THÔNG TIN & TRUYỀN THÔNG

CƠ QUAN CỦA BỘ THÔNG TIN VÀ TRUYỀN THÔNG

ISSN 1859-3526

CÁC CÔNG TRÌNH NGHIÊN CỨU PHÁT TRIỂN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

Tập 2019
Số 2
Tháng 12

Journal of Research and Development on Information and Communication Technology

THÔNG TIN VÀ TRUYỀN THÔNG
CƠ QUAN CỦA BỘ THÔNG TIN VÀ TRUYỀN THÔNG
ISSN: 1859-3526

TẬP 2019, SỐ 2, THÁNG 12

TỔNG BIÊN TẬP: VŨ CHÍ KIÊN

CÁC CÔNG TRÌNH NGHIÊN CỨU PHÁT TRIỂN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

THƯỜNG TRỰC BAN BIÊN TẬP

Trưởng Ban

Nguyễn Linh Trung, Trường Đại học Công nghệ, Đại học Quốc gia Hà Nội

Các đồng Trưởng ban

Nguyễn Hoàng Hà, Đại học Saskatchewan, Canada
Đỗ Ngọc Minh, Đại học Illinois, Hoa Kỳ

Các Phó trưởng ban

Nguyễn Văn Đức, Trường Đại học Bách khoa Hà Nội
Lê Vũ Hà, Trường Đại học Công nghệ, Đại học Quốc gia Hà Nội
Nguyễn Nam Hoàng, Trường Đại học Công nghệ, Đại học Quốc gia Hà Nội
Lê Hoàng Sơn, Viện Công nghệ Thông tin, Đại học Quốc gia Hà Nội
Trần Xuân Tú, Trường Đại học Công nghệ, Đại học Quốc gia Hà Nội
Nguyễn Khánh Văn, Đại học Bách khoa Hà Nội

Thư ký khoa học: Trương Minh Chính, Trường Đại học Sư phạm, Đại học Huế

Thư ký hành chính: Nguyễn Lan Phương, Tạp chí Thông tin và Truyền thông

BIÊN TẬP VIÊN LĨNH VỰC

Trịnh Quốc Anh, Trường Đại học Khoa học Tự nhiên, Đại học Quốc gia Hà Nội
Võ Đình Bảy, Trường Đại học Công nghệ TP. Hồ Chí Minh
Ejder Bastug, NOKIA Bell Labs, Pháp
Huỳnh Thị Thanh Bình, Trường Đại học Bách khoa Hà Nội
Nguyễn Việt Dũng, Trường Công nghệ Tiên tiến quốc gia Bretagne, Pháp
Lê Đức Hậu, Vingroup
Phan Dương Hiệu, Trường Đại học Limoges, Pháp
Đậu Sơn Hoàng, Trường Đại học RMIT, Úc
Bùi Quang Hưng, Trường Đại học Công nghệ, Đại học Quốc gia Hà Nội
Nguyễn Tấn Hưng, Đại học Đà Nẵng
Phan Quốc Huy, Trường Đại học Kent, Anh
Trương Trung Kiên, Học viện công nghệ Bưu chính Viễn thông
Raghvendra Kumar, LNCT Group of Colleges, Ấn Độ
Nguyễn Lê Minh, Viện Khoa học và Công nghệ Tiên tiến Nhật Bản
Trần Minh Quang, Trường Đại học Bách khoa, Đại học Quốc gia TP. Hồ Chí Minh
Nguyễn Thái Sơn, Trường Đại học Trà Vinh
Lê Nhật Thăng, Học viện công nghệ Bưu chính Viễn thông
Quản Thành Thơ, Trường Đại học Bách khoa, Đại học Quốc gia TP. Hồ Chí Minh

ĐỊA CHỈ LIÊN HỆ:

115 Trần Duy Hưng, Hà Nội
Tel: (84)-24-37737136
Fax: (84)-24-37737130, Website http://ictmag.vn
Email: ict.research@mic.gov.vn (biên tập) chuyensanbcvt@mic.gov.vn (hành chính)
Chi nhánh TP. Hồ Chí Minh
27 Nguyễn Bỉnh Khiêm, Quận 1, TP. Hồ Chí Minh
Tel/Fax: (84)-28-38992898

LIÊN HỆ QUẢNG CÁO PHÁT HÀNH

Phạm Thị Lâm
Email: ptlam@mic.gov.vn, Tel: 84-948503999

Giấy phép xuất bản số 365/GP-BTTTT, ngày 19/12/2014; sửa đổi số 233/GP-BTTTT ngày 23/5/2017.
In tại Công ty CP In Hà Nội.
In xong và nộp lưu chiểu tháng 12/2019.

Giá: 40.000 đồng

Các công trình nghiên cứu phát triển
Công nghệ Thông tin và Truyền thông
http://www.ictmag.vn/cntt-tt
Tập 2019, Số 2
Tháng 12

MỤC LỤC

FGenHUSM: Một thuật toán hiệu quả khai thác các chuỗi sinh phổ biến lợi ích cao 57
Trương Chí Tín, Trần Ngọc Anh, Dương Văn Hải, Lê Hoài Bắc

Tăng tốc độ phát hiện dị thường trên ảnh đa phổ và siêu phổ ứng dụng trong tìm kiếm cứu nạn .. 70
Nguyễn Văn Phương, Đào Khánh Hoài, Tống Minh Đức

Song song hóa thuật toán Lamport trong loại trừ tương hỗ phân tán 83
Đặng Hùng Vĩ, Lê Văn Sơn, Nguyễn Xuân Huy

Toán tử lân cận mới cho thuật toán Tabu Search và PSO giải bài toán lập lịch luồng công việc ... 93
trong môi trường điện toán đám mây ..... ***Phan Thanh Toàn, Đặng Quốc Hữu, Nguyễn Thế Lộc***

Một phương pháp sinh luật mờ dựa trên cây quyết định và đại số gia tử xây dựng hệ luật mờ 102
giải bài toán hồi quy ..... ***Nguyễn Đức Dư, Hoàng Văn Thông***

Research and Development on
Information and Communication Technology *Vietnamese Edition*
http://www.ictmag.vn/cntt-tt **Volume 2019, Number 2**
December

TABLE OF CONTENTS

FGenHUSM: An Efficient Algorithm For Mining Frequent Generator High Utility Sequences 57
..... ***Truong Chi Tin, Tran Ngoc Anh, Duong Van Hai, Le Hoai Bac***

Acceleration of Anomaly Detection in Multispectral and Hyperspectral Images for Search and 70
Rescue Situations ..... ***Nguyen Van Phuong, Dao Khanh Hoai, Tong Minh Duc***

A Parallelization of the Lamport Algorithm for Distributed Mutual Exclusion 83
..... ***Dang Hung Vi, Le Van Son, Nguyen Xuan Huy***

New Effective Neighborhoods for Tabu Search and Particle Swarm Optimization to Schedule 93
Workflow in Cloud Computing ..... ***Phan Thanh Toan, Dang Quoc Huu, Nguyen The Loc***

A Method to Generate Fuzzy Rules based on Decision Tree and Hedge Algebras for Building 102
Fuzzy Rule-based Systems for Regression ..... ***Nguyen Duc Du, Hoang Van Thong***

FGenHUSM: Một thuật toán hiệu quả khai thác các chuỗi sinh phổ biến lợi ích cao

Trương Chí Tín[1], Trần Ngọc Anh[1], Dương Văn Hải[1,2], Lê Hoài Bắc[2]

[1] Khoa Toán – Tin học, Trường Đại học Đà Lạt

[2] Khoa Công nghệ Thông tin, Trường Đại học Khoa học Tự nhiên, Đại học Quốc gia Tp. Hồ Chí Minh

Tác giả liên hệ: Trần Ngọc Anh, anhtn@dlu.edu.vn

Ngày nhận bài: 15/07/2019, ngày sửa chữa: 09/10/2019, ngày duyệt đăng: 28/10/2019

Định danh DOI: 10.32913/mic-ict-research-vn.v2019.n2.872

Biên tập lĩnh vực điều phối phản biện và quyết định nhận đăng: PGS.TS. Lê Hoàng Sơn

Tóm tắt: Khai thác các chuỗi phổ biến và các chuỗi lợi ích cao có mức độ quan trọng khác nhau trong các ứng dụng thực tế. Gần đây, các nghiên cứu tập trung giải quyết bài toán tổng quát hơn, là khai thác tập *FHUS* chuỗi phổ biến lợi ích cao. Tuy nhiên, thời gian và bộ nhớ dùng để khai thác *FHUS* vẫn còn quá lớn. Bài báo đề xuất khái niệm tập *FGHUS* các chuỗi sinh phổ biến lợi ích cao, là một biểu diễn súc tích của *FHUS*, và một thuật toán mới hiệu quả để khai thác nó. Dựa vào hai chặn trên của độ đo lợi ích, hai chiến lược tỉa theo chiều rộng và sâu được thiết kế để loại bỏ nhanh các chuỗi ít phổ biến hoặc lợi ích thấp. Sử dụng một chặn dưới mới của lợi ích, một chiến lược tỉa địa phương mới được đề xuất để loại bỏ sớm các chuỗi không là chuỗi sinh phổ biến lợi ích cao. Dựa vào các chiến lược này, một thuật toán mới *FGenHUSM* được thiết kế để khai thác *FGHUS* mà tính hiệu quả của nó được thể hiện qua các thử nghiệm trên các cơ sở dữ liệu lớn.

