TẠP CHÍ KHOA HỌC
TRƯỜNG ĐẠI HỌC SƯ PHẠM TP HỒ CHÍ MINH
Tập 23, Số 1 (2026): 189-200
Website: https://journal.hcmue.edu.vn
ISSN: 2734-9918

HO CHI MINH CITY UNIVERSITY OF EDUCATION
JOURNAL OF SCIENCE
Vol. 23, No. 1 (2026): 189-200
https://doi.org/10.54607/hcmue.js.23.1.5278(2026)

Bài báo nghiên cứu

KHAI THÁC MẪU CHIẾM DỤNG CAO TRÊN CƠ SỞ DỮ LIỆU TRỌNG SỐ

Lê Tấn Long

Trường Đại học Sài Gòn, Việt Nam

Tác giả liên hệ: Lê Tấn Long – Email: nhatld@hcmue.edu.vn

Ngày nhận bài: 29-9-2025; Ngày nhận bài sửa: 14-11-2025; Ngày duyệt đăng: 18-11-2025

TÓM TẮT

Khai thác mẫu chiếm dụng cao (High Occupancy Itemset – HOI) là một hướng nghiên cứu mới, hiện đang thu hút nhiều sự quan tâm trong lĩnh vực khai phá dữ liệu. Không giống như các mẫu phổ biến vốn dựa trên tần suất xuất hiện, HOI được định nghĩa là những tập danh mục chiếm tỉ lệ lớn trong độ dài của các giao dịch. So với các mẫu phổ biến, số lượng HOI thường ít hơn nhưng lại mang những đặc trưng có ý nghĩa hơn. Tuy nhiên, HOI chỉ chú trọng đến sự có mặt của các danh mục, mà chưa phản ánh sự khác biệt về trọng số giữa chúng. Để khắc phục hạn chế này, bài báo giới thiệu khái niệm mẫu chiếm dụng trọng số cao (High Weighted Occupancy Pattern – HWOP) và đề xuất thuật toán HWOP-ROL nhằm khai thác HWOP. Ngoài ra, chúng tôi cũng xây dựng một ngưỡng chặn trên UBWO để cắt tỉa không gian tìm kiếm. Kết quả thực nghiệm trên nhiều bộ dữ liệu có trọng số chứng minh tính hiệu quả vượt trội của phương pháp đề xuất so với Baseline.

Từ khóa: mẫu chiếm dụng cao; mẫu chiếm dụng trọng số cao; thuật toán HWOP-ROL; ngưỡng chặn trên UBWO

1. Giới thiệu

1.1. Tổng quan về khai phá mẫu chiếm dụng cao

Khai thác mẫu là một lĩnh vực quan trọng trong khai phá dữ liệu, nhằm rút trích đặc trưng phục vụ phân tích và làm đầu vào cho các hệ thống thông minh. Loại mẫu được nghiên cứu nhiều nhất là mẫu phổ biến, lần đầu tiên được giới thiệu bởi Agrawal và Srikant (1994), với nhiều thuật toán tiêu biểu như Apriori (Agrawal & Srikant, 1994), Eclat (Zaki, 2000), FP-Growth (Grahne & Zhu, 2005) và PrePost (Deng et al., 2012). Các phương pháp này đều dựa trên độ hỗ trợ – số lần xuất hiện của một tập mục trong cơ sở dữ liệu (CSDL) giao dịch – để đánh giá tính phổ biến. Một tập mục được coi là phổ biến nếu độ hỗ trợ của nó vượt qua ngưỡng cho trước. Tuy nhiên, độ hỗ trợ chỉ phản ánh sự xuất hiện của các mục và xem chúng như nhau, trong khi trên thực tế mỗi mục có tầm quan trọng hoặc giá trị khác nhau. Ví dụ, các mặt hàng thiết yếu như gạo, muối có thể xuất hiện nhiều nhưng giá trị lợi nhuận lại thấp hơn hàng cao cấp như yến sào hoặc nhân sâm.

Cite this article as: Le, T. L. (2026). High Occupancy Pattern Mining on Weighted Transactional Databases. *Ho Chi Minh City University of Education Journal of Science, 23*(1), 189-200. https://doi.org/10.54607/hcmue.js.23.1.5278(2026)

Để giải quyết vấn đề này, khái niệm mẫu phổ biến có trọng số đã được đề xuất (Ramkumar et al., 1998), với các thuật toán điển hình như WIT-FWIDiff (Vo et al., 2013), IWS (Nguyen et al., 2016) và NFWI (Bui et al., 2018). Các phương pháp này dựa trên độ phổ biến có trọng số, được tính từ tổng trọng số của các giao dịch chứa mẫu so với toàn bộ CSDL.

Năm 2012, Tang và cộng sự (2012) đưa ra khái niệm độ chiếm dụng trung bình, thể hiện tỉ lệ trung bình của một tập mục trong các giao dịch chứa nó. Các mẫu chiếm dụng cao đã được chứng minh hữu ích trong nhiều lĩnh vực như y tế, thương mại. Tuy nhiên, việc xác định đồng thời hai ngưỡng hỗ trợ (α) và chiếm dụng (β) là không đơn giản đối với người dùng.

