ĐẠI HỌC HOA SEN

Tạp chí Khoa học Trường Đại học Hoa Sen
https://www.hoasen.edu.vn/qlkh/ tapchidhhs.vn

THUẬT TOÁN KHAI THÁC MẪU HỮU ÍCH CAO LIÊN QUAN DỰA TRÊN BỘ ĐỆM

Lý Quang Vinh*[1], Nguyễn Quí Trung[2], Nguyễn Hữu Hoàng[2], Phạm Bửu Tài[3], Nguyễn Huy Cường[4]

[1]Khoa Thiết Kế Nghệ Thuật, Trường Đại học Hoa Sen
[2]Phòng Công Nghệ Thông Tin, Trường Đại học Hoa Sen
[3]Trường Đại học Luật TP. Hồ Chí Minh
[4]Trường Đại học Công nghệ TP. Hồ Chí Minh

Thông tin bài báo

Nhận bài: 07/2025
Chấp nhận: 08/2025
Xuất bản online: 12/2025

TÓM TẮT

Trong lĩnh vực khai thác dữ liệu, khai thác tập hữu ích cao (HUI) là một bài toán thu hút nhiều quan tâm từ những nhà khoa học cùng với nhiều thuật toán và phương pháp khai thác (HUI) hiệu quả. Tuy vậy trong các thuật toán trước đây ngoài việc chú trọng đến các phương pháp cắt tỉa nhằm tìm ra kiếm các tập HUI nhanh nhất, chính xác nhất, thường ít quan tâm đến việc sử dụng bộ nhớ đệm trong các quá trình đó.

Nét chính của thuật toán này khác biệt so với các thuật toán trước đây ở chỗ là sử dụng bộ đệm trong quá trình khai thác tìm các tập hữu ích cao. Việc sử dụng bộ nhớ đệm không chỉ làm giảm bộ nhớ mà còn giúp cho thời gian khai thác tìm kiếm hiệu quả hơn. Chúng tôi đề xuất cải tiến thuật toán ULB-Miner để khai thác các tập HUI có sử dụng bộ đệm gọi là ULBi. Với kết quả thu được cho thấy thuật toán đề xuất có thời gian và bộ nhớ hiệu quả hơn với thuật toán ULB-Miner

ABSTRACT

In the field of Data Mining, the task high utility set (HUI) mining has always been attract attention from scientists along with many effective algorithms and mining methods (HUI). However, in the previous algorithms, in addition to focusing on pruning methods to find the fastest and most accurate HUI sets, they often paid little attention to the use of cache in those processes.

The main feature of this algorithm differs from previous algorithms in that it uses a buffer in the mining process to find highly useful sets. The use of caching not only reduces memory but also makes the search mining time more efficient. We propose an improved ULB-Miner algorithm to mine HUI sets using a buffer called ULBi. The obtained results show that the proposed algorithm is more efficient in time and memory than the ULB-Miner algorithm

Keywords: *Khai thác dữ liệu, tập hữu ích cao, tập liên quan.*

* Tác giả liên hệ:
email: *vinh.lyquang@hoasen.edu.vn*

1. TỔNG QUAN

Khai thác mẫu hữu ích cao (HUI) giúp tổ chức và doanh nghiệp đưa ra quyết định dựa trên thông tin về mối quan hệ giữa các sản phẩm. Điều này hỗ trợ họ xây dựng kế hoạch và chiến lược kinh doanh phù hợp để đáp ứng nhu cầu thị trường hiệu quả hơn.

Trước khi khai thác dữ liệu, việc lựa chọn phương pháp thích hợp cho hệ thống lưu trữ là cần thiết. Trong quá trình khai thác từ cơ sở dữ liệu giao dịch, các tập dữ liệu hữu ích giúp doanh nghiệp nắm bắt tồn kho, sở thích mua sắm của khách hàng và thay đổi giá sản phẩm theo thời gian. Tìm ra các tập dữ liệu này nhanh chóng là ưu tiên quan trọng.

