

ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

Lê Hoàng Bình Nguyên

PHÁT TRIỂN THUẬT TOÁN KHAI THÁC
HIỆU QUẢ MÃ TRỌNG SỐ HỮU ÍCH PHỔ BIẾN
TRÊN LUỒNG DỮ LIỆU

Chuyên ngành: Cơ sở toán học cho tin học

Mã số: 9460110

TÓM TẮT LUẬN ÁN TIẾN SĨ TOÁN TIN

Hà Nội - 2026

Công trình được hoàn thành tại: **Trường Đại học Khoa học Tự nhiên,
Đại học Quốc gia Hà Nội.**

Người hướng dẫn khoa học 1: PGS.TS. Nguyễn Thị Hồng Minh

Người hướng dẫn khoa học 2: PGS.TS. Nguyễn Duy Hàm

Phản biện: PGS.TS. Ngô Quốc Dũng
Học viện Công nghệ Bưu chính Viễn thông

Phản biện: PGS.TS. Trần Thị Lượng
Học viện Kỹ thuật mật mã

Phản biện: PGS.TS. Đỗ Năng Toàn
Viện Hàn lâm Khoa học và Công nghệ Việt Nam

Luận án sẽ được bảo vệ trước Hội đồng cấp Đại học Quốc gia chấm
luận án tiến sĩ họp tại
.....

vào hồi giờ ngày tháng năm 20...

Có thể tìm hiểu luận án tại:

- Thư viện Quốc gia Việt Nam
- Trung tâm Thông tin - Thư viện, Đại học Quốc gia Hà Nội

MỞ ĐẦU

1. Lý do chọn đề tài

Khai thác dữ liệu là quá trình phát hiện các mẫu tiềm ẩn và tri thức có giá trị từ khối lượng dữ liệu lớn, hỗ trợ dự đoán xu hướng và ra quyết định. Khai thác mẫu phổ biến (Frequent Pattern - FP) là một trong những bài toán quan trọng với mục tiêu tìm các tập mẫu thỏa mãn ngưỡng hỗ trợ tối thiểu trong cơ sở dữ liệu (CSDL). Tuy nhiên, mô hình FP truyền thống bộc lộ hạn chế khi giả định các mục có tầm quan trọng như nhau, điều này không phù hợp với dữ liệu thực tế khi các mục có sự khác biệt về giá trị, số lượng hoặc lợi nhuận. Do đó, nhiều hướng mở rộng đã được nghiên cứu như mẫu có trọng số, mẫu hữu ích cao (High Utility Pattern - HUP), mẫu tuần tự.

Khai thác mẫu trọng số hữu ích phổ biến (Frequent Weighted Utility Pattern - FWUP) trên CSDL định lượng là một hướng tiếp cận quan trọng, khi mỗi mục được gán trọng số và số lượng cụ thể. Bài toán này nhằm tìm các mẫu vừa xuất hiện phổ biến, vừa có tầm quan trọng cao. Đã có nhiều thuật toán như được đề xuất nhằm nâng cao hiệu quả khai thác, nhưng các nghiên cứu hiện tại chủ yếu giả định trọng số của các mục là cố định. Trên thực tế, các giá trị này có thể thay đổi theo thời gian do ảnh hưởng của hành vi người dùng, yếu tố mùa vụ hoặc chiến lược kinh doanh. Việc bỏ qua đặc tính động này làm giảm tính ứng dụng, đặt ra yêu cầu cần phát triển các thuật toán khai thác FWUP thích ứng với sự thay đổi trọng số. Bên cạnh đó, số lượng mẫu sinh ra trong quá trình khai thác thường rất lớn, gây khó khăn cho lưu trữ và phân tích. Khai thác mẫu đóng là một giải pháp hiệu quả giúp nén không gian mẫu mà vẫn bảo toàn thông tin. Mặc dù đã có nhiều nghiên cứu về mẫu đóng trong các bối cảnh khác nhau, nhưng bài toán khai thác mẫu trọng số hữu ích phổ biến đóng

(Frequent Weighted Utility Closed Pattern - FWUCP), đặc biệt trong môi trường dữ liệu động, vẫn chưa được nghiên cứu đầy đủ.

Cùng với đó, sự phát triển của các hệ thống sinh dữ liệu liên tục đã dẫn đến nhu cầu khai thác trên luồng dữ liệu. Trong môi trường này, việc lưu trữ toàn bộ dữ liệu là không khả thi, đòi hỏi các thuật toán phải cập nhật theo thời gian thực, sử dụng bộ nhớ hạn chế và chỉ quét dữ liệu một lần. Đối với luồng dữ liệu định lượng động (dynamic Quantitative Data Stream - dQDS), bài toán phải xử lý cả nội dung dữ liệu và trọng số của các mục có thể thay đổi theo thời gian. Việc áp dụng mô hình cửa sổ trượt để khai thác FWUP trên dQDS là hướng tiếp cận tiềm năng nhưng còn nhiều thách thức. Đồng thời, việc xác định ngưỡng khai thác phù hợp trong môi trường biến động là khó khăn, do đó khai thác top- k FWUP là giải pháp hiệu quả giúp người dùng tiếp cận trực quan và linh hoạt hơn.

Xuất phát từ các vấn đề lý thuyết và thực tiễn nêu trên, nghiên cứu sinh lựa chọn đề tài: **“Phát triển thuật toán khai thác hiệu quả mẫu trọng số hữu ích phổ biến trên luồng dữ liệu”** làm đề tài nghiên cứu cho luận án tiến sĩ.

2. Mục tiêu nghiên cứu

- Nghiên cứu bài toán tổng quát về khai thác FWUP từ CSDL định lượng động, với trọng số của các mục có thể thay đổi. Từ đó, phát triển các cấu trúc dữ liệu, phương pháp và thuật toán để giải quyết bài toán này. Đồng thời mở rộng sang bài toán khai thác FWUCP, là dạng rút gọn nhưng không làm mất thông tin của FWUP.

- Nghiên cứu bài toán khai thác FWUP trên dQDS sử dụng mô hình cửa sổ trượt với dữ liệu và trọng số các mục thay đổi theo thời gian. Trên cơ sở đó, phát triển các thuật toán và phương pháp cập nhật, khai thác hiệu quả, đồng thời mở rộng sang bài toán khai thác top- k

FWUP trên dQDS nhằm giảm sự phụ thuộc của ngưỡng tối thiểu.

3. Đối tượng và phạm vi nghiên cứu

- **Đối tượng nghiên cứu:** Các thuật toán khai thác FWUP; các phương pháp khai thác mẫu đóng và top- k trên CSDL nhị phân, có trọng số và định lượng; các kỹ thuật khai thác mẫu trên luồng dữ liệu.

- **Phạm vi nghiên cứu:** Tập trung trên CSDL định lượng tĩnh và động với trọng số thay đổi, cũng như dQDS; sử dụng các bộ dữ liệu định lượng đã được chuẩn hóa phục vụ nghiên cứu.

4. Phương pháp luận và phương pháp nghiên cứu

- **Phương pháp lý thuyết:** Tổng hợp và phân tích các công trình nghiên cứu liên quan; khảo sát đặc trưng của CSDL định lượng động và luồng dữ liệu có trọng số thay đổi; đánh giá các phương pháp hiện có nhằm xác định ưu điểm và hạn chế, từ đó đề xuất các mô hình và kỹ thuật khai thác FWUP hiệu quả.

- **Phương pháp thực nghiệm:** Xây dựng và cài đặt các thuật toán đề xuất; tiến hành thực nghiệm trên các bộ dữ liệu nhằm đánh giá hiệu năng, tính hiệu quả và khả năng mở rộng của các phương pháp.