Để khắc phục, Deng (2020) đã đề xuất khái niệm mẫu chiếm dụng cao (High Occupancy Itemset – HOI), trong đó độ chiếm dụng được tính bằng tổng tỉ lệ độ dài của tập mục trong các giao dịch chứa nó. Một tập mục được coi là HOI nếu độ chiếm dụng vượt ngưỡng ξ. Mặc dù số lượng HOI nhỏ hơn nhiều so với mẫu phổ biến, nhưng HOI không thỏa mãn tính chất bao đóng giảm, dẫn đến không gian tìm kiếm rất lớn. Để giải quyết, Deng (2020) giới thiệu ngưỡng chặn trên UBO cùng thuật toán HEP dựa trên cấu trúc Occupancy-list. Sau đó, Nguyen và cộng sự (2023) đã cải tiến với các thuật toán FHOI và DFHOI, áp dụng chiến lược loại bỏ ứng viên, kiểm tra độ dài Occupancy-list, phân lớp tương đương và DFS, từ đó tăng tốc khai thác và giảm mức tiêu thụ bộ nhớ. Gần đây, Zhang và cộng sự (2024) thiết kế một cấu trúc nén gọi là sequence projection (seqPro) và đề xuất một thuật toán hiệu quả có tên HUSP-SP (discovering High-Utility Sequential Patterns with the seqPro structure) để khai thác mẫu tiện ích cao trên cơ sở dữ liệu tuần tự.

Trong bài báo này, chúng tôi đề xuất bài toán khai thác mẫu trọng số chiếm dụng cao trên CSDL trọng số *WD* (Weighted database) và đề xuất thuật toán hiệu quả để giải quyết, đồng thời thực hiện các thực nghiệm trên nhiều bộ dữ liệu chuẩn (benchmark datasets) để chứng minh tính hiệu quả của phương pháp đề xuất.

1.2. Cơ sở lí thuyết và nghiên cứu liên quan

a. Cơ sở lí thuyết

Định nghĩa 1. (CSDL có trọng số): Một CSDL có trọng số WD là một bộ ba ⟨T, I, W⟩ trong đó:

- T = {tj | j [1, m]} là tập các giao dịch,
- I = {ij | j [1, n]} là tập các mục,
- W = {wj | j [1, n] ∧ wj là trọng số của mục ij.

Ví dụ về CSDL có trọng số ở Bảng 1: CSDL này gồm sáu giao dịch t1,t2,t3, t4, t5, t6 được hình thành từ tập các mặt hàng A, B, C, D, E với các trọng số tương ứng lần lượt là 1.2, 0.9, 0.6, 1.8, 2.4.

Bảng 1. *Ví dụ về CSDL trọng số*

A		B	
CSDL giao dịch		**Trọng số mục**	
Giao dịch	**Danh mục**	**Mục**	**Trọng số**
t1	A, C, D, E	*A*	1.2
t2	A, B, C	*B*	0.9
t3	C, D	*C*	0.6
t4	A, C	*D*	1.8
t5	A, B, E	*E*	2.4

Trong mục này, chúng tôi định nghĩa các khái niệm cơ bản và mô tả bài toán khai thác HWOP.

Định nghĩa 2. Trọng số giao dịch *tw* (Transaction Weighted) của một giao dịch tkđược tính như sau:

tw_{tk}= iStkwi |S(tk)| (1)

Trong đó $S(t_k)$ là tập hợp các danh mục thuộc giao dịch $_{tk}$, *i* và wi lần lượt là một mục và trọng số của nó trong tập hợp trên. |P| đại diện cho số phần tử trong tập hợp P.

Ví dụ 1. Dựa trên Công thức (1), ta có Bảng 2 chứa cách tính chi tiết trọng số giao dịch của WD ở Bảng 1.

Bảng 2. *Trọng số giao dịch của từng giao dịch*

Giao dịch	Công thức	*tw*
t1	(1.2 + 0.6 + 1.8 + 2.4) / 4	1.5
t2	(1.2 + 0.9 + 0.6) / 3	0.9
t3	(0.6 + 1.8) / 2	1.2
t4	(1.2 + 0.6) / 2	0.9
t5	(1.2 + 0.9 + 2.4) / 3	1.5
	Tổng trọng số giao dịch (sumtw)	6

Định nghĩa 3. Độ hỗ trợ trọng số *ws* (Weighted Support) của mẫu (pattern) *X* được tính như sau:

wsX= tktXtwtktkTtwtk (2)

trong đó tX là tập các giao dịch có chứa mẫu X.

Ví dụ 2. Giá trị *ws* của tập mục *CD* được xác định bằng Công thức (2) như sau:

ws(*CD*)= twt1+ twt3] / ∑*tw* = (1.5 + 1.2) / 6 = 0.45.

Định nghĩa 4. Trọng số *w* (Weighted) của mẫu danh mục X chứa nhiều phần tử được tính như sau:

wX= i ∈ Xwi (3)

Ví dụ 3. Giá trị trọng số của mẫu CD được xác định bằng công thức (3) như sau:

WCD=wc + wd=0.6+1.8=2.4

Định nghĩa 5. Độ chiếm dụng trọng số riêng của một mẫu đơn X trong giao dịch t là so (Singleton occupancy) được tính như sau:

soX,t= w(X)w(St)×twt (4)

Định nghĩa 6. Độ chiếm dụng trọng số của một mẫu X (gồm nhiều phần tử) trong giao dịch t là *wot* (Weighted occupancy in transaction) được tính như sau:

wotX,t= w(X)w(St) ×twt (5)

Từ Định nghĩa 5 và 6 có thể thấy độ chiếm dụng trọng số của một mẫu X trong giao dịch t chính là tổng độ chiếm dụng trọng số riêng của từng phần tử trong *X*. Điều này giúp việc tính toán độ chiếm dụng trọng số của một mẫu nhiều phần tử trong giao dịch t trở nên nhanh hơn. Đặc biệt là khi ta có độ chiếm dụng trọng số của tất cả các phần tử tiền tố trước đó trong giao dịch t. Ví dụ, wotPY, t=wotP,t+so(Y,t). Công thức này làm cho việc tính toán độ hộ trợ trong số trong thuật toán đề xuất ở Hình 1 nhanh hơn.