Vào năm 2003, thuật toán HUIM (Chan et al., 2003) được đề xuất bởi nhóm tác giả Chan, Yang và Shen, mục tiêu tìm ra các HUI, một phiên bản cải tiến của thuật toán HUI-Miner (Liu & Qu, 2012) với cách sử dụng trọng số giao dịch(Transaction-Weighted Utility-TWU), nhờ vào TWU đã giúp cắt tỉa sớm ứng viên. Fournier-Viger (2014) đã đề xuất thuật toán FHM (Fournier-Viger et al., 2014). Năm 2017, Zida và cộng sự phát triển thuật toán EFIM (Zida et al., 2017) để khai thác HUI trong CSDL bằng nhiều giải pháp trong đó có việc cải thiện không gian tìm kiếm cũng như kỹ thuật chiếu (Krishnamoorthy, 2017).Cùng năm 2017 tác giả Krishnamoorthy đã đưa ra thuật toán HMiner (Krishnamoorthy, 2017) dùng cấu trúc CUL (Compact utility list data).

Thực trạng nhu cầu thiết thực trong phân tích cũng như khai thác dữ liệu ngày càng nhiều và đa chủng loại. Hiện nay bài toán về khai thác dữ liệu để tìm ra tập hữu ích cao trên cơ sở dữ liệu giao dịch rất cần được các doanh nghiệp, người sử dụng đầu tư và tập trung nghiên cứu vì các lợi ích đạt được rõ ràng là không nhỏ.

Để làm giảm quá trình thao tác và nâng cao tốc độ cũng như giảm thiểu sử dụng không gian tìm kiếm của thuật toán khai thác tập hữu ích cao, thuật toán ULB-Miner (Duong et al., 2018) được đề xuất bởi Duong và đồng sự. Thuật toán là một bước phát triển về mặt sử dụng lại bộ nhớ đệm dựa trên cấu trúc UTL Buff và cấu trúc SUL.

Vào năm 2019 nhóm tác giả Nguyen và các đồng sự đưa ra thuật toán iMEFIM (Nguyen et al., 2019) nhằm cải thiện nhược điểm của thuật toán EFIM (Zida et al., 2017) bằng cách sử dụng giải pháp chiếu ngược P-set để giảm số lượng giao dịch cần xét và giảm thời gian khai phá các tập hữu ích cao.

Dựa trên các nghiên cứu trên, bài báo này đề xuất thuật toán ULBi, với mục tiêu HUI được tìm ra bằng cách sử dụng bộ đệm. Kết quả thực nghiệm (KQ-TN) cho thấy thuật toán ULBi đề xuất đã đặt thời gian chạy hiệu quả nhưng vẫn đảm bảo độ chính xác việc tìm kiếm tập HUI.

2. PHƯƠNG PHÁP

Trong thuật toán sử dụng bộ đệm, các tác giả đề xuất một cấu trúc danh sách hữu ích được cải tiến gọi là bộ đệm danh sách hữu ích, để giảm tiêu thụ bộ nhớ và tăng tốc hoạt động. Cấu trúc này được tích hợp vào một thuật toán mới có tên ULB-Miner (Duong et al., 2018) (Bộ đệm danh sách hữu ích cho người khai thác tập phổ biến hữu ích cao) cùng với việc kết hợp sử dụng lại bộ nhớ <ULBReusing Memory Construct> đưa các phần tử trong Primary đã kết hợp và xác định thành nhánh để dễ dàng trong việc xác định các tập có giá trị hữu ích cao. Việc thuật toán sử dụng bộ đệm UTLBuf, SULs đã giảm thao tác quét lại dữ liệu nhiều lần, nhưng bên cạnh đó vẫn còn mặt hạn chế nếu dữ liệu có nhiều giao dịch trùng lấp thì việc thể hiện bộ đệm đó sẽ tiêu tốn về mặt thời gian quét cũng như thời gian dành cho việc cắt tỉa tìm tập hữu ích cao

Thuật toán ULBi chúng tôi sử dụng phương pháp cắt tỉa của thuật toán iMEFIM (Nguyen et al., 2019) hiệu quả hơn, bên trong thuật toán đó có sử dụng phương pháp tính (local utility) và (sub-tree utility), P-set như vậy thì xuyên suốt quá trình cắt tỉa đều sử dụng lại các giá trị cần thiết từ các trường <item,prefix,remaining,offset> dựa vào đó, chúng tôi kết hợp cải tiến bộ đệm kết hợp UTLBuf (Duong et al., 2018) với phương pháp cắt tỉa iMEFIM (Nguyen et al., 2019).

