TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA CÔNG NGHỆ THÔNG TIN

HUỲNH PHÚC NGUYÊN – 52000091
MA NHẬT BIỂN – 52000015

NGHIÊN CỨU KHAI PHÁ K TẬP PHỔ BIẾN TIỆN ÍCH CAO ĐẦU TIÊN TỪ CƠ SỞ DỮ LIỆU KHÔNG CHẮC CHẮN

CHUYÊN ĐỀ NGHIÊN CỨU 1

CHUYÊN NGÀNH KHOA HỌC MÁY TÍNH

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2024

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA CÔNG NGHỆ THÔNG TIN

HUỲNH PHÚC NGUYÊN – 52000091
MA NHẬT BIỂN – 52000015

NGHIÊN CỨU KHAI PHÁ K TẬP PHỔ BIẾN TIỆN ÍCH CAO ĐẦU TIÊN TỪ CƠ SỞ DỮ LIỆU KHÔNG CHẮC CHẮN

CHUYÊN ĐỀ NGHIÊN CỨU 1

CHUYÊN NGÀNH KHOA HỌC MÁY TÍNH

Người hướng dẫn
Nguyễn Chí Thiện

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2024

LỜI CẢM ƠN

Chúng em xin chân thành cảm ơn Khoa Công nghệ thông tin đã tạo điều kiện cho chúng em được tiếp cận và hoàn thành bài báo cáo. Chúng em xin chân thành cảm ơn thầy Nguyễn Chí Thiện đã hướng dẫn hoàn thành bài báo cáo. Những sự hướng dẫn, quan tâm tận tình trong quá trình thực hiện báo cáo của thầy chính là động lực và là tiền đề để chúng em có được thành quả ngày hôm nay.

Trong quá trình làm bài báo cáo, do kiến thức cũng như kinh nghiệm còn nhiều hạn chế nên bài báo cáo không thể tránh khỏi những thiếu sót, chúng em rất mong nhận được ý kiến đóng góp của thầy để chúng em có thể học hỏi được nhiều kĩ năng và kinh nghiệm, từ đó có được nền tảng để phát triển hơn cho các dự án sau.

Chúng em xin chân thành cảm ơn!

TP. Hồ Chí Minh, ngày 20 tháng 3 năm 2024

Tác giả

Huỳnh Phúc Nguyên

Ma Nhật Biển

CÔNG TRÌNH ĐƯỢC HOÀN THÀNH
TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

Tôi xin cam đoan đây là công trình nghiên cứu của riêng tôi và được sự hướng dẫn khoa học của thầy Nguyễn Chí Thiện. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong Dự án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung Dự án của mình. Trường Đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

TP. Hồ Chí Minh, ngày 20 tháng 3 năm 2024

Tác giả

(Ký tên và ghi rõ họ tên)

Huỳnh Phúc Nguyên

Ma Nhật Biển

NGHIÊN CỨU KHAI PHÁ K TẬP PHỔ BIẾN TIỆN ÍCH CAO ĐẦU TIÊN TỪ CƠ SỞ DỮ LIỆU KHÔNG CHẮC CHẮN

TÓM TẮT

Khai thác top-k tập mẫu tiện ích cao từ bộ dữ liệu không chắc chắn là đề tài được đưa ra thảo luận trong thời gian gần đây và đã có nhiều cách tiếp cận được đưa ra để giải quyết bài toán. Tuy nhiên, những cách giải quyết thông thường đều gặp các vấn đề khi tìm kiếm quá nhiều mẫu tiện ích cao không chắc chắn do áp dụng các cách tiếp cận truyền thống.

Bài báo cáo này sẽ nghiên cứu về khai thác top-k tập mẫu phổ biến tiện ích cao từ bộ dữ liệu không chắc chắn. Ngoài việc khai thác các mẫu có tiện ích cao thì sẽ thêm tính phổ biến của mẫu và ngưỡng thêm vào top-k sẽ là ngưỡng phổ biến. Trong nghiên cứu này, chúng tôi sẽ áp dụng một giải pháp khai thác hiệu quả có tên TUHUFP đi cùng với các chiến lược nhằm nâng cao hiệu suất thời gian và giảm thiểu không gian lưu trữ. Các thực nghiệm sẽ được tiến hành nhằm kiểm tra mức độ hiệu quả của giải pháp, gồm số lượng mẫu tham gia, thời gian khai thác và bộ nhớ được sử dụng.

MỤC LỤC

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG BIỂU

DANH MỤC CÁC CHỮ VIẾT TẮT

expSup	Expected support
HUP	High utility pattern
TUHUFP	Top-k uncertain high utility frequent pattern
TID	Transaction identity
CUP	Candidate uncertain pattern
Prob	Probability
Uti	Utility
TWU	Transaction Weight Utilization

CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI

Trong thời đại số hóa ngày nay, dữ liệu đã trở thành một nguồn tài nguyên vô cùng quý báu, đóng vai trò quan trọng trong mọi lĩnh vực của xã hội và kinh tế. Sự tích tụ lớn mạnh của dữ liệu từ các nguồn đa dạng như thiết bị di động, cảm biến, mạng xã hội và nhiều nguồn khác, tạo ra một biển số dữ liệu ngày càng phong phú và phức tạp. Khai thác dữ liệu (data mining) đã trở thành một công cụ quan trọng, giúp chúng ta phân tích, tìm kiếm mẫu, và trích xuất thông tin quan trọng từ lượng lớn dữ liệu này, từ đó doanh nghiệp và tổ chức có thể đưa ra tổng kết và dự đoán xu hướng, tìm ra các mối liên kết ẩn sau dữ liệu, và tạo ra giá trị thực sự từ thông tin

Một khía cạnh hấp dẫn của thách thức trong khai thác dữ liệu là việc trích xuất các mẫu thường xuyên hàng đầu (top-K frequent patterns) từ các cơ sở dữ liệu không chắc chắn [10], nơi tính không chắc chắn tự nhiên của dữ liệu tạo ra những phức tạp trong quá trình khai thác mẫu. Đồng thời, việc tìm kiếm thông tin giá trị mở rộng đến lĩnh vực các mẫu có tiện ích cao nhất hàng đầu (top-K high utility patterns) [22], nhấn mạnh sự quan trọng của việc khám phá các mẫu đóng góp đáng kể vào tổng giá trị. Khác với các mẫu thường xuyên truyền thống, mẫu có tiện ích cao xem xét không chỉ tần suất xuất hiện mà còn giá trị liên quan đến mỗi mẫu. Giá trị này có thể bao gồm nhiều yếu tố, chẳng hạn như giá trị tiền tệ, điểm quan trọng hoặc các chỉ số quan trọng khác, tùy thuộc vào lĩnh vực ứng dụng.

Nghiên cứu này sẽ khám phá sâu sắc về những phức tạp khi khai thác các mẫu thường xuyên hàng đầu từ các cơ sở dữ liệu không chắc chắn và khám phá các mẫu có ích cao hàng đầu. Bằng cách phát triển các phương pháp tiên tiến được điều chỉnh để xử lý sự không chắc chắn và xem xét các yếu tố tiện ích, chúng tôi hướng đến việc khám phá các mẫu không chỉ thường xuyên mà còn có giá trị cao trong nhiều tình huống thực tế.

CHƯƠNG 2. CÔNG TRÌNH LIÊN QUAN

2.1 Khai thác tập mẫu phổ biến

Khai thác mẫu phổ biến là quá trình tìm kiếm và xác định các phần dữ liệu xuất hiện thường xuyên và có mối quan hệ chặt chẽ với nhau. Các mẫu phổ biến này có thể xuất hiện trong nhiều lĩnh vực khác nhau, từ thị trường mua sắm đến nghiên cứu sinh học và phân tích dữ liệu mạng.

