TỔNG QUAN

1. Tính cấp thiết

Bài toán khai thác tập mục hữu ích cao (High-Utility Itemset Mining – HUIM) ra đời từ năm 2004 và luôn có được sự quan tâm nghiên cứu, cải tiến với hàng loạt các tiếp cận được đề xuất. Các cải tiến đều nhằm giảm độ phức tạp thời gian và không gian, tuy nhiên đều chưa tận dụng được hết khả năng của các bộ vi xử lý đa nhân, vốn đã và đang phổ biến hiện nay.

Đa phần các tiếp cận đều là các dạng tuần tự, hoặc song song hóa nhưng còn hạn chế. Ngoài ra, các giải thuật đều đặt ra một số giả định không thực tế, nhằm giảm bớt độ phức tạp.

Với sự tiến bộ của công nghệ chế tạo vi xử lý, ngày nay hầu như tất cả máy tính điện tử đều được trang bị các CPU có nhiều hơn một nhân tính toán – còn được gọi là vi xử lý đa nhân. Các CPU này phổ biến ở mức từ 2 đến 12 nhân xử lý vật lý (hoặc từ 4 đến 24 nhân luận lý) với chi phí phù hợp cho nhiều đối tượng người sử dụng. Việc tích hợp nhiều nhân tính toán vào một đế CPU giúp giảm đáng kể thời gian chờ của các tác vụ có thời gian xử lý dài và giúp các ứng dụng thực thi được hiệu quả hơn.

Sự ra đời của các vi xử lý đa nhân cũng là động lực thúc đẩy sự phát triển của các kỹ thuật tính toán, xử lý song song. Ngày càng có nhiều các ứng dụng, giải thuật, mô hình được triển khai để tận dụng khả năng xử lý song song của vi xử lý. Thế nhưng đối với bài toán khai thác dữ liệu, việc áp dụng các mô hình xử lý song song vào quá trình khai thác vẫn còn hạn chế, đặc biệt là bài toán HUIM.

Trong thời gian gần đây, một số các nghiên cứu áp dụng mô hình xử lý song song đa nhân vào bài toán HUIM cũng đã được đề xuất [57], [58]. Tuy nhiên các tiếp cận này vẫn còn nhược điểm cần phải khắc phục, như chiến lược điều phối, khả năng tận dụng tối đa sức tính toán của các vi xử lý đa nhân, v.v…

Với các vấn đề nêu trên, các nội dung trọng tâm sau đây được đặt ra cho quá trình nghiên cứu nhằm tìm ra giải pháp khắc phục.

Nội dung 1: Làm thế nào tăng hiệu quả cho quá trình khai thác các tập mục hữu ích cao đóng để áp dụng, phục vụ cho bài toán tìm luật kết hợp hữu ích cao, và bảo toàn lượng thông tin đã khai thác.

Nội dung 2: Tìm hiểu các kỹ thuật, chiến lược kết hợp bài toán khai thác tập mục hữu ích cao, tập mục hữu ích cao đóng với định dạng CSDL có phân cấp các hạng mục. Và đề xuất mô hình song song hóa để cải thiện hiệu năng khai thác của bài toán thực tiễn này.

Nội dung 3: Tối ưu hóa quá trình khai thác HUI trên định dạng CSDL phân cấp với không gian tìm kiếm lớn, tận dụng hiệu quả hơn sức mạnh tính toán của các vi xử lý đa nhân phổ biến hiện nay để tăng hiệu quả.

Thông qua luận án, NCS tập trung nghiên cứu, đề xuất các phương pháp để cải thiện thời gian khai thác tập mục hữu ích cao bằng việc áp dụng linh hoạt mô hình xử lý song song để cải thiện hiệu quả của quá trình xử lý song song trên các CPU đa nhân.

2. Mục tiêu nghiên cứu

Luận án tập trung nghiên cứu, tìm kiếm các cấu trúc dữ liệu phù hợp, các chiến lược để tận dụng hiệu quả hơn nữa năng lực xử lý của các CPU đa nhân để cải thiện thời gian khai thác các tập mục hữu ích cao. Mục tiêu nghiên cứu của luận án có thể được tóm gọn vào ba nội dung:

- Đề xuất các chiến lược, kỹ thuật để cải thiện hiệu quả của bài toán khai thác tập mục hữu ích cao và bảo toàn lượng tri thức khai thác được.
- Áp dụng mô hình xử lý song song để nâng cao hiệu quả khai thác tập mục hữu ích cao.
- Mở rộng các chiến lược song song hóa để tận dụng tốt hơn các CPU đa nhân.

3. Đối tượng nghiên cứu

Luận án tập trung nghiên cứu các chủ thể sau:

- Các giải thuật khai thác tập mục hữu ích cao truyền thống hiện có như cùng các chiến lược hiệu quả đi kèm như Two-Phase, FHM, EFIM, iMEFIM.
- Các giải thuật khai thác biểu diễn cô đọng của tập mục hữu ích cao như CHUI-Miner, CLS-Miner, EFIM-Closed cùng các kỹ thuật liên quan.
- Các giải thuật khai thác tập mục hữu ích cao dựa trên cấu trúc phân cấp các hạng mục như MLHUI-Miner, CLH-Miner.
- Các giải thuật khai thác tập mục hữu ích cao sử dụng mô hình xử lý song song đa nhân như pEFIM, MCH-Miner.
- Định dạng CSDL có độ hữu ích tĩnh và định dạng CSDL có độ hữu ích biến đổi.
- Định dạng CSDL có sự phân cấp các hạng mục cùng với cấu trúc phân cấp.
- Các CSDL thực nghiệm chuẩn và công khai, thường sử dụng trong các báo cáo khoa học.

4. Các đóng góp của luận án

Luận án nghiên cứu và đề xuất các tiếp cận sau, tập trung vào việc cải thiện thời gian khai thác của bài toán khai thác tập mục hữu ích cao:

- Các tiếp cận cho phép khai thác hiệu quả các tập mục hữu ích cao sử dụng biểu diễn thu gọn và bảo toàn.
- Các mô hình song song hóa sử dụng kiến trúc xử lý đa nhân để rút ngắn thời gian thực hiện của các giải thuật khai thác tập mục hữu ích cao.

5. Ý nghĩa thực tiễn của luận án

Kết quả nghiên cứu của luận án có thể được áp dụng vào các bài toán sau:

- Giảm thiểu kích thước tập kết quả của bài toán khai thác luật kết hợp hữu ích cao trên CSDL thông thường và cả trên các CSDL có phân cấp các hạng mục mà vẫn bảo toàn độ chính xác. Và có thể đưa ra được các hạng mục có liên

quan với nhau, mang lại mức độ hữu ích cao trong quá trình vận hành các cửa hàng bán lẻ, siêu thị, v.v…

- Mở rộng các phương pháp khai thác tập mục hữu ích cao hiện có để tận dụng tối đa sức mạnh của các vi xử lý đa nhân đang được sử dụng rộng rãi hiện nay nhằm nâng cao hiệu quả của bài toán với chi phí thấp.

6. Bố cục của luận án

Luận án được cấu trúc thành năm chương. Trong đó, các đóng góp chính của luận án được trình bày tại **Chương 2**, **Chương 3** và **Chương 4**.

CHƯƠNG 1 TỔNG QUAN VỀ LĨNH VỰC NGHIÊN CỨU

1.1 Lịch sử của bài toán khai thác tập mục hữu ích cao

Bài toán khai thác tập mục hữu ích cao khởi nguồn từ bài toán khai thác tập mục phổ biến (FIM) (Agrawal & Srikant, 1994). Mục đích của FIM là nhằm xác các kết hợp hạng mục xuất hiện phổ biến bên trong một CSDL giao dịch. Mức độ phổ biến của các tập mục được xác định dựa trên một ngưỡng độ phổ biến tối thiểu ($minsup$) do người khai thác chỉ định. Kết quả của bài toán là các tri thức có dạng luật kết hợp, hỗ trợ cho việc ra quyết định. Do FIM chỉ xét sự xuất hiện của các hạng mục trong giao dịch và bỏ qua nguồn thông tin quan trọng về mức độ hữu ích của các hạng mục, đa phần các tập mục tìm được lại không mang lại giá trị hữu ích cao.