5. Các đóng góp của luận án

Luận án tập trung nghiên cứu bài toán khai thác FWUP trong môi trường dữ liệu động và phức tạp. Các đóng góp chính bao gồm:

- Phát triển các phương pháp khai thác FWUP từ CSDL định lượng động. Khác với các nghiên cứu trước đây giả định trọng số là cố định, luận án đã xây dựng cơ chế xử lý sự biến động của trọng số theo thời gian. Trên cơ sở đó, các thuật toán dFWUT sử dụng cấu trúc Tidset và dFWUNL sử dụng cấu trúc WUNList được đề xuất nhằm giải quyết hiệu quả bài toán này. Tiếp đó, hướng nghiên cứu được mở rộng sang bài toán khai thác FWUCP với thuật toán FWUCWP và các chiến lược cắt tia nhằm thu hẹp không gian tìm kiếm.

- Nghiên cứu bài toán khai thác FWUP trên dQDS sử dụng mô hình cửa sổ trượt và đề xuất hai thuật toán FWUDS-CT và FWUDS-DWT để giải quyết bài toán này. Thuật toán FWUDS-CT sử dụng cấu trúc Circular Tidset cải tiến nhằm quản lý hiệu quả định danh giao dịch, tránh hiện tượng tràn số. Thuật toán FWUDS-DWT dựa trên cấu trúc DSWUN-tree và sự hiệu quả của cấu trúc WUNList, cho phép cập nhật dữ liệu nhanh chóng và thực hiện khai thác chỉ với một lần quét dữ liệu. Từ đó, luận án phát triển thuật toán TKFWUDS để khai thác top- k FWUP trên dQDS, sử dụng cơ chế nâng ngưỡng động kết hợp các chiến lược cắt tia, giúp loại bỏ sự phụ thuộc vào ngưỡng tối thiểu.

- Kết quả thực nghiệm trên nhiều bộ dữ liệu khác nhau cho thấy các phương pháp đề xuất đạt hiệu quả cao hơn về thời gian xử lý, sử dụng bộ nhớ và khả năng mở rộng so với các phương pháp hiện có.

6. Cấu trúc luận án

Ngoài phần Mở đầu và Kết luận, cấu trúc luận án bao gồm ba chương và tài liệu tham khảo:

Chương 1: “Tổng quan về lĩnh vực nghiên cứu” trình bày bài toán khai thác mẫu; khảo sát và phân tích các công trình liên quan; chỉ ra các hạn chế của phương pháp hiện có làm cơ sở cho các đề xuất.

Chương 2: “Khai thác mẫu trọng số hữu ích phổ biến từ cơ sở dữ liệu định lượng động” trình bày các cấu trúc dữ liệu Tidset và WUNList; đề xuất các thuật toán dFWUT, dFWUNL cho bài toán FWUP và FWUCWP cho bài toán FWUCP.

Chương 3: “Khai thác mẫu trọng số hữu ích phổ biến trên luồng dữ liệu định lượng động” xây dựng mô hình, đề xuất các cấu trúc cải tiến và thuật toán khai thác FWUP và top- k FWUP trên dQDS dựa trên mô hình cửa sổ trượt.

CHƯƠNG 1: TỔNG QUAN VỀ LĨNH VỰC NGHIÊN CỨU

1.1 Bài toán khai thác mẫu trên CSDL định lượng

1.1.1 Bài toán khai thác mẫu hữu ích cao

Khai thác HUP tập trung tìm kiếm các mẫu có giá trị hữu ích không nhỏ hơn ngưỡng hữu ích tối thiểu. Tuy nhiên, do không thỏa mãn tính chất bao đóng giảm, bài toán này có không gian tìm kiếm rất lớn. Nhiều phương pháp đã được đề xuất như Two-Phase, UP-Growth, HUI-Miner và các cải tiến như FHM, EFIM. Các nghiên cứu gần đây tiếp tục tối ưu bằng các cấu trúc dữ liệu hiệu quả và chiến lược cắt tỉa tiên tiến, giúp giảm chi phí tính toán và bộ nhớ, đồng thời mở rộng ứng dụng của HUP sang nhiều bài toán khác.

1.1.2 Bài toán khai thác mẫu trọng số hữu ích phổ biến

Khai thác FWUP kết hợp đồng thời tần suất xuất hiện và tầm quan trọng của các mẫu. Phương pháp này được Khan và đồng sự đề xuất sử dụng hai độ đo mới là trọng số hữu ích giao dịch (transaction weighted utility - *twu*) và độ hỗ trợ trọng số hữu ích (weighted utility support - *wus*), đồng thời thỏa mãn tính chất bao đóng giảm, giúp thu hẹp không gian tìm kiếm. Nhờ đó, bài toán FWUP có ưu thế hơn so với HUP về khả năng kiểm soát số lượng mẫu sinh ra và chi phí tính toán. Tác giả cũng đề xuất thuật toán dựa trên Apriori nhưng còn hạn chế về hiệu năng. Các nghiên cứu sau đó tập trung cải tiến cấu trúc dữ liệu và thuật toán. Phương pháp dựa trên Tidset như MWIT-tree giúp giảm số lần quét dữ liệu; cấu trúc MBiS tối ưu hóa lưu trữ thông tin giao dịch liên tiếp, cải thiện hiệu quả bộ nhớ. Trong khi đó, các phương pháp lai như WUN-Miner, FWUP-SWUNL kết hợp ưu điểm của cấu trúc cây và danh sách để tăng tốc khai thác, giảm đáng kể tài nguyên sử dụng. Ngoài ra, FWUP còn được mở rộng sang nhiều bối cảnh khác như dữ liệu phân cấp hoặc trong phân cụm văn bản, cho thấy tiềm năng

ứng dụng rộng rãi của hướng tiếp cận này.

1.2 Bài toán khai thác mẫu trên luồng dữ liệu

Luồng dữ liệu có các đặc trưng như kích thước vô hạn, tốc độ cao, liên tục theo thời gian và biến động nhanh, khiến các phương pháp khai thác truyền thống không còn phù hợp do hạn chế về lưu trữ và chi phí tính toán. Các nghiên cứu hiện nay tập trung vào ba mô hình chính: mô hình điểm mốc, mô hình suy giảm và mô hình cửa sổ trượt. Mô hình điểm mốc xem xét toàn bộ dữ liệu từ một thời điểm bắt đầu cho đến hiện tại, phù hợp khi dữ liệu có mức độ quan trọng như nhau. Mô hình suy giảm gán trọng số giảm dần cho dữ liệu cũ, phản ánh tốt tính thời gian nhưng đòi hỏi chi phí cập nhật và quản lý cao. Trong khi đó, mô hình cửa sổ trượt chỉ duy trì một tập dữ liệu gần nhất trong cửa sổ có kích thước cố định, cho phép khai thác hiệu quả các xu hướng mới nhất và được sử dụng rộng rãi trong thực tế.

Nhiều thuật toán đã được đề xuất dựa trên các mô hình này nhằm khai thác FP và các biến thể như HUP, mẫu có thể xóa, với các cấu trúc dữ liệu và kỹ thuật cắt tía giúp giảm số lần quét dữ liệu, tối ưu bộ nhớ và hỗ trợ xử lý thời gian thực. Tuy nhiên, cho đến nay chưa có nghiên cứu nào giải quyết bài toán FWUP trên dQDS, nơi vừa tồn tại tính chất dòng dữ liệu liên tục, vừa có sự thay đổi của trọng số theo thời gian. Đây chính là khoảng trống nghiên cứu quan trọng, làm cơ sở cho các đề xuất của luận án.

1.3 Các biến thể của bài toán khai thác mẫu

1.3.1 Bài toán khai thác mẫu đóng

Bài toán khai thác FP đóng được đề xuất nhằm tìm tập mẫu không dư thừa nhưng vẫn bảo toàn đầy đủ thông tin của các FP. Một mẫu được gọi là đóng nếu không tồn tại tập cha có cùng độ hỗ trợ. Bài toán khai thác mẫu đóng cũng được mở rộng sang nhiều biến thể khác. Tuy

nhien, hiện không có nghiên cứu nào liên quan đến khai thác FWUCP.