2.1. Thuật toán ULBi

Input: Database D, set of items I, minimum utility threshold minutil
Output: High utility itemsets

1. $X = \emptyset$;
2. Scan D to calculate lu(X,i) for all item $i \in I$;
3. $Secondary(X) = \{i \mid i \in I \wedge lu(X,i) \geq minutil\}$;
4. Sort Secondary(X) in ascending order of lu(X,i) values;
5. Scan D to remove each item $i \notin Secondary(X)$ from all transactions. Then remove empty transactions
6. Sort each transaction T in ascending order according of lu(X,i) values when read backward;
7. Scan D again build the intal UTLBuf and calculate su(X,i) and Pex-set (X, i) for $I \in Secondary(X)$;
8. $Primary(X) = \{i \mid i \in Secondary(X) \wedge su(X,i) \geq minutil\}$;
9. Seach (X, UTLBuf, Primary(X), Secondary(X), minutil, Pex-set(X, i));

2.2. Thủ tục Search của ULBi

Input: Itemset X, database cDx, primary items Primary(X), secondary items Secondary(X), minimum utility threshold, minutil, the extensions of the TIDs projection Pex-set(X, i)

Output: High utility itemsets found by appending items to X

1. Set Huilist=Ø;
2. Foreach item i ∈ Primary(X) do
3. β= X ∪ {i};
4. Scan Dx in Pex-set(X,i) to calculate u(β) and construct Dβ; using transaction merging
5. If u(β) ≥ minutil then Huilist= Huilist ∪ β;
6. ULBReusingMemory-contrust (UTLBuf, su(β, z), lu(β, z), Pex-set(β, z)) for all z ∈ Secondary(X) after i ;
7. Primary(β) = { z ∈ Secondary(X) | su(β, z) ≥ minutil};
8. Secondary(β) = {z ∈ Secondary(X) | lu(β, z) ≥ minutil};
9. Search (X, UTLBuf, Primary(X), Secondary(X), minutil, Pex-set (X,i));
10. End;
11. Return Huilist;

Minh họa thuật toán: Ta có bảng CSDL giao dịch như sau:

Bảng 1: Dữ liệu giao dịch

T_{id}	a	b	c	d	e	f	g
T_1	1	-	1	1	-	-	-
T_2	2	-	6	-	2	-	5
T_3	1	2	1	6	1	5	-
T_4	-	4	3	3	1	-	-
T_5	-	2	2	-	1	-	2

Bảng 2: Bảng giá trị lợi nhuận phần tử

I	a	b	c	d	e	f	g
Utility	5	2	1	2	3	1	1

Sau khi tính lợi nhuận của các mặt hàng trong giao dịch và tính lần lượt U(X) theo công thức (Nguyen et al., 2019)1, TU(T_j) theo công thức (Nguyen et al., 2019)2, TWU(Cambria Math) theo công thức (Nguyen et al., 2019)3 ta được các bảng sau:

Bảng 3: Lợi nhuận theo giao dịch.

	a	b	c	d	e	f	g
	5	-	1	2	-	-	-
	10	-	6	-	6	-	5
	5	4	1	12	3	5	-
	-	8	3	6	3	-	-
	-	4	2	-	3	-	2

Bảng 4: Lợi ích các mặt hàng trong giao dịch

	a	b	c	d	e	f	g	
T_1	5	-	1	2	-	-	-	**8**
T_2	10	-	6	-	6	-	5	**27**
T_3	5	4	1	12	3	5	-	**30**
T_4	-	8	3	6	3	-	-	**20**
T_5	-	4	2	-	3	-	2	**11**
TU(X)	20	16	13	20	15	5	7	
TWU(X)	**65**	**61**	**96**	**58**	**88**	**30**	**38**	