Từ khóa: *Chuỗi lợi ích cao, khai thác chuỗi sinh phổ biến lợi ích cao, chặn trên và chặn dưới của độ đo lợi ích.*

Title: **FGenHUSM: An Efficient Algorithm For Mining Frequent Generator High Utility Sequences**

Abstract: Mining the set of all frequent high utility sequences (*FHUS*) in quantitative sequential databases (QSDBs) plays an important role in many real-life applications. However, for huge QSDBs and low minimum support and utility thresholds, algorithms for discovering *FHUS* often exhibit poor performance in terms of runtime and memory consumption due to the enormous cardinality of the *FHUS* set. To address this issue, this paper introduces a novel set of all frequent generator high utility sequences (*FGHUS*). This set is a concise representation of *FHUS* having a cardinality that is often much less than that of *FHUS*. Thus, it is more convenient for users to analyze the information provided by the *FGHUS* set. This paper proposes a novel algorithm named *FGenHUSM* to efficiently mine *FGHUS*. The algorithm adopts the depth and width pruning strategies to quickly eliminate infrequent or low utility sequences. In addition, it also uses a novel local pruning strategy to prune non-frequent generator high utility sequences early, which is based on a new lower bound on the utility measure. Experimental results on large QSDBs show that the proposed algorithm is efficient in terms of runtime for mining *FGHUS*, and that the pruning strategies can greatly reduce the search space.

Keywords: *High utility sequence, frequent generator high utility sequence, upper and lower bounds on utility measure.*

I. GIỚI THIỆU

Khi khai thác tập các chuỗi phổ biến (*FSM*: Frequent Sequence Mining) trên các cơ sở dữ liệu chuỗi tuần tự (SDB: Sequential Datadase), ta có thể đánh mất nhiều chuỗi lợi ích cao (HU: High Utility) quan trọng trong nhiều ứng dụng thực tế khi chúng ít phổ biến. Chẳng hạn, lợi ích của các mặt hàng bán được là thông tin rất hữu ích khi ra các quyết định trong kinh doanh. Vì vậy, bài toán khai thác các chuỗi HU (*HUSM*: High Utility Sequence Mining) trên các SDB lượng hóa (QSDB: Quantitative SDB) ra đời sau đó như một nhu cầu bức thiết.

HUSM tổng quát hơn *FSM* và có nhiều ứng dụng như phân tích hành vi duyệt web [1], dữ liệu thương mại di động [2], hiệu chỉnh gen [3], v.v. Ta xét một ứng dụng điển hình của *HUSM* là phân tích dữ liệu mua hàng. Xét cơ sở dữ liệu chuỗi biểu diễn các đơn mua hàng của khách hàng trong một cửa hàng bán lẻ. Khi đó một chuỗi chứa các mặt hàng được mua bởi một khách hàng ở các thời điểm khác nhau. Chi tiết hơn, nó là một danh sách có thứ tự của các tập mặt hàng, mỗi tập mặt hàng chứa các mặt hàng được mua cùng nhau. Lấy ví dụ, ta có chuỗi ⟨{kem đánh răng, bàn chải đánh răng}, {bánh Kinh đô, phô mai}, {nhang}⟩. Chuỗi mặt hàng có thể được mua bởi một phụ nữ. Người

này mua kem đánh răng và bàn chải đánh răng cùng nhau, sau đó mua bánh Kinh đô với phô mai, cuối cùng mua nhang. Mỗi mặt hàng có một giá bán ra. Khi đó, ta có thể xác định được giá trị của một tập các mặt hàng được mua cùng nhau cũng như giá trị của một danh sách con các tập mặt hàng trong chuỗi mua hàng của một khách hàng nào đó. Khi khai thác trên một tập rất lớn các đơn mua hàng, ta có thể biết được các thông tin như: (i) các mặt hàng nào thường được mua cùng nhau, (ii) lợi ích đem lại của một chuỗi các tập mặt hàng nào đó, v.v. Các thông tin này rất có ích cho việc ra các quyết định kinh doanh của cửa hàng.

Cho Ψ' là một QSDB chứa các chuỗi đầu vào, trong mỗi chuỗi có các sự kiện, trong mỗi sự kiện có các thuộc tính, mỗi thuộc tính gắn với một số lượng và một giá trị lợi ích. Do mỗi chuỗi α có thể xuất hiện ở các vị trí khác nhau (với các lợi ích khác nhau) trong Ψ', nên lợi ích của α trong Ψ' có thể được tính dưới dạng tổng [4], giá trị lớn nhất $u_{\max}(\alpha)$ [5, 6] hoặc giá trị nhỏ nhất $u_{\min}(\alpha)$ [7, 8] (theo nghĩa an toàn và ít rủi ro cho việc phát triển các chiến lược kinh doanh hay ra quyết định). Khi đó, lợi ích của α là tổng lợi ích trên tất cả các chuỗi Ψ' chứa α. Nếu lợi ích của α lớn hơn hoặc bằng một ngưỡng lợi ích tối thiểu mu, nó được gọi là chuỗi HU, ngược lại α được gọi là chuỗi lợi ích thấp (LU: Low Utility). Mục đích của *HUSM* là khai thác tập chuỗi lợi ích cao (*HUS*) chứa các chuỗi HU trên một QSDB.

Thuận lợi chính khi giải *FSM* là tính 'a priori', còn gọi là tính đơn điệu giảm (anti-monotonic) hay DCP (Downward-Closure Property), của độ đo hỗ trợ: Mọi chuỗi cha của một chuỗi ít phổ biến (LF: Low Frequent) cũng LF [5, 9]. Đặc tính này cho phép rút gọn đáng kể không gian tìm kiếm khi tiến hành khai thác các chuỗi lớn dần trên cây tiền tố. Đáng tiếc là trong *HUSM*, độ đo lợi ích không có tính DCP. Để khắc phục, các chặn trên (UB: Upper Bound) lợi ích được thiết kế để thu hẹp phạm vi tìm kiếm.

Chặn trên SWU [5] (thỏa DCP nhưng giá trị lớn) và các chặn trên khác chặt hơn (có giá trị nhỏ và gần với lợi ích hơn) lần lượt được thiết kế và sử dụng (mặc dù chỉ thỏa mãn các tính chất tựa DCP). Với $u_{\max}(\alpha)$, có thể kể ra SPU và SRU (2013), PEU và RSU (2016), gần đây là MEU [6] và SEU [10]. Với $u_{\min}(\alpha)$, các tác giả trong [7] đã đề xuất hai chặn trên, RBU và LRU, thiết kế hai chiến lược tỉa theo chiều sâu (DPS: Depth Pruning Strategy) và rộng (WPS: Width Pruning Strategy), và tích hợp chúng vào thuật toán *EHUSM* để khai thác hiệu quả *HUS*. Mặc dù *EHUSM* có thể khai thác nhanh *HUS*, lực lượng tập kết quả thường rất lớn, việc quản lý và phân tích chúng gây khó khăn đối với người sử dụng. Một tiếp cận thường được dùng trong *FSM* là khai thác các biểu diễn súc tích của chúng, chẳng hạn như các chuỗi tối đại, đóng và sinh [11, 12]. Chú ý rằng, bằng việc tích hợp *FSM* và *HUSM*, ta xét bài toán tổng quát hơn, là *FHUSM*: Khai thác tập *FHUS* các chuỗi phổ biến lợi ích cao (FHU: Frequent High Utility), đối với độ đo lợi ích $u_{\min}$. Theo cách tiếp cận trên, nhóm tác giả trong [13] đã đề xuất các thuật toán hiệu quả nhằm khai thác hai biểu diễn súc tích của *FHUS*, bao gồm các chuỗi phổ biến tối đại lợi ích cao (FMaxHU) và tập *FCHUS* các chuỗi phổ biến đóng lợi ích cao (FCHU). Tuy nhiên, chiều dài các chuỗi FMaxHU và FCHU thường khá lớn. Vì vậy, bài báo đề xuất một biểu diễn súc tích khác *FGHUS* của *FHUS*.

Trước đây, đã xuất hiện nhiều công trình nhằm khai thác các tập (tập thuộc tính) sinh phổ biến có/không có lợi ích cao [14–16]. Các chuỗi (danh sách có thứ tự các tập thuộc tính) đóng/tối đại/sinh phổ biến cũng đã là đối tượng của nhiều nghiên cứu gần đây [12, 17, 18]. Tập chứa các chuỗi sinh phổ biến lợi ích cao (*FGHUS*) là mở rộng tự nhiên của chuỗi sinh phổ biến truyền thống. Một chuỗi FHU được gọi là chuỗi sinh phổ biến lợi ích cao (FGHU) nếu không tồn tại chuỗi con FHU nào có cùng độ hỗ trợ.

Vì các chuỗi FGHU có chiều dài thường bé hơn nhiều so với các chuỗi FMaxHU và FCHU, nên chúng có các ưu điểm sau. Thứ nhất, ta có thể xem nó như một biểu diễn nén của *FHUS*. Điều này cũng rất phù hợp với nguyên lý chiều dài mô tả bé nhất (MDL: Minimum Description Length) [19]. Thứ hai, nó cho độ chính xác cao trong các nhiệm vụ chọn mô hình (so với *FHUS* hay tập *FCHUS*). Ngoài ra, khai thác các mẫu sinh (với chiều dài tối thiểu) còn là một bước quan trọng trong việc tìm các luật tuần tự quan trọng, chẳng hạn, các luật với ít giả thiết (vế trái) nhưng dẫn ra nhiều kết luận (vế phải), hoặc các luật không dư thừa [20]. Khi đó, các mẫu sinh chính là vế trái, vế phải có thể là các mẫu đóng hoặc không. Do đó, các mẫu sinh trong *FGHUS* thường được ưa thích hơn đối với người dùng khi cần phân tích tập kết quả, vì số lượng và chiều dài của chúng khá bé so với *FHUS*. Tập *FHUS* thường được sử dụng khi lực lượng của chúng khá bé, chẳng hạn khi ngưỡng hỗ trợ (ms) và ngưỡng lợi ích tối thiểu (mu) khá cao. Ngược lại, khi các ngưỡng này khá thấp và đặc biệt trên các tập dữ liệu lớn, để vượt qua khó khăn trong việc sử dụng cũng như phân tích tập kết quả *FHUS* với kích thước quá lớn, tập *FGHUS* sẽ là một lựa chọn phù hợp hơn với người dùng.