1.3.2 Bài toán khai thác top- k mẫu

Bài toán khai thác top- k mẫu được đề xuất để trực tiếp tìm ra k mẫu tốt nhất, loại bỏ sự phụ thuộc vào ngưỡng cố định. Cho đến nay, chưa có nghiên cứu nào giải quyết bài toán khai thác top- k FWUP.

1.4 Khoảng trống nghiên cứu, thách thức và hướng giải quyết

Từ việc khảo sát các công trình liên quan, luận án xác định hai hướng nghiên cứu chính có tính kế thừa và phát triển: (1) Khai thác FWUP trên CSDL định lượng động nhằm khắc phục hạn chế của các phương pháp truyền thống với trọng số cố định, đồng thời mở rộng sang khai thác FWUCP để giảm số lượng mẫu mà vẫn bảo toàn thông tin; (2) Khai thác FWUP và top- k FWUP trên dQDS sử dụng mô hình cửa sổ trượt, đáp ứng yêu cầu cập nhật liên tục, xử lý một lần, đồng thời khắc phục khó khăn trong việc xác định ngưỡng khai thác.

1.5 Kết luận chương

Chương 1 đã tiến hành khảo sát các nghiên cứu về khai thác mẫu trên CSDL định lượng và luồng dữ liệu. Kết quả cho thấy các phương pháp hiện tại còn hạn chế khi giả định trọng số cố định và chưa xử lý tốt trong môi trường dữ liệu động. Bên cạnh đó, các hướng như khai thác mẫu đóng và top- k mẫu nhằm giảm dư thừa và loại bỏ phụ thuộc vào ngưỡng tối thiểu cũng được phân tích. Tuy nhiên, việc kết hợp các yếu tố như trọng số động, mẫu đóng, top- k và luồng dữ liệu trong bài toán FWUP vẫn chưa được nghiên cứu đầy đủ. Đây là cơ sở để luận án đề xuất các phương pháp và thuật toán trong các chương tiếp theo.

CHƯƠNG 2: KHAI THÁC MẪU TRỌNG SỐ HỮU ÍCH PHỔ BIẾN TỪ CƠ SỞ DỮ LIỆU ĐỊNH LƯỢNG ĐỘNG

2.1 Giới thiệu bài toán

Bài toán khai thác FWUP là mở rộng của khai thác FP trên CSDL định lượng. Các phương pháp hiện tại chủ yếu giả định trọng số các mục là cố định, chưa phù hợp với thực tế khi giá trị này có thể thay đổi theo do các yếu tố như giá cả, khuyến mãi hoặc nhu cầu thị trường. Để giải quyết vấn đề này, luận án mở rộng mô hình CSDL định lượng sang dạng động, trong đó trọng số được tổ chức theo các lô tương ứng với từng giai đoạn, phản ánh sự biến đổi theo thời gian. Trên cơ sở đó, chương này đề xuất hai thuật toán dFWUT và dFWUNL dựa trên các cấu trúc dữ liệu Tidset và WUNList nhằm khai thác hiệu quả FWUP từ CSDL định lượng động. Đồng thời, luận án nghiên cứu bài toán khai thác mẫu FWUCP nhằm giảm số lượng mẫu nhưng vẫn bảo toàn thông tin, thông qua các chiến lược cắt tĩa và kiểm tra tính đóng hiệu quả. Kết quả thực nghiệm cho thấy các phương pháp đề xuất cải thiện đáng kể hiệu năng so với các hướng tiếp cận trước đó.

2.2 Một số định nghĩa cơ bản

Định nghĩa 2.1 (CSDL định lượng động). CSDL định lượng động, ký hiệu là $dQDB$, là một bộ gồm ba thành phần (T, I, DW) . Trong đó $T = \{t_1, t_2, \dots, t_m\}$ là tập các giao dịch định lượng với m là số lượng giao dịch, $I = \{i_1, i_2, \dots, i_n\}$ là tập các mục với n là số lượng mục, và $DW = \{dw_1, dw_2, \dots, dw_q\}$ là các tập trọng số với q là số lượng lô. Mỗi giao dịch t_k có một định danh duy nhất là Tid , có dạng $t_k = \{x_{k1}, x_{k2}, \dots, x_{kn}\}$, $k = 1, 2, \dots, m$, trong đó x_{ki} là số lượng của mục thứ i trong giao dịch t_k , và tương ứng với một $dw_j \in DW$. Mỗi lô $dw_j = \{w_{j1}, w_{j2}, \dots, w_{jn}\}$, $j = 1, 2, \dots, q$, là một tập trọng số của các mục trong I .

Định nghĩa 2.2 (Trọng số hữu ích giao dịch). Giá trị trọng số hữu ích giao dịch, ký hiệu là twu , của một giao dịch t_k với tập trọng số $dw_p \in DW$ được xác định như sau:

$$twu(t_k) = \frac{\sum_{i \in t_k} w_{pi} \times x_{ki}}{|t_k|} \quad (2.1)$$

trong đó, w_{pi} là trọng số của mục i trong giao dịch t_k , x_{ki} là số lượng của mục i trong t_k , và $|t_k|$ là số lượng các mục xuất hiện trong t_k .

Định nghĩa 2.3 (Độ hỗ trợ trọng số hữu ích). Với $sumtwu = \sum_{t_k \in T} twu(t_k)$, độ hỗ trợ trọng số hữu ích, ký hiệu là wus , của một mẫu P được xác định như sau:

$$wus(P) = \frac{\sum_{t_k \in T(P)} twu(t_k)}{sumtwu} \quad (2.2)$$

trong đó, $T(P)$ là tập các giao dịch trong CSDL có chứa P .

Định nghĩa 2.4 (Mẫu trọng số hữu ích phổ biến). Cho một ngưỡng trọng số hữu ích tối thiểu, ký hiệu là $minwus$, và một mẫu $P \subseteq I$, nếu $wus(P) \geq minwus$ thì P được gọi là FWUP.

Định nghĩa 2.5 (Bài toán khai thác FWUP). Khai thác FWUP từ CSDL định lượng động là bài toán tìm tất cả các FWUP thỏa mãn giá trị $minwus$ cho trước do người dùng chỉ định.

Định nghĩa 2.6 (Mẫu trọng số hữu ích phổ biến đóng). Cho một mẫu $P \subseteq I$ là một FWUP, P được gọi là FWUCP khi và chỉ khi không tồn tại một mẫu Q nào mà $P \subset Q$ và $wus(P) = wus(Q)$.

Định nghĩa 2.7 (Bài toán khai thác FWUCP). Khai thác FWUCP từ CSDL định lượng động là bài toán tìm tất cả các FWUCP thỏa mãn ngưỡng $minwus$ cho trước.

2.3 Thuật toán khai thác mẫu trọng số hữu ích phổ biến từ cơ sở dữ liệu định lượng động

2.3.1 Thuật toán dFWUT

Thuật toán dFWUT sử dụng cấu trúc Tidset để khai thác FWUP từ CSDL định lượng động được mô tả ở Hình 2.1.

Thuật toán 2.1. Thuật toán dFWUT

Input: CSDL định lượng động <i>dQDB</i> và ngưỡng <i>minwus</i> .
Output: Tập tất cả các mẫu trọng số hữu ích phổ biến <i>FWUPs</i> .
1. Quét <i>dQDB</i> để tính <i>twu</i> của mỗi giao dịch, giá trị <i>wus</i> và <i>Tidset</i> của mỗi mục 2. Lọc các mục có <i>wus</i> $\geq$ <i>minwus</i> và thêm vào C_1 3. $FWUPs \leftarrow C_1$ 4. $dWUPT\text{-}tree \leftarrow \emptyset$ 5. Tid_Mine($[C_1]$) 6. return <i>FWUPs</i>
Procedure Tid_Mine ($[C]$) 1. $dWUPT\text{-}tree \leftarrow [C]$ 2. foreach $c_i \in [C]$ do 3. $[C_{next}] \leftarrow \emptyset$ 4. foreach $c_j \in [C]$, với $j > i$ do 5. $P \leftarrow c_i \cup c_j$ 6. $t(P) \leftarrow t(c_i) \cap t(c_j)$ 7. if <i>wus</i>(P) $\geq$ <i>minwus</i> then 8. $FWUPs \leftarrow P$ 9. $[C_{next}] \leftarrow \{P, t(P), wus(P)\}$ 10. Tid_Mine($[C_{next}]$)

Hình 2.1. Thuật toán dFWUT.