Định nghĩa 7. Độ chiếm dụng trọng số *wo* (Weighted Occupancy) của mẫu X trong CSDL được tính như sau:

wo_x= tktXwotX,tktkTtwtk (6)

Ví dụ 4. Áp dụng Công thức (4) ta tính được độ chiếm dụng trọng số của mẫu CD như sau.

woCD= twt1WCDWACDE+twt3WCDWCDsumtw=1.5×2.46+1.2× 2.42.4 6= 0.3

Định lí 1. Với cùng một mẫu X thì wo(X) ≤ ws(X).

Chứng minh: Vì X⊆S(tk) ta có:

W(X)W(Stk)≤1 (7)

Từ (*7*) ta có:

tktXtwtkW(X)W(Stk) ≤ tktXtwtk (8)

Từ (*8*) ta có:

tktXtwtkW(X)W(Stk)tkTtwtk tktXtwtktkTtwtk (9)

Từ (*9*) ta có:

tktXwotX,tktkTtwtk tktXtwtktkTtwtk (10)

Từ (*10*) và công thức (2), (6) ta có:

woX≤ws(X)

Định lí 1 đã được chứng minh.

Định nghĩa 8. Cho một ngưỡng chiếm dụng trọng số *ξ*. Một mẫu danh mục X được gọi là mẫu chiếm dụng trọng số cao *HWOP* (High Weighted Occupancy Pattern) khi và chỉ khi woX≥ *ξ*.

Từ Định lí 1 và Định nghĩa 8 ta có thể suy ra hệ quả, nếu một mẫu X không phải là mẫu phổ biến ở ngưỡng *ξ* (mẫu có ws *ξ*) thì không thể là mẫu chiếm dụng cao.

b. Một số nghiên cứu liên quan

Năm 1998, (Ramkumar et al., 1998) lần đầu tiên giới thiệu bài toán khai thác các mẫu có trọng số phổ biến (FWP). Sau đó, Tao và cộng sự (2003) đã đề xuất một khung khái niệm mới với các khái niệm trọng số giao dịch và độ hỗ trợ có trọng số để khai thác FWP.

Trọng số giao dịch (*tw* – transactions weighted) là trung bình có trọng số của các mặt hàng trong một giao dịch, còn độ hỗ trợ có trọng số là tổng các giá trị tw của các giao dịch chứa mẫu đó chia cho tổng các giá trị tw của tất cả các giao dịch. Do đó, việc khai thác FWP là tìm tất cả các mẫu có độ hỗ trợ có trọng số không nhỏ hơn một ngưỡng độ hỗ trợ có trọng số tối thiểu do người dùng chỉ định trong CSDL.

Với cách tiếp cận này, Vo và cộng sự (2013) đã đề xuất cấu trúc WIT-tree như một phần mở rộng của cấu trúc IT-tree để khai thác FWP. Phương pháp này được chứng minh là hiệu quả vì nó chỉ cần quét CSDL một lần để tạo các tập hợp ID giao dịch (tidset) và thực hiện khai thác trên WIT-tree. Ngoài ra, chiến lược diffset cũng được áp dụng để tăng tốc quá trình khai thác. Tuy nhiên, phương pháp này không hiệu quả trên các CSDL lớn vì tốn nhiều bộ nhớ để lưu trữ tidset.

Tiếp theo, Nguyen và cộng sự (2016) đề xuất sử dụng cấu trúc bitset IWS và bitset mở rộng EIWS (Nguyen et al., 2022) cùng với hướng tiếp cận Eclat để tối ưu hóa việc lưu trữ và tính giao tidset của các itemsets. Ở hướng tiếp cận FP-growth, Lee và cộng sự (2017) đề xuất hai thuật toán, PWP-WSD và PWP-TCD, để khai thác FWP dựa trên cấu trúc FP-tree. Phương pháp này nén CSDL thành cấu trúc cây tiền tố và không cần lưu trữ tidset. Tuy nhiên, nó phải quét cây nhiều lần để khai thác FWP.

Gần đây, Bui và cộng sự (2018) đã đề xuất cấu trúc WN-list và thuật toán NPWP để khai thác FWP. Cấu trúc WN-list mở rộng cấu trúc N-list (Deng et al., 2012) với khả năng nén dữ liệu rất hiệu quả. Kết quả thực nghiệm cho thấy thuật toán NPWP hiện là thuật toán khai thác FWP tốt nhất hiện nay (State – Of – The – Art), phương pháp này tiếp tục được mở rộng áp dụng có hiệu quả cho các bài toán khai thác top-rank-k (Vo et al., 2020), khai thác tập đóng (Bui et al., 2018) hay CSDL dạng chuỗi (Bui et al., 2018).

2. Nội dung nghiên cứu

Trong phần này chúng tôi sẽ trình bày về thuật toán khai thác mẫu chiếm dụng cao cho WD.

2.1. Thuật toán khai thác mẫu phổ biến trên WD.

Trong phần này chúng tôi đề xuất Thuật toán *HWOP-ROL* (High Weighted Occupancy Patterns mining using the *RO-list* (Remain Occupancy list) structure.) sử dụng cấu trúc dữ liệu *RO-list* để khai thác hiệu quả mẫu chiếm dụng cao trên WD.