Mục tiêu của bài báo này là khai thác tập *FGHUS*. Để khai thác hiệu quả nó, do $FGHUS \subseteq FHUS$, nên một chuỗi LU hoặc LF sẽ không là FGHU. Do đó, tính chất DCP của độ hỗ trợ và hai chiến lược WPS và DPS dựa vào RBU và LRU [7, 8] có thể được sử dụng để tỉa hiệu quả các chuỗi LF hoặc LU không những từ *FHUS* mà cả *FGHUS*. Tuy nhiên, vì *FGHUS* không là tập con của tập tất cả các chuỗi sinh phổ biến (*FGS*), nên ta không thể áp dụng trực tiếp các điều kiện tỉa 3E trong [12] để loại bỏ các chuỗi không là chuỗi sinh lợi ích cao (GHU).

Bài báo có một số đóng góp sau đây: (i) đề xuất chặn dưới SF của $u_{\min}$; (ii) dựa vào điều kiện tỉa sớm tổng quát *GEP* [13] và SF, chiến lược tỉa địa phương (*LPG*) được thiết kế để loại bỏ sớm các ứng viên (và các mở rộng của chúng) không là GHU; (iii) tích hợp ba chiến lược *DPS*, *WPS* và *LPG* vào thuật toán *FGenHUSM* để khai thác các chuỗi sinh phổ biến lợi ích cao (FGHU); (iv) các thử nghiệm trên hai cơ sở dữ liệu lớn, thực tế và tổng hợp, đã chỉ ra tính hiệu quả của *FGenHUSM* (so với thuật toán cơ sở không áp dụng các chiến lược tỉa) về mặt thời gian chạy và lực lượng của *FGHUS* thường rất bé so với *FHUS*. Đây là thuật toán đầu tiên khai thác biểu diễn súc tích *FGHUS* của *FHUS* với độ đo lợi ích $u_{\min}$.

Phần còn lại của bài báo được tổ chức như sau. Phần II trình bày các khái niệm và kết quả cơ bản. Phần xây dựng chiến lược tỉa địa phương *LPG*. Phần IV đưa ra thuật toán *FGenHUSM* và các kết quả thử nghiệm. Phần V đưa ra các kết luận của bài báo.

II. CÁC KHÁI NIỆM VÀ KẾT QUẢ CƠ BẢN

1. Định nghĩa bài toán

Mục này giới thiệu vài khái niệm cơ sở liên quan đến bài toán *HUSM* với $u_{\min}$ trong [7].

Định nghĩa 1 ([7]): Gọi $\mathcal{A} \stackrel{\text{def}}{=} \{a_1, a_2, \ldots, a_M\}$ là tập các thuộc tính phân biệt. Mỗi thuộc tính a gắn liền với một số dương $\mathrm{P}(a)$ thể hiện giá trị lợi ích đơn vị của nó. Khi đó, ta có véctơ $\mathrm{P}(A) \stackrel{\text{def}}{=} \langle \mathrm{P}(a_1), \mathrm{P}(a_2), \ldots, \mathrm{P}(a_M) \rangle$. Một thuộc tính số lượng/định lượng (q-thuộc tính) là một cặp (a, q), với $a \in \mathcal{A}$ và $q \in R_+$ là một số lượng dương. Tập con A của $\mathcal{A}$, $A \subseteq \mathcal{A}$, được gọi là một sự kiện. Không mất tổng quát, giả sử rằng các thuộc tính trong một sự kiện được sắp tăng theo thứ tự từ điển $\prec$. Một sự kiện số lượng (q-sự kiện) ứng với A được định nghĩa là

$$A' \stackrel{\text{def}}{=} \{(a_i, q_i) \mid a_i \in A, q_i \in R_+\}.$$

A được gọi là sự kiện chiếu của A', ký hiệu là $A = \mathrm{proj}(A')$. Danh sách các q-sự kiện $\langle A'_k, k = 1, 2, \ldots, p \rangle$ ký hiệu là $\alpha' = A'_1 \to A'_2 \cdots \to A'_p$, được gọi là một q-chuỗi. Chuỗi chiếu α của q-chuỗi α' được định nghĩa là

$$\alpha = \mathrm{proj}(\alpha') \stackrel{\text{def}}{=} \mathrm{proj}(A'_1) \to \mathrm{proj}(A'_2) \to \cdots \to \mathrm{proj}(A'_p).$$

Để thuận tiện, ta ký hiệu $\alpha'[k] \stackrel{\text{def}}{=} A'_k$ và $\alpha[k] \stackrel{\text{def}}{=} \mathrm{proj}(A'_k)$. Một q-chuỗi là rỗng (<>) nếu tất cả các sự kiện của nó là rỗng. Một cơ sở dữ liệu chuỗi lượng hóa (QSDB) $\mathcal{D}'$ chứa hữu hạn các q-chuỗi đầu vào, $\mathcal{D}' = \{\Psi'_i, i = 1, 2, \ldots, N\}$ và $P(A)$. Mỗi q-chuỗi Ψ'_i gắn với một định danh (SID) i. Cơ sở dữ liệu chuỗi (không lượng hóa SDB) $\mathcal{D}$ ứng với $\mathcal{D}'$ được định nghĩa là

$$\mathcal{D} = \mathrm{proj}(\mathcal{D}') \stackrel{\text{def}}{=} \{\mathrm{proj}(\Psi'_i), i = 1, 2, \ldots, N\}.$$

Kích thước của q-chuỗi α', ký hiệu là $\mathrm{size}(\alpha')$, là số các q-sự kiện (p) của nó.

Từ đây về sau, ta xét hai q-chuỗi bất kỳ $\alpha' = A'_1 \to A'_2 \to \cdots \to A'_p$ và $\beta = B'_1 \to B'_2 \to \cdots \to B'_q$ cùng hai chuỗi tương ứng $\alpha = A_1 \to A_2 \to \cdots \to A_p$ và $\beta = B_1 \to B_2 \to \cdots \to B_q$.

Định nghĩa 2 ([7]): Xét hai q-sự kiện A' và B' sau:

$$A' = \{(a_{i_1}, q_{i_1}), (a_{i_2}, q_{i_2}), \ldots, (a_{i_m}, q_{i_m})\},$$
$$B' = \{(a_{j_1}, q_{j_1}), (a_{j_2}), q_{j_2}), \ldots, (a_{j_n}, q_{j_n})\},$$

với $m \le n$. A' được gọi là chứa trong B', ký hiệu là $A' \sqsubseteq B'$, nếu tồn tại các số tự nhiên $1 \le k_1 < \cdots < k_m \le n$ sao cho $a_{i_l} = a_{j_{k_l}}$ và $q_{i_l} = q_{j_{k_l}}$, với mọi $l = 1, 2, \ldots, m$.

Ngoài ra, ta nói α' chứa trong β', ký hiệu là $\alpha' \sqsubseteq \beta'$, nếu $p \le q$ và tồn tại p số nguyên dương, $1 \le j_1 < \cdots < j_p \le q$ sao cho $A'_k \sqsubseteq B'_{j_k}$, với mọi $k = 1, 2, \ldots, p$. Đồng thời, $\alpha' \sqsubset \beta'$ tương đương với $((\alpha' \sqsubset \beta') \wedge (\alpha' \ne \beta'))$.

Tương tự, ta cũng dùng ký hiệu $\sqsubseteq$ để định nghĩa quan hệ chứa trong trên tập tất cả các chuỗi. Ta nói $\alpha \sqsubseteq \beta$ hoặc $\beta \sqsupseteq \alpha$ (β được gọi là chuỗi cha của α) nếu tồn tại p số nguyên dương, $1 \le j_1 < \cdots < j_p \le q$ sao cho $A_k \subseteq B_{j_k}$, với mọi $k = 1, 2, \ldots, p$. Đồng thời, $\alpha \sqsubset \beta$ tương đương với $((\alpha \sqsubseteq \beta) \wedge (\alpha \ne \beta))$.

Ta nói β' chứa α, ký hiệu là $\alpha \sqsubseteq \beta'$ (hay $\beta' \sqsupseteq \alpha$, β' được gọi là q-chuỗi cha của α) nếu $\mathrm{proj}(\beta') \sqsupseteq \alpha$.