2.3.2 Thuật toán dFWUNL

Định nghĩa 2.8 (WUN-tree). WUN-tree là một cấu trúc cây tiền tổ đa nhánh với một nút gốc và một tập hợp các nút con. Mỗi nút lưu trữ một bộ năm giá trị (*name*, *childlist*, *pre*, *post*, *weight*). Trong đó, *name* là tên mục đại diện cho nút, *childlist* là tập các nút con của nó, *pre* là thứ hạng tiền thứ tự của nút, *post* là thứ hạng hậu thứ tự của nút và *weight* là tổng giá trị *twu* của các giao dịch đi qua nút này.

Định nghĩa 2.9 (WUN-code). WUN-code của một nút *N* trong WUN-tree có dạng (*N.pre*, *N.post*, *N.weight*).

Định lý 2.1. Xét hai nút N_1 và N_2 thuộc WUN-tree với WUN-code lần lượt là ($pre_1, post_1, weight_1$) và ($pre_2, post_2, weight_2$). N_1 là tổ tiên của N_2 khi và chỉ khi $pre_1 < pre_2$ và $post_1 > post_2$.

Định nghĩa 2.10 (WUNList của một mục). WUNList của một mục *x*, ký hiệu là *wunl*(*x*), là dãy tất cả các WUN-code của nút đại diện cho mục *x* trong WUN-tree. WUNList được sắp xếp tăng dần

theo giá trị *pre*.

Định nghĩa 2.11 (WUNList của một *k*-pattern). Cho P_x và P_y là hai $(k-1)$ -pattern có cùng tiền tố P , và x đứng sau y theo thứ tự trong I_1 (ký hiệu là $y < x$). $wunl(P_x)$ và $wunl(P_y)$ lần lượt là WUNList của P_x và P_y . $wunl(P_{xy})$ là WUNList của k -pattern P_{xy} được tạo theo hai bước sau: (1) Với mỗi $C_u \in wunl(P_x)$ và $C_v \in wunl(P_y)$, nếu C_v là tổ tiên của C_u , ta thêm $(C_v.pre, C_v.post, C_u.weight)$ vào $wunl(P_{xy})$. (2) Duyệt tất cả WUN-code trong $wunl(P_{xy})$ và hợp nhất các WUN-code có cùng giá trị *pre* vào một WUN-code với giá trị trọng số mới bằng tổng trọng số của các nút đang xét.

Định lý 2.2. Cho $P = x_1x_2 \dots x_n$ là một mẫu với x_i đứng sau x_{i+1} theo I_1 , có WUNList là $wunl(P) = \{(pre_1, post_1, weight_1), (pre_2, post_2, weight_2), \dots, (pre_m, post_m, weight_m)\}$. Độ hỗ trợ trọng số hữu ích, $wus(P)$, của mẫu P được tính như sau:

$$wus(P) = \frac{\sum_{i=1}^m weight_i}{sumtwu} \quad (2.3)$$

Định lý 2.3. Cho một mẫu P và một mục i ($i \notin P$), $wus(P) = wus(P \cup \{i\})$; khi đó, với mọi mẫu X thoả mãn $(X \cap P) = \emptyset \wedge i \notin X$, thì $wus(X \cup P) = wus(X \cup P \cup \{i\})$.

Thuật toán dFWUNL sử dụng cấu trúc WUN-tree và WUNList để khai thác FWUP từ CSDL định lượng động được mô tả ở Hình 2.3.

Thuật toán 2.4. Thuật toán dFWUNL

Input: CSDL định lượng động $dQDB$ và ngưỡng *minwus*

Output: Tập $FWUPs$

1. Gọi **Build_WUN-tree**($dQDB, minwus$) để tạo I_1 và WUN-tree
2. $FWUPs \leftarrow I_1$
3. Duyệt WUN-tree để tạo WUNList của các mục trong I_1
4. Khởi tạo $F2[|I_1|][|I_1|]$
5. Duyệt WUN-tree theo phép duyệt tiền thứ tự pre-order
6. **foreach** nút N lưu mục i_x **do**
7. **foreach** nút tổ tiên N_a của N lưu mục i_y **do**
8. $F2[i_x][i_y] += N.weight$

<pre> 9. for $x = I_1 - 1$ downto 1 do 10. $Prefix_2 \leftarrow \{i_x\}, L_2 \leftarrow \emptyset, S_2 \leftarrow \emptyset$ 11. for $y = 0$ to $x - 1$ do 12. $wus(i_{xy}) \leftarrow F2[i_x][i_y]/sumtwu$ 13. if $wus(i_{xy}) = wus(i_x)$ then $S_2 \leftarrow i_y$ 14. else if $wus(i_{xy}) \geq minwus$ then 15. $i_{xy} \leftarrow i_y$ 16. $wunl(i_{xy}) \leftarrow \text{Construct_WUNList}(wunl(i_x), wunl(i_y))$ 17. $L_2 \leftarrow i_{xy}$ 18. $FWUPs \leftarrow i_x i_y$ 19. if $S_2 > 0$ then $\text{Find_FWUP_sameWUS}(Prefix_2, S_2)$ 20. if $L_2 > 0$ then $\text{Find_FWUP}(Prefix_2, L_2, S_2)$ 21. return $FWUPs$ </pre>
Procedure Find_FWUP ($Prefix_k, L_k, S_k$) <pre> 22. $Prefix_{next} \leftarrow Prefix_k$ 23. for $i = L_k - 1$ downto 1 do 24. $Prefix_{next} \leftarrow L_k[i], L_{next} \leftarrow \emptyset, S_{next} \leftarrow S_k$ 25. for $j = 0$ to $i - 1$ do 26. $C \leftarrow L_k[j]$ 27. $wunl(C) \leftarrow \text{Construct_WUNList}(wunl(L_k[i]), wunl(L_k[j]))$ 28. if $wunl(C) \neq NULL$ then 29. if $wus(C) == wus(L_k[i])$ then $S_{next} \leftarrow L_k[j]$ 30. else 31. $L_{next} \leftarrow C$ 32. $FWUPs \leftarrow \{Prefix_k \cup L_k[i] \cup L_k[j]\}$ 33. if $S_{next} > 0$ then $\text{Find_FWUP_sameWUS}(Prefix_{next}, S_{next})$ 34. if $L_{next} > 0$ then $\text{Find_FWUP}(Prefix_{next}, L_{next}, S_{next})$ 35. Xoá mục cuối cùng của $Prefix_{next}$ 36. $Prefix_{next} \leftarrow L_k[0]$ 37. if $S_k > 0$ then $\text{Find_FWUP_sameWUS}(Prefix_{next}, S_k)$ </pre>
Procedure Find_FWUP_sameWUS ($Prefix, S$) <pre> 38. Sinh tập <i>Child</i> lưu tất cả tập con của S 39. foreach $S_i \in Child$ do 40. $FWUPs \leftarrow \{Prefix \cup S_i\}$ </pre>

Hình 2.2. Thuật toán dFWUNL.