HUIM khắc phục vấn đề này bằng cách xét thêm các thông tin ứng với mỗi hạng mục. Mỗi hạng mục trong giao dịch sẽ được bổ sung số lượng và đơn giá của hạng mục tương ứng. Để tổng quát hóa, các thông tin này lần lượt được gọi là độ hữu ích nội và độ hữu ích ngoại. Tích của hai thông tin này cho biết mức độ hữu ích nhận được khi bán mặt hàng tương ứng. Khi một hạng mục hoặc tập mục có độ hữu ích thỏa một ngưỡng độ hữu ích tối thiểu ($minutil$) cho trước, chúng được gọi là các tập mục hữu ích cao.

Tuy nhiên, việc xem xét nhiều thông tin hơn mang lại không ít khó khăn và thách thức cho bài toán HUIM do độ đo mức độ hữu ích không thỏa tính chất bao đóng giảm (DCP) (Y. Liu et al., 2005). Đặc điểm này làm chi phí tính toán và không gian tìm kiếm của bài toán trở nên lớn hơn so với FIM.

Định nghĩa 1.1 (Bài toán khai thác tập mục hữu ích cao) (Y. Liu et al., 2005).

Cho CSDL giao dịch $\mathbb{D}$ và một ngưỡng độ hữu ích tối thiểu $minutil$, xác định tập đầy đủ tất cả tập mục hữu ích cao từ CSDL đã cho.

Đầu vào: CSDL giao dịch $\mathbb{D}$, ngưỡng $minutil$.

Đầu ra: tập $\mathbb{H}$ chứa đầy đủ các tập mục hữu ích cao trong $\mathbb{D}$.

Định nghĩa 1.2 (Độ hữu ích của một hạng mục trong giao dịch) (Y. Liu et al., 2005).

Cho hạng mục $i \in \mathbb{I}$ xuất hiện trong giao dịch T_c, độ hữu ích của i trong T_c được xác định như sau: $u(i, T_c) = iu(i, T_c) \times eu(i)$.

1.2 Các công trình liên quan

Một số tiếp cận để giải quyết bài toán HUIM gồm: Two-Phase (Y. Liu et al., 2005), TWU-Mining (Le et al., 2009), HUI-Miner (M. Liu & Qu, 2012), FHM (Fournier-Viger et al., 2014), EFIM (Fournier-Viger et al., 2014), v.v… Tuy nhiên, bài toán khai thác mẫu trả về một lượng lớn các tập mục, đòi hỏi thêm thời gian để phân tích kết quả. Các phương pháp thu gọn kích thước tập kết quả cũng đã được giới thiệu, tiêu biểu như: CHUI-Miner (Wu et al., 2016), CLS-Miner (Dam et al., 2019), EFIM-Closed (Fournier-Viger et al., 2016), v.v…

Các tiếp cận HUIM nói trên chỉ tập trung vào các hạng mục đơn lẻ và các kết hợp của chúng. Các tiếp cận này chỉ đáp ứng một phần của bài toán trong thực tế. Thông thường, các hạng mục được phân loại thành các nhóm, thuộc nhiều chủng loại khác nhau. Việc phân loại này giúp cho công việc quản lý được dễ dàng và thuận tiện.

Để tận dụng các thông tin này trong CSDL giao dịch, bài toán HUIM được mở rộng kể từ năm 2017. Bài toán được gọi là khai thác tập mục hữu ích cao tổng quát (Generalized HUIM – GHUIM). Một số phương pháp tiêu biểu gồm: MLHUI-Miner (Cagliero et al., 2017), CLH-Miner (Fournier-Viger et al., 2020), TKC (Nouioua et al., 2020). Bài toán GHUIM là một bài toán có độ phức tạp cao do không gian tìm kiếm được mở rộng và chưa có nhiều các phương pháp để giải quyết hiệu quả.

Để thời gian khai thác của bài toán HUIM, mô hình xử lý song song dựa trên CPU đa nhân đã được xem xét. Mức độ hiệu quả của việc áp dụng mô hình xử lý song song vào bài toán HUIM phụ thuộc vào nhiều yếu tố như: việc phân hoạch không gian tìm kiếm, chiến lược điều phối các tác vụ song song, kích thước CSDL, v.v… Một số các giải thuật song song hóa tiêu biểu gồm: pEFIM (Nguyen

et al., 2019), MCH-Miner (Vo et al., 2020), pCLH-Miner (Tung et al., 2021). Các giải thuật song song này có chung đặc điểm là mô hình song song hóa và chiến lược điều phối còn đơn giản, chưa tận dụng triệt để khả năng tính toán của các CPU đa nhân.

Mô hình phân tán cũng được áp dụng vào bài toán HUIM để tận dụng năng lực từ nhiều máy tính khác nhau. Một số phương pháp gồm: PHUI-Miner (Chen & An, 2016), P-FHM+ (Sethi et al., 2018), v.v… Tuy nhiên, điểm chung của các phương pháp này quá trình khai thác diễn ra tuần tự trên các máy tính.

1.3 Các vấn đề mở trong việc áp dụng xử lý song song vào HUIM

Trong quá trình tìm hiểu và khảo sát, NCS nhận thấy việc áp dụng mô hình xử lý song song vào bài toán HUIM gặp phải một số thách thức sau, và bài toán vẫn còn chưa có giải pháp hoặc giải pháp vẫn còn chưa triệt để:

- Chi phí quét CSDL lớn, đặc biệt đối với bài toán khai thác HUIM với biểu diễn đóng trên các CSDL có mật độ thấp, kích thước lớn và số lượng hạng mục nhiều. Khi đó, các kỹ thuật ghép, chiếu giao dịch sẽ không hiệu quả.
- Mô hình song song hóa chỉ được áp dụng vào duy nhất giai đoạn khai thác thông qua việc phân hoạch không gian tìm kiếm tĩnh, áp dụng chiến lược điều phối đơn giản. Vì vậy, không tận dụng tối đa năng lực xử lý của phần cứng đa nhân mang lại.
- Bài toán GHUIM có không gian tìm kiếm rất lớn và vẫn chưa có một phương pháp song song hóa hiệu quả để tăng tốc quá trình khai thác.

CHƯƠNG 2 KHAI THÁC HIỆU QUẢ TẬP MỤC HỮU ÍCH CAO ĐÓNG TỪ CƠ SỞ DỮ LIỆU CÓ ĐỘ HỮU ÍCH BIẾN ĐỘNG

2.1 Giới thiệu bài toán

Bài toán HUIM có không gian tìm kiếm lớn và độ phức tạp tính toán cao. Ngoài ra, kết quả trả về từ các giải thuật khai thác đều có số lượng tập mục cũng rất lớn. Bên cạnh đó, các giải thuật khai thác tập mục hữu ích cao trước đây đều giả định là độ hữu ích của các hạng mục là tĩnh. Điều này tạo ra các kết quả sai lệch so với thực tế và không khả dụng khi áp dụng lên các CSDL giao dịch có độ hữu ích thay đổi. Các vấn đề này đều được khắc phục, tuy nhiên, chỉ ở từng góc nhìn độc lập nhau chứ chưa được kết hợp để trở thành một giải pháp hoàn chỉnh.