Gọi

$$\rho(\alpha) \stackrel{\text{def}}{=} \{\Psi' \in \mathcal{D}' \mid \Psi' \sqsupseteq \alpha\}$$

là tập tất cả các q-chuỗi đầu vào chứa α. Độ hỗ trợ của α được định nghĩa là số các q-chuỗi đầu vào chứa α,

$$\mathrm{supp}(\alpha) \stackrel{\text{def}}{=} |\rho(\alpha)|.$$

Định nghĩa 3 ([7]): Các lợi ích của q-thuộc tính (a, q), q-sự kiện $A' = (a_{i_1}, q_{i_1}), (a_{i_2}, q_{i_2}), \ldots, (a_{i_m}, q_{i_m})$, q-chuỗi α' và QSDB $\mathcal{D}'$ lần lượt được định nghĩa là

$$u((a, q)) \stackrel{\text{def}}{=} P(a) * q,$$
$$u(A') \stackrel{\text{def}}{=} \sum_{j=1}^{m} u((a_{i_j}, q_{i_j})),$$
$$u(\alpha') \stackrel{\text{def}}{=} \sum_{i=1}^{p} u(A'_i),$$
$$u(\mathcal{D}') \stackrel{\text{def}}{=} \sum_{\Psi' \in \mathcal{D}'} u(\Psi').$$

Để tránh tính toán lặp lại lợi ích u của mỗi q-thuộc tính (a, q) trong các q-chuỗi Ψ' của $\mathcal{D}'$, ta tính chúng một lần và thay q bởi $u((a, q)) = \mathrm{P}(a) * q$ trong cơ sở dữ liệu. Biểu diễn tương đương này của QSDB $\mathcal{D}'$ được gọi là QSDB tích hợp của $\mathcal{D}'$, cũng được ký hiệu vắn tắt là $\mathcal{D}'$. Từ đây về sau ta chỉ xét các QSDB tích hợp.

Bảng I
$\mathcal{D}'$ – MỘT QSDB MINH HỌA

SID	Chuỗi
1	$\Psi'_1 = (a,9) \to (c,2)(e,20) \to (c,1)(g,50)$ $\to (c,8)(d,16) \to (c,8)(d,16)(e,35)(g,60)$ $\to (a,3)(c,5)(e,2)$
2	$\Psi'_2 = (c,9)(f,28) \to (a,15)(c,10)$ $\to (d,16)(g,40)(a,15)(e,20) \to (c,6)(d,24)(e,25)$
3	$\Psi'_3 = (g,20) \to (e,40)(f,12) \to (b,45)(c,1) \to (d,56)$
4	$\Psi'_4 = (e,40) \to (c,2)(f,20) \to (c,3)$

Xét QSDB minh họa trong Bảng I, sẽ được dùng cho các ví dụ trong suốt bài báo. Chuỗi $\alpha = e \to ce$ chứa trong Ψ'_1, $\alpha \sqsubseteq \Psi'_1$, vì nó xuất hiện trong Ψ'_i. Lần xuất hiện đầu tiên của α trong Ψ'_1 là q-chuỗi con $\alpha' = (e,20) \to (c,8)(e,35)$ của Ψ'_1. Vậy $\text{proj}(\alpha') = \alpha$ và $u(\alpha') = 20+8+35 = 63$. Ngoài ra, do $\alpha \sqsubseteq \Psi'_2$, nên $\rho(\alpha) = \{\Psi'_1, \Psi'_2\}$ và $\text{supp}(\alpha) = 2$. SDB $\mathcal{D}$ ứng với $\mathcal{D}'$ là

$$\begin{aligned}\mathcal{D} = \{&\Psi_1 = a \to ce \to cg \to cdeg \to ace,\\ &\Psi_2 = cf \to ac \to dg \to ae \to cde,\\ &\Psi_3 = g \to ef \to bc \to d,\\ &\Psi_4 = e \to cf \to c\}.\end{aligned}$$

Định nghĩa 4 ([7]): Giả sử $\alpha \sqsubseteq \beta'$. Gọi

$$O(\alpha, \beta') \stackrel{\text{def}}{=} \{\alpha' \mid (\alpha' \sqsubseteq \beta') \wedge (\text{proj}(\alpha') = \alpha)\}$$

là tập tất cả các lần xuất hiện α' của α trong β'. Lợi ích bé nhất (gọi tắt là lợi ích) của α trong β' được định nghĩa là

$$u_{\min}(\alpha, \beta') \stackrel{\text{def}}{=} \min\{u(\alpha') \mid \alpha' \in O(\alpha, \beta')\}.$$

Lợi ích của α trong $\mathcal{D}'$ được định nghĩa là

$$u_{\min}(\alpha) \stackrel{\text{def}}{=} \sum_{\Psi' \in \rho(\alpha)} u_{\min}(\alpha, \Psi')).$$

Lúc đó, α được gọi là chuỗi lợi ích cao (HU) nếu $u_{\min}(\alpha) \geq mu$.

Từ nay, ta viết gọn α_s^u để diễn tả rằng $u_{\min}(\alpha) = u$ và $\text{supp}(\alpha) = s$. Lý do của việc sử dụng $u_{\min}$ có thể tham khảo thêm trong [7, 8, 13]. Lấy ví dụ, xét $\alpha = e \to ce$ với $\rho(\alpha) = \{\Psi'_1, \Psi'_2\}$ và $\text{supp}(\alpha) = 2$. Dễ thấy rằng,

$$\begin{aligned}O(\alpha, \Psi'_4) = \{&(e,20) \to (c,8)(e,35),\ (e,20) \to\\ &(c,5)(e,20),\ (e,35) \to (c,5)(e,20)\}.\end{aligned}$$

Do đó, $u_{\min}(\alpha, \Psi'_1) = \min\{63, 45, 60\} = 45$ và tương tự, $u_{\min}(\alpha, \Psi'_1) = 51$. Vì vậy, ta có α_2^{96}.

Định nghĩa 5: Một chuỗi HU γ được gọi là chuỗi sinh lợi ích cao, GHU, nếu không tồn tại chuỗi con HU có cùng độ hỗ trợ với nó. Tập tất cả các chuỗi GHU được định nghĩa và ký hiệu là

$$\begin{aligned}GHUS \stackrel{\text{def}}{=} \big\{&\gamma \in HUS \mid \nexists \gamma^* \in HUS :\\ &(\gamma^* \sqsubset \gamma) \wedge (\text{supp}(\gamma) = \text{supp}(\gamma^*))\big\}.\end{aligned}$$

Tập các chuỗi sinh phổ biến lợi ích cao, FGHU, được định nghĩa và ký hiệu là

$$\begin{aligned}FGHUS \stackrel{\text{def}}{=} \big\{&\gamma \in FHUS \mid \nexists \gamma^* \in FHUS :\\ &(\gamma^* \sqsubset \gamma) \wedge (\text{supp}(\gamma) = \text{supp}(\gamma^*))\big\}.\end{aligned}$$

Khai thác tập *FGHUS* là mục tiêu của bài báo này. Ví dụ, xét $mu = 226$ và $ms = 2$. Với chuỗi $\gamma = a \to g \to cde$, ta có $\gamma_2^{228} \in FHUS$. Hơn nữa, γ là một chuỗi FGHU, vì không tồn tại $\gamma^* \in FHUS$ sao cho $\gamma^* \sqsubset \gamma$ và $\text{supp}(\gamma^*) = \text{supp}(\gamma)$. Đây cũng là chuỗi FGHU duy nhất, tức là $FGHUS = \{\gamma\}$. Để ý rằng, lực lượng của *FGHUS* thường bé hơn rất nhiều so với *FHUS*, đặc biệt khi mu và ms bé. Chẳng hạn, với $mu = 1$ và $ms = 1$, $|FHUS| = 5235$, trong khi đó $|FGHUS| = 105$ (khoảng 2% của $|FHUS|$).

Định nghĩa 6 ([7]): Ta định nghĩa s-mở rộng và i-mở rộng của α và β lần lượt là

$$\alpha \diamond_s \beta \stackrel{\text{def}}{=} A_1 \to A_2 \to \cdots \to A_p \to B_1 \to B_2 \to \cdots \to B_q,$$

$$\alpha \diamond_i \beta \stackrel{\text{def}}{=} A_1 \to A_2 \to \cdots \to (A_p \cup B_1) \to B_2 \to \cdots \to B_q,$$

trong đó $a \prec b$ với mọi $a \in A_p$ và $b \in B_1$. Một mở rộng tiến (hay mở rộng) của α với β, ký hiệu là $\gamma = \alpha \diamond \beta$, là một i-mở rộng hoặc là s-mở rộng, tức là $\alpha \diamond_i \beta$ hay $\alpha \diamond_s \beta$. Khi đó, α được gọi là một tiền tố của γ và β là một hậu tố (đối với α) của γ. Ngoài ra, nếu δ là tiền tố bé nhất (của γ với $\sqsubseteq$) chứa α, ta ký hiệu δ là $\text{pref}(\gamma, \alpha)$. Hậu tố β của γ (đối với δ, tức là $\gamma = \delta \diamond \beta$) được ký hiệu là $\text{suf}(\gamma, \alpha)$. Cơ sở dữ liệu chiếu (PDB) của α được định nghĩa là

$$\mathcal{D}_\alpha \stackrel{\text{def}}{=} \{\text{suf}(\Psi, \alpha) | (\Psi \in \mathcal{D}) \wedge (\Psi \sqsupseteq \alpha)\}.$$

Ta nói $\mathcal{D}_\beta$ chứa trong $\mathcal{D}_\alpha$, ký hiệu là $\mathcal{D}_\beta \sqsubseteq D_\alpha$, nếu $\rho(\beta) \subseteq \rho(\alpha)$ và với mọi $\Psi \in \rho(\beta)$ ta có $\text{suf}(\Psi, \beta) \sqsubseteq \text{suf}(\Psi, \alpha)$. Khi $\mathcal{D}_\beta \sqsubseteq \mathcal{D}_\alpha$ và $\mathcal{D}_\alpha \sqsubseteq \mathcal{D}_\beta$, ta nói rằng $\mathcal{D}_\beta$ bằng $\mathcal{D}_\alpha$ và ký hiệu $\mathcal{D}_\beta = \mathcal{D}_\alpha$. Dễ thấy rằng $\text{supp}(\alpha) = |\mathcal{D}_\alpha|$.