2.3.3 Kết quả thực nghiệm

Các thực nghiệm so sánh hai thuật toán dFWUT và dFWUNL với WUN-Miner-Mod được tiến hành trên các bộ dữ liệu Accident, BMS-POS, Chess, Retail, Kosarak và Chainstore. Kết quả cho thấy, thuật toán dFWUNL luôn đạt hiệu năng tốt nhất về thời gian chạy trên tất

cả các bộ dữ liệu. Về bộ nhớ, dFWUNL sử dụng bộ nhớ hiệu quả hơn đáng kể so với dFWUT và thường tốt hơn WUN-Miner-Mod, ngoại trừ một số trường hợp dữ liệu rất thưa. Đồng thời, thuật toán dFWUNL thể hiện khả năng mở rộng tốt với thời gian chạy tăng chậm và ổn định, trong khi mức sử dụng bộ nhớ trong giới hạn chấp nhận được.

2.4 Thuật toán khai thác mẫu trọng số hữu ích phổ biến đóng từ cơ sở dữ liệu định lượng động

Luận án đề xuất thuật toán có tên là FWUCWP được mô tả ở Hình 2.19 để khai thác FWUCP từ CSDL định lượng động. Thuật toán này được xây dựng dựa trên sự kế thừa trực tiếp và phát huy tối đa tính ưu việt của cấu trúc dữ liệu WUNList. Hơn nữa, FWUCWP còn được trang bị thêm các cơ chế nhằm loại bỏ sớm các nhánh không tiềm năng thông qua các chiến lược cắt tỉa chuyên biệt bao gồm:

Chiến lược cắt tỉa dựa trên quan hệ cấu trúc tổ tiên WUNList: Nếu WUNList của mẫu X có quan hệ cấu trúc tổ tiên với WUNList của mẫu Y ($X \subset Y$), thì điều đó ngụ ý rằng mỗi khi X xuất hiện thì Y cũng xuất hiện trong cùng các giao dịch. Do đó, X không thể là một FWUCP, bởi vì nó có một tập cha Y có cùng wus .

Chiến lược cắt tỉa kết hợp *twu* tối đa (Maximal Twu Combination Pruning - MTCP): Trước khi thực hiện phép kết hợp tồn kém giữa hai mẫu, MTCP đánh giá xem giá trị wus tối đa có thể đạt được khi kết hợp hai mẫu đó có còn thỏa mãn ngưỡng $minwus$ hay không. Nếu không đạt ngưỡng yêu cầu, mẫu mới kết hợp đó có thể được loại bỏ ngay lập tức một cách an toàn. Chiến lược này dựa trên tính chất phản đơn điệu và một cấu trúc mới gọi là Precomputed Twu Pairwise (PTP) lưu trữ các giá trị cận trên twu cho từng cặp mục.

Chiến lược cắt tỉa dựa trên khái niệm Enclosure: tận dụng mối quan hệ đồng xuất hiện chặt chẽ giữa các mục. Cụ thể, nếu mục x luôn

xuất hiện trong mọi giao dịch có chứa mục y , thì mọi mẫu đóng chứa x bắt buộc cũng phải chứa y . Ta có thể khai thác tính chất này để mở rộng mẫu trực tiếp mà không cần thực hiện tìm kiếm.

Thuật toán 2.8. FWUCWP
Input: CSDL định lượng động $dQDB$ và một ngưỡng $minwus$;
Output: Tập $FWUCPs$
1. $R \leftarrow \text{Build_WUN-tree}(dQDB, minwus)$ 2. Construct-PTP-Enclosure(R) 3. Duyệt WUN-tree R để tạo WUNList cho các mục trong I_1 4. $FWUCPs \leftarrow \emptyset$ 5. Mine_FWUCPs($\emptyset, \emptyset, I_1$) 6. return $FWUCPs$
Procedure Mine_FWUCPs ($Prefix, S, C$) 1. $Prefix_{next} \leftarrow Prefix$ 2. for $i \leftarrow C - 1$ downto 0 do 3. $Prefix_{next} \leftarrow Prefix \cup C_i, S_{next} \leftarrow S, C_{next} \leftarrow \emptyset$ 4. for $j \leftarrow i - 1$ downto 0 do 5. if $PTP[C_i][C_j]/sumtwu < minwus$ then continue 6. if $C_j \in C_i. Enc$ or Improved_WUNList_Check($C_i. wunl, C_j. wunl$) then 7. $S_{next} \leftarrow S_{next} \cup C_j$ 8. if $C_i. wus = C_j. wus$ then 9. Xoá C_j 10. $i--$ 11. continue 12. $Cand \leftarrow C_j$ 13. $Cand. wunl \leftarrow \text{Construct_WUNList}(C_i. wunl, C_j. wunl)$ 14. if $Cand. wunl \neq NULL$ then 15. $Cand. Enc \leftarrow C_i. Enc \cup C_j. Enc$ 16. $C_{next} \leftarrow Cand \cup C_{next}$ 17. $P \leftarrow Prefix_{next} \cup S_{next}, P. wus \leftarrow C_i. wus$ 18. if P là mẫu đóng trong $FWUCPs$ then 19. $FWUCPs \leftarrow FWUCPs \cup P$ 20. Mine_FWUCPs($Prefix_{next}, S_{next}, C_{next}$) 21. Xoá mục cuối cùng của $Prefix_{next}$

Hình 2.3. Thuật toán FWUCWP.

2.4.1 Kết quả thực nghiệm

Thuật toán FWUCWP được so sánh với thuật toán NFWCI để khai thác FWUCP trên các bộ dữ liệu định lượng động là Pumsb,

Retail, Accidents, BMS-POS, Kosarak và Chainstore. Kết quả thực nghiệm khẳng định thuật toán FWUCWP hiệu quả hơn so với NFWCI về thời gian chạy và khả năng mở rộng, đặc biệt là trên các tập dữ liệu thưa. Phân tích chuyên sâu cho thấy chiến lược MTCP đóng góp nhiều nhất vào việc loại bỏ các mẫu không thoả mãn, tiếp theo là chiến lược WSR, và cuối cùng là chiến lược dựa trên Enclosure.

2.5 Kết luận chương

Chương 2 đã giải quyết bài toán khai thác FWUP và FWUCP trên CSDL định lượng động với trọng số các mục có thể thay đổi. Luận án đề xuất hai thuật toán là dFWUT dựa trên cấu trúc Tidset và dFWUNL dựa trên WUNList để khai thác FWUP. Trong đó, dFWUNL cho hiệu năng vượt trội nhờ khả năng nén dữ liệu hiệu quả giúp tăng tốc tính toán. Cùng với đó, thuật toán FWUCWP được phát triển để khai thác FWUCP, nhằm giảm số lượng kết quả nhưng vẫn bảo toàn thông tin. Thuật toán kết hợp hiệu quả cấu trúc WUNList với các chiến lược cắt tĩa, qua đó thu hẹp đáng kể không gian tìm kiếm và nâng cao hiệu suất. Các kết quả đạt được không chỉ giải quyết hiệu quả bài toán trên CSDL định lượng động mà còn tạo nền tảng cho việc mở rộng sang dQDS.

CHƯƠNG 3: KHAI THÁC MẪU TRỌNG SỐ HỮU ÍCH PHỔ BIẾN TRÊN LUỒNG DỮ LIỆU ĐỊNH LƯỢNG ĐỘNG

3.1 Giới thiệu bài toán

Chương 3 mở rộng nghiên cứu từ CSDL định lượng động sang dQDS, nơi dữ liệu được sinh ra liên tục với tốc độ cao và khối lượng lớn. Trong bối cảnh này, việc khai thác FWUP trở nên quan trọng nhằm hỗ trợ hiệu quả cho việc ra các quyết định nhanh chóng. So với môi trường tĩnh, luồng dữ liệu đặt ra nhiều thách thức như chỉ cho phép quét dữ liệu một lần, yêu cầu cập nhật liên tục và giới hạn bộ

nhớ. Đồng thời, cả số lượng và trọng số của các mục đều biến động theo thời gian, khiến các phương pháp truyền thống không còn phù hợp. Mô hình cửa sổ trượt được lựa chọn để tập trung khai thác các dữ liệu mới nhất thông qua cơ chế cập nhật linh hoạt. Bên cạnh đó, việc xác định ngưỡng *minwus* cố định là khó khăn trong môi trường động, do đó bài toán khai thác top- k FWUP được đề xuất như một giải pháp thay thế, giúp người dùng dễ dàng kiểm soát số lượng mẫu quan trọng cần quan tâm. Chương này đề xuất các thuật toán FWUDS-CT và FWUDS-DWT nhằm khai thác FWUP hiệu quả trên dQDS sử dụng mô hình cửa sổ trượt, cùng với thuật toán TKFWUDS cho bài toán khai thác top- k FWUP. Các phương pháp này kế thừa và phát triển các cấu trúc dữ liệu từ Chương 2, đảm bảo khả năng cập nhật nhanh, xử lý hiệu quả và mở rộng tốt trong môi trường luồng dữ liệu động.

