Tree Structures for Highutility Pattern Mining in Incremental Databases, 2009*.

[2] G. CHENG LAN, T. PEI HONG, AND V. S. TSENG, *An efficient projection-based indexing approach for mining high utility itemsets*, 2013.

[3] Y. LIU, W. LIAO, AND A. CHOUDHARY, *A Fast High Utility Itemsets Mining Algorithm*, Proceedings of the 1st International Workshop on Utility-based Data Mining, New York, NY, USA, 2005, pp. 90–99.

[4] M. LIU AND J. QU, *Mining High Utility Itemsets Without Candidate Generation*, Proceedings of the 21st ACM International Conference on Information

and Knowledge Management, New York, NY, USA, 2012, pp. 55–64.

[5] K. SUBRAMANIAN, P. KANDHASAMY, AND S. SUBRAMANIAN, *A Novel Approach to Extract High Utility Itemsets from Distributed Databases*, Comput. Inform., vol. 31, no. 6+, pp. 1597–1615, 2013.

[6] V. S. TSENG, B.-E. SHIE, C.-W. WU, AND P. S. YU, *Efficient Algorithms for Mining High Utility Itemsets from Transactional Databases*, IEEE Trans Knowl Data Eng, vol. 25, no. 8, pp. 1772–1786, Aug. 2013.

[7] R. AGRAWAL AND R. SRIKANT, *Fast Algorithms for Mining Association Rules in Large Databases*, Proceedings of the 20th International Conference on Very Large Data Bases, 1994, pp. 487–499.

[8] Y. LIU, W. LIAO, AND A. CHOUDHARY, *A Two-phase Algorithm for Fast Discovery of High Utility Itemsets*, Proceedings of the 9th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining, Berlin, Heidelberg, 2005, pp. 689–695.

[9] W. SONG, Y. LIU, AND J. LI, *Vertical mining for high utility itemsets*, 2012 IEEE International Conference on Granular Computing, 2012, pp. 429–434.

[10] 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, no. 2, pp. 181–198, Apr. 2011.

[11] V. S. TSENG, C.-W. WU, B.-E. SHIE, AND P. S. YU, *UP-Growth: An Efficient Algorithm for High Utility Itemset Mining*, Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 2010, pp. 253–262.

[12] A. ERWIN, R. P. GOPALAN, AND N. R. ACHUTHAN, *CTU-Mine: An Efficient High Utility Itemset Mining Algorithm Using the Pattern Growth Approach*, 7th IEEE International Conference on Computer and Information Technology (CIT 2007), 2007, pp. 71–76.

[13] D. PHONG AND N. HUNG, *Một mô hình hiệu quả khai phá tập mục lợi ích cao*, Các Công Trình Nghiên Cứu Phát Triển Và Ứng Dụng CNTT-TT, pp. 26–36, Jun. 2015.

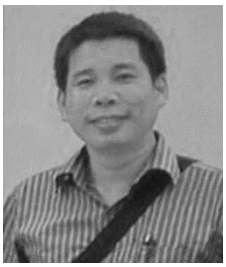
[14] A. ERWIN, R. P. GOPALAN, AND N. R. ACHUTHAN, *Efficient Mining of High Utility Itemsets from Large Datasets*, Advances in Knowledge Discovery and Data Mining, T. Washio, E. Suzuki, K. M. Ting, and A. Inokuchi, Eds. Springer Berlin Heidelberg, 2008, pp. 554–561.

[15] B. VO, H. NGUYEN, T. B. HO, AND B. LE, *Parallel Method for Mining High Utility Itemsets from Vertically Partitioned Distributed Databases*, Proceedings of the 13th International Conference on Knowledge-Based and Intelligent Information and Engineering Systems: *Part I*, Berlin, Heidelberg, 2009, pp. 251–260.

Nhận bài ngày: 09/12/2016

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 trường ĐH Thăng Long; về 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 ĐH 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