Bài toán đặt ra là làm thế nào để giải quyết một cách triệt để bài toán khai thác tập mục hữu ích cao đóng với CSDL có độ hữu ích động. Từ đó có thể mở rộng để áp dụng các chiến lược tối ưu hóa, xử lý song song để tăng tốc quá trình khai thác của bài toán.

2.2 Mô hình độ hữu ích biến đổi

Để áp dụng các tiếp cận khai thác HUIM lên CSDL có độ hữu ích động, cách xác định độ hữu ích của các hạng mục ở **Định nghĩa 1.2** phải thay đổi hoàn toàn. Định nghĩa sau sẽ phản ánh sát hơn tính chất động của độ hữu ích các hạng mục trong thực tế.

Định nghĩa 2.1 (Độ hữu ích của hạng mục trong giao dịch)

Độ hữu ích của hạng mục i trong giao dịch T_c, ký hiệu $u(i, T_c)$, được định nghĩa lại như sau: $u(i, T_c) = iu(i, T_c) \times eu(i, T_c)$. Trong đó, $eu(i, T_c)$ là độ hữu ích ngoại của hạng mục i khi xuất hiện trong giao dịch T_c.

2.3 Giải thuật iEFIM-Closed

Giải thuật được đề xuất để giải quyết hiệu quả bài toán đã đặt ra, có tên gọi là iEFIM-Closed (Giải thuật 2.1). Giải thuật này có các đặc điểm sau:

- Giải thuật một pha, kế thừa từ giải thuật EFIM-Closed.
- Tích hợp mô hình độ hữu ích động của giải thuật iMEFIM.

- Giảm chi phí quét CSDL bằng việc áp dụng cấu trúc dữ liệu P-Set.

Input: tập mục α, CSDL chiếu $\alpha\mathbb{D}$, tập primary $Primary(\alpha)$, tập $Secondary(\alpha)$, ngưỡng $minutil$, tập chiếu $P\text{-}set(\alpha, i)$ mở rộng.
Output: tập các tập mục hữu ích cao là mở rộng của α.

FOREACH $i \in Primary(\alpha)$ **DO**
  $\beta = \alpha \cup \{i\}$;
  Duyệt $\alpha\mathbb{D}$ trong $P\text{-}set(\alpha, i)$ để tính $u(\beta)$ và tạo ra $\beta\mathbb{D}$; // gồm HTM.
  IF β không có mở rộng lùi **THEN**
    Tính $sup(\beta, z), su(\beta, z), lu(\beta, z), P\text{-}set(\beta, z)$ với mỗi hạng mục $z \in Secondary(\alpha)$ bằng cách duyệt $\beta\mathbb{D}$.
    IF $sup(\beta) = sup(\beta \cup \{z\}), \forall z \succ i \wedge z \in E(\alpha)$ **THEN**
      RETURN $\beta \cup \bigcup_{z \succ i \wedge z \in E(\alpha)}\{z\}$ nếu là HUI; // nhảy bao đóng
    ELSE
      $Primary(\beta) = \{z \in Secondary(\alpha) | su(\beta, z) \geq minutil\}$;
      $Secondary(\beta) = \{z \in Secondary(\alpha) | lu(\beta, z) \geq minutil\}$;
      $Search(\beta,\ \beta\mathbb{D},\ Primary(\beta),\ Secondary(\beta),\ minutil,\ P\text{-}set(\beta, z))$;
    IF β không có mở rộng tiến **AND** $u(\beta) \geq minutil$ **THEN**
      RETURN β;
    END ELSE
  END IF
END FOR

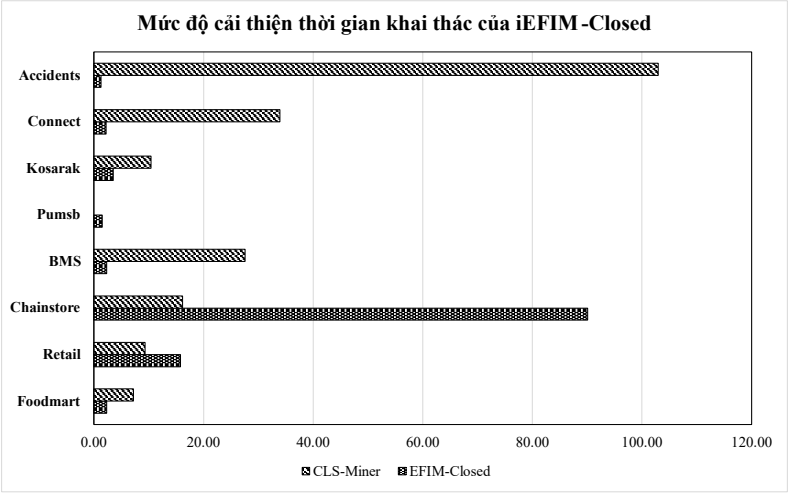
Giải thuật 2.1. Mã giả giải thuật iEFIM-Closed

2.4 Thực nghiệm

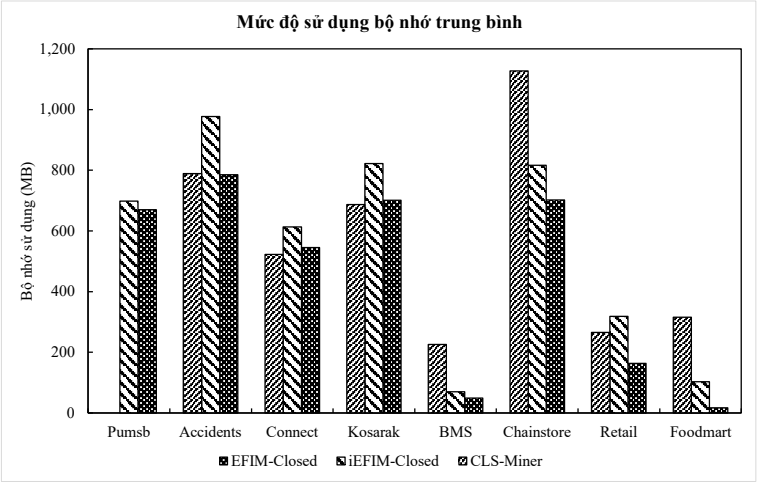
Giải thuật iEFIM-Closed được thử nghiệm trên nhiều CSDL giao dịch để đánh giá hiệu năng. Tuy nhiên, để phù hợp với định dạng CSDL có độ hữu ích biến đổi, tất cả CSDL này đều được phát sinh ngẫu nhiên giá trị độ hữu ích ngoại trong phạm vi [1,10] sử dụng phân bố chuẩn.

Để đánh giá hiệu năng khai thác của iEFIM-Closed, giải thuật được so sánh với hai giải thuật là EFIM-Closed và CLS-Miner. Hai giải thuật này cũng

được áp dụng **Định nghĩa 3.1** để có khả năng hoạt động trên các CSDL có độ hữu ích biến đổi.



Hình 2.1. So sánh về mức độ cải thiện thời gian trung bình



Hình 2.2. So sánh mức độ sử dụng bộ nhớ trung bình

Hình 2.1 thể hiện kết quả mức độ cải thiện thời gian khai thác của iEFIM-Closed so với EFIM-Closed và CLS-Miner. Có thể thấy iEFIM-Closed cho hiệu năng khai thác vượt trội cả EFIM-Closed và CLS-Miner. Thời gian khai thác mà

iEFIM-Closed được cải thiện lên đến 90 lần so với EFIM-Closed trên CSDL Chainstore và hơn 100 lần đối với CLS-Miner trên CSDL Accidents.

Mức độ sử dụng bộ nhớ trung bình của cả ba giải thuật được cho tại Hình 2.2. iEFIM-Closed cho thấy luôn có mức độ sử dụng bộ nhớ trung bình cao hơn EFIM-Closed để lưu trữ cấu trúc P-Set, tuy nhiên sự chênh lệch là không đáng kể so với hiệu năng lớn mà giải thuật mang lại.