3.2 Một số định nghĩa

Định nghĩa 3.1 (Luồng dữ liệu định lượng động). Một dQDS là danh sách các giao dịch định lượng động được sắp xếp theo thứ tự thời gian. dQDS không giới hạn số lượng các giao dịch.

Định nghĩa 3.2 (Mô hình cửa sổ trượt). Một cửa sổ trượt trên luồng dữ liệu được xác định bởi hai giá trị là kích thước cửa sổ và kích thước pane. Trong đó, một pane chứa số lượng giao dịch cố định và một cửa sổ chứa nhiều pane hiện tại. Cập nhật cửa sổ là quá trình xoá pane cũ nhất và thêm dữ liệu từ pane mới vào cửa sổ.

Định nghĩa 3.3 (Bài toán khai thác FWUP trên luồng dữ liệu). Bài toán khai thác FWUP trên dQDS sử dụng mô hình cửa sổ trượt, là tìm tất cả các FWUP có *wus* lớn hơn hoặc bằng *minwus* do người dùng chỉ định trong mỗi cửa sổ dữ liệu.

Định nghĩa 3.4 (Bài toán khai thác top- k FWUP trên luồng dữ liệu). Bài toán khai thác top- k FWUP trên dQDS sử dụng mô hình cửa

sổ trượt là việc tìm ra chính xác k FWUP có wus lớn nhất trong mỗi cửa sổ dữ liệu.

3.3 Thuật toán khai thác mẫu trọng số hữu ích phổ biến trên luồng dữ liệu định lượng động

3.3.1 Thuật toán FWUDS-CT

Định nghĩa 3.5 (Định danh giao dịch cục bộ). Định danh giao dịch cục bộ, ký hiệu là $lcTid$, là giá trị định danh của giao dịch trong cửa sổ hiện tại. Ban đầu, $lcTid$ có giá trị bằng 0. Nếu có một giao dịch mới được thêm vào thì $lcTid$ của giao dịch đó được tính như sau:

$$lcTid = \begin{cases} lcTid + 1, & \text{if } lcTid < |W| \\ 1, & \text{if } lcTid = |W| \end{cases} \quad (3.1)$$

trong đó, $|W|$ là kích thước của cửa sổ tính theo số lượng giao dịch.

Định nghĩa 3.6 (Circular Tidset). Circular Tidset, ký hiệu là CTset, của một mẫu X trong cửa sổ W_i là danh sách lưu trữ các $lcTid$ của các giao dịch có chứa X trong cửa sổ W_i .

$$CTset(X, W_i) = \{lcTid(t) | t \in W_i, X \subseteq t\} \quad (3.2)$$

Giá trị wus của mẫu X trong cửa sổ W_i được tính dựa trên $CTset(X, W_i)$ như sau.

$$wus(X, W_i) = \frac{\sum_{j \in CTset(X, W_i)} twu(t^{(j)})}{sumtwu(W_i)}, \quad (3.3)$$

với $t^{(j)} \in W_i$ và $lcTid(t^{(j)}) = j$.

CTset của các mục được cập nhật khi trượt khung cửa sổ bằng cách thêm các $lcTid$ mới vào cuối danh sách và loại bỏ các $lcTid$ thuộc pane cũ nhất ở đầu danh sách. Tương tự Tidset, các mẫu mới có thể được tạo ra thông qua phép giao giữa các CTset. Do CTset không đảm bảo tính đơn điệu như Tidset, luận án đề xuất phương pháp giúp thực hiện phép giao hiệu quả với độ phức tạp tuyến tính.

Thuật toán FWUDS-CT khai thác FWUP trên dQDS sử dụng mô hình cửa sổ trượt được mô tả ở Hình 3.6.

Thuật toán 3.3. Thuật toán FWUDS-CT	
Input: Pane mới nhất P , giá trị $lcTid$ hiện tại $Curr_lcTid$, tập $CTset$ của các mục CTS , kích thước cửa sổ w , kích thước pane p và ngưỡng $minwus$;	
Output: Tập $FWUPs$ trong cửa sổ mới	
1. RemoveOldestPane($Curr_lcTid, CTS, w, p$) 2. InsertNewPane($P, Curr_lcTid, CTS, w$) 3. Lọc các mục có wus không nhỏ hơn $minwus$ và lưu vào C_1 4. $FWUPs \leftarrow C_1$ 5. MineFWUPs_CTset($C_1, Curr_lcTid, w, p$) 6. return $FWUPs$	
Procedure MineFWUPs_CTset ($C, Curr_lcTid, w, p$) 1. foreach $c_i \in C$ do 2. $C_{next} \leftarrow \emptyset$ 3. foreach $c_j \in C$ ($j > i$) do 4. $X \leftarrow c_i \cup c_j$ 5. $CTset(X) \leftarrow CTset_Intersection(CTset(c_i), CTset(c_j),$ $Curr_lcTid, w, p)$ 6. if $wus(X) \geq minwus$ then 7. $FWUPs \leftarrow X$ 8. $C_{next} \leftarrow \{X, CTset(X), wus(X)\}$ 9. if $C_{next} > 1$ then 10. MineFWUPs_CTset($C_{next}, Curr_lcTid, w, p$)	

Hình 3.1. Thuật toán FWUDS-CT.

3.3.2 Thuật toán FWUDS-DWT

Định nghĩa 3.7 (DSWUN-tree). DSWUN-tree là cấu trúc cây bao gồm một nút gốc và tập các nút con. Mỗi nút của DSWUN-tree bao gồm 6 trường (*name*, *weight*, *pre*, *post*, *children*, *parent*). Trong đó, *name* là tên của mục đại diện cho nút, *weight* là tổng *twu* của các giao dịch đi qua nút này, *pre* là thứ tự của nút khi duyệt tiền thứ tự, *post* là thứ tự của nút khi duyệt hậu thứ tự, *children* là danh sách các nút con, và *parent* là nút cha của nút hiện tại.

DSWUN-tree lưu trữ toàn bộ CSDL, không loại bỏ các mục không thoả mãn và chỉ mất một lần quét để cập nhật cây, sau đó có thể duy trì và khai thác nhiều lần với nhiều ngưỡng khác nhau. Ngoài ra, DSWUN-tree còn sử dụng thêm một danh sách tên là TAILLIST, để lưu trữ các nút cuối cùng của từng giao dịch khi được thêm vào cây.

Trong giai đoạn trượt khung cửa sổ, sử dụng TAILLIST và thông tin nút cha được lưu trữ tại mỗi nút, ta có thể dễ dàng trích xuất các giao dịch trong pane cũ nhất và xóa chúng khỏi DSWUN-tree.

Sau khi cập nhật DSWUN-tree cho cửa sổ dữ liệu mới, thuật toán FWUDS-DWT trích xuất WUNList và tiến hành khai thác dựa trên cấu trúc này. Thuật toán FWUDS-DWT được mô tả ở Hình 3.11.