Có nhiều thuật toán nổi tiếng được dùng để khai thác mẫu phổ biến, bao gồm Apriori [18], FP-growth, và Eclat. Kết quả của khai thác mẫu phổ biến có thể được áp dụng rộng rãi, từ việc đề xuất sản phẩm cho đến dự đoán xu hướng thị trường, từ phân tích dữ liệu y tế đến phát hiện gian lận mạng.

2.2 Khai thác tập mẫu top-k và top-rank-k

Khai thác mẫu top-k và top-rank-k là quá trình tìm kiếm và xác định k mẫu quan trọng nhất từ một tập dữ liệu lớn. Trong trường hợp top-rank-k, mẫu được xác định dựa trên mức độ quan trọng của chúng như tần suất xuất hiện, giá trị thông tin, hoặc các yếu tố khác.

Việc khai thác mẫu top-k và top-rank-k mang lại nhiều tiện ích trong các ứng dụng thực tế như tìm kiếm sản phẩm phổ biến nhất trong thị trường hay phân loại tin tức theo yêu cầu đề ra.

2.3 Khai thác tập mẫu tiện ích cao

Khai thác tập mẫu có tiện ích cao là quá trình xác định và trích ra những mẫu dữ liệu đặc biệt có giá trị lớn hoặc quan trọng. Trong môi trường dữ liệu không chắc chắn, khi mỗi mẫu dữ liệu đi kèm với một độ không chắc chắn về giá trị, khai thác mẫu có giá trị cao trở nên cực kỳ hữu ích. Điều này giúp chúng ta hiểu rõ hơn về sự quan trọng của mỗi mẫu và cách chúng đóng góp vào giá trị tổng của dữ liệu.

Việc xác định tổ hợp sản phẩm với số lượng bán ra có lợi nhuận cao hơn một ngưỡng cho trước có ý nghĩa lớn trong tính toán doanh thu và tìm ra chiến lược kinh doanh.

2.4 Khai thác tập mẫu từ bộ dữ liệu không chắc chắn

Cơ sở dữ liệu không chắc chắn xuất hiện khi chúng ta không chắc chắn về giá trị thực sự của một số liệu trong cơ sở dữ liệu. Điều này có thể phát sinh từ nhiều lý do: nhiễu, đo lường không chắc chắn hoặc không chắc chắn tự nhiên trong quá trình thu thập dữ liệu. Phương pháp khai thác tập mẫu từ bộ dữ liệu không chắc chắn thường phụ thuộc vào các kỹ thuật như lấy mẫu ngẫu nhiên hay khai thác tập mẫu không giám sát và bán giám sát. Khai thác mẫu từ cơ sở dữ liệu không chắc chắn có ứng dụng rộng rãi trong nhiều lĩnh vực. Nghiên cứu về khai thác tập mẫu từ bộ dữ liệu không chắc chắn thường liên quan đến việc phát triển các mô hình xác suất, thuật toán xử lý không chắc chắn và phương pháp thống kê.

CHƯƠNG 3. MÔ HÌNH TOÁN HỌC CỦA ĐỀ TÀI

Một siêu thị bán lẻ kinh doanh hàng nghìn sản phẩm khác nhau và mỗi sản phẩm có mức lợi nhuận cùng độ phổ biến không chắc chắn. Mỗi khách hàng khi tham gia mua sản phẩm tạo thành một giao dịch. Tất cả giao dịch được siêu thị lưu trữ, tạo ra một cơ sở dữ liệu. Từ cơ sở dữ liệu đó, những người lãnh đạo cần tìm ra tập hợp các sản phẩm mang lại lợi nhuận cao nhất, từ đó đưa ra chiến lược kinh doanh phù hợp.

Bảng 3.1 Danh sách giao dịch và bảng lợi nhuận của từng sản phẩm.

Giao dịch	Danh sách sản phẩm trong giao dịch	Giá trị lợi nhuận trong giao dịch
T_1	a: 0.9, c: 0.9, d: 0.6	a, c, d : 9 : 3, 1, 5
T_2	a: 0.9, b: 0.9, c: 0.7, d: 0.6, e: 0.4	a, b, c, d, e : 20 : 6, 2, 3, 5, 4
T_3	b: 0.5, c: 0.8, d: 0.9, f: 0.2	b, c, d, f : 17 : 2, 6, 5, 4
T_4	c: 0.9, e: 0.1, f: 0.5	c, e, f : 15 : 5, 4, 6
T_5	a: 0.4, b: 0.5, c: 0.9, d: 0.3	a, b, c, d : 23 : 6, 4, 3, 10
T_6	d: 0.9, e: 0.1, f: 0.6	d, e, f : 13 : 5, 4, 4
T_7	a: 0.9, b: 0.7	a, b: 5 : 3, 2

Mô hình toán học của bài toán được phát biểu như sau:

Cho:

- **I** = $\{i_1, i_2, ..., i_m\}$ là một tập hợp hữu hạn của m sản phẩm khác nhau.
- **D** = $\{T_1, T_2, ..., T_n\}$ là một tập các giao dịch, mỗi giao dịch gồm danh sách sản phẩm và tiện ích từ giao dịch, $pr(i, T_q) \in$ **D** là xác suất của item i trong giao dịch T_q $(1 \leq q \leq n)$ thuộc cơ sở dữ liệu **D** và $u(i, T_q) \in$ **D** là tiện ích của item i trong giao dịch T_q $(1 \leq q \leq n)$ thuộc cơ sở dữ liệu **D**.
- δ là ngưỡng tiện ích tối thiểu theo tùy chọn của người dùng.

Định nghĩa 1: Xác suất mong đợi (expected probabitily) của mẫu $\boldsymbol{x}$ trong một giao dịch T_q, ký hiệu là pr($\boldsymbol{x}$, T_q), được định nghĩa là:

$$pr(\boldsymbol{x}, T_q) = \prod_{i \in \boldsymbol{x}} pr(i, T_q)$$

Ví dụ: dựa theo Bảng 3.1, xác suất mong đợi của mẫu $\boldsymbol{x}$ = {a, b} trong giao dịch T_2 được tính tương ứng:

$$pr(\boldsymbol{x}, T_2) = \prod_{i \in \boldsymbol{x}} pr(i, T_2)$$

$$= pr(a, T_2) * pr(b, T_2)$$

$$= 0.9 * 0.9 = 0.81$$

Định nghĩa 2: Độ tin cậy mong đợi (expected support) của một mẫu $\boldsymbol{x}$ trong bộ dữ liệu **D**, ký hiệu là expSup($\boldsymbol{x}$), được định nghĩa là:

$$expSup(\boldsymbol{x}) = \sum_{T_q \in \boldsymbol{D_x}} pr(\boldsymbol{x}, T_q)$$

Trong đó: D_x là tập các giao dịch có chứa mẫu $\boldsymbol{x}$ trong cơ sở dữ liệu D.

Ví dụ: dựa theo Bảng 3.1, ta có:

expSup(a) = 0.9 + 0.9 + 0.4 + 0.9 = 3.1

expSup(ab) = (0.9*0.9) + (0.4*0.5) + (0.9*0.7) = 1.64

Định nghĩa 3: Tiện ích của một mẫu $\boldsymbol{x}$ trong giao dịch T_q, ký hiệu là u($\boldsymbol{x}$, T_q) và được định nghĩa là:

$$u(\boldsymbol{x}, T_q) = \sum_{i \in \boldsymbol{x}} u(i, T_q)$$

Ví dụ: dựa theo Bảng 3.1, tiện ích của mẫu $\boldsymbol{x}$ = {a, c} trong giao dịch T_1 được tính là:

$$ul(\boldsymbol{x}, T_1) = ul(a, T_1) + ul(c, T_1) = 3 + 1 = 4$$

Định nghĩa 4: Tiện ích của một mẫu $\boldsymbol{x}$ trong cơ sở dữ liệu D, ký hiệu là u($\boldsymbol{x}$) và được định nghĩa là:

$$u(\boldsymbol{x}) = \sum_{T_q \in \boldsymbol{D_x}} u(\boldsymbol{x}, T_q)$$

Trong đó: D_x là tập các giao dịch có chứa mẫu $\boldsymbol{x}$ trong cơ sở dữ liệu D.