Bảng 2.1 thể hiện chi phí quét CSDL của cả iEFIM-Closed và EFIM-Closed và được xác định dựa trên tổng số giao dịch được quét trong toàn bộ quá trình khai thác. Trong tất cả CSDL, số lượt giao dịch được quét bởi iEFIM-Closed luôn thấp hơn so với EFIM-Closed và mức độ cải thiện lên đến 99.4%.

Bảng 2.1. So sánh số giao dịch được quét

CSDL	iEFIM-Closed	EFIM-Closed	Giảm	Quét
Chainstore	10,021,074	6,141,636,120	**99.8%**	0.2%
Retail	861,798	134,978,290	**99.4%**	0.6%
Kosarak	8,101,676	872,102,126	**99.1%**	0.9%
Foodmart	75,484	6,716,739	**98.9%**	1.1%
BMS	175,270	10,002,024	**98.2%**	1.8%
Accidents	12,357,609	22,724,687	**45.6%**	54.4%
Connect	1,348,154	1,732,230	**22.2%**	77.8%
Pumsb	18,679,150	22,393,751	**16.6%**	83.4%

2.5 Kết chương

Đóng góp thứ nhất đề xuất giải thuật iEFIM-Closed phục vụ cho bài toán khai thác các tập mục hữu ích cao đóng và có khả năng thực hiện:

- Khai thác hiệu quả tất cả tập mục hữu ích cao đóng trong CSDL giao dịch.
- Có thể áp dụng lên các cơ sở dữ liệu có lợi nhuận thay đổi.
- Áp dụng một kỹ thuật mới nhằm tăng hiệu năng thực hiện của giải thuật, giảm thiểu chi phí quét CSDL nhiều lần khi áp dụng lên các CSDL thực tế.

- Giải thuật có hiệu năng khai thác tốt trên cả hai định dạng CSDL có mật độ cao và mật độ thấp. Sự cải thiện đáng kể về hiệu năng khai thác được thể hiện trên các CSDL thưa.

Thông qua thực nghiệm cho thấy iEFIM-Closed đã giải quyết được nội dung nghiên cứu đầu tiên của luận án. Tuy nhiên, với cơ chế kiểm tra tính đóng của các tập mục kế thừa từ EFIM-Closed đều đặt ra các phụ thuộc dữ liệu. Vì vậy, việc triển khai mô hình song song hoá trên iEFIM-Closed là không khả thi.

Do đó, hướng phát triển của giải thuật có thể được xem xét là:

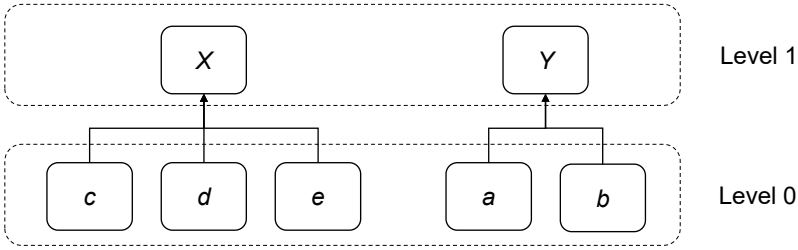
- Mở rộng giải thuật để có thể hoạt động trên các CSDL có phân cấp các hạng mục, phục vụ cho bài toán GHUIM.
- Nghiên cứu áp dụng cơ chế song song hoá theo từng mức của CSDL giao dịch có sự phân cấp các hạng mục.

CHƯƠNG 3 MÔ HÌNH SONG SONG HÓA KHAI THÁC TẬP MỤC HỮU ÍCH CAO ĐA MỨC

3.1 Giới thiệu bài toán

Bài toán khai thác tập mục hữu ích cao trên CSDL phân cấp được đề xuất vào năm 2017 và mở rộng của bài toán HUIM. Bài toán này xem xét sự phân cấp các hạng mục trong CSDL thông qua cấu trúc taxonomy. Hình 3.1 minh họa một taxonomy, trong đó các hạng mục a, b được phân vào một nhóm có tên gọi là Y; tương tự, c, d, e được phân vào nhóm X. Hạng mục X, Y được gọi là các hạng mục tổng quát và a, b, c, d, e được gọi là các hạng mục chuyên biệt. Thông qua taxonomy, có thể thấy được sự xuất hiện của các tri thức mà trước đây, bài toán HUIM truyền thống không thể phát hiện ra.

Cho đến nay chưa có một giải thuật nào được đề xuất để giải quyết bài toán khai thác tập mục hữu ích cao đóng trên CSDL phân cấp nhằm khắc phục vấn đề số lượng tập mục kết quả.



Hình 3.1. Cấu trúc phân cấp các hạng mục

3.2 Mô hình song song hóa

Để tận dụng được sức mạnh của các CPU đa nhân, các giải thuật cần phải được triển khai ở dạng song song hoá. Có nhiều phương pháp để thực hiện song song hoá, ví dụ như song song hoá thông qua tập lệnh vi xử lý (SIMD, MIMD), song song hoá thông qua cơ chế xử lý đa luồng (multi-threaded), v.v… Nhiều ứng dụng đều đã được triển khai dưới dạng song song hoá, tận dụng mô hình xử lý đa luồng để tăng hiệu quả. Đối với lĩnh vực khai thác dữ liệu, nhiều tiếp cận khai thác cũng được mở rộng theo hướng song song hoá.

Đối với bài toán khai thác tập mục hữu ích cao, một số tiếp cận song song hoá cũng được đề xuất. Tuy nhiên chỉ dừng ở song song hoá ở mức đơn giản. Đối với bài toán khai thác tập mục hữu ích cao trên CSDL phân cấp, hoàn toàn chưa có tiếp cận song song hoá nào được đề xuất để giảm thời gian khai thác. Vì vậy, một số nhận định sau được rút ra:

- Quá trình khai thác tập mục hữu ích cao trên CSDL phân cấp diễn ra trên taxonomy của CSDL.
- Đối với bài toán khai thác tập mục hữu ích cao đa mức, các tập mục khai thác là độc lập theo từng cấp trên taxonomy.
- Kích thước taxonomy càng lớn (độ sâu và chiều rộng mỗi level), thời gian khai thác sẽ gia tăng đáng kể.

- **Input:** CSDL $\mathbb{D}$, taxonomy τ, ngưỡng $minutil$
- **Output:** $\mathcal{R}$ tập các tập mục hữu ích cao đa mức

1 $\mathcal{R} \leftarrow \emptyset$; $EUCS \leftarrow \emptyset$;
2 Quét $\mathbb{D}$ và τ để xây dựng tập $\mathbb{I}$ và $\mathbb{G}$, xác định TWU của các $i \in \mathbb{I}$.
3 Quét $\mathbb{D}$ và τ để tính TWU cho các hạng mục tổng quát $g \in \mathbb{G}$.
4 Tỉa các hạng mục chuyên biệt: $\mathbb{I}^+ \leftarrow \{i \in \mathbb{I} \mid TWU(i) \geq minutil\}$.
5 Tỉa các hạng mục tổng quát: $\mathbb{G}^+ \leftarrow \{g \in \mathbb{G} \mid TWU(g) \geq minutil\}$.
6 Xây dựng $EUCS$ cho các hạng mục thuộc $\mathbb{I}^+ \cup \mathbb{G}^+$
7 **FOR EACH** level $l \in \tau$ **DO PARALLEL**
8 $\mathbb{IG}^+ \leftarrow$ các hạng mục tiềm năng thuộc $\mathbb{I}^+ \cup \mathbb{G}^+$ tại cấp l.
9 $\mathcal{R}_l \leftarrow$**FHM_Search** $(\emptyset, \mathbb{IG}^+, \emptyset, minutil, EUCS)$;
10 **END**
11 **RETURN** $\mathcal{R} \leftarrow \cup_{l \in \tau} \mathcal{R}_l$.