Thuật toán 4.6. Thuật toán FWUDS-DWT	
Input:	Pane mới nhất P , DSWUN-tree của cửa sổ hiện tại R và ngưỡng $minwus$;
Output:	Tập $FWUPs$ của cửa sổ mới
1.	$R \leftarrow \text{Update_DSWUN-tree}(P, R)$
2.	Lưu các mục thỏa mãn $minwus$ theo thứ tự vào I_1
3.	Tạo WUNList cho các mục trong I_1
4.	$FWUPs \leftarrow I_1$
5.	MineFWUPs_WUNList (I_1)
6.	return $FWUPs$
Procedure MineFWUPs_WUNList (C)	
1.	for $i \leftarrow C - 1$ downto 1 do
2.	$C_{next} \leftarrow \emptyset$;
3.	for $j \leftarrow i - 1$ downto 0 do
4.	$X \leftarrow C[i] \cup C[j]$
5.	$wunl(X) \leftarrow \text{Construct_WUNList}(wunl(C[i]), wunl(C[j]))$
6.	if $wunl(X) \neq NULL$ then
7.	$FWUPs \leftarrow X$
8.	$C_{next} \leftarrow \{X, wunl(X), wus(X)\}$
9.	if $ C_{next} > 1$ then
10.	MineFWUPs_WUNList (C_{next})

Hình 3.2. Thuật toán FWUDS-DWT.

3.3.3 Kết quả thực nghiệm

Phần này trình bày kết quả thực nghiệm so sánh hai thuật toán đề xuất FWUDS-CT và FWUDS-DWT với phiên bản cải tiến dFWUNL-stream trên các bộ dữ liệu Retail, Accidents, BMS-POS, Kosarak và Susy với các tham số cửa sổ, kích thước pane và ngưỡng $minwus$ được thiết lập phù hợp cho từng trường hợp. Quy trình thực nghiệm bao gồm xây dựng cửa sổ ban đầu, khai thác FWUP, sau đó trượt cửa sổ bằng cách loại bỏ pane cũ và thêm pane mới, lặp lại trong 50 lần.

Kết quả thực nghiệm cho thấy FWUDS-CT và FWUDS-DWT vượt trội hơn dFWUNL-stream trong việc xây dựng và cập nhật dữ liệu nhờ cơ chế cập nhật cục bộ, giúp thời gian xử lý gần thời gian thực khi trượt cửa sổ. Về thời gian khai thác, FWUDS-DWT đạt hiệu năng tốt nhất, đặc biệt khi ngưỡng *minwus* nhỏ, trong khi FWUDS-CT kém hiệu quả hơn và có thể gặp hạn chế về bộ nhớ trên dữ liệu lớn. Về khả năng mở rộng, thuật toán FWUDS-DWT thể hiện tốt và ổn định nhất, trở thành giải pháp hiệu quả nhất tổng thể.

3.4 Khai thác top- k mẫu trọng số hữu ích phổ biến trên luồng dữ liệu định lượng động

Việc xác định ngưỡng *minwus* trong môi trường dữ liệu biến động liên tục là một thách thức lớn. Do đó, luận án đề xuất thuật toán TKFWUDS nhằm giải quyết bài toán khai thác top- k FWUP trên luồng dữ liệu, tận dụng cấu trúc DSWUN-tree kết hợp với WUNList và các chiến lược cắt tĩa hiệu quả để thu hẹp không gian tìm kiếm.

Định nghĩa 3.8 (Ngưỡng hỗ trợ trọng số hữu ích động). Ngưỡng hỗ trợ trọng số hữu ích động của khung cửa sổ W_i , ký hiệu là $dWus(W_i)$, là giá trị *wus* của mẫu lớn thứ k tính đến thời điểm khai thác hiện tại.

Chiến lược 1. Tăng ngưỡng hỗ trợ trọng số hữu ích động. $dWus$ được duy trì thông qua tập top- k hiện tại. Khi tìm thấy mẫu mới có giá trị lớn hơn $dWus$, mẫu này được đưa vào tập kết quả và ngưỡng $dWus$ được cập nhật theo giá trị nhỏ nhất trong top- k .

Chiến lược 2. Loại bỏ nhanh các mẫu không thỏa mãn. Dựa trên ngưỡng động $dWus$, các mẫu không thỏa mãn sẽ bị loại bỏ sớm trong quá trình sinh ứng viên. Nếu một mẫu có *wus* không vượt quá $dWus$ thì các mẫu mở rộng từ chúng cũng không thể thuộc top- k .

Chiến lược 3. Cắt tĩa tăng cường dựa trên cấp mục. Chiến lược

này dựa trên ma trận PTP để ước lượng nhanh giá trị của các 2-pattern. Nếu giá trị này không vượt ngưỡng $dWus$, các mẫu kết hợp tương ứng sẽ bị loại bỏ mà không cần thực hiện phép kết hợp tốn kém.

Chiến lược 4. Sử dụng hàng đợi ưu tiên lưu trữ top- k FWUP.

Hàng đợi ưu tiên, ký hiệu là PQ , có tối đa k phần tử để duy trì top- k FWUP của cửa sổ hiện tại.

3.4.1 Kết quả thực nghiệm

Thuật toán TKFWUDS được so sánh với dFWUNL-mod trên bảy bộ dữ liệu gồm Chess, Connect, Accidents, Retail, BMS-POS, Kosarak và Susy. Kết quả cho thấy khi giá trị k tăng, thời gian khai thác của các thuật toán đều tăng. Tuy nhiên, TKFWUDS luôn đạt hiệu năng tốt hơn dFWUNL-mod trên tất cả các bộ dữ liệu. Về khả năng mở rộng, khi kích thước cửa sổ và pane tăng, thời gian chạy của cả hai thuật toán đều tăng, nhưng TKFWUDS tăng chậm và ổn định hơn.

3.5 Kết luận chương

Chương 3 tập trung nghiên cứu bài toán khai thác FWUP trên dQDS sử dụng mô hình cửa sổ trượt, đề xuất hai thuật toán FWUDS-CT và FWUDS-DWT. Thuật toán FWUDS-CT sử dụng cấu trúc Circular Tidset để tối ưu việc quản lý định danh giao dịch, trong khi FWUDS-DWT khai thác ưu thế nén dữ liệu của DSWUN-tree và WUNList. Kết quả thực nghiệm cho thấy FWUDS-DWT hiệu quả hơn về thời gian xử lý và khả năng mở rộng. Bên cạnh đó, thuật toán TKFWUDS cũng được đề xuất để giải quyết bài toán khai thác top- k FWUP, sử dụng ngưỡng hỗ trợ trọng số hữu ích động kết hợp với các chiến lược cắt tia hiệu quả. Thực nghiệm chứng minh TKFWUDS đạt hiệu năng cao và khả năng mở rộng tốt, góp phần hoàn thiện giải pháp khai thác FWUP trên dQDS.

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

1. Kết luận

Luận án đã tập trung nghiên cứu, phát triển các cấu trúc dữ liệu và đề xuất các thuật toán để giải quyết hiệu quả một số bài toán khai thác FWUP trong môi trường dữ liệu định lượng động.

Đối với khai thác FWUP trên CSDL định lượng động, luận án đã xây dựng hệ thống định nghĩa và mô hình hóa bài toán này để giải quyết hạn chế của các mô hình truyền thống vốn mặc định trọng số các mục là cố định. Luận án đã đề xuất hai thuật toán nền tảng là dFWUT sử dụng cấu trúc Tidset và thuật toán dFWUNL sử dụng cấu trúc WUNList. Là mở rộng của cấu trúc N-list, WUNList thể hiện sự hiệu quả nhờ khả năng nén dữ liệu, cơ chế giao các WUNList và khả năng tự giảm kích thước sau mỗi lần giao mẫu. Hơn nữa, thuật toán dFWUNL còn kết hợp các phương pháp tối ưu hóa như tính toán trước mẫu có hai phần tử và cơ chế xác định nhanh giá trị *wus* của các mẫu. Kết quả thực nghiệm trên các bộ dữ liệu dày và thưa cho thấy thuật toán dFWUNL mang lại hiệu suất vượt trội so với dFWUT và hướng tiếp cận truyền thống về cả thời gian thực thi lẫn bộ nhớ sử dụng.