Định nghĩa 9. Độ chiếm dụng trọng số còn lại *ro* (Remain Occupancy) của một mẫu đơn X trong giao dịch t được tính như sau:

$$roX,t=Y\in S(t)\land X < Yw(Y)w(St) \times twt \qquad (11)$$

Trong đó S(t) là tập hợp các danh mục đã được sắp xếp của giao dịch t, thứ tự sắp xếp có thể là theo thứ tự bảng chữ cái hoặc bất kì một loại thứ tự phù hợp nào khác, trong nghiên cứu này chúng tôi chỉ sắp xếp dựa trên thứ tự bảng chữ cái. Kí hiệu '<' đại diện cho việc danh mục X đứng trước danh mục Y theo thứ tự đã sắp xếp.

Định nghĩa 10. Danh sách độ chiếm dụng trọng số còn lại ROL (Remain Occupancy list) của một mẫu đơn x là một cấu trúc dữ liệu dạng danh sách liên kết, mỗi node của danh sách bao gồm các trường {tid, ro, wot,so}. Trong đó tid (transaction *id*) là số thứ tự của giao dịch chứa mẫu đơn x trong WD. Các trường ro, wot, so lần lượt là độ chiếm dụng trọng số, độ chiếm dụng trọng số còn lại và độ chiếm dụng trọng số riêng của danh mục X trong giao dịch thứ tid. Các node được sắp xếp theo thứ tự tăng dần của tid.

Ví dụ 5. Danh sách độ chiếm dụng còn lại của mẫu C:

ROLC={1, 1.05, 0.15, 0.15,2, 0, 0.2, 0.2, {3, 0.9, 0.3, 0.3}, {4, 0, 0.3, 0.3}}

Ở giao dịch t1 phía sau danh mục C còn lại danh mục DE, độ chiếm dụng trọng số còn lại của danh mục này trong t1 là 1.05. Đặc biệt ở giao dịch t2 và t4 thì C là danh mục cuối nên chỉ số chiếm dụng còn lại bằng 0. Các trường wot và so sẽ bằng nhau khi mẫu X là mẫu chỉ có một phần tử.

Định nghĩa 11. Danh sách độ chiếm dụng trọng số còn lại của mẫu có k phần tử. Cho PX, PY là hai mẫu có k – 1 danh mục và có chung tiền tố P, trong đó Y xếp sau X theo thứ tự đã được sắp xếp. Với ROL(PX) và ROL(PY) lần lượt là danh sách độ chiếm dụng trọng số còn lại của hai mẫu PX, PY thì ROL(PXY) sẽ được tính bằng thuật toán như sau:

Với mỗi node tidi, roi, woti, soi ∈ROL(PX) và tidj, roj, wotj, soj ∈ ROLPY nếu tidi= tidj một phần tử mới sẽ được tạo và thêm vào ROL(PXY) với các thuộc tính tương ứng là {tidj, roj, woti+ soj, soj}, nếu tidi> tidj ta sẽ duyệt qua node tiếp theo của ROLPY ngược lại ta duyệt node tiếp theo của ROLPX cho đến khi một trong hai danh sách không còn phần tử để duyệt. Độ phức tạp của thuật toán là tuyến tính, do các node chỉ được duyệt qua một lần duy nhất.

Ví dụ 6. Danh sách độ chiếm dụng còn lại của mẫu CD được tạo thành thì danh sách của hai mẫu thành phần là C và D:

ROLC=1, 1.05, 0.15, 0.15,2, 0, 0.2, 0.2, 3, 0.9, 0.3, 0.3, 4, 0, 0.3, 0.3

ROLD={{1, 0.6, 0.45, 0.45},{3, 0, 0.9, 0.9}}

ROLCD={{1, 0.6, 0.6, 0.45},{3, 0,1.2, 0.9}}

Bước 1. Xét hai phần tử đầu tiên của ROL(C) và ROL(D).

Hai phần tử này có cùng chỉ số tid bằng 1 nên sẽ hợp lại với nhau. ROL(CD) được thêm một phần tử mới bằng cách kết hợp giữa 2 phần tử bên trên. Các màu đỏ, lục, lam, tím lần lượt được đánh dấu trên phần tử mới của ROL(CD) để chỉ ra thành phần tương ứng tạo ra nó từ ROL(C) và ROL(D), trong đó màu tím là sự kết hợp giữa phần tử màu xanh dương và đỏ.

ROLC=1, 1.05, 0.15, 0.15,2, 0, 0.2, 0.2, 3, 0.9, 0.3, 0.3, 4, 0, 0.3, 0.3

ROLD={{1, 0.6, 0.45, 0.45},{3, 0, 0.9, 0.9}}

1, 0.6,0.15+0.45, 0.45

ROLCD={{1, 0.6, 0.6, 0.45}}

Bước 2. Xét hai phần tử tiếp theo của ROL(C) và ROL(D). Hai phần tử này không cùng chỉ số tid nên chúng ta sẽ bỏ qua phần tử có tid nhỏ hơn trong ROL(C) để xét phần tử tiếp theo. Do các phần tử trong ROL được sắp xếp theo thứ tự tăng dần nên việc loại bỏ này hoàn toàn phù hợp vì chắc chắn không có phần tử nào của ROL(D) có thể có cùng chỉ số tid với phần tử hiện tại của ROL(C).

ROLC=1, 1.05, 0.15, 0.15,~~2, 0, 0.2, 0.2~~, 3, 0.9, 0.3, 0.3, 4, 0, 0.3, 0.3

ROLD={{1, 0.6, 0.45, 0.45},{3, 0, 0.9, 0.9}}

ROLCD={{1, 0.6, 0.6, 0.45}}

Bước 3. Xét hai phần tử mới có cùng chỉ số tid bằng 3 nên ta hợp lại thành phần tử mới của ROLCD tương tự bước 1.