Ví dụ: Dựa theo Bảng 3.1, tiện ích của mẫu $\boldsymbol{x}$ = {a, c} được tính bằng:

$ul(\boldsymbol{x}) = ul(\boldsymbol{x}, T_1) + ul(\boldsymbol{x}, T_2) + ul(\boldsymbol{x}, T_5)$

$= (3 + 1) + (6 + 3) + (6 + 3) = 22$

Định nghĩa 5: Tiện ích của một giao dịch T_q được ký hiệu là tu(T_q), và được định nghĩa là:

$$tu(T_q) = \sum_{i \in Tq} u(i, T_q)$$

Ví dụ: dựa theo Bảng 3.1, ta có:

tu(T_1) = u(a, T_1) + u(c, T_1) + u(d, T_1)

= 3 + 1 + 5 = 9

Định nghĩa 6: Tổng tiện ích của cơ sở dữ liệu **D**, ký hiệu là TU, được định nghĩa như sau:

$$TU = \sum_{T_q \in \boldsymbol{D}} tu(T_q)$$

Ví dụ: Tổng tiện ích của cơ sở dữ liệu D theo Bảng 3.1 được tính bằng:

TU = 9 + 20 + 17 + 15 + 23 + 13 + 5 = 102

Định nghĩa 7: Mức tiện ích trọng số của một mẫu $\boldsymbol{x}$ là tổng các tiện ích của giao dịch mà chứa $\boldsymbol{x}$, được kí hiệu là twu($\boldsymbol{x}$) và có định nghĩa như sau:

$$twu(\boldsymbol{x}) = \sum_{\boldsymbol{x} \subseteq Tq \wedge T_q \in \boldsymbol{D}} tu(T_q)$$

Định nghĩa 8: Một mẫu $\boldsymbol{x}$ trong cơ sở dữ liệu D là một mẫu tiện ích cao, ký hiệu là HUP nếu và chỉ nếu tiện ích của nó cao hơn ngưỡng tiện ích tối thiểu và được tính là:

$$HUP \leftarrow \{\boldsymbol{x} | u(\boldsymbol{x}) \geq TU \times \delta\}$$

Ví dụ: dựa theo Bảng 3.1, với ngưỡng tối thiểu δ bằng 20%, TU = 102, xét tiện ích của tập mẫu {a} và tập mẫu {a, c}, ta thấy:

Tập mẫu {a} không phải là một HUP vì u(a) = 18 < 102 x 20% = 20,4

Tập mẫu {a, c} là một HUP vì u(a) = 22 > 102 x 20% = 20,4

Định nghĩa 9: Top-k tập mẫu phổ biến không chắc chắn có tiện ích cao trong cơ sở dữ liệu D, ký hiệu là TUHUFP, gồm các tập mẫu $\boldsymbol{x}$ thuộc cơ sở dữ liệu D và được định nghĩa là:

$$TUHUFP \leftarrow \{\boldsymbol{x} | \boldsymbol{x} \in HUP \cap expSup(\boldsymbol{x}) \geq expSup_{min}(TUHUFP)\}$$

Trong đó: $expSup_{min}(TUHUFP)$ là độ tin cậy mong đợi nhỏ nhất trong tập TUHUFP và được tính bằng:

$$expSup_{min}(TUHUFP) = \min\ (expSup(\boldsymbol{x}) |\ \boldsymbol{x} \in\ TUHUFP)$$

CHƯƠNG 4. PHƯƠNG PHÁP THỰC HIỆN

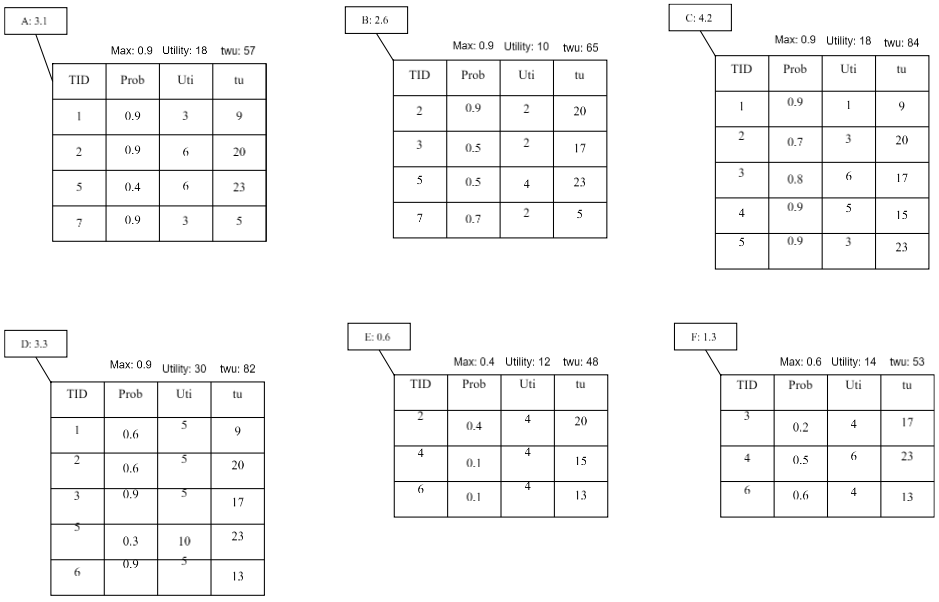
Có rất nhiều phương pháp đã được tìm ra nhằm khai thác các tập mẫu không chắc chắn cũng như khai thác các tập mẫu có tiện ích cao. Trong báo cáo này, chúng tôi sẽ đưa ra các phương pháp đã áp dụng để giải quyết bài toán khai thác top-k tập mẫu phổ biến tiện ích cao trong bộ dữ liệu không chắc chắn.

4.1 CUP-lists

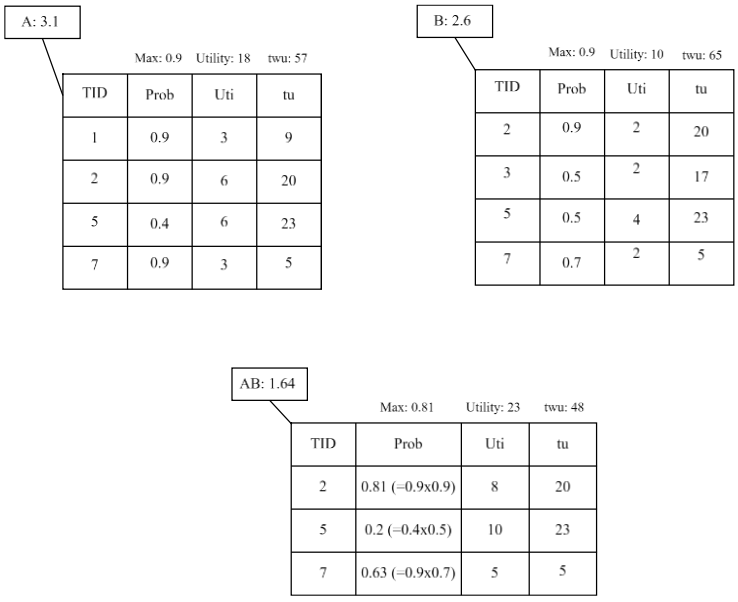
CUP-list là cấu trúc dữ liệu được sử dụng để phục vụ khai thác top-k tập mẫu không chắc chắn có tiện ích cao (TUHUFP). Cấu trúc CUP-list gồm có:

- Tên tập mẫu (pattern name)
- Độ tin cậy mong đợi (expected support) của mẫu
- Một tập hợp chứa dữ liệu của mẫu gọi là TEP-list
- Thuộc tính max để lưu giá trị xác suất mong đợi lớn nhất của mẫu trong TEP-List. Ngưỡng tối đa (overestimate) của một tập XY được xác định bằng expSup(X) x max(Y). Bằng việc tính toán trước ngưỡng tối đa của một tập mẫu và so sánh với ngưỡng đặt ra trong top-k, chúng ta có thể tính toán các tập kết hợp có tồn tại top-k hay không, từ đó cắt giảm không gian lưu trữ và nâng cao hiệu suất khai thác.
- Thuộc tính utility để lưu giá trị tiện ích của mẫu trong cơ sở dữ liệu.
- Thuộc tính TWU (transaction-weight-utility) là tổng các tiện ích của giao dịch trong TEP-List.
- Dữ liệu của TEP-list gồm:
    - Địa chỉ giao dịch (TID)
    - Xác suất mong đợi (expected probability) của mẫu trong mỗi giao dịch
    - Tiện ích của mẫu trong mỗi giao dịch
    - Tiện ích của mỗi giao dịch

Dựa theo dữ liệu ví dụ ở Bảng 3.1, Hình 4.1 mô tả cấu trúc CUP-list của các sản phẩm có trong cơ sở dữ liệu. Bằng việc lưu trữ dữ liệu các mẫu theo cấu trúc này, chúng ta có thể tính toán độ tin cậy mong đợi và tiện ích của mẫu một cách hiệu quả mà không phải duyệt lại cơ sở dữ liệu.



Hình 4.1 Mô hình CUP-list



Hình 4.2 Mô hình liên kết CUP-list cho mẫu AB

Hình 4.2 thể hiện quá trình kết hợp mẫu {A, B} từ CUP-list item A và CUP-list item B. Đầu tiên, ta xác định TEP-list của mẫu từ các TID trùng lặp (2, 5, 7). Dựa

vào các định nghĩa 1 và 3 để xác định các giá trị xác suất mong đợi và tiện ích của mẫu trong TEP-list.

4.2 Các chiến lược nâng cao hiệu quả khai thác

Chiến lược 1 – nâng ngưỡng: Top-k tập mẫu không chắc chắn có tiện ích cao trong cơ sở dữ liệu được lưu trong một mảng sắp xếp giảm dần độ tin cậy mong đợi (expSup). Từ đó độ tin cậy mong đợi của phần tử cuối cùng trong mảng chính là ngưỡng của top-k. Khi có một tập mẫu có độ tin cậy cao hơn độ tin cậy của phần tử cuối mảng, tập mẫu mới sẽ được thêm vào top-k và xóa đi phần tử cuối, đồng thời cập nhật ngưỡng tin cậy mong đợi theo phần tử cuối hiện tại. Chiến lược này được áp dụng để giảm thiểu thời gian tìm kiếm và thay thế tập mẫu trong danh sách top-k.

Chiến lược 2 – cắt tỉa theo ngưỡng được nâng: trong quá trình tạo các tập mẫu mới trong phương thức **TUHUFP_Search**, thuật toán sẽ không tạo ra tập mẫu mới nếu độ tin cậy mong đợi của mẫu tham gia thấp hơn ngưỡng tin cậy mong đợi. Chiến lược này sẽ giúp giảm thiểu đáng kể thời gian xử lý và không gian lưu trữ của thuật toán.

Chiến lược 3 – cắt tỉa theo TWU: trong quá trình khai thác các tập mẫu, thuật toán sẽ có một danh sách chứa các tập mẫu gọi là danh sách ứng viên sẽ tham gia vào thuật toán. Thuật toán sẽ không thêm tập mẫu mới vừa được tạo vào danh sách ứng viên nếu TWU của tập mẫu đó nhỏ hơn ngưỡng tiện ích. Điều này giúp giảm số lượng ứng viên tham gia một cách đáng kể.

4.3 Thuật toán TUHUFP

Bước quan trọng trong thuật toán đề xuất được gọi là TUHUFP và được mô tả như sau:

1. Tìm kiếm tất cả CUP-list của tập mẫu đơn (1 sản phẩm) trong cơ sở dữ liệu D và sắp xếp theo thứ tự giảm dần độ tin cậy mong đợi.

2. Đưa danh sách trên vào danh sách top-k.

3. Gọi phương thức TUHUFP_Search. TUHUFP_Search sử dụng chiến lược chia để trị để tạo ra các CUP-list của tập mẫu lớn hơn từ các CUP-list của tập mẫu

tham gia. Với mỗi tập mẫu, nếu độ tin cậy mong đợi thấp hơn ngưỡng tin cậy mong đợi hoặc tiện ích của tập mẫu thấp hơn ngưỡng tiện ích thì sẽ bỏ qua. Nếu không, thuật toán sẽ thêm tập mẫu vào kết quả và kết hợp chúng để tìm kiếm các tập mẫu lớn hơn. Chiến lược này sẽ tìm kiếm đến khi tất cả mẫu được xem xét.

CHƯƠNG 5. TRIỂN KHAI GIẢI PHÁP

5.1 Class Diagram

Giải pháp được triển khai với các lớp sau:

- Tep<T1, T2>: lớp này sử dụng để tạo đối tượng Tep bên trong TEP-List của CUP; bao gồm các thuộc tính như TID, xác suất tồn tại, giá trị tiện ích, và giá trị tiện ích của giao dịch tại TID đó, các phương thức cơ bản như khởi tạo, các getters và setters.
- Cup<T1, T2, T3>: lớp được sử dụng để tạo nên CUP tham gia vào quá trình khi thác của thuật toán, các thuộc tính bao gồm: tên, độ mong đợi tin cậy, giá trị tiện ích, TEP-List, giá trị max trong TEP-List, tổng các tiện ích giao dịch trong TEP-List; cùng với các phương thức khởi tạo, setters, getters.
- AlgorithmTUHUFP<T1, T2, T3>: lớp này chứa thuật toán TUHUFP để khai thác k tập mẫu tiện ích cao phổ biến đầu tiên.

  Các thuộc tính trong lớp này bao gồm: danh sách kết quả, một Map chứa các 1-mẫu tham gia, ngưỡng tiện ích và ngưỡng phổ biến.

  Các phương thức:

  + readData(filePath: String, percentage: double, k: int): đọc file chứa các cơ sở dữ liệu để chuẩn bị cho quá trình xử lý.

  + processData(itemName: String, probLine: String, utilLine: String, TID: Integer): xử lý dữ liệu được truyền từ readData() và tạo ra CUP-list 1-mẫu.