Giải thuật 3.1. Mã giả của giải thuật MCML-Miner

3.3 Giải thuật MCML-Miner

Giải thuật đầu tiên được đề xuất để giải quyết bài toán này là giải thuật MCML-Miner (Giải thuật 3.1) và là giải thuật đầu tiên triển khai mô hình song song cho bài toán khai GHUIM. Mô hình song song hoá được lựa chọn áp dụng là mô hình xử lý đa nhân. MCML-Miner được mở rộng dựa trên giải thuật

MLHUI-Miner. Quá trình khai thác chính của MLHUI-Miner trên từng cấp của taxonomy được triển khai song song hóa. Taxonomy với chiều sâu lớn kết hợp với các vi xử lý có nhiều nhân sẽ càng làm tăng hiệu quả của quá trình khai thác.

Các đặc điểm của giải thuật MCML-Miner gồm:

- Là giải thuật một pha mở rộng dựa trên MLHUI-Miner.
- Song song hóa quá trình khai thác ở mỗi cấp của taxonomy.

3.4 Giải thuật MLC-Miner

Nhằm thu gọn và bảo toàn tập kết quả của quá trình khai thác, biểu diễn đóng được xem xét áp dụng vào quá trình khai thác. Giải thuật MLC-Miner (Giải thuật 3.2) được đề xuất. Từ đó là cơ sở áp dụng mô hình song song hoá.

Trong suốt quá trình khai thác tập mục hữu ích cao đóng đa mức, mỗi một tập mục ứng viên được phát sinh, giải thuật cần phải tiến hành kiểm tra tính đóng của tập mục này. Giải thuật CHUI-Miner đề xuất việc kiểm tra tính đóng đơn giản thông qua việc kiểm tra bao đóng của các tập mục. Tuy nhiên, do độ phức tạp của phép kiểm tra rất lớn sẽ làm gia tăng đáng kể thời gian khai thác, và làm giảm đi hiệu suất của toàn bộ giải thuật. Vì vậy, một kỹ thuật tối ưu hoá có tên gọi Rapid Utility Closure Checker (RUCK) được đề xuất. Đây là đóng góp chính của toàn bộ giải thuật, giúp cải thiện hiệu năng khai thác một cách đáng kể. Chiến lược RUCK được đề xuất như sau:

- Chuyển đổi các utility-list từ dạng danh sách trở thành dạng bảng băm.
- Các phần tử của bảng băm này là một bộ $\langle TID,\ item \rangle$ trong utility-list.
- Không lưu trữ trường TID vì đã có trong bảng băm: giảm chi phí về bộ nhớ.
- Quá trình kiểm tra tính đóng được thực hiện thông qua kiểm tra $TidSet$ của các tập mục, chính là tập khoá của các bảng băm.

Đây là sự kết hợp các ưu điểm của hai cấu trúc dữ liệu là utility-list và P-Set. Giải thuật MLC-Miner (Multi-Level Closed HUI Miner) được đề xuất để khai thác các tập mục hữu ích cao đóng từ CSDL phân cấp, với các đặc điểm sau:

- Giải thuật một pha, kế thừa từ CHUI-Miner.
- Áp dụng kỹ thuật EUCP để thu nhỏ không gian tìm kiếm.
- Áp dụng chiến lược Rapid Utility Closure Checker (RUCK).

Input: CSDL $\mathbb{D}$, taxonomy τ, ngưỡng $minutil$, cờ useEUCP
Output: tập các tập mục hữu ích cao đóng, đa mức

1 $\mathcal{R} \leftarrow \emptyset$; $EUCS \leftarrow \emptyset$;
2 Quét $\mathbb{D}$ và τ để xây dựng tập $\mathbb{I}$ và $\mathbb{G}$, tính TWU của các $i \in \mathbb{I}$.
3 Tính TWU cho các hạng mục tổng quát $g \in \mathbb{G}$.
4 Tỉa các hạng mục chuyên biệt: $\mathbb{I}^+ \leftarrow \{i \in \mathbb{I} \mid TWU(i) \geq minutil\}$.
5 Tỉa các hạng mục tổng quát: $\mathbb{G}^+ \leftarrow \{g \in \mathbb{G} \mid TWU(g) \geq minutil\}$.
6 Quét $\mathbb{D}$ một lần nữa để xây dựng EU-List, kết hợp xây dựng $EUCS$ cho các hạng mục thuộc $\mathbb{I}^+ \cup \mathbb{G}^+$ nếu cờ $useEUCP =$ **TRUE**.
7 **FOR EACH** level $l \in \tau$ **DO**
8 $\mathbb{IG}^+ \leftarrow$ các hạng mục tiềm năng thuộc $\mathbb{I}^+ \cup \mathbb{G}^+$ tại cấp l.
9 $\mathcal{R}_l \leftarrow$**Explore**($\emptyset$, $\mathbb{IG}^+$, $\emptyset$, $minutil$, $EUCS$);
10 **END**
11 **RETURN** $\mathcal{R} \leftarrow \cup_{l \in \tau} \mathcal{R}_l$.

Giải thuật 3.2. Mã giả của giải thuật MLC-Miner

3.5 Giải thuật PMLC-Miner

MLC-Miner giải quyết hiệu quả bài toán khai thác tập mục hữu ích cao đóng trên CSDL phân cấp. Tuy nhiên, MLC-Miner là một giải thuật tuần tự. Vì vậy, phiên bản song song hóa có tên PMLC-Miner (Giải thuật 3.3) được triển khai để cải thiện thời gian khai thác, sử dụng cùng mô hình song song đã nêu. Giải thuật PMLC-Miner có các đặc điểm như sau:

- Giải thuật một pha mở rộng dựa trên MLC-Miner.
- PMLC-Miner kế thừa các toàn bộ đặc trưng của MLC-Miner.
- Rút ngắn thời gian khai thác thông qua mô hình song song hóa đa nhân.

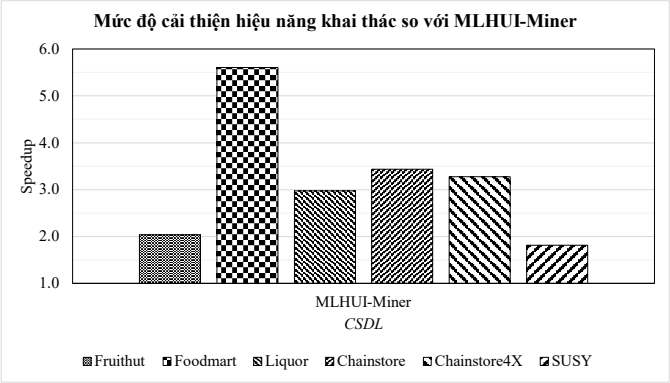
Input: CSDL $\mathbb{D}$, taxonomy τ, ngưỡng $minutil$
Output: tập các tập mục hữu ích cao đóng, đa mức $\mathcal{R}$

1	$\mathcal{R} \leftarrow \emptyset$; $EUCS \leftarrow \emptyset$;
2	Quét $\mathbb{D}$ và τ để xây dựng tập $\mathbb{I}$ và $\mathbb{G}$, tính TWU của $i \in \mathbb{I}$.
3	Tính TWU cho các hạng mục tổng quát $g \in \mathbb{G}$.
4	Tỉa các hạng mục chuyên biệt: $\mathbb{I}^+ \leftarrow \{i \in \mathbb{I} \mid TWU(i) \geq minutil\}$.
5	Tỉa các hạng mục tổng quát: $\mathbb{G}^+ \leftarrow \{g \in \mathbb{G} \mid TWU(g) \geq minutil\}$.
6	Quét $\mathbb{D}$ để xây dựng EU-List, xây dựng $EUCS$ cho $i \in \mathbb{I}^+ \cup \mathbb{G}^+$.
7	**FOR EACH** level $l \in \tau$ **DO PARALLEL**
8	$\mathbb{IG}^+ \leftarrow$ các hạng mục tiềm năng thuộc $\mathbb{I}^+ \cup \mathbb{G}^+$ tại cấp l.
9	$\mathcal{R}_l \leftarrow$**Explore**$(\emptyset, \mathbb{IG}^+, \emptyset, minutil, EUCS)$;
10	**END**
11	**RETURN** $\mathcal{R} \leftarrow \cup_{l \in \tau} \mathcal{R}_l$.