Định nghĩa 12. Giới hạn trên về độ chiếm dụng trọng số UBWO (Upper-bound Weight Occupancy) của mẫu X là tổng của wo(X) và ro(X) trong đó roX là tổng tất cả ro trong ROL(X).

Dễ thấy một mẫu X có UBWOX< ξ thì tất cả các siêu tập (superset) Y của X đều không thể là mẫu trọng số chiếm dụng cao. Đây cũng chính là điều kiện để cắt nhánh sớm khi khai thác mẫu trọng số có độ chiếm dụng cao.

Các bước thực hiện thuật toán HWOP-ROL

Bước 1. Xây dựng RO-list cho các mẫu có một danh mục

- Quét WD để tính toán tw của từng giao dịch. Sắp xếp lại các danh mục trong từng giao dịch theo thứ tự từ điển.
- Quét WD lần 2, tính toán ws và xây dựng *RO-list* cho từng danh mục đơn.

- Loại tất cả các danh mục đơn có ws < ξ. Tạo ra tập F1, mỗi phần tử chứa tên, *RO-list*, ws, wo và ro, tương ứng của các tập một danh mục, sắp xếp các phần tử của F1 theo thứ tự từ điển.

Bước 2. Khai thác bằng cây liệt kê (Thủ tục Find_HWOP)

- Sử dụng cây liệt kê để khai thác tập chiếm dụng trọng số cao theo phương pháp duyệt theo chiều sâu (DFS).
- Các phần tử trong các lớp tương đương lần lượt được giao tuần tự với nhau để hình thành các mẫu mới. Thuật toán sẽ kiểm tra khả năng sinh ra mẫu chiếm dụng cao của mẫu X bằng cách sử dụng giá trị UBWO(X). Nếu UBWO(X) < ξ cắt nhánh X sớm, các nhánh có ws(X) < ξ loại trực tiếp ứng viên *X*, ngược lại ta thêm mẫu X vào cây liệt kê và tiếp tục mở rộng.

Các bước trên được thể hiện qua giả mã trong Hình 1 như sau:

HWOP-ROL Algorithm

Input: A weighted database WD and a threshold ξ.
Output: HWOPs, the set of all high weighted occupancy patterns
Algorithm 1: The *HWOP-ROL* Algorithm
Input: A weighted database WD, a minimum weighted occupancy threshold ξ.
Output: HWOPs, the set of all high weighted occupancy patterns.

```
--------------------------------------------------------------------------------
Scan WD to calculate transaction weights (tw) for all transactions.
Scan WD a second time to compute ws and construct the RO-list for each 1-pattern.
F1 ← {p | p is a 1-pattern and p.ws ≥ ξ}.
Sort F1 in lexicographical order
HWOPs ← Ø
Call Find_HWOP(Ø, F1, tw, HWOPs, ξ)
Return HWOPs

Procedure Find_HWOP(Pre, F , tw, HWOPs, ξ)
for each candidate X in F do
    if X.ro + X.wo < ξ then   // Pruning with UBWO
        continue
    if X.wo ≥ ξ then
        HWOPs ← HWOPs ∪ (Pre ∪ {X.name})
    F_next ← Ø
    for each candidate Y in F that appears after X do
        C ← ROL_intersect(X, Y, tw)
        if C.ws ≥ ξ then
            F_next ← F_next ∪ {C}
    if F_next ≠ Ø then
        Call Find_HWOP(Pre ∪ {X.name}, F_next, tw, HWOPs, ξ)
end for

Function ROL_intersec (U, V, tw)
C ← new Candidate
C.name ← V.name   // Assumes U and V share a prefix
C.ROL ← Ø
C.ws ← 0; C.wo ← 0; C.ro ← 0
i ← 0; j ← 0
while i < |U.ROL| and j < |V.ROL| do
    a ← U.ROL[i]; b ← V.ROL[j]
```

```
        if a.tid = b.tid then
                n ← new Node({b.tid, b.ro, a.wot + b.so, b.so})
                add n to C.ROL
                C.ro ← C.ro + b.ro
                C.wo ← C.wo + (a.wot + b.so)
                C.ws ← C.ws + tw[a.tid]
                i ← i + 1; j ← j + 1
        else if a.tid > b.tid then
                j ← j + 1
        else
                i ← i + 1
        end if
    end while
return C
```

Hình 1. *Mã giả thuật toán HWOP-ROL*

2.2. *Một ví dụ cụ thể*

Trong phần này, chúng tôi sẽ trình bày một ví dụ cụ thể đối với thuật toán đề xuất trong phần 2.1.

Với WD trong Bảng 1 và $\xi = 0.3$, thuật toán *HWOP-ROL* duyệt CSDL 2 lần để tạo ra *RO-list* của từng danh mục đơn như sau:

ROLA={1,1.2, 0.3, 0.3, 2, 0.5,0.4,0.4, 4, 0.3, 0.6, 0.6, 5, 1.1, 0.4, 0.4}

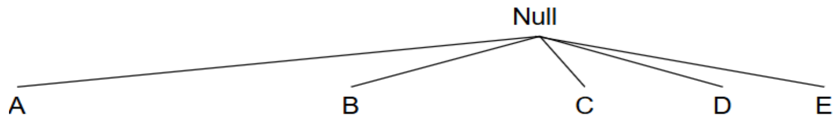
ROLB={2, 0.2, 0.3, 0.3, 5, 0.8, 0.3, 0.3}

ROLC=1, 1.05, 0.15, 0.15,2, 0, 0.2, 0.2, 3, 0.9, 0.3, 0.3, 4, 0, 0.3, 0.3

ROLD={{1, 0.6, 0.45, 0.45},{3, 0, 0.9, 0.9}}

ROLE={{1, 0, 0.6, 0.6},{5, 0, 0.8, 0.8}}

Ta xây dựng cây liệt kê ban đầu như Hình 2:



Hình 2. *Cây liệt kê ban đầu*

+ Đi vào nhánh A ta tạo được các nhánh con AB, AC, AD, AE với *RO-list* như sau:

ROLA={1,1.2, 0.3, 0.3, 2, 0.5,0.4,0.4, 4, 0.3, 0.6, 0.6, 5, 1.1, 0.4, 0.4}

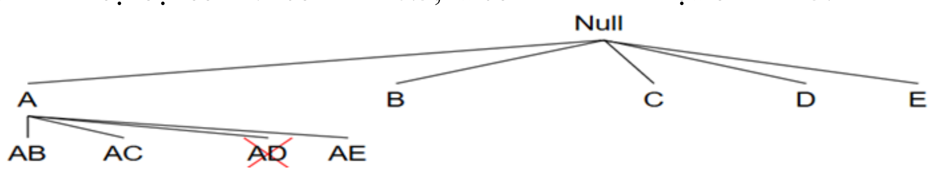
ROLAB=2,0.2, 0.7, 0.3, 5, 0.8, 0.7, 0.3

ROLAC={1,1.05, 0.45, 0.15, 2, 0, 0.6, 0.2, 4, 0, 0.9, 0.3}

ROLAD={{1, 0.6, 0.75, 0.45}}

ROLAE={{1, 0, 0.9, 0.6},{5, 0, 1.2, 0.8}}

Nhánh AD bị loại sớm vì có ws < 0.3, ta có hình minh họa ở Hình 3.



Hình 3. *Cây liệt kê khi đi vào nhánh A*

Tiếp theo, đi vào nhánh con AB, hai nhánh con ABC và ABE được hình thành nhưng đều loại, và tương tự các nhanh AC, AE…

Cuối cùng thuật toán kết thúc khi không có nhánh mới được sinh ra. $HWOP_s$chứa 4 mẫu trọng số chiếm dụng cao lần lượt là {AB, AC, AE, CD}.

3. Kết quả và thảo luận

3.1. Cơ sở thực nghiệm

Tất cả các thí nghiệm trong phần tiếp theo đều được thực hiện trên một hệ thống có cấu hình CPU Intel Core i5 2.4 GHz, bộ nhớ RAM 8G, chạy Windows 10 và sử dụng Java JDK 23.0.2. Bảng 3 sẽ trình bày chi tiết về các CSDL.

Bảng 3. *CSDL thực nghiệm*

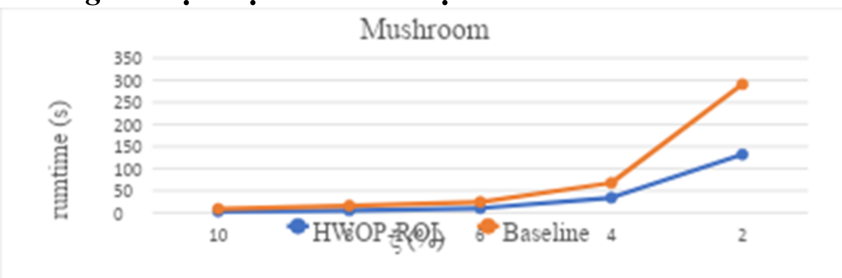
Tên	Số giao dịch	Số danh mục
Mushrooms	8,416	119
Kosarak	990,002	41,270
Retail	88,162	16,470

Ở phần này chúng tôi trình bày kết quả thực nghiệm của thuật toán *HWOP-ROL* so với thuật toán nền chúng tôi gọi là Baseline. Thuật toán Baseline dựa trên hướng tiếp cận Eclat khai thác tất cả các mẫu trọng số phổ biến sau đó lọc ra các mẫu trọng số chiếm dụng cao. Cả thuật toán Eclat và HWOP đều sử dụng kĩ thuật nén bít với cấu trúc bit vector EIWS (Nguyen et al., 2022), đây là kĩ thuật nén bít hiệu quả nhất trên Eclat hiện nay.

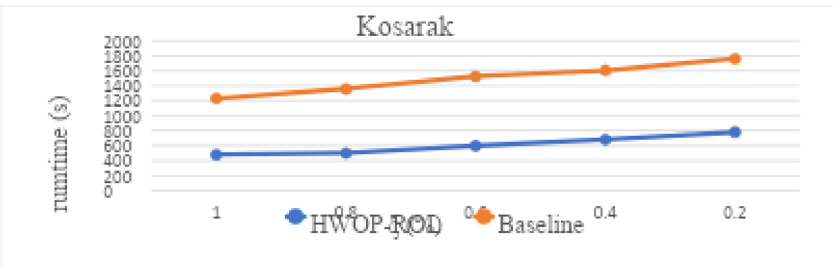
Trước khi khai thác, các mục được tái sắp xếp (theo thứ tự từ điển) và trong Eclat, mục cũng được sắp theo độ phổ biến trước khi nén lên IT-tree. Vì vậy, ở góc nhìn thuật toán, phân bố sinh trọng số không thay đổi cơ chế cắt tỉa của UBWO hay luồng xử lí chính. Trên cơ sở đó, chúng tôi sinh ngẫu nhiên trọng số các mục trong khoảng [0.5,5] đối với CSDL thực nghiệm trình bày trong Bảng 3. Trước khi thêm trọng số cho các mục, các

CSDL thực nghiệm này là các CSDL nhị phân lấy từ https://www.philippe-fournier-viger.com/spmf/index.php?link=datasets.php