Bảng 5: Bộ đệm UTLBuf trong thao tác ULBReusingMemory-contrust

UTLBuf						
Tid	**Iutil**	**Rutil**	**Item**	**Prefix**	**Remaining**	**Offset**
T_1	2	6	d	0	8	0
T_1	5	1	a	2	6	0
T_1	1	0	c	7	1	0
T_5	2	9	g	0	11	0
T_5	4	5	b	2	9	0
T_5	3	2	e	6	5	0
T_5	2	0	c	9	2	0
T_4	6	14	d	0	20	0
T_4	8	6	b	6	14	0
T_4	3	3	e	14	6	0
T_4	3	0	c	17	3	0
T_3	12	13	d	0	25	0
T_3	4	9	b	12	13	0
T_3	5	4	a	16	9	0
T_3	3	1	e	21	4	0
T_3	1	0	c	24	1	0
T_2	5	22	g	0	27	0
T_2	10	12	a	5	22	0
T_2	6	6	e	15	12	0
T_2	6	0	c	2	6	0

[1] Công thức tính U(X) "Lợi ích của một tập mục trong tập dữ liệu": $U(X) = \Sigma_{T_j \in D \, X \subseteq T_j} U(X, T_j)$

[2] Công thức tính TU(Tj) (Lợi ích giao dịch): $TU(T_j) = \Sigma_{i_k \subseteq T_j} U(i_k, T_j)$

[3] Công thức tính TWU(X) (Trọng số lợi ích giao dịch): $TWU(X) = \Sigma_{X \subseteq T_j \wedge T_j \in T_j} TU(T_j)$

Trên đây là toàn bộ các bước thực hiện của thuật toán ULBi để mô tả lại cách khai thác tìm ra tập hui, luận văn với ý tưởng ban đầu dùng bộ đệm UTLBuff của ulb-miner kết hợp với thuật toán iMEFIM, để kết hợp được thì phải đồng bộ hóa cấu trúc bộ đệm UTLBuff chứa dữ liệu giao dịch sao cho khi thực thủ tục đệ quy của IMEFIM phát huy được chiến lược cắt tỉa mà vẫn giữ nguyên cấu trúc UTLBuff bằng việc thêm vào các trường <prefix, remaining, offset> có sẵn trong thuật toán IMEFIM. kết quả ta đã thấy đều tìm ra đúng tập HUI

3. KẾT QUẢ THỰC NGHIỆM

Thuật toán ULBi được cài đặt bằng ngôn ngữ lập trình Java và thử nghiệm trên máy tính "Intel core i7 – 7x 2.40GHz, Ram 8GB", Hệ điều hành Windows 10 Pro. Các CSDL thử nghiệm được tải từ thư viện SPMF là các CSDL giao dịch gồm Chainstore, Chess, Foodmart, Mushroom, Connect. Thực nghiệm của thuật toán ULBi được so sánh với thuật toán ULB-Miner (Liu & Qu, 2012) khai thác tập HUI.

KQ-TN được đánh giá dựa trên kết quả các tập hữu ích cao có sử dụng bộ đệm thu được.

Bảng 6: Dữ liệu thực nghiệm

Cơ sở dữ liệu	Số giao dịch	Số phần tử	Độ dài trung bình
Chainstore	1,112,949	46,086	7.2
Chess	3,196	75	37
Mushroom	8,124	119	23
Connect	67,557	129	43

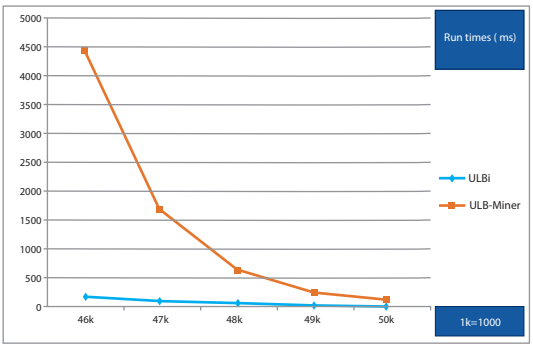
Trong bài báo này, dữ liệu thực nghiệm được chọn từ thư viện SPMF với các đặc trưng khác nhau:

- Chainstore: 1.112.949 giao dịch, 46.086 phần tử, độ dài trung bình 7.2, đại diện cho dữ liệu thương mại điện tử quy mô lớn.
- Chess: 3.196 giao dịch, 75 phần tử, độ dài trung bình 37, thách thức ở số lượng nhỏ nhưng giao dịch rất dài.
- Mushroom: 8.124 giao dịch, 119 phần tử, độ dài trung bình 23, dữ liệu trung bình, điển hình cho lĩnh vực thực phẩm.
- Connect: 67.557 giao dịch, 129 phần tử, độ dài trung bình 43, đại diện cho dữ liệu có độ phức tạp cao.

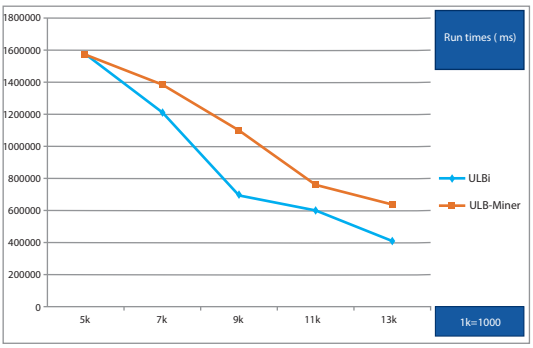
Những bộ dữ liệu này với đặc điểm đa dạng giúp kiểm chứng tính ổn định và hiệu quả của thuật toán ULBi trong nhiều tình huống.

Hình 1, 2, 3, 4, thể hiện KQ-TN so sánh giữa thuật toán ULBi được so sánh với thuật toán ULB-Miner (Liu & Qu, 2012) về số lượng tập HUI tìm được với ngưỡng miniutil = 46000 được sử dụng cho tất cả các tập dữ liệu.

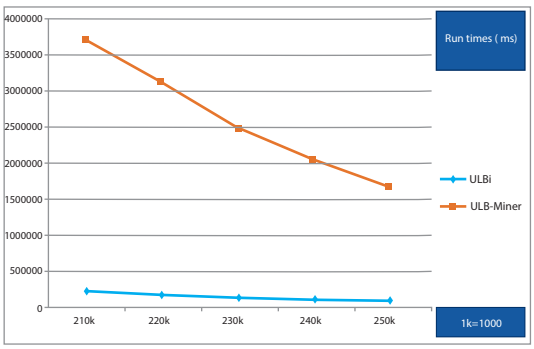
KQ-TN cho thấy thuật toán ULBi hiệu quả về mặt thời gian chạy hiệu quả hơn thuật toán ULB-Miner (Liu & Qu, 2012) trên tất cả các CSDL: Chainstore, Chess, Mushroom, Connect.



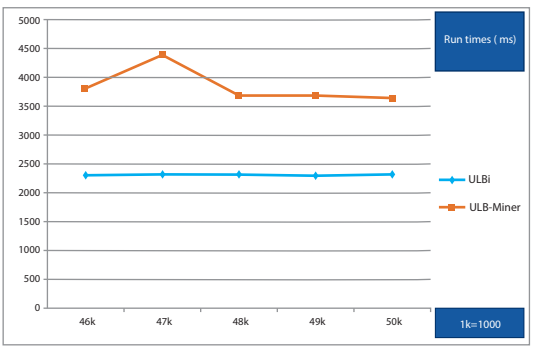
Hình 1: Kết quả trên bộ dữ liệu Chess



Hình 2: Kết quả trên bộ dữ liệu Mushroom



Hình 3: Kết quả trên bộ dữ liệu Connect



Hình 4: Kết quả trên bộ dữ liệu Chainstore

Từ biểu đồ kết quả, ta thấy đối với cả bốn cơ sở dữ liệu như Chainstore, Chess, Mushroom, Connect thì với giá trị minutil càng cao thì số tập HUI tìm được càng ít mà vẫn bằng nhau ở 2 thuật toán.