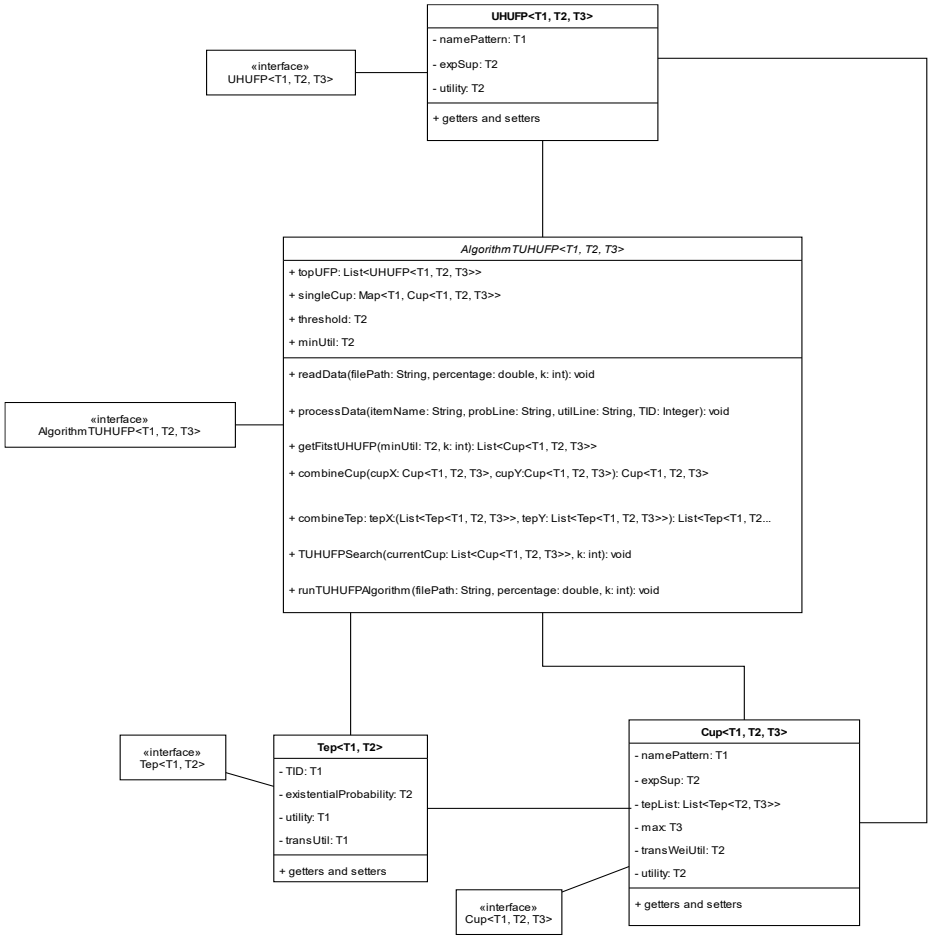
  + getFitstUHUFP(minUtil: T2, k: int): phương thức này trả về một danh sách các mẫu tham gia vào thuật toán, đồng thời sẽ thêm các mẫu thỏa điều kiện về ngưỡng lợi ích vào kết quả với độ dài k

+ combineCup(cupX: Cup<T1, T2, T3>, cupY:Cup<T1, T2, T3>): phương thức sử dụng để ghép 2 CUP lại với nhau và trả về một CUP mới.

+ combineTep: tepX:(List<Tep<T1, T2, T3>>, tepY: List<Tep<T1, T2, T3>>): phương thức sử dụng để ghép 2 TEP-List của 2 CUP lại với nhau và trả về một TEP-List mới.

+ TUHUFPSearch(currentCup: List<Cup<T1, T2, T3>>, k: int): phương thức chính của thuật toán, thực hiện khai thác các tập mẫu và thêm vào kết quả cuối cùng khi thỏa các ngưỡng và trả về k tập mẫu không chắc chắn tiện ích cao phổ biến đầu tiên.

+ runTUHUFPAlgorithm(filePath: String, percentage: double, k: int): phương thức được dùng để chạy thuật toán với các cài đặt cần thiết.



Hình 5.1 Class Diagram

5.2 Xử lý dữ liệu

Để triển khai khai thác k tập mẫu tiện ích cao phổ biến đầu tiên, đề tài yêu cầu hai bộ dữ liệu:

- Bộ dữ liệu không chắc chắn: gồm danh sách các giao dịch và danh sách các sản phẩm trong mỗi giao dịch, ứng với mỗi sản phẩm trong giao dịch chứa xác suất tồn tại (existential probability) của sản phẩm.
- Bộ dữ liệu tiện ích: gồm danh sách các sản phẩm, tiện ích tương ứng của sản phẩm trong các giao dịch và tiện ích của mỗi giao dịch.

Các bộ dữ liệu trên được tham khảo từ thư viện nguồn dữ liệu mở SPMF. Trong đó, bộ dữ liệu không chắc chắn sẽ được bổ sung thêm xác suất tồn tại của sản phẩm bằng phương pháp sinh ngẫu nhiên để phù hợp với yêu cầu của đề tài.

5.3 Áp dụng cấu trúc dữ liệu CUP-lists

Từ hai bộ dữ liệu trên, các thông tin trong bộ dữ liệu sẽ được đưa vào các thuộc tính tương ứng trong CUP-lists. Bộ dữ liệu không chắc chắn cung cấp tên tập mẫu, danh sách giao dịch và xác suất mong đợi của tập mẫu tại giao dịch đó. Bộ dữ liệu tiện ích cung cấp giá trị tiện ích của tập mẫu trong mỗi giao dịch.

Tiếp đến, các thông tin trên được sử dụng để tính toán độ tin cậy mong đợi của tập mẫu, xác suất mong đợi lớn nhất trong TEP-list và tiện ích của tập mẫu.

Sau khi duyệt cơ sở dữ liệu, ta có được danh sách CUP-lists 1-mẫu.

5.4 Thiết lập thuật toán TUHUFP

Bằng cách sắp xếp danh sách CUP-lists 1-mẫu theo thứ tự giảm dần của độ tin cậy mong đợi, đưa ra k số lượng tập mẫu đầu tiên là tập mẫu có tiện ích cao vào danh sách kết quả.

Đồng thời, kiểm tra 1-mẫu trong danh sách theo chiến lược 3 (TWU lớn hơn hay nhỏ hơn ngưỡng tiện ích tối thiểu) để có được một danh sách ứng viên ($\mathbf{u}_1$).

TUHUFP_Search là phương thức sử dụng để khai thác các tập mẫu phổ biến không chắc chắn có tiện ích cao từ danh sách **u**.

Gọi phương thức TUHUFP_Search với chiến lược chia để trị. Đầu tiên với danh sách ứng viên k-mẫu, thuật toán sẽ kết hợp các phần tử trong danh sách với nhau để tạo ra (k+1)-mẫu. Nếu (k+1)-mẫu này có độ tin cậy mong đợi nhỏ hơn ngưỡng tin cậy mong đợi thì thuật toán sẽ bỏ qua. Ngược lại, thuật toán sẽ kiểm tra:

1) Nếu giá trị tiện ích của mẫu có lớn hơn ngưỡng tiện ích thì thêm vào danh sách kết quả;

2) Gọi $\mathbf{u}_{k+1}$ là danh sách ứng viên mới, nếu TWU của mẫu lớn hơn ngưỡng tiện ích thì thêm mẫu vào $\mathbf{u}_{k+1}$, tiếp tục gọi TUHUFP_Search với danh sách ứng viên để tạo danh sách (k+2)-mẫu. Chiến lược này được sử dụng cho đến khi tất cả các mẫu đã được xem xét.