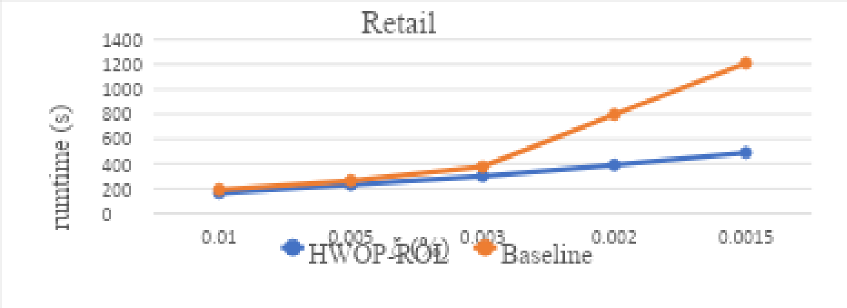
2. So sánh thời gian thực hiện của hai thuật toán



Hình 4. *So sánh về thời gian giữa 2 thuật toán trên CSDL Mushrooms*



Hình 5. *So sánh về thời gian giữa 2 thuật toán trên CSDL Kosarak*

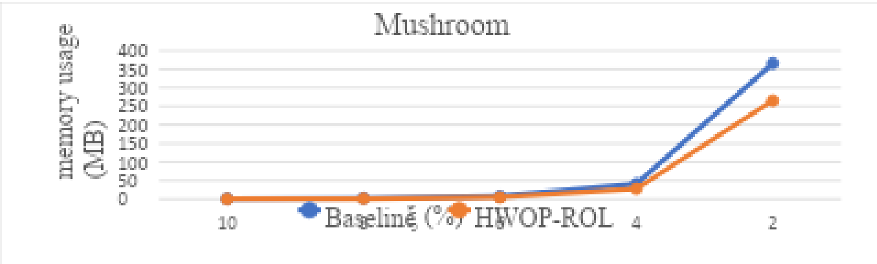


Hình 6. *So sánh về thời gian giữa 2 thuật toán trên CSDL Retail*

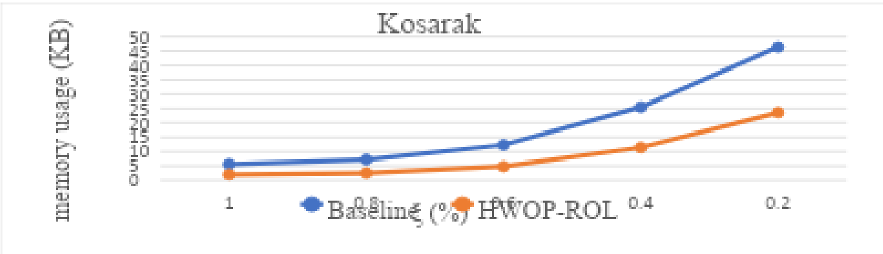
Kết quả thực nghiệm trong các hình 4-6 cho thấy thuật toán *HWOP-ROL* tối ưu rất đáng kể về mặt thời gian trong quá trình khai thác mẫu trọng số chiếm dụng cao so với thuật toán nền tảng.

Thuật toán giao giữa hai *RO-list* cũng được xây dựng tối ưu để đạt được độ phức tạp tuyến tính, giúp cho thuật toán hoạt động hiệu quả. Khi mở rộng mẫu trong cùng giao dịch, *wot* được cập nhật tức thời nhờ *so*, giúp tránh bao hàm–loại trừ và không phải đọc lại giao dịch. Kết hợp với giao ROL tuyến tính theo tid, phần lõi giảm về một phép cộng O(1) mỗi lần mở rộng. Điều này lí giải việc HWOP-ROL duy trì thời gian tốt hơn Baseline trong các thí nghiệm.

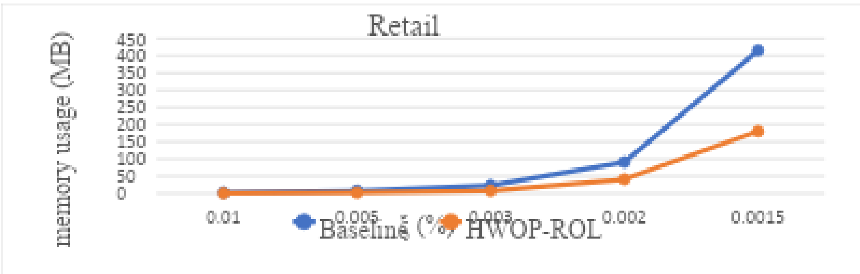
3. So sánh Bộ nhớ của hai thuật toán



Hình 7. *So sánh về bộ nhớ giữa 2 thuật toán trên CSDL Mushroom*



Hình 8. *So sánh về bộ nhớ giữa 2 thuật toán trên CSDL Kosarak*



Hình 9. *So sánh về bộ nhớ giữa 2 thuật toán trên CSDL Retail*

So sánh về thời gian thực thi của hai thuật toán thể hiện trên các Hình 7-9. Các biểu đồ cho thấy ưu thế của *HWOP-ROL* so với Baseline. Điều này có thể giải thích như sau: Cấu trúc *RO-list* của *HWOP-ROL* có chi phí lưu trữ trên mỗi node cao hơn, do phải duy trì ba trường thông tin (*ro*, *wot*, *so*) so với cấu trúc tidset cơ bản của Baseline (Eclat) nhưng giảm cấu trúc trung gian nhờ cắt tỉa và cập nhật O(1) do sử dụng *so*.