Bên cạnh việc khai thác tập FWUP đầy đủ, luận án cũng đã đề xuất thuật toán FWUCWP để giải quyết bài toán khai thác mẫu trọng số hữu ích phổ biến đóng từ CSDL định lượng động. Thuật toán FWUCWP được đề xuất không chỉ kế thừa sức mạnh của cấu trúc WUNList mà còn tích hợp một hệ thống các chiến lược cắt tĩa chuyên sâu bao gồm cắt tĩa dựa trên quan hệ cấu trúc tổ tiên WUNList, cắt tĩa kết hợp *twu* tối đa và chiến lược cắt tĩa dựa trên khái niệm Enclosure. Những cải tiến này giúp thu hẹp nhanh chóng không gian tìm kiếm và cải thiện hiệu năng rõ rệt, đặc biệt là trên các tập dữ liệu thưa thớt với tốc độ xử lý tốt hơn so với thuật toán tốt nhất hiện nay, đồng thời vẫn

bảo toàn trọn vẹn tri thức cần thiết cho việc ra quyết định.

Đối với bài toán khai thác trên luồng dữ liệu định lượng động, luận án đã mở rộng thành công mô hình khai thác sang bối cảnh dữ liệu đến liên tục, đòi hỏi xử lý một lần với thời gian và tài nguyên bộ nhớ hạn chế. Thông qua việc sử dụng mô hình cửa sổ trượt, luận án đã đề xuất thuật toán FWUDS-CT sử dụng cấu trúc Circular Tidset, giải quyết hiệu quả vấn đề tràn số định danh giao dịch khi luồng dữ liệu kéo dài. Song song với đó, luận án cũng phát triển thuật toán FWUDS-DWT với cấu trúc DSWUN-tree cho phép cập nhật các dữ liệu mới và loại bỏ dữ liệu lỗi thời trong cửa sổ trượt chỉ với một lần quét dữ liệu. Với sự kết hợp giữa cấu trúc DSWUN-tree và WUNList đã giúp thuật toán FWUDS-DWT đạt được tốc độ khai thác FWUP và khả năng cập nhật dữ liệu hiệu quả, đáp ứng yêu cầu khắt khe của luồng dữ liệu.

Đóng góp cuối cùng của luận án là phát triển thuật toán TKFWUDS để khai thác top- k FWUP trên luồng dữ liệu định lượng động sử dụng mô hình cửa sổ trượt. Luận án đã đề xuất khái niệm ngưỡng hỗ trợ trọng số hữu ích động trên mỗi cửa sổ và kỹ thuật nâng ngưỡng tự động. Các chiến lược cắt tía dựa trên ngưỡng động này cũng được áp dụng giúp trích xuất nhanh chóng và chính xác k FWUP giá trị nhất mà không cần người dùng thiết lập ngưỡng hỗ trợ trọng số hữu ích tối thiểu, là một rào cản lớn trong khai thác luồng dữ liệu thực tế.

Các kết quả nghiên cứu nêu trên đã được công bố trên các tạp chí và hội thảo trong nước và quốc tế uy tín. Những kết quả này đóng góp về mặt học thuật trong lĩnh vực khai thác dữ liệu và cung cấp bộ giải pháp thực tiễn mạnh mẽ cho các hệ thống hỗ trợ ra quyết định thông minh dựa trên dữ liệu định lượng và luồng dữ liệu lớn.

2. Hướng phát triển

Mặc dù đã đạt được những kết quả khả quan, luận án vẫn mở ra

nhiều hướng nghiên cứu tiềm năng nhằm đáp ứng sự phức tạp ngày càng tăng của dữ liệu lớn. Trong tương lai, một trong những ưu tiên hàng đầu là việc tối ưu hóa hiệu năng tính toán thông qua các kiến trúc lập trình song song như bộ xử lý đa lõi, CUDA hoặc các hệ thống tính toán phân tán như Spark, Hadoop. Việc áp dụng các kỹ thuật này không chỉ giúp xử lý các CSDL ở quy mô lớn hơn mà còn cho phép thuật toán thực thi trong các môi trường đòi hỏi độ trễ thấp. Bên cạnh việc tiếp tục cải thiện hiệu năng, một hướng nghiên cứu trong tương lai là mở rộng phạm vi đánh giá các phương pháp đề xuất trên những luồng dữ liệu thực tế với các đặc trưng và quy mô khác nhau. Việc thực nghiệm trong các điều kiện tốc độ dữ liệu đến không đồng đều, khối lượng dữ liệu lớn và yêu cầu khác nhau về độ trễ xử lý sẽ giúp đánh giá toàn diện hơn khả năng đáp ứng và tính khả thi của các thuật toán trong các hệ thống xử lý luồng dữ liệu thực tế. Cùng với đó, nghiên cứu sẽ tập trung vào việc phát triển các cơ chế thích ứng cho luồng dữ liệu, cho phép hệ thống tự điều chỉnh kích thước pane dữ liệu hoặc tích hợp các chiến lược giảm tải để duy trì sự ổn định khi luồng dữ liệu xuất hiện các biến động bất thường hoặc bùng nổ về lưu lượng.

Một hướng đi quan trọng khác là mở rộng mô hình khai thác lên các loại hình dữ liệu không chắc chắn, nơi sự tồn tại của các mục hoặc trọng số của chúng mang tính xác suất. Điều này đòi hỏi việc tái cấu trúc các định nghĩa và chiến lược cắt tía để đảm bảo tính chính xác của tri thức trích xuất được. Cuối cùng, việc đa dạng hóa các mô hình cửa sổ khai thác như cửa sổ dựa trên thời gian thực tế, đồng thời tích hợp các cơ chế khai thác tăng trưởng để tối ưu hóa việc tái sử dụng tri thức cũ cũng là một hướng đi tiềm năng. Những nghiên cứu này không chỉ hoàn thiện bức tranh về khai thác FWUP mà còn góp phần xây dựng các hệ thống thông minh, linh hoạt và mạnh mẽ hơn trong tương lai.

DANH MỤC CÔNG TRÌNH KHOA HỌC CỦA TÁC GIẢ LIÊN QUAN ĐẾN LUẬN ÁN

- [I] **Le N.**, Bui H., Nguyen M., Nguyen H., and Le T. (2022), “Mining frequent weighted utility patterns in dynamic quantitative databases”, *2022 RIVF International Conference on Computing and Communication Technologies (RIVF)*, pp. 536–541.
- [II] Nguyen H., **Le N.**, Bui H., and Le T. (2023), “Mining frequent weighted utility patterns with dynamic weighted items from quantitative databases”, *Applied Intelligence* 53(16), pp. 19629–19646.
- [III] **Lê Hoàng Bình Nguyên**, Nguyễn Duy Hàm, Nguyễn Thị Hồng Minh (2023), “Thuật toán song song khai thác nhanh các mẫu trọng số hữu ích phổ biến từ cơ sở dữ liệu định lượng động”, *Chuyên san Các công trình nghiên cứu, phát triển và ứng dụng Công nghệ Thông tin và Truyền thông* 2023(2).
- [IV] **Le N.**, Nguyen H., Bui H., and Nguyen M. (2025), “The Problem Of Mining Top-K Frequent Pattern Over Dynamic Quantitative Data Stream”, *The 18th National Conference on Fundamental and Applied Information Technology Research (FAIR'2025)*, pp. 615-621.
- [V] **Le N.**, Nguyen H., Nguyen M., Bui H., Yun U., and Vo B. (2026), “Efficiently mining frequent weighted utility closed patterns with pruning strategies from dynamic quantitative databases”, *The Journal of Supercomputing* 82(3), pp. 129.
- [VI] **Le N.**, Nguyen H., Nguyen M., Bui H., Yun U., and Vo B. (2026), “Fast and scalable sliding-window-based algorithms for mining frequent weighted utility patterns over dynamic quantitative data streams”, *Information Sciences* 748, pp. 123488.