Về độ lệch thời gian của thuật toán cải tiến ULBi và thuật toán ULB-Miner (Liu & Qu, 2012) thì ta thấy kết quả càng tiến lại gần nhau lại

4. KẾT LUẬN

Trong bài báo này, chúng tôi đã đề xuất một thuật toán mới mang tên ULBi nhằm khai thác tập lợi ích cao có sử dụng bộ đệm. Thuật toán sử dụng cấu trúc chiếu ngược P-set, danh sách tóm tắt SUL. Cùng với chiến lược cắt tỉa thông minh như ma trận EUCS, đã được áp dụng để giảm thiểu không gian tìm kiếm và đảm bảo số lượng tập hữu ích cao.

KQ-TN trên các bộ dữ liệu chuẩn từ thư viện SPMF đã cho thấy thuật toán ULBi vượt trội hơn so với ULB-Miner (Liu & Qu, 2012) về cả hiệu suất và tính hiệu quả. Cụ thể, thuật toán không chỉ giảm thời gian xử lý mà còn loại bỏ được các tập dư thừa không mang lại ý nghĩa thực tiễn

Hướng nghiên cứu tiếp theo

Dựa trên những kết quả đạt được, chúng tôi đề xuất các hướng nghiên cứu trong tương lai như sau:

1. Cải tiến việc sử dụng bộ nhớ: tối ưu hóa cấu trúc bộ đệm để thích nghi với nhiều loại CSDL
2. Nâng cao mức ngưỡng: Khám phá các mức ngưỡng ở các loại CSDL có kích thước lớn
3. Ứng dụng trên CSDL theo khung thời gian: Mở rộng thuật toán để khai thác HUI trên CSDL theo khung thời gian

Bài báo đã cung cấp một cách tiếp cận mới và hiệu quả trong việc xử lý dữ liệu giao dịch có dung bộ đệm góp phần mở rộng tiềm năng ứng dụng của thuật toán tìm tập hữu ích cao trong môi trường kinh doanh thực tế.

TÀI LIỆU THAM KHẢO

[1] Chan, R., Yang, Q., & Shen, Y.-D. (2003). Mining high utility itemsets. Third IEEE international conference on data mining,

[2] Duong, Q. H., Fournier-Viger, P., Ramampiaro, H., Nørvåg, K., & Dam, T. L. (2018). Efficient high utility itemset mining using buffered utility-lists. *Applied Intelligence, 48*, 1859-1877.

[3] Fournier-Viger, P., Wu, C.-W., Zida, S., & Tseng, V. S. (2014). FHM: Faster high-utility itemset mining using estimated utility co-occurrence pruning. Foundations of Intelligent Systems: 21st International Symposium, ISMIS 2014, Roskilde, Denmark, June 25-27, 2014. Proceedings 21,

[4] Krishnamoorthy, S. (2017). HMiner: Efficiently mining high utility itemsets. *Expert Systems with Applications, 90*, 168-183.

[5] Liu, M., & Qu, J. (2012). Mining high utility itemsets without candidate generation. Proceedings of the 21st ACM international conference on Information and knowledge management,

[6] Nguyen, L. T., Nguyen, P., Nguyen, T. D., Vo, D. B., Fournier-Viger, P., & Tseng, V. S. (2019). Mining high-utility itemsets in dynamic profit databases. *Knowledge-Based Systems, 175*, 130-144.

[7] Zida, S., Fournier-Viger, P., Lin, J. C.-W., Wu, C.-W., & Tseng, V. S. (2017). EFIM: a fast and memory efficient algorithm for high-utility itemset mining. *Knowledge and Information Systems, 51*(2), 595-625.

[8] Võ, Đ. B., & Nguyễn, T. P. (2017). Nâng cao hiệu quả khai phá tập hữu ích cao bằng giải pháp chiếu ngược P-set. *Bản B của Tạp chí Khoa học và Công nghệ Việt Nam, 59*(11).