Giải thuật 3.3. Mã giả của giải thuật PMLC-Miner

3.6 Thực nghiệm

Ba giải thuật đề xuất được so sánh với các giải thuật tương ứng dựa trên: thời gian khai thác và mức độ sử dụng bộ nhớ. MCML-Miner được so sánh với MLHUI-Miner. MLC-Miner và PMLC-Miner sẽ được so sánh với CHUI-Miner* (CHUI-Miner mở rộng để làm việc với CSDL phân cấp) và phiên bản song song sử dụng cùng mô hình là CHUI-Miner**.



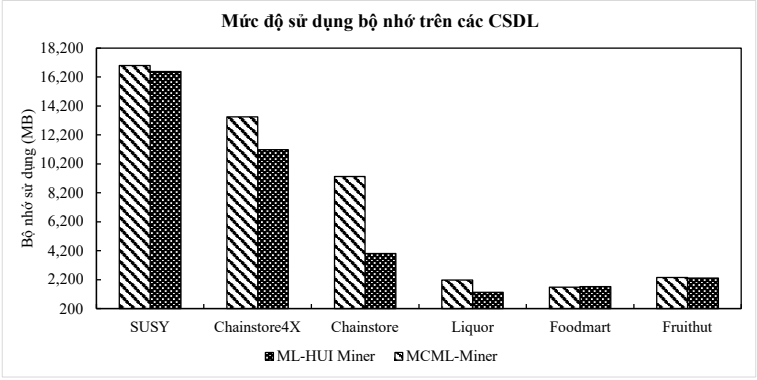
Hình 3.2. Mức độ cải thiện thời gian giữa MCML-Miner và MLHUI-Miner

3.6.1 Giải thuật MCML-Miner

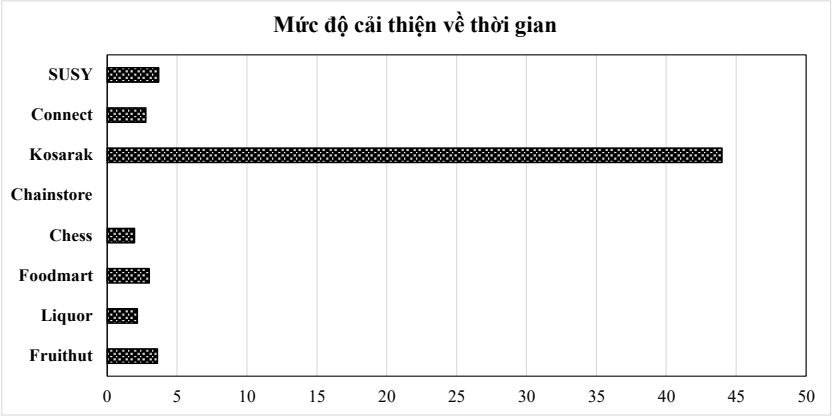
Hình 3.2 cho thấy tính hiệu quả về thời gian của mô hình song song hoá đa nhân của MCML-Miner: hiệu quả cải thiện đến 5.6 lần. Tuy nhiên, chiến lược

song song hoá phụ thuộc vào taxonomy và quá trình song song hoá chỉ diễn ra trong lúc khai thác. Vì vậy, mặc dù hiệu năng khai thác được cải thiện, MCML-Miner vẫn chưa tận dụng hết khả năng của các vi xử lý đa nhân.

MCML-Miner cũng cho thấy lượng bộ nhớ mà giải thuật yêu cầu luôn cao hơn MLHUI-Miner. Càng nhiều tác vụ song song, lượng bộ nhớ cần để quản lý chúng cũng gia tăng (Hình 3.3).



Hình 3.3. Mức độ sử dụng bộ nhớ trung bình

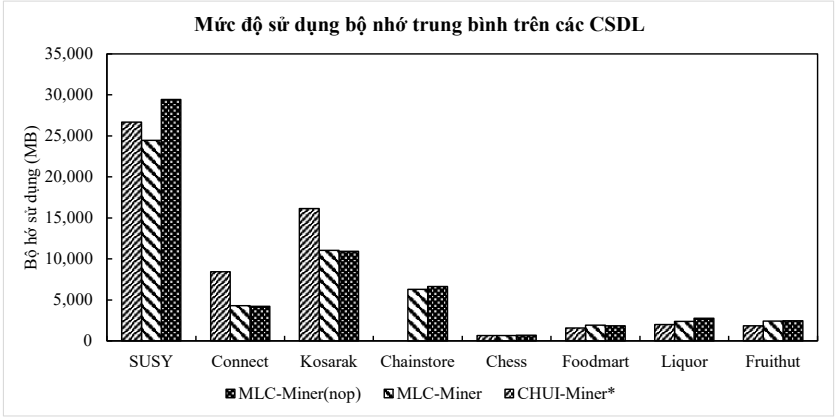


Hình 3.4. Mức độ cải thiện về thời gian giữa MLC-Miner và CHUI-Miner*

3.6.2 Giải thuật MLC-Miner

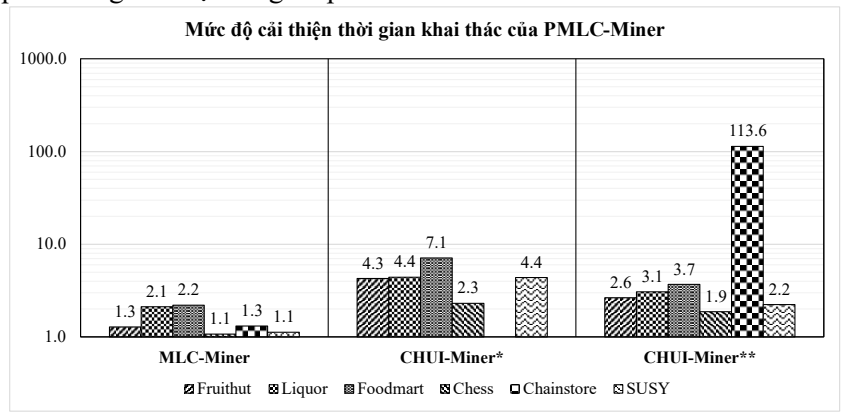
Mức độ cải thiện về thời gian thực hiện của MLC-Miner so với CHUI-Miner* được cho tại Hình 3.4. MLC-Miner cho thấy có hiệu năng tốt trên các

CSDL lớn. Đối với CSDL Chainstore, trong khi MLC-Miner hoàn thành dưới150 giây và CHUI-Miner* không thể hoàn thành trong phạm vi 7200 giây.



Hình 3.5. Bộ nhớ sử dụng trung bình của MLC-Miner và các giải thuật

Hình 3.5 thể hiện sự hiệu quả trong việc sử dụng bộ nhớ giữa MLC-Miner và CHUI-Miner*. Mức độ sử dụng bộ nhớ của MLC-Miner là thấp hơn so với CHUI-Miner*, kể cả không áp dụng EUCP là MLC-Miner(nop), lượng bộ nhớ cấp phát cho giải thuật cũng thấp hơn so với CHUI-Miner*.