Ví dụ, với $\alpha = e \to e \sqsubset \beta = e \to ce$, ta có $\rho(\alpha) = \rho(\beta) = \{\Psi_1, \Psi_2\}$, $\text{suf}(\Psi_1, \alpha) = _g \to ace$ và $\text{suf}(\Psi_2, \alpha) = <>$. Do đó, PDB của α là $\mathcal{D}_\alpha = \{_g \to ace, <>\}$. Tương tự, ta có $\mathcal{D}_\beta = \mathcal{D}_\alpha$.

Không gian tìm kiếm chuỗi lời giải được biểu diễn bởi một cây tiền tố với gốc là chuỗi rỗng, mỗi nút biểu diễn một chuỗi ứng viên, mỗi nút con biểu diễn một chuỗi mở rộng. Ta ký hiệu $\text{branch}(\alpha)$ là nhánh có gốc tại nút biểu diễn α, nó chứa α và các mở rộng của nó. Với một tiền tố khác rỗng α, $\beta = \alpha \diamond y$, chuỗi $\gamma = \alpha \diamond \varepsilon \diamond y$ mà $\gamma \sqsupseteq \beta$ được gọi là một mở rộng lùi (BE) của β bởi ε, với y là thuộc

tính cuối của β, ký hiệu là lastItem(β). Chẳng hạn, với $\alpha = cd$ thì branch(α) chứa các chuỗi: $cd \rightarrow \varepsilon$, $cde \rightarrow \varepsilon$ và $cdeg \rightarrow \varepsilon$, với mọi ε, kể cả <>. Cho $\beta = ce$, các chuỗi cde và $c \rightarrow cg \rightarrow cde$ là các BE của β. Tuy nhiên, $ce \rightarrow cg \rightarrow e$ là BE của $c \rightarrow e$, nhưng không là BE của β.

Thách thức chính của *HUSM* là $u_{\min}$ không đơn điệu tăng cũng không đơn điệu giảm, tức là

$$\exists \beta, \alpha, \gamma, \delta : (\beta \sqsupset \alpha) \wedge (\gamma \sqsubset \delta) \\ (u_{\min}(\beta) > u_{\min}(\alpha)) \wedge (u_{\min}(\gamma) > u_{\min}(\delta)).$$

Nói cách khác, $u_{\min}$ không thỏa mãn DCP (tính chất của supp được sử dụng trong *FSM*). Thật vậy, với $\alpha = cd$, $\beta = cdg$, và $\delta = c \rightarrow c \rightarrow cd$, dễ thấy rằng $\delta \sqsupset \alpha \sqsubset \beta$ và $u_{\min}(\beta) = 84 > u_{\min}(\alpha) = 54 > u_{\min}(\delta) = 27$. Để khắc phục, các chặn trên $u_{\min}$ thỏa mãn các tính chất tựa đơn điệu giảm (có thể yếu hơn DCP) được đề xuất trong mục tiếp theo.

2. Hai chiến lược tỉa các chuỗi LU và LF theo chiều sâu và chiều rộng

Giả sử $\alpha' \sqsubseteq \Psi'$, $\alpha = \text{proj}(\alpha') \sqsubseteq \Psi'$, với $\Psi' = B'_1 \rightarrow B'_2 \rightarrow \cdots \rightarrow B'_q \in \mathcal{D}'$, tức là tồn tại p số nguyên, $1 \le i_1 < i_2 < \cdots < i_p \le q$ sao cho $A'_k \sqsubseteq B'_{i_k}$ và $A_k = \text{proj}(A'_k) \subseteq \text{proj}(B'_{i_k})$, với mọi $k = 1, 2, \ldots, p$. Chỉ số cuối i_p được gọi là điểm cuối của α (hay α') trong Ψ', ký hiệu là end(α, Ψ') (hay end(α', Ψ')). Thuộc tính cuối của α trong B'_{i_p} được gọi là thuộc tính cuối ứng với i_p, ký hiệu là e_{i_p}. Khi đó, q-chuỗi còn lại của α trong Ψ' (đối với i_p) là phần còn lại của Ψ' sau e_{i_p}, ký hiệu là rem(α, Ψ', i_p). Gọi $i^*_p \stackrel{\text{def}}{=} \text{FEnd}(\alpha, \Psi')$ là điểm cuối đầu tiên của α trong Ψ'. Cơ sở dữ liệu chiếu lượng hóa (PQDB) $\mathcal{D}'_\alpha$ của α chứa tất cả các chuỗi còn lại rem(α, Ψ', i^*_p), với $\Psi' \in \mathcal{D}'$. Nếu $\alpha =<>$, ta quy ước $i^*_p \stackrel{\text{def}}{=} \text{FEnd}(<>, \Psi') \stackrel{\text{def}}{=} 0$, $\mathcal{D}'_{<>} \stackrel{\text{def}}{=} \mathcal{D}'$ và rem($<>, \Psi', i^*_p$) = Ψ'. Với mỗi điểm cuối $i_p = \text{end}(\alpha, \Psi')$, ta định nghĩa

$$u(\alpha, \Psi', i_p) \stackrel{\text{def}}{=} \min\{u(\alpha') \mid \alpha' \in O(\alpha, \Psi') \wedge \text{end}(\alpha', \Psi') = i_p\}.$$

Chặn trên dựa vào lợi ích còn lại của $u_{\min}(\alpha, \Psi')$ được định nghĩa và ký hiệu:

$$ub_{\text{rem}}(\alpha, \Psi') \stackrel{\text{def}}{=} \begin{cases} u(\alpha, \Psi', i^*_p) + u(\text{rem}(\alpha, \Psi', i^*_p)), & \alpha \neq <>, \\ u(\Psi'), & \alpha =<>. \end{cases}$$

Ví dụ, xét $\beta = c \rightarrow d$. Do lần xuất hiện đầu tiên của β trong Ψ'_1 là q-chuỗi con $\beta' = (c, 2) \rightarrow (d, 16) \sqsubseteq \Psi'_1$, nên

$$i^*_p \stackrel{\text{def}}{=} \text{FEnd}(\beta, \Psi'_1) = 4, \\ \text{rem}(\beta, \Psi'_1, i^*_p) = (e, 35)(g, 60) \rightarrow (a, 3)(c, 5)(e, 20), \\ u(\text{rem}(\beta, \Psi'_1, i^*_p)) = 123, \\ u(\beta, \Psi'_1, i^*_p) = \min\{u(\beta'), u((c, 1) \rightarrow (d, 16))\} = 17.$$

Vì vậy

$$ub_{\text{rem}}(\beta, \Psi'_1) = 17 + 123 = 140.$$

Một độ đo ub được gọi là một chặn trên của $u_{\min}$ nếu $u_{\min}(\alpha) \le ub(\alpha)$, với mọi α. Ngoài chặn trên truyền thống $\text{SWU}(\alpha) \stackrel{\text{def}}{=} \sum_{\Psi' \in \rho(\alpha)} u(\Psi')$ [5], ta còn có các chặn trên chặt hơn sau đây.

Định nghĩa 7 ([7]): Xét $y \in \mathcal{A}$ và α khác rỗng. Ta định nghĩa các chặn trên sau đây:

$$\text{RBU}(\alpha) \stackrel{\text{def}}{=} \sum_{\Psi' \in \rho(\alpha)} ub_{\text{rem}}(\alpha, \Psi'), \\ \text{LRU}(\alpha \diamond y) \stackrel{\text{def}}{=} \sum_{\Psi' \in \rho(\alpha \diamond y)} ub_{\text{rem}}(\alpha, \Psi'), \\ \text{LRU}(y) \stackrel{\text{def}}{=} \text{SWU}(y).$$

Với hai chặn trên ub_1 và ub_2, ub_1 được gọi là chặt hơn ub_2, ký hiệu là $ub_1 \ll ub_2$, nếu $ub_1(\alpha) \le ub_2(\alpha)$ với mọi α. Theo [7, Định lý 1], ta có

$$u_{\min} \ll \text{RBU} \ll \text{LRU} \ll \text{SWU}.$$

Ví dụ, xét chuỗi $\beta = c \rightarrow cd$, $\rho(\beta) = \{\Psi'_1, \Psi'_2\}$. Khi đó

$$u_{\min}(\beta, \Psi'_1) = 25, \quad u_{\min}(\beta, \Psi'_2) = 39, \\ \text{FEnd}(\beta, \Psi'_1) = 4, \quad u(\beta, \Psi'_1, 4) = 25, \\ u(\text{rem}(\beta, \Psi'_1, 4)) = 123.$$

Vì vậy

$$u_{\min}(\beta) = u_{\min}(\beta, \Psi'_1) + u_{\min}(\beta, \Psi'_2) = 64, \\ ub_{\text{rem}}(\beta, \Psi'_1) = 150, ub_{\text{rem}}(\beta, \Psi'_2) = 64, \\ \text{RBU}(\beta) = ub_{\text{rem}}(\beta, \Psi'_1) + ub_{\text{rem}}(\beta, \Psi'_2) = 212.$$

Ngoài ra, với $\alpha = c \rightarrow c$, ta dễ thấy rằng $\beta = \alpha \diamond_i d$ và

$$\text{LRU}(\beta) = UB_{\text{rem}}(\alpha, \Psi'_1) + ub_{\text{rem}}(\alpha, \Psi'_2) \\ = 200 + 165 = 365, \\ \text{SWU}(\beta) = u(\Psi'_1) + u(\Psi'_2) = 229 + 208 = 437.$$

Vì vậy ta có

$$u_{\min}(\beta) < \text{RBU}(\beta) < \text{LRU}(\beta) < \text{SWU}(\beta).$$

Để loại nhanh các chuỗi có lợi ích thấp hoặc ít phổ biến, dựa vào [7, Định lý 2], ta thiết kế hai chiến lược *DPS* và *WPS* nhằm thu hẹp hiệu quả không gian tìm kiếm. Trước hết, ta có chiến lược tỉa theo chiều sâu dựa vào RBU, viết là *DPS*(RBU), như sau: "*Nếu* RBU(α) < *mu thì* branch(α) *có thể được tỉa*".