THUẬT TOÁN 1: Thuật toán TUHUFP
Đầu vào: U: một cơ sở dữ liệu không chắc chắn, H: cơ sở dữ liệu tiện ích, percentage: ngưỡng tiện ích mong muốn của người dùng (%), k: số lượng UHUFP cần tìm. **Đầu ra:** k tập mẫu phổ biến không chắc chắn có tiện ích cao đầu tiên
1. Đọc cơ sở dữ liệu U và H để tạo danh sách CUP (1-mẫu) 2. Tính tiện ích của cơ sở dữ liệu: *databaseUtil* ← *SUM(tu(Tq)* ∈ *H)* 3. *minUtil* ← *databaseUtil*percentage* 4. *threshold* ← *0* 5. Sắp xếp lại danh sách CUP (1-mẫu) theo *expSup* 5. *Candidates* ← *Candidates* ∪ *CUPx* ∈ *CUP, CUPx.TWU* ≥ *minUtil* 6. *topUHUFP* ← *topUHUFP* ∪ *CUPx* ∈ *CUP, CUPx.utility* ≥ *minUtil* 7. *threshold* ← phần tử cuối *CUPx* ∈ *CUP, topUHUFP.size = k* 8. *TUHUFPSearch(Candidates, threshold, minUtil, k)*

THUẬT TOÁN 2: Phương thức TUHUFP_Search

Đầu vào: P: một danh sách chứa các Cup, threshold: ngưỡng phổ biến, minUtil: ngưỡng tiện ích mong muốn của người dùng, và k là số lượng UHUFP cần tìm.

Đầu ra: k tập mẫu phổ biến không chắc chắn có tiện ích cao đầu tiên

```
1.  Nếu P.size > 1 thì:
2.      Với mỗi Px ∈ P, thực hiện:
3.          ListofnewCUP ← ∅
4.          Với mỗi Py ∈ P, y > x thực hiện
5.              overestimate ← Px.expSup*Py.max
6.              Nếu overetimate < threshold , thì
7.                  Thực hiện Px tiếp theo, quay lại 2
8.              Kết thúc
9.              combined ← combineCup(Px, Py)
10.             Nếu combined.expSup > threshold, thì
11.                 // thêm vào kết quả khi combined là HUP
12.                  và đặt lại threshold khi cần
13.                 TUHUFPSearchHelper(combined, k)
14.                 Nếu combined.TWU ≥ minUtil, thì
15.                     ListofnewCUP ← ListofnewCUP ∪ combined
16.                 Kết thúc
17.             Kết thúc
18.         Kết thúc
```

19. *TUHUFP_Search(ListofnewCUP, threshold, minUtil, k)*
20. **Kết thúc**
21. **Kết thúc**

5.5 Một số cài đặt khác

5.5.1 Lưu trữ kết quả

Trong quá trình cài đặt thuật toán, có 2 cách để lưu trữ kết quả:

- Sử dụng List để lưu trữ các UHUFP
- Sử dụng Priority Queue

Kết quả chứa các k UHUFP và khi thêm phần tử mới, phải kiểm tra rằng có thỏa số lượng k hay không, nếu quá số lượng phải xóa đi phần tử có giá trị độ mong đợi tin cậy nhỏ nhất và cập nhật lại threshold (ngưỡng phổ biến). Tất nhiên, cách dễ dàng được nghĩ đến là sử dụng List để lưu trữ kết quả và sắp kết quả giảm dần theo độ mong đợi tin cậy. Khi vượt quá số lượng k thì phần tử cuối cùng sẽ bị xóa và threshold sẽ được đặt lại là phần tử liền trước. Để duy trì một danh sách luôn giảm dần thì danh sách đó sẽ được sắp xếp lại sau mỗi lần thêm phần tử mới vào danh sách.

Ngược lại, trong Priority Queue các phần tử trong hàng đợi được sắp xếp theo độ mong đợi tin cậy của chúng. Khi thêm một phần tử mới, hàng đợi ưu tiên sẽ được đặt vào vị trí thích hợp trong hàng đợi dựa trên độ ưu tiên của nó. Nếu số lượng phần tử vượt quá k, chúng ta loại bỏ phần tử có độ ưu tiên thấp nhất từ hàng đợi.

Việc sử dụng Priority Queue thường hiệu quả hơn về mặt thời gian so với việc sử dụng danh sách, đặc biệt là khi số lượng phần tử lớn. Việc sử dụng Priority Queue giúp giảm thiểu thời gian xử lý và tăng hiệu suất của thuật toán.

5.5.2 Thuộc tính phần tử cuối của CUP

Giả sử khi ta ghép 2 mẫu CD và CA để tạo thành CDA, được hiểu rằng C, D, A sẽ được ghép với nhau. Tuy nhiên, CD và CA lại có chung phần tử C vì vậy ta sẽ lượt

bỏ đi C trong CA và thực hiện ghép CD với A (phần tử cuối) để tạo thành CDA. Điều này cũng tương tự với việc ghép 2 CUP lại với nhau.

Để thực hiện truy suất phần tử cuối, có 2 giải pháp được đề ra:

- Sử dụng Map để lưu danh sách các phần tử 1-mẫu với value là 1-mẫu và key là tên của mẫu. Truy suất phần tử cuối cùng của CUP trong Map để thực hiện ghép.
- Thêm một thuộc tính bên trong CUP là last: là phần tử cuối cùng của k-mẫu, các 1-mẫu shẽ có phần tử cuối cùng là rỗng. Khi ghép 2 *CUP x* và *y* với nhau thì thì ta sẽ ghép *CUPx* với phần tử cuối cùng của *CUPy* khi *CUPx* và *CUPy* có chung phần tử đầu. Điều này sẽ dễ dàng hơn khi việc ghép diễn ra trực tiếp mà không cần truy suất.

Sử dụng thuộc tính last cho thấy được tính hữu dụng và dễ sử dụng hơn, còn sử dụng Map có thể tốn nhiều bộ nhớ hơn, đặc biệt là khi có nhiều mẫu.

5.5.3 Hướng dẫn người dùng

Link github chứa thuật toán:

https://github.com/huynhphucnguyen/CDNC1_Mining-top-k-high-utility-frequent-itemsets-from-uncertain-databases

Hướng dẫn chạy thực ngiệm trên terminal/command prompt:

Sau khi tải file nén về, tiến hành giải nén và thực hiện các bước sau:

1. Vào thư mục TUHUFP.
2. Giải nén data.zip (extract here) trong TUHUFP/data.
3. Tại thư mục TUHUFP nhập cmd trên thanh đường dẫn để mở command prompt.
4. Vào thư mục chứa thuật toán bằng câu lệnh:
   > cd algorithm/src

Chạy các bộ dữ liệu từ bài báo trong thư mục test:

1. Chess

> javac -d bin test/MainTestTUHUFPChess.java

> java -cp bin test.MainTestTUHUFPChess

Nhập số lượng k và ngưỡng tiện ích mong muốn.

Các mức testcase gồm: k = {100, 500, 900} và percentage = {12%, 14%, 16%, 18%, 20%, 22%}.

2. Foodmart

> javac -d bin test/MainTestTUHUFPFoodmart.java

> java -cp bin test.MainTestTUHUFPFoodmart

Nhập số lượng k và ngưỡng tiện ích mong muốn.

Các mức testcase gồm: k = {100, 500, 900} và percentage = {0.009%, 0.0095%, 0.01%, 0.0105%, 0.011%, 0.0115%}.

3. Retail

> javac -d bin test/MainTestTUHUFPRetail.java

> java -cp bin test.MainTestTUHUFPRetail

Nhập số lượng k và ngưỡng tiện ích mong muốn.

Các mức testcase gồm: k = {100, 500, 900} và percentage = {0.02%, 0.025%, 0.03%, 0.035%, 0.04%, 0.045%}

4. Pumsb

> javac -d bin test/MainTestTUHUFPPumsb.java

> java -cp bin test.MainTestTUHUFPPumsb

Nhập số lượng k và ngưỡng tiện ích mong muốn.

- Các mức testcase gồm: k = {100, 500, 900} và percentage = {0.2%, 0.4%, 0.6%, 0.8%, 1%, 1.2%}

CHƯƠNG 6. THỰC NGHIỆM

6.1 Thiết lập thực nghiệm

Thực nghiệm được thực hiện trên Laptop Dell với cầu hình Intel Core-i7 6700HQ CPU 2.60Hz, bộ nhớ RAM 16GB và hệ điều hành Windows 10. Thuật toán được cài đặt bằng ngôn ngữ Java trên IntelliJ IDEA Community.