Hình 3.6. Mức độ cải thiện thời gian của PMLC-Miner

3.6.3 Giải thuật PMLC-Miner

Đối với phiên bản song song hóa của MLC-Miner là PMLC-Miner, kết quả so sánh về mức độ cải thiện hiệu năng được thể hiện tại Hình 3.6 cho thấy

hiệu năng tăng gần gấp bảy lần so với CHUI-Miner*, hơn hai lần so với MLC-Miner. Đặc biệt với CHUI-Miner**, là phiên bản song song hóa của CHUI-Miner** sử dụng cùng mô hình song song nói trên , PMLC-Miner cho hiệu quả cao hơn đến hơn 110 lần ở CSDL Chainstore.

Về mức độ sử dụng bộ nhớ, kết quả tương tự như MCML-Miner. PMLC-Miner luôn có mức độ sử dụng bộ nhớ cao hơn so với phiên bản tuần tự.

3.7 Kết chương

Nội dung nghiên cứu này tập trung vào bài toán khai thác tập song song các mục hữu ích cao đa mức, đa mức - đóng trên định dạng CSDL phân cấp và đề xuất hàng loạt các phương pháp để giải quyết. Các giải thuật được đề xuất lần lượt có tên gọi là MCML-Miner, MLC-Miner và PMLC-Miner.

Nội dung nghiên cứu thứ hai của luận án đã cơ bản hoàn thành bằng việc đề xuất một mô hình song song hoá như đã trình bày. Tuy nhiên, có thể thấy được quá trình xử lý song song của mô hình đề xuất chỉ diễn ra duy nhất trong pha quét không gian tìm kiếm của giải thuật. Bên cạnh đó, mô hình song song hoá này cũng chưa tận dụng triệt để các nhân xử lý nhàn rỗi của các vi xử lý có nhiều nhân và còn phụ thuộc chính vào chiều sâu của taxonomy. Nếu áp dụng mô hình song song hóa này trên các CSDL thông thường, các CSDL không chứa taxonomy, thì quá trình song song hoá sẽ bị triệt tiêu.

Hướng phát triển tiếp theo của luận án sẽ tập trung vào việc cải thiện sâu hơn nữa mô hình song song hoá này. Từ đó, tận dụng tốt hơn, triệt để hơn sức mạnh tính toán song song của các vi xử lý đa nhân phục vụ cho bài toán khai thác các tập mục hữu ích cao hiệu quả hơn.

CHƯƠNG 4 CẢI THIỆN HIỆU QUẢ MÔ HÌNH KHAI THÁC SONG SONG CÁC TẬP MỤC HỮU ÍCH CAO ĐA MỨC

4.1 Giới thiệu bài toán

Nội dung nghiên cứu thứ 2 đã đề xuất một mô hình song song hoá dựa trên taxonomy để tiến hành khai thác. Mô hình này được áp dụng vào giải thuật MCML-Miner và PMLC-Miner để khai thác lần lượt các tập mục hữu ích cao đa mức và đa mức đóng. Tuy nhiên, mô hình này chỉ tập trung triển khai song song hoá quá trình khai thác và phụ thuộc vào chiều sâu của taxonomy.

Nội dung nghiên cứu thứ 3 này tập trung vào việc mở rộng, cải thiện sâu hơn nữa mô hình song song hoá nói trên để tận dụng triệt để các vi xử lý đa nhân.

4.2 Giải thuật MCML+

Nhược điểm của MCML-Miner là khai thác không gian tìm kiếm từng cấp trong taxonomy một cách độc lập. Thông thường với các CPU có 2 đến 4 nhân, MCML-Miner cho hiệu quả khai thác tốt. Nếu CPU có số nhân xử lý nhiều hơn chiều sâu của taxonomy, chiến lược sẽ này trở nên kém hiệu quả vì không sử dụng các nhân CPU còn rãnh rỗi.

Do đó cần có tiếp cận tốt hơn để giải quyết vấn đề của MCML-Miner, tận dụng tốt hơn năng lực xử lý của các CPU. Mô hình xử lý song song có thể áp dụng ở tại các giai đoạn khác nhau của giải thuật để tăng hiệu quả. Đặc điểm của giải thuật MCML+ gồm:

- Giải thuật một pha cải tiến từ giải thuật MCML-Miner.
- Áp dụng chiến lược song song hoá đa nhân ở các giai đoạn khác nhau cải thiện hiệu năng của quá trình khai thác.

4.3 Giải thuật MCML++

Tiếp cận của MCML+ (Giải thuật 4.1) đạt được mục đích cải thiện thời gian khai thác, đặc biệt trên các CSDL lớn. Tuy nhiên quá trình khai thác chính của MCML+ vẫn dựa trên MCML-Miner và chưa đạt được hiệu quả. Cần phải

xử lý triệt để nhược điểm này: tất cả nhân CPU phải được sử dụng cho quá trình quét không gian bài toán, thay vì chỉ tận dụng một vài nhân để quét taxonomy.

	Input: CSDL $\mathbb{D}$, taxonomy τ, ngưỡng $minutil$
	Output: tập các tập mục hữu ích cao đóng, đa mức $\mathcal{R}$
1	$\mathcal{R} \leftarrow \emptyset$; $EUCS \leftarrow \emptyset$;
2	Quét $\mathbb{D}$ và τ để xây dựng tập $\mathbb{I}$ và $\mathbb{G}$, tính TWU của $i \in \mathbb{I}$.
3	Tính TWU cho các hạng mục tổng quát $g \in \mathbb{G}$.
4	Song song tỉa các hạng mục: $\mathbb{I}^+ \leftarrow \{i \in \mathbb{I} \mid TWU(i) \geq minutil\}$.
5	Song song tỉa các hạng mục: $\mathbb{G}^+ \leftarrow \{g \in \mathbb{G} \mid TWU(g) \geq minutil\}$.
6	Song song hóa, quét $\mathbb{D}$ để xây dựng EU-List, xây dựng $EUCS$ cho các hạng mục thuộc $\mathbb{I}^+ \cup \mathbb{G}^+$ sử dụng taxonomy τ.
7	Sắp thứ tự các hạng mục tiềm năng trong sử dụng chiến lược SJF.
8	Đẩy các hạng mục $\mathbb{I}^+ \cup \mathbb{G}^+$ vào hàng đợi ưu tiên dựa trên thứ tự SJF.
9	**FOR EACH** $i \in \mathbb{I}^+ \cup \mathbb{G}^+$ **DO**$\mathbb{I}^+ \cup \mathbb{G}^+$ **PARALLEL**
10	$\mathbb{IG}_i \leftarrow$ các hạng mục tiềm năng có thể mở rộng i.
11	$\mathcal{R} \leftarrow \mathcal{R} \cup$ **Explore**$(\emptyset, \mathbb{IG}_i, \emptyset, minutil, EUCS)$;
12	**END**
13	**RETURN** $\mathcal{R}$.

Giải thuật 4.1. Mã giả giải thuật MCML++

Vấn đề này được giải quyết với việc song song hóa không gian tìm kiếm của từng hạng mục đơn, tương tự giải thuật pEFIM, MCH-Miner. Chiến lược điều phối được sử dụng trong các giải thuật này là FCFS. Việc ấn định các không gian tìm kiếm một cách ngẫu nhiên cho các nhân CPU mà không xem xét đặc điểm của chúng là không hiệu quả. Xét heuristic: các không gian tìm kiếm có kích thước nhỏ cần được đẩy lên thực hiện trước, rồi đến các không gian có kích thước lớn hơn. Chiến lược điều phối sử dụng heuristic này được gọi là Shortest Jobs First.