Gọi hai tập thuộc tính ứng viên cho các $i-$ và s-mở rộng lần lượt là:

$$I_{\text{LRU}}(\alpha) \stackrel{\text{def}}{=} \{z \mid (\text{LRU}(\alpha \diamond_i z) \ge mu) \wedge (\text{supp}(\alpha \diamond_i z) \ge ms)\}, \\ S_{\text{LRU}}(\alpha) \stackrel{\text{def}}{=} \{z \mid (\text{LRU}(\alpha \diamond_s z) \ge mu) \wedge (\text{supp}(\alpha \diamond_s z) \ge ms)\}.$$

Nếu $u_{\min}(\alpha \diamond_i z) \geq mu$ và $\text{supp}(\alpha \diamond_i z) \geq ms$, thì

$$\text{LRU}(\alpha \diamond_i z) \geq u_{\min}(\alpha \diamond_i z) \geq mu,$$

tức là $z \in I_{\text{LRU}}(\alpha)$. Vì $\text{LRU}(\alpha \diamond_i y \diamond_i z) \leq \text{LRU}(\alpha \diamond_i z)$,

$$\max\{\text{LRU}(\gamma) \mid \gamma \in \{\alpha \diamond_i y \diamond_s z, \alpha \diamond_s y \diamond_i z, \alpha \diamond_s y \diamond_s z\}\} \leq \text{LRU}(\alpha \diamond_s z),$$

$$\text{supp}(\alpha \diamond y \diamond z) \leq \text{supp}(\alpha \diamond z), \quad \text{với } \alpha \diamond y \diamond z \sqsupseteq \alpha \diamond z.$$

Cho nên ta có

$$I_{\text{LRU}}(\alpha \diamond_i y) \subseteq I_{\text{LRU}}(\alpha),$$
$$(S_{\text{LRU}}(\alpha \diamond_i y) \cup S_{\text{LRU}}(\alpha \diamond_i y)) \subseteq S_{\text{LRU}}(\alpha).$$

Từ đó, ta có chiến lược tỉa theo chiều rộng dựa vào LRU, viết là *WPS*(LRU), như sau: "*Nếu* $\text{LRU}(\alpha \diamond y) < mu$, *ta không cần xét tất cả các mở rộng tiến của* $\alpha \diamond y$, *tức là* $\text{branch}(\alpha \diamond y)$, *và các mở rộng lùi của nó.*

Lấy ví dụ, với $mu = 230$ và $ms = 2$, ta có $\text{supp}(b) < ms$ và $\text{LRU}(b) = 174 < mu$. Các giá trị LRU của các thuộc tính còn lại, trong tập chứa các thuộc tính ứng viên cho các mở rộng, $I_{\text{LRU}}(<>) = S_{\text{LRU}}(<>) = \{a, c, d, e, f, g\}$, đều lớn hơn hay bằng mu (chẳng hạn $\text{LRU}(a) = 447$). Khi đó, do *WPS*(LRU), ta có thể loại thuộc tính b khỏi $\mathcal{D}'$.

Để minh họa tác dụng tỉa theo chiều sâu của RBU, xét hai giá trị $\text{RBU}(ac) = 199$ và $\text{RBU}(a \to c) = 229$ đều bé hơn mu. Sử dụng *DPS*(RBU), toàn bộ nhánh $\text{branch}(ac)$ và $\text{branch}(a \to c)$ bị tỉa. Dễ thấy rằng, sử dụng supp, $I_{\text{LRU}}(a) = \{c, e\}$ và $S_{\text{LRU}}(a) = \{a, c, d, e, g\}(\subseteq S_{\text{LRU}}(<>))$. Vì $S_{\text{LRU}}(a \to c) \subseteq S_{\text{LRU}}(a)$ và với bất kỳ $x \in S_{\text{LRU}}(a)$, $\text{RBU}(a \to c \to x) < \text{LRU}(a \to c \to x) = 229 < mu$ hay $\text{supp}(a \to c \to x) = 1 < ms$, $S_{\text{LRU}}(a \to c) = \emptyset$. Do đó, bởi *DPS*(RBU), ta chỉ có thể tỉa nhánh $\text{branch}(a \to c \to x)$, với mọi $x \in \{a, c, d, e, g\}$.

Tuy nhiên, sử dụng *WPS*(LRU), ta vẫn có thể loại bỏ tất cả các nhánh mở rộng lùi của $\text{branch}(a \to c \to x)$, ví dụ như $\text{branch}(a \to c \to g \to d)$, $\text{branch}(a \to ce \to cg \to cd)$ (của $\text{branch}(a \to c \to d)$), v.v. Thật vậy, lý do là $I_{\text{LRU}}(a \to ce \to cg \to c) \subseteq S_{\text{LRU}}(a \to ce) \subseteq S_{\text{LRU}}(a \to c) = \emptyset$. Ngoài ra, mặc dù chặt hơn LRU, nhưng RBU không có tác dụng tỉa theo chiều rộng. Nếu dùng RBU để tỉa theo chiều rộng, ta có thể tỉa nhầm một số lời giải. Thật vậy, với $mu = 225$ và $ms = 2$, vì $I_{\text{RBU}}(a \to g \to e \to c)(\subseteq S_{\text{RBU}}(a \to g \to e) = \{c\})$, nên $I_{\text{RBU}}(a \to g \to e \to c) = \emptyset$. Tuy nhiên, $a \to g \to e \to ce_2^{225} \in FHUS$. Có thể kết luận rằng, mặc dù LRU lỏng hơn RBU, tác dụng tỉa của *WPS* thật sự mạnh hơn *DPS*.

Vì vậy, *DPS* và *WPS* được dùng để tỉa các nhánh LU (và LF) trên cây tìm kiếm tiền tố, có hiệu quả với các giá trị mu cao [7]. Tuy nhiên, nhiều chuỗi HU có thể không là các chuỗi sinh HU (non-GHU). Do đó, trong phần tiếp theo, chiến lược *LPG* tỉa các ứng viên non-GHU sẽ được đề xuất. Để ý rằng, *LPG* sẽ hiệu quả với các giá trị mu thấp.

III. CHIẾN LƯỢC LPG TỈA SỚM CÁC CHUỖI KHÔNG LÀ SINH LỢI ÍCH CAO

1. Điều kiện tỉa sớm tổng quát

Trước hết, chúng tôi nhắc lại hai độ đo SE, SLIP và điều kiện tỉa sớm tổng quát đã được dùng để tỉa các chuỗi phổ biến đóng và các chuỗi phổ biến đóng có lợi ích cao [13], hoặc các chuỗi sinh phổ biến [12].

Định nghĩa 8 ([12]): Tổng các sự kiện còn lại trong một PDB $\mathcal{D}_\alpha$ được định nghĩa và ký hiệu là:

$$\text{SE}(\alpha) \stackrel{\text{def}}{=} \sum_{\substack{\Psi \in \mathcal{D}: \\ \text{suf}(\Psi, \alpha) \in \mathcal{D}_\alpha}} \big(\text{size}(\Psi) - \text{size}(\text{pref}(\Psi, \alpha)) + 1\big).$$

Với chuỗi α, gọi δ là tiền tố bé nhất của Ψ chứa α, $\Psi \in \rho(\alpha)$. Đặt $fi(\alpha, \Psi)$ là chỉ số (trong Ψ) của sự kiện cuối của δ và $\text{lastEvent}(\alpha)$ là sự kiện cuối của α. Gọi $\text{LP}(\alpha, \Psi)$ là danh sách các vị trí thứ i khác nhau trong Ψ mà $\text{lastEvent}(\alpha) \subseteq \Psi[i]$ và $fi(\alpha, \Psi) \leq i \leq \text{size}(\Psi)$. Không mất tổng quát, giả sử rằng các chỉ số trong $\text{LP}(\alpha, \Psi)$ được sắp tăng. Khi đó,

$$\text{slip}(\alpha, \Psi) \stackrel{\text{def}}{=} |\text{LP}(\alpha, \Psi)|$$

là số các vị trí xuất hiện của sự kiện cuối của δ trong Ψ.

Định nghĩa 9 ([13]): Số đo SLIP của PDB $\mathcal{D}_\alpha$ được định nghĩa và ký hiệu là

$$\text{SLIP}(\alpha) \stackrel{\text{def}}{=} \sum_{\Psi \in \rho(\alpha)} \text{slip}(\alpha, \Psi).$$

Ví dụ, xét $\alpha = e \to e$. Vì $\mathcal{D}_\alpha = \{_g \to ace, <>\}$, nên $\text{SE}(\alpha) = 3$. Ta có $\rho(\alpha) = \{\Psi_1, \Psi_2\}$. Vì sự kiện cuối e xuất hiện hai lần trong Ψ_1 tại vị trí thứ 4 và thứ 5, nên $\text{LP}(\alpha, \Psi_1) = \{4, 5\}$. Tương tự, $\text{LP}(\alpha, \Psi_2) = \{5\}$. Vì vậy, $\text{SLIP}(\alpha) = 3$. Từ SE và SLIP, ta có điều kiện tỉa sớm tổng quát (GEP) trong Định lý 1 sau đây.