Các cơ sở dữ liệu được sử dụng trong thực nghiệm này gồm: Foodmart, Retail, Chess, Pumsb. Đây là các cơ sở dữ liệu phổ biến được sử dụng nhiều cho việc thực nghiệm khai thác dữ liệu và được tham khảo trên thư viện nguồn dữ liệu mở SPMF: https://www.philippe-fournier-viger.com/spmf/index.php?link=datasets.php#d3

Mỗi cơ sở dữ liệu sẽ có một khoảng ngưỡng tiện ích riêng và phù hợp để tối ưu thực nghiệm. Thực hiện với 6 ngưỡng tiện ích khác nhau cho các cơ sở dữ liệu lần lượt với các top 100, 500 và 900.

Retail: 0.02%, 0.025%, 0.03%, 0.035%, 0.04%, 0.045%

Foodmart: 0.009%, 0.0095%, 0.01%, 0.0105%, 0.011%, 0.0115%

Chess: 12%, 14%, 16%, 18%, 20%, 22%

Pumsb: 0.2%, 0.4%, 0.6%, 0.8%, 1%, 1.2%

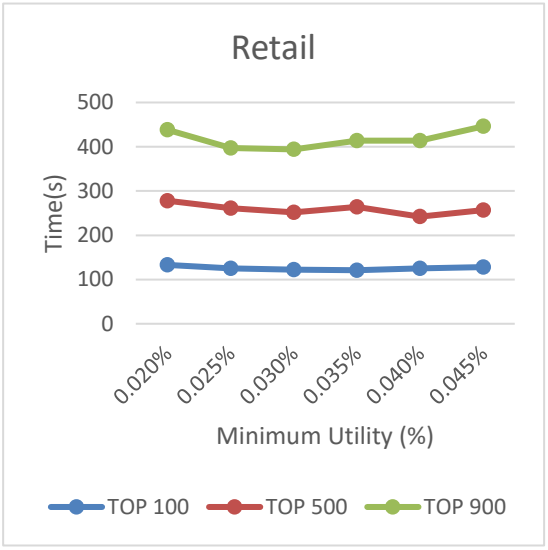
Cơ sở dữ liệu	Số lượng giao dịch	Số lượng sản phẩm	Độ dài trung bình
Retail	88,162	16,470	10.3
Foodmart	4,141	1,559	4.4
Chess	3,196	76	37
Pumsb	49,046	2113	74

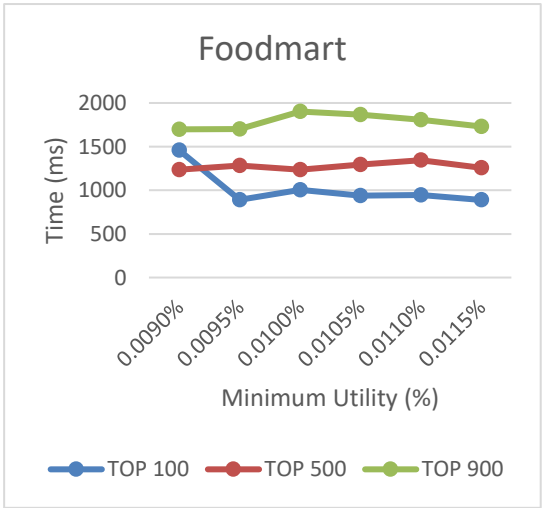
Bảng 6.1 Thông tin các cơ sở dữ liệu

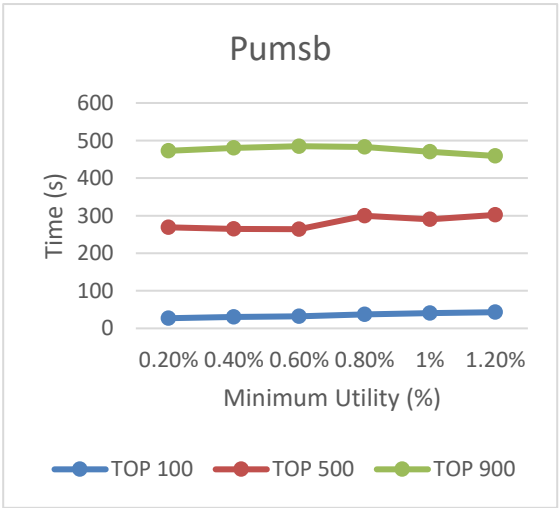
6.2 Kết quả thực nghiệm

6.2.1 Thời gian khai thác

Hình 7.1 cho ta thấy thời gian khai thác của thuật toán TUHUFP trên 4 bộ dữ liệu với 3 mức k và 6 ngưỡng tiện ích khác nhau. Cả 4 bộ, ta có thể thấy sự phân bố rõ ràng giữa các mức và ngưỡng. Thời gian khai thác của Chess có sự chênh lệch rõ ràng theo từng ngưỡng lợi ích. Ngược lại, Foodmart, Pumsbvà Retail không có sự chênh lệch nhiều giữa các ngưỡng tiện ích nhưng lại chênh lệch giữa các mức rất rõ ràng.







Hình 6.2.1 Thời gian khai thác của TUHUFP trên các điều kiện khác nhau

6.2.2 Số lượng ứng viên tham gia

Thông qua Hình 7.2, số lượng ứng viên tham gia của 4 bộ tại mức 100 gần như không có sự thay đổi qua các ngưỡng tiện ích và cả Foodmart và Retail cũng cho ra kết quả tương tự đối với 2 mức còn lại nhưng Retail lại tăng mạnh ở mức 900 với 0.04%. Số lượng ứng viên của Chess tăng dần khi tăng ngưỡng tiện ích và có lượng ứng viên chênh lệch nhau khá nhiều giữa các ngưỡng. Trong khi đó Pumsb thì lại cho kết quả giảm dần.

Hình 6.2.2 Số ứng viên tham gian vào TUHUFP trên các điều kiện khác nhau

6.2.3 Bộ nhớ sử dụng

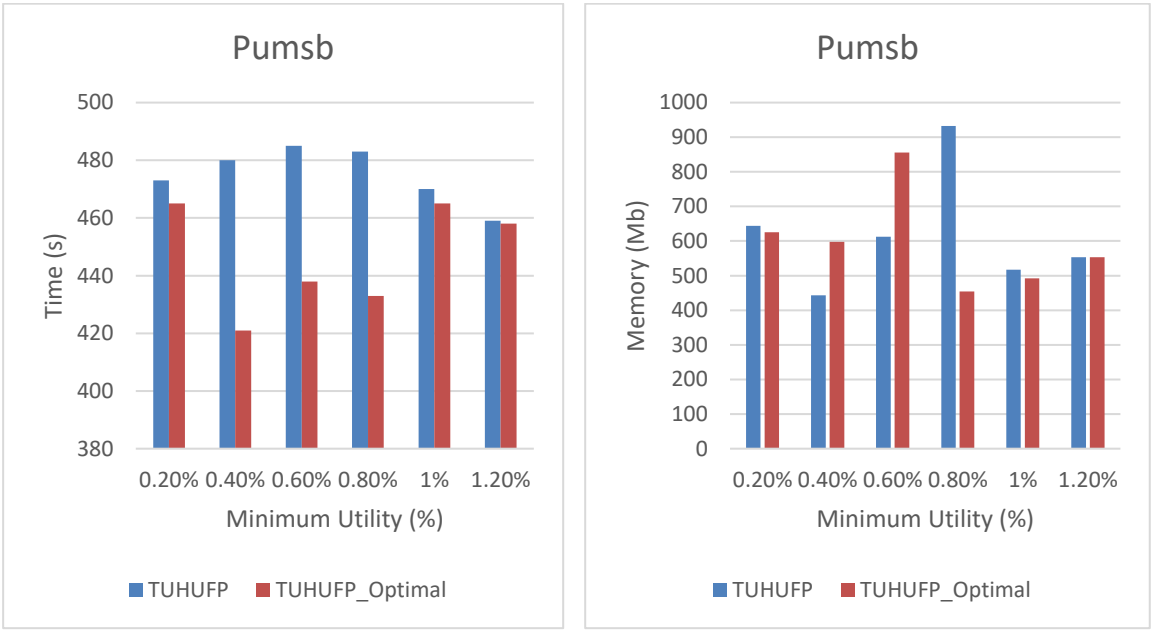
Hình 6.2.3 cho thấy rằng bộ nhớ sử dụng của các cơ sở dữ liệu đều biến thiên liên tục. Bộ nhớ sử dụng của Retail, Chess và Pumsb đều trên 100Mb. Trong đó Pumsb có bộ nhớ sử dụng cao nhất và nhỏ nhất là Foodmart dao động từ 80Mb đến 140Mb.