Định nghĩa 5.1. (Trình tự xử lý)

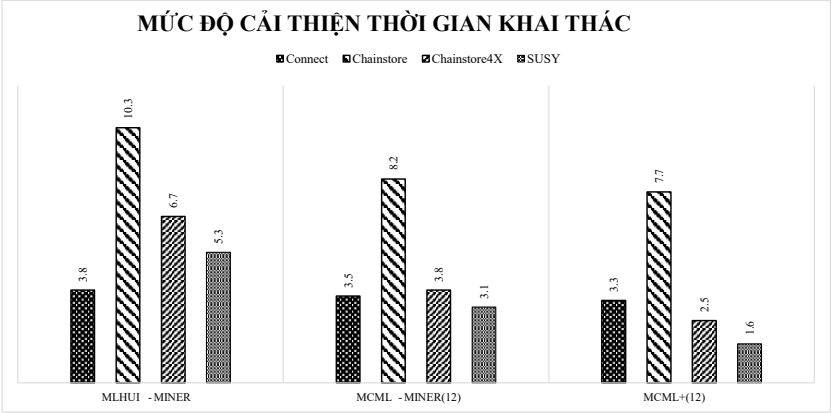
Cho $\mathcal{A}$ và $\mathcal{B}$ là hai tập mục cần được mở rộng. Không gian tìm kiếm của $\mathcal{A}$ là nhỏ hơn không gian tìm kiếm của $\mathcal{B}$ nếu $|E(\mathcal{A})| < |E(\mathcal{B})|$, trong đó $E(\mathcal{A})$ và $E(\mathcal{B})$ là tập các hạng mục có thể dùng để mở rộng lần lượt tập $\mathcal{A}$ và $\mathcal{B}$. Thứ tự xử lý có thể triển khai bằng một cấu trúc hàng đợi ưu tiên.

Giải thuật MCML++ tích hợp tất cả kỹ thuật song song hoá có đặc điểm:

- Giải thuật một pha sử dụng mô hình xử lý song song đa nhân.
- Loại bỏ phụ thuộc chiều sâu taxonomy để tận dụng tối đa CPU đa nhân.
- Áp dụng chiến lược điều phối Shortest Jobs First.

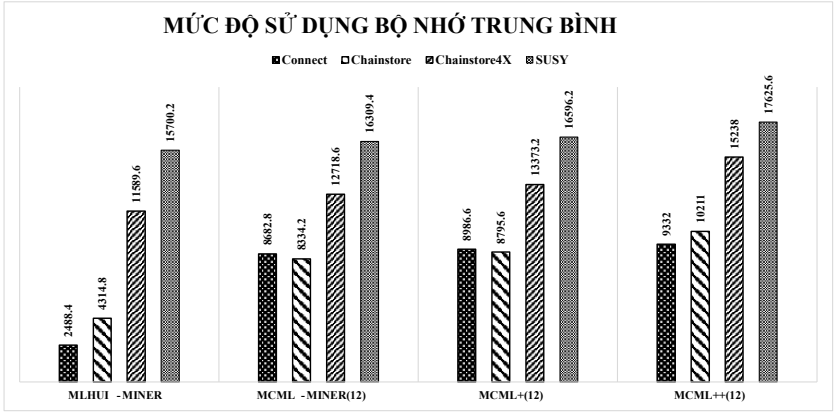
4.4 Thực nghiệm

Các thực nghiệm được triển khai để đánh giá hiệu quả của MCML+ và MCML++. Giải thuật MCML-Miner được ký hiệu là MCML(12). Giải thuật MCML+ và MCML++ được thử nghiệm với $\rho = 12$, được ký hiệu là MCML+(12) và MCML++(12). Hình 4.1 cho thấy MCML++ có mức độ cải thiện tốt hơn so với MCML+, MCML-Miner và MLHUI-Miner.



Hình 4.1. So sánh mức độ cải thiện thời gian khai thác

Hình 4.2 thể hiện lượng bộ nhớ trung bình của các giải thuật. MCML++ cho thấy lượng bộ nhớ cao hơn các giải thuật còn lại. Tuy nhiên mức độ gia tăng bộ nhớ so với với hiệu năng được cải thiện là không lớn. Ví dụ, MCML++ với 12 nhân trên CSDL SUSY có mức tăng bộ nhớ chỉ xấp xỉ 2.0GB trong khi mức độ cải thiện hiệu năng là hơn gấp 5.3 lần so với MLHUI-Miner (Hình 4.1).



Hình 4.2. So sánh mức độ sử dụng bộ nhớ trung bình

4.5 Kết chương

Nội dung nghiên cứu thứ 3 nghiên cứu bài toán khai thác song song và hiệu quả tập mục hữu ích cao đa mức từ CSDL phân cấp. Giải thuật MCML+ và MCML++ đề xuất dựa trên MCML-Miner. Mô hình xử lý song song được áp dụng lên các giai đoạn khác nhau của giải thuật. Chiến lược điều phối Shortest Jobs First cũng được đề xuất để điều khiển quá trình khai thác song song.

Thực nghiệm cho thấy MCML+, MCML++ có hiệu quả tốt hơn đáng kể so với MCML-Miner. Mức độ sử dụng bộ nhớ của MCML+ và MCML++ cao hơn so với MCML-Miner nhưng không lớn so với hiệu năng cải thiện.

Mặc dù áp dụng hàng loạt các kỹ thuật nâng cao hiệu quả khai thác, các phương pháp đã đề xuất này vẫn có khả năng mở rộng trong tương lai.

- Việc phân chia không gian tìm kiếm của các giải thuật trên đều bất cân bằng. Các tác vụ khai thác sẽ kết thúc ở các thời điểm khác nhau. Vì vậy cần tìm hiểu kỹ thuật phân chia tải các công việc cân bằng hơn.
- Các giải thuật đã đề xuất chỉ hoạt động trên một hệ thống tính toán đơn, có năng lực xử lý và bộ nhớ giới hạn. Vì vậy, giải pháp tiếp theo là mở rộng giải thuật để có thể hoạt động trên các CSDL có kích thước lớn.

CHƯƠNG 5 KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1 Kết luận

Trong quá trình nghiên cứu, NCS đã đạt được một số kết quả bằng việc đề xuất hàng loạt các giải thuật khai thác tập mục hữu ích cao, tập mục hữu ích cao tổng quát, và kết hợp các bài toán này với mô hình xử lý song song đa nhân. Mục tiêu chính là để tận dụng tốt hơn nữa khả năng tính toán của các vi xử lý đa nhân với chi phí thấp. Các kết quả thực nghiệm cho thấy sự cải thiện hiệu năng của các phương pháp đã đề xuất. Đặc biệt là các tiếp cận sử dụng mô hình tính toán song song đa nhân. Hiệu năng cải thiện của các phương pháp này so với các tiếp cận không song song là đáng kể.

Về mặt phương pháp luận, các mô hình song song hoá đã được chứng minh thông qua mô hình toán học cũng như dựa trên thực nghiệm là cho khả năng cải thiện thời gian khai thác lên nhiều lần so với các phương pháp không áp dụng mô hình tính toán song song.

5.2 Hướng phát triển

Các giải thuật đã đề xuất mặc dù có hiệu quả rất tốt, chúng vẫn tồn tại các một số nhược điểm. Đặc điểm chung của các giải thuật khai thác tập mục theo chiều sâu là không gian tìm kiếm bị mất cân bằng, dẫn đến các tác vụ khai thác song song sẽ có thời gian thực hiện không tương đương nhau. Vì vậy, các kỹ thuật phân chia tải một cách cân bằng hơn cần phải được xem xét.

Ngoài ra, các mô hình xử lý song song đã đề xuất chỉ bó hẹp vào một hệ thống tính toán đơn với năng lực xử lý và lưu trữ giới hạn. Do đó, hướng phát triển tiếp theo là mở rộng giải thuật để có thể hoạt động trên các CSDL cực lớn thông qua môi trường phân tán.