Định lý 1 (Điều kiện tỉa sớm tổng quát [13]): Xét hai chuỗi α và β thỏa mãn $\alpha \sqsubseteq \beta$. Khi đó:

a) Nếu $\text{SE}(\alpha) = \text{SE}(\beta)$, thì $\text{supp}(\alpha) = \text{supp}(\beta)$ và $\mathcal{D}_\gamma = \mathcal{D}_\lambda$ với mọi s-mở rộng γ của α và λ của β với cùng một chuỗi khác rỗng;

b) Nếu $\text{SE}(\alpha) = \text{SE}(\beta)$ và $\text{SLIP}(\alpha) = \text{SLIP}(\beta)$, thì $\text{supp}(\alpha) = \text{supp}(\beta)$ và $\mathcal{D}_\gamma = \mathcal{D}_\lambda$ với tất cả mở rộng γ của α và λ của β với cùng một chuỗi khác rỗng.

2. Chiến lược LPG

Dựa vào GEP và chặn dưới SF sau đây của $u_{\min}$, chiến lược *LPG* tỉa sớm các chuỗi non-GHU được đề xuất.

Định nghĩa 10: Cho chuỗi phổ biến α, tức là $\text{supp}(\alpha) \geq k$, với $k \stackrel{\text{def}}{=} ms$. Sau khi sắp xếp tăng dần dãy

$$\mathcal{U}(\alpha) \stackrel{\text{def}}{=} \{u_{\min}(\alpha, \Psi') \mid \Psi' \in \rho(\alpha)\},$$

ta thu được dãy $\{u_i, 1 \le i \le n\}$ với $n \ge k$. Chặn dưới SF của $u_{\min}$ được định nghĩa là tổng k giá trị bé nhất của $\mathcal{U}(\alpha)$,

$$\mathrm{SF}(\alpha) \stackrel{\text{def}}{=} \sum_{1 \le i \le k} u_i.$$

Định lý 2: Ta có:

(a) SF là một chặn dưới của $u_{\min}$, tức là $\mathrm{SF}(\alpha) \le u_{\min}(\alpha)$, với mọi α;

(b) SF là đơn điệu tăng đối với các mở rộng, tức là $\mathrm{SF}(\beta) \ge \mathrm{SF}(\alpha)$, với mọi $\beta = \alpha \diamond \delta \sqsupseteq \alpha$.

Chứng minh: (a) Với mọi α, ta có $\mathrm{SF}(\alpha) = \sum_{i=1}^{k} u_i \le \sum_{i=1}^{n} u_i = u_{\min}(\alpha)$.

(b) Với mở rộng bất kỳ $\beta = \alpha \diamond \delta \sqsupseteq \alpha$, ta có $\rho(\beta) \subseteq \rho(\alpha)$ và $u_{\min}(\beta, \Psi') \ge u_{\min}(\alpha, \Psi')$, với mọi $\Psi' \in \rho(\beta)$. Thật vậy $u_{\min}(\beta, \Psi') \stackrel{\text{def}}{=} \min\{u(\beta') \mid \beta' \in O(\beta, \Psi')\} = u(\beta'_{\min})$, $\exists \alpha'_* \in O(\alpha, \Psi')$, $\varepsilon'_* \sqsubseteq \beta'_{\min}$ sao cho $\alpha \sqsubseteq \beta'_{\min} = \alpha'_* \diamond \varepsilon'_* \in O(\beta, \Psi')$ và $u(\beta'_{\min}) = u(\alpha'_*) + u(\varepsilon'_*) \ge u(\alpha'_*) \ge \min\{u(\alpha') \mid \alpha' \in O(\alpha, \Psi')\} \stackrel{\text{def}}{=} u_{\min}(\alpha, \Psi')$. Sau khi sắp tăng $\mathcal{U}(\alpha)$ và $\mathcal{U}(\beta)$, ta có được hai dãy $\{u_i, 1 \le i \le n\}$, $\{u'_j, 1 \le j \le m\}$ với $n \ge m \ge k$. Khi đó, tồn tại các số nguyên $1 \le i_1 < i_2 < \cdots < i_m$ sao cho $u_{i_j} \le u'_j$, với $1 \le j \le m$, nên ta có

$$\mathrm{SF}(\alpha) = \sum_{i=1}^{k} u_i \le \sum_{j=1}^{k} u_{i_j} \le \sum_{j=1}^{k} u'_j = \mathrm{SF}(\beta).$$

■

Ví dụ, khi xét hai ngưỡng $mu = 85$, $ms = 2$ và mở rộng $\beta = e \to e$ của $\alpha = e$, ta có $\rho(\beta) = \{\Psi'_1, \Psi'_2\} \subset \rho(\alpha) = \mathcal{D}'$, $u_{\min}(\alpha) = 120$, $u_{\min}(\beta) = 85$, và $\mathcal{U}(\alpha) = \{20; 20; 40; 40\}$, $\mathcal{U}(\beta) = \{40; 45\}$, nên $\mathrm{SF}(\alpha) = 20 + 20 = 40$ và $\mathrm{SF}(\beta) = 40 + 45 = 85$. Khi đó, $\mathrm{SF}(\alpha) < u_{\min}(\alpha)$, $\mathrm{SF}(\beta) \le u_{\min}(\beta)$, và $\mathrm{SF}(\alpha) < \mathrm{SF}(\beta)$.

Trên cây tiền tố với gốc *Root* =<>, với mỗi nút q (1-thuộc tính), ta có chuỗi tương ứng $\alpha = \mathrm{path}(q)$, trong đó $\mathrm{path}(q)$ là chuỗi đầy đủ thu được bằng cách duyệt từ *Root* đến q. Nút q có i-mở rộng và s-mở rộng bởi hai thuộc tính u và v tương ứng, ký hiệu là, $\alpha \diamond_i u$ và $\alpha \diamond_s v$. Khi đó, ta gán $u.type$ = *i-ext* và $v.type$ = *s-ext*. Ký hiệu tập chứa α và tất cả các mở rộng (hay các s-mở rộng, các i-mở rộng) cũng như tất cả các hậu duệ của các mở rộng này (hay các s-mở rộng và các i-mở rộng) là $\mathrm{branch}(\alpha)$ (hay tương ứng là *s-child branches*(α) và *i-child branches*(α)). Nếu tất cả các chuỗi trong các nhánh này không là chuỗi sinh HU, ta ký hiệu chúng là *non-GHU branch*(α) (hay tương ứng là *non-GHU s-child branches*(α) và *non-GHU i-child branches*(α)).

Định lý 3 (Chiến lược *LPG*): Xét q, u, v là các nút có cùng tiền tố và $\alpha = \mathrm{path}(q)$ là chuỗi ứng với q.

(a) Với mỗi i-mở rộng, $i_{\text{new}} = \alpha \diamond_i u$:

(i) Nếu $\mathrm{SE}(\alpha) = \mathrm{SE}(i_{\text{new}})$ và $\mathrm{SF}(\alpha) \ge mu$, thì *non-GHU s-child branches*(i_{new}) có thể được tỉa. Hơn nữa, nếu $\mathrm{SLIP}(\alpha) = \mathrm{SLIP}(i_{\text{new}})$, thì toàn bộ nhánh *non-GHU branch*(i_{new}) cũng được tỉa;

(ii) Nếu $\mathrm{SE}(\mathrm{path}(u)) = \mathrm{SE}(i_{\text{new}})$ và $\mathrm{SF}(\mathrm{path}(u)) \ge mu$, thì *non-GHU s-child branches*(i_{new}) được tỉa. Ngoài ra, nếu $\mathrm{SLIP}(\mathrm{path}(u)) = \mathrm{SLIP}(i_{\text{new}})$, thì *non-GHU branch*$(i_{\text{new}})$ được tỉa;

(iii) Nếu $q.type$ = *s-ext*, $q.Parent.Item$ = $q.Item$ (tồn tại $r \in q.Parent.iChildren$ sao cho $r.Item$ = $u.Item$), $q.Parent.type$ = *s-ext*, $\mathrm{SE}(\mathrm{path}(r)) = \mathrm{SE}(i_{\text{new}})$ và $\mathrm{SF}(\mathrm{path}(r)) \ge mu$, thì ta có thể tỉa *non-GHU branch*(i_{new}).

(b) Với mỗi s-mở rộng, $s_{\text{new}} = \alpha \diamond_s v$ sao cho $v.type$ = *s-ext*, nếu $\mathrm{SE}(\mathrm{path}(v)) = \mathrm{SE}(s_{\text{new}})$ và $\mathrm{SF}(\mathrm{path}(v)) \ge mu$, thì *non-GHU branch*$(s_{\text{new}})$ có thể được tỉa.