4. Kết luận

Trong bài báo này chúng tôi đã trình bày một bài toán mới, với cách tiếp cận mới về khai thác mẫu trọng số chiếm dụng cao trên WD. Để giải quyết bài toán này, chúng tôi đã đề xuất thuật toán mới mang tên *HWOP-ROL*. Bằng cách sử dụng cấu trúc dữ liệu *RO-list* và ngưỡng chặn trên *UBWO*. Các kết quả thực nghiệm về thời gian thực thi và bộ nhớ đều cho thấy sự hiệu quả của thuật toán đề xuất.

Trong thời gian tới, chúng tôi sẽ phát triển bài toán trên các loại CSDL phức tạp hơn như CSDL có trọng số động, CSDL định lượng. Ngoài ra chúng tôi sẽ tìm cách cải tiến thêm về mặt hiệu suất khai thác bằng cách song song hóa thuật toán.

❖ ***Tuyên bố về quyền lợi:*** *Tác giả xác nhận hoàn toàn không có xung đột về quyền lợi.*

❖ ***Lời cảm ơn:*** *Nghiên cứu này được tài trợ bởi Trường Đại học Sài Gòn theo đề tài CSB2025-01.*

TÀI LIỆU THAM KHẢO

Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules. *Proceedings of the 20th International Conference on Very Large Data Bases (VLDB'94)*, 487–499.

Bui, H., Vo, B., Nguyen, H., Nguyen-Hoang, T. A., & Hong, T. P. (2018). A weighted N-list-based method for mining frequent weighted itemsets. *Expert Systems with Applications*, *96*, 388–405. https://doi.org/10.1016/j.eswa.2017.10.039

Deng, Z. H. (2020). Mining high occupancy itemsets. *Future Generation Computer Systems*, *102*, 222–229. https://doi.org/10.1016/j.future.2019.07.039

Deng, Z. H., Wang, Z. H., & Jiang, J. J. (2012). A new algorithm for fast mining frequent itemsets using N-lists. *Science China Information Sciences*, *55*(9), 2008–2030. https://doi.org/10.1007/s11432-012-4638-z

Grahne, G., & Zhu, J. (2005). Fast algorithms for frequent itemset mining using FP-trees. *IEEE Transactions on Knowledge and Data Engineering*, *17*(10), 1347–1362. https://doi.org/10.1109/TKDE.2005.166

Nguyen, H., Le, T., Nguyen, M., Fournier-Viger, P., Tseng, V. S., & Vo, B. (2022). Mining frequent weighted utility itemsets in hierarchical quantitative databases. *Knowledge-Based Systems*, *237*, 107709. https://doi.org/10.1016/j.knosys.2021.107709

Nguyen, H., Vo, B., Nguyen, M., & Pedrycz, W. (2016). An efficient algorithm for mining frequent weighted itemsets using interval word segments. *Applied Intelligence*, *45*(4), 1008–1020. https://doi.org/10.1007/s10489-016-0799-6

Nguyen, L. T., Mai, T., Pham, G. H., Yun, U., & Vo, B. (2023). An efficient method for mining high occupancy itemsets based on equivalence class and early pruning. *Knowledge-Based Systems*, *267*, 110441. https://doi.org/10.1016/j.knosys.2023.110441

Ramkumar, G. D., Ranka, S., & Tsur, S. (1998). Weighted association rules: Model and algorithm. In *Proceedings of the Fourth ACM International Conference on Knowledge Discovery and Data Mining (KDD'98)* (pp. 1–13).

Tang, L., Zhang, L., Luo, P., & Wang, M. (2012). Incorporating occupancy into frequent pattern mining for high-quality pattern recommendation. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management (CIKM'12)* (pp. 75–84). https://doi.org/10.1145/2396761.2396775

Vo, B., Coenen, F., & Le, B. (2013). A new method for mining frequent weighted itemsets based on WIT-trees. *Expert Systems with Applications*, *40*(4), 1256–1264. https://doi.org/10.1016/j.eswa.2012.08.065

Zaki, M. J. (2000). Scalable algorithms for association mining. *IEEE Transactions on Knowledge and Data Engineering*, *12*(3), 372–390. https://doi.org/10.1109/69.846291

Zhang, C., Yang, Y., & Du, Z. (2024). HUSP-SP: Faster utility mining on sequence data. *ACM Transactions on Knowledge Discovery from Data*, *18*, 1–21. https://doi.org/10.1145/359793

HIGH OCCUPANCY PATTERN MINING ON WEIGHTED TRANSACTIONAL DATABASES

Le Tan Long

Saigon University, Vietnam

Corresponding author: Le Tan Long – Email: dungnt@hcmue.edu.vn

Received: September 29, 2025; Revised: November 14, 2025; Accepted: November 18, 2025

ABSTRACT

High Occupancy Itemset (HOI) mining is an emerging research direction in data mining that has garnered considerable attention. In contrast to frequent patterns, which are measured by their occurrence frequency, HOIs are defined as itemsets that occupy a significant proportion of the lengths of transactions in which they appear. While typically less numerous than frequent patterns, HOIs often possess more meaningful characteristics, effectively supporting tasks such as data analysis and visualization in intelligent systems. However, a key limitation of HOIs is that they only consider the presence of items, failing to reflect the differences in importance or weight among them. To address this limitation, this paper introduces the concept of High Weighted Occupancy Patterns (HWOPs) and proposes the HWOP-ROL algorithm for their efficient discovery. Furthermore, we introduce a tight upper-bound, named UBWO, to effectively prune the search space. Experimental results on various benchmark weighted datasets demonstrate the superior efficiency of the proposed approach when compared to a baseline algorithm.

Keywords: High Occupancy Pattern; high weighted-occupancy pattern; HWOP-ROL algorithm; upper bound threshold of weighted occupancy (UBWO)