Hình 6.2.3 Bộ nhớ sử dụng của TUHUFP trên các điều kiện khác nhau

6.2.4 Thực nghiệm với cài đặt khác (TUHUFP-Optimal)

Thực nghiệm này chạy trên cơ sở dữ liệu Pumsb với mức k = 900 .

Hình 6.2.4 đã cho thấy được rằng TUHUFP_Optimal cớ thời gian chạy nhanh hơn và bộ nhớ sử dụng cũng tương đối it hơn TUHUFP.



Hình 6.2.4 Thời gian và bộ nhớ sử dụng của TUHUFP với TUHUFP_Optimal

CHƯƠNG 7. KẾT LUẬN

Thuật toán khai thác k tập mẫu phổ biến tiện ích cao (TUHUFP) là một công cụ quan trọng trong việc khám phá và nắm bắt thông tin từ các cơ sở dữ liệu không chắc chắn. Trong bối cảnh các bài toán phát hiện mẫu phổ biến thường mang tính NP-hard và có ứng dụng rộng rãi trong thực tiễn, TUHUFP nổi bật với khả năng xử lý dữ liệu không chắc chắn một cách hiệu quả, giúp phát hiện các k tập mẫu phổ biến có độ tin cậy cao.

Một trong những điểm mạnh của TUHUFP là khả năng đưa ra những kết quả có tính tiện ích cao. Sử dụng thành công chiến lược nâng ngưỡng và giảm số lượng mẫu tham gia. Các kết quả đầu ra của thuật toán cung cấp thông tin quan trọng về mẫu phổ biến trong dữ liệu, từ đó giúp định hình chiến lược kinh doanh, tối ưu hóa các quy trình hoặc dự đoán xu hướng tương lai.

Ngoài ra, TUHUFP còn đảm bảo tính linh hoạt và hiệu quả trong việc áp dụng cho nhiều loại cơ sở dữ liệu khác nhau. Sự kết hợp giữa tính linh hoạt, hiệu quả và tính tiện ích cao của TUHUFP mở ra nhiều cơ hội và tiềm năng trong nghiên cứu và ứng dụng thực tiễn. Trong tương lai, chúng tôi sẽ tiếp tục nghiên cứu những đề tài liên quan và cải thiện thêm về hiện năng cho TUHUFP khi cần thiết.

TÀI LIỆU THAM KHẢO

[1] Dam, T.-L., Li, K., Fournier-Viger, P., & Duong, Q.-h. (2016). *An efficient algorithm for mining top-rank-k frequent patterns.* Applied intelligence (Boston).

[2] Davashi, R. (2021). *ILUNA: Single-pass incremental method for uncertain frequent pattern mining without false positives.* Information Sciences.

[3] Davashi, R. (2022). *ITUFP: A fast method for interactive mining of Top-K frequent patterns from uncertain data.* Expert systems with applications.

[4] DŨNG, N. Đ. (2019). *KHAI THÁC TẬP MỤC LỢI ÍCH CAO SỬ DỤNG PHƯƠNG PHÁP TỐI ƯU ĐÀN KIẾN.* TRƯỜNG ĐẠI HỌC CÔNG NGHỆ.

[5] Duong, Q.-h., Liao, B., Fournier-Viger, P., & Dam, T.-L. (2016). *An efficient algorithm for mining the top-k high utility itemsets, using novel threshold raising and pruning strategies.* Knowledge-Based Systems.

[6] Fournier-Viger, P., Lin, J. C.-W., Chi, T. T., & Nkambou, R. (2019). *A Survey of High Utility Itemset Mining.* Springer.

[7] HAN, J., PEI, J., YIN, Y., & MAO, R. (2004). *Mining frequent patterns without candidate* . Data Mining and Knowledge Discovery.

[8] Krishnamoorthy, S. (2015). *Pruning strategies for mining high utility itemsets.* Expert systems with applications.

[9] Le, T., Vo, B., & Baik, S. (2018). *Efficient algorithms for mining top-rank-k erasable patterns using pruning strategies and the subsume concept.* Engineering applications of artificial intelligence.

[10] Le, T., Vo, B., Huynh, V.-N., Nguyen, N., & Baik, S. (2020). *Mining top-k frequent patterns from uncertain databases.* Springer.

[11] Lee, G., & Yun, U. (2017). *A new efficient approach for mining uncertain frequent patterns using minimum data structure without false positives.* Future generations computer systems.

[12] Lee, G., Yun, U., & Ryang, H. (2015). *An uncertainty-based approach: Frequent itemset mining from uncertain data with different item importance.* Knowledge-Based Systems.

[13] Lin, C.-W., Gan, W., Fournier-Viger, P., Hong, T.-P., & Tseng, V. (2016). *Efficient algorithms for mining high-utility itemsets in uncertain databases.* Knowledge-Based Systems.

[14] Lin, C.-W., Gan, W., Fournier-Viger, P., Hong, T.-P., & Tseng, V. (2016). *Efficient Mining of Uncertain Data for High-Utility Itemsets.* Interational Conference on Web-Age Information Management.

[15] Lin, C.-W., Gan, W., Fournier-Viger, P., Hong, T.-P., & Zhan, J. (2016). *Efficient mining of high-utility itemsets using multiple minimum utility thresholds.* Knowledge-Based Systems.

[16] Lin, J. C.-W., Li, T., Fournier-Viger, P., & Su, T.-P. H.-H. (2016). *Efficient Mining of High Average-Utility Itemsets with Multiple Minimum Thresholds.* SpringerLink.

[17] Qu, J.-F., & Fournier-Viger, M. L. (2019). *Efficient Algorithms for High Utility Itemset Mining Without Candidate Generation.* Springer.

[18] R. Agrawal, R. S. (1998). *Fast algorithms for mining association.*

[19] Ryang, H., & Yun, U. (2015). *Top-k high utility pattern mining with effective threshold raising strategies.* Knowledge-Based Systems.

[20] Tseng, V., Wu, C.-W., Fournier-Viger, P., & Yu, P. S. (2016). *Efficient Algorithms for Mining Top-K High Utility Itemsets.* IEEE Transactions on Knowledge and Data Engineering.

[21] VU, V. V., LAM, M. T., T. T., MANH, L. T., NGUYEN, T. T., NGUYEN, L. V., . . . (Senior Member, I. A. (2023). *FTKHUIM: A Fast and Efficient Method for Mining Top-K High-Utility Itemsets.* IEEE.

[22] Wu, C. W., Shie, B.-E., Yu, P. S., & Tseng, V. S. (2024). *Mining Top-K High Utility Itemsets.*