Chứng minh: Ta sẽ chứng minh a(i). Các khẳng định còn lại có thể được chứng minh tương tự. Giả sử rằng $\mathrm{SE}(\alpha) = \mathrm{SE}(i_{\text{new}})$ và $\mathrm{SF}(\alpha) \ge mu$. Vì $\alpha \sqsubset i_{\text{new}}$, với mọi s-mở rộng γ và λ của α và i_{new} tương ứng với cùng một chuỗi ε (kể cả chuỗi rỗng), $\gamma = \alpha \diamond_s \varepsilon$ và $\lambda = i_{\text{new}} \diamond_s \varepsilon$, thì $\gamma \sqsubset \lambda$ và $\mathrm{supp}(\gamma) = \mathrm{supp}(\lambda)$, vì do phần (a) của định lý 1, với ε khác rỗng, $\mathcal{D}_\gamma = \mathcal{D}_\lambda$, nên $|\mathcal{D}_\gamma| = |\mathcal{D}_\lambda|$. Hơn nữa, nếu $\mathrm{SLIP}(\alpha) = \mathrm{SLIP}(i_{\text{new}})$, do phần (b) của định lý 1, ta cũng có $\gamma \sqsubset \lambda$ và $\mathrm{supp}(\gamma) = \mathrm{supp}(\lambda)$, với bất kỳ i-mở rộng $\lambda = i_{\text{new}} \diamond_i \varepsilon$ của i_{new} và i-mở rộng tương ứng $\gamma = \alpha \diamond_i \varepsilon$ của α với cùng ε. Vì vậy, với bất kỳ mở rộng $\lambda = i_{\text{new}} \diamond \varepsilon$ của i_{new} và $\gamma = \alpha \diamond \varepsilon$ của α, ta luôn có $\gamma \sqsubset \lambda$ và $\mathrm{supp}(\gamma) = \mathrm{supp}(\lambda)$. Ngoài ra, do tính đơn điệu tăng của SF đối với phép toán mở rộng (phần (b) của định lý 2), nên $u_{\min}(\gamma) \ge \mathrm{SF}(\gamma) \ge \mathrm{SF}(\alpha) \ge mu$, nghĩa là $\exists \gamma \in HUS$: $\gamma \sqsubset \lambda$ và $\mathrm{supp}(\gamma) = \mathrm{supp}(\lambda)$. Vì vậy, $\lambda \notin GHUS$. ■

Chiến lược *LPG* cho phép tỉa sớm các chuỗi non-GHU ngay khi chúng vừa được tạo ra trong hai mức liền kề nhau trên cây tiền tố. Ngoài ra, ta không cần kiểm tra quan hệ cha con trên các chuỗi sử dụng trong định lý 3, chẳng hạn, $\alpha \sqsubset i_{\text{new}} = \alpha \diamond_i u$ trong a(i). Do đó, nó giúp rút ngắn đáng kể thời gian thực thi cũng như giảm lượng bộ nhớ lưu trữ trong quá trình khai thác. Sau đây, ta sẽ minh họa các trường hợp áp dụng của định lý 3.

3. Minh họa chiến lược *LPG*

Trước hết, ta minh họa trường hợp a(i) của định lý 3. Xét hai ngưỡng $mu = 85$, $ms = 2$ và $\alpha = a \to dg \to a \sqsubseteq i_{\text{new}} = \alpha \diamond_i e = a \to dg \to ae$. Ta có $\mathrm{SE}(i_{\text{new}}) = \mathrm{SE}(\alpha) = 3$, $\mathrm{SF}(\alpha) = 174 \ge mu$, và $\mathrm{SLIP}(i_{\text{new}}) = \mathrm{SLIP}(\alpha) = 2$. Theo phần a(i) của định lý 3, toàn bộ nhánh *non-GHU branch*(i_{new}) được tỉa.

Bây giờ, ta minh họa trường hợp a(ii). Đặt $mu = 75$ và $ms = 2$. Cho trước hai nút $q = a$ và $u = e$ với cùng tiền tố $a \to d$, ta có $\alpha \stackrel{\text{def}}{=} \mathrm{path}(q) = a \to d \to a$ và $\gamma \stackrel{\text{def}}{=} \mathrm{path}(u) = a \to d \to e$. Khi đó, $i_{\text{new}} = a \to d \to ae \sqsupseteq \gamma$, $\rho(i_{\text{new}}) = \rho(\gamma) = \{\Psi_1, \Psi_2\}$. Ta có $u_{\min}(i_{\text{new}}) = 96 \ge mu$

Thuật toán 1: Thuật toán FGenHUSM

Dữ liệu vào: $\mathcal{D}'$, ms, mu
Dữ liệu ra: $FGHUS$

```
1  FGHUS ← ∅;
2  S ← I ← {i ∈ 𝒜 | (LRU(i) ≥ mu) ∧ supp(i) ≥ ms)};
3  Loại các thuộc tính không thuộc S khỏi 𝒟′;
4  for i ∈ S do
5  |  DfsFGHUS(i, S, I, ms, mu);
6  end
```

và $\text{supp}(i_{\text{new}}) = 2 \geq ms$, nên i_{new} là chuỗi FHU. Tuy nhiên, i_{new} và các s-mở rộng của nó không là các chuỗi FGHU. Thật vậy, ta có $\text{SF}(\gamma) = 78 \geq mu$ và $\text{SE}(i_{\text{new}}) = \text{SE}(\gamma) = 3$. Tuy nhiên, mặc dù $\text{LP}(i_{\text{new}}, \Psi_1) = \text{LP}(\gamma, \Psi_1) = 5$, nhưng do $\text{LP}(i_{\text{new}}, \Psi_2) = 5 \neq \text{LP}(\gamma, \Psi_2) = \{4, 5\}$, cho nên $\text{SLIP}(i_{\text{new}}) = 2 \neq \text{SLIP}(\gamma) = 3$. Theo phần a(ii) của định lý 3, các nhánh *non-GHU s-child branches*(i_{new}) được tỉa.

Để minh họa trường hợp (b), xét hai nút $q = g$ và $v = e$ với cùng tiền tố $a \rightarrow d$, ta có $\alpha \overset{\text{def}}{=} \text{path}(q) = a \rightarrow dg$, $\gamma \overset{\text{def}}{=} \text{path}(v) = a \rightarrow d \rightarrow e$ và $v.type = s\text{-}ext$. Khi đó, $s_{\text{new}} = \alpha \diamond_s e = a \rightarrow dg \rightarrow e \sqsupseteq \gamma$, $\text{SE}(s_{\text{new}}) = \text{SE}(\gamma) = 3$ và $\text{SF}(\gamma) = 78 \geq mu$. Vì $\text{lastEvent}(\gamma) = \text{lastEvent}(s_{\text{new}}) = e$, nên $\text{SLIP}(s_{\text{new}}) = \text{SLIP}(\gamma) = 3$. Theo phần (b) của định lý 3, ta có thể tỉa ngay nhánh *non-GHU branch*(s_{new}).

Bây giờ, đặt $mu = 65$, $ms = 2$ và xét hai nút $q = c$ và $(u \equiv) r = e$ ứng với các chuỗi $\alpha \overset{\text{def}}{=} \text{path}(q) = c \rightarrow c \rightarrow c$ và $\delta \overset{\text{def}}{=} \text{path}(r) = c \rightarrow ce$ có cùng chuỗi cha $parent = c \rightarrow c$ và $parent.type = q.type = s\text{-}ext$. Khi đó, ta có $i_{\text{new}} = \alpha \diamond_i u = c \rightarrow c \rightarrow ce \sqsupseteq \delta$, $\text{SE}(i_{\text{new}}) = \text{SE}(\delta) = 3$, $\text{SF}(\delta) = 66 \geq mu$ và $\text{SLIP}(i_{\text{new}}) = \text{SLIP}(\delta) = 3$. Vì vậy, theo phần a(iii) của định lý 3, toàn bộ nhánh *non-GHU branch*(i_{new}) có thể được tỉa.

Để ý rằng, điều kiện "$\text{SF}(\alpha) \geq mu$" là cần thiết cho việc tỉa. Thật vậy, với $mu = 99$ và $ms = 3$, hai nút $q = c$ và $v = d$ có cùng tiền tố <>, ta có $\alpha \overset{\text{def}}{=} \text{path}(q) = c$, $\gamma \overset{\text{def}}{=} \text{path}(v) = d$ và $v.type = s\text{-}ext$, $s_{\text{new}} = \alpha \diamond_s d = c \rightarrow d \sqsupseteq \gamma$ với $\text{SE}(s_{\text{new}}) = \text{SE}(\gamma) = 6$ và $\text{SLIP}(s_{\text{new}}) = \text{SLIP}(\gamma) = 3$. Do đó, nếu dùng phần (b) của định lý 3 mà bỏ qua phép kiểm tra "$\text{SF}(\gamma) = 88 \geq mu = 99$", nhánh $\text{branch}(s_{\text{new}})$ sẽ bị tỉa nhầm. Thật vậy, s_{new} là chuỗi FGHU, vì $u_{\min}(s_{\text{new}}) = 99 \geq mu$, $\text{supp}(s_{\text{new}}) = 3 \geq ms$ và không tồn tại $\gamma^* \in HUS$ mà $\gamma^* \sqsubset s_{\text{new}}$ và $\text{supp}(\gamma^*) = \text{supp}(s_{\text{new}})$. Thật vậy, với mọi $c, d \sqsubset s_{\text{new}}$, $u_{\min}(c) = 10 < u_{\min}(d) = 88 < mu$.

IV. THUẬT TOÁN VÀ THỬ NGHIỆM

1. Thuật toán FGenHUSM

Thuật toán *FGenHUSM* khai thác tập *FGHUS* được cho trong thuật toán 1. Đầu vào của nó là QSDB $\mathcal{D}'$, hai ngưỡng ms và mu; đầu ra là *FGHUS* (để tiện trình bày, ta xem nó như biến toàn cục đối với các thủ tục và hàm con). Ở các

Thuật toán 2: Thủ tục DfsFGHUS

Dữ liệu vào: γ, S, I, ms, mu
Dữ liệu ra: $FGHUS$

```
1  update ← UpdateFGHUS(γ, mu);
2  if update is true then
3  |  return;
4  end
5  S_new ← I_new ← ∅;
6  for i ∈ I mà i > lastItem(γ) do
7  |  if LRU(γ ⋄_i i) ≥ mu và supp(γ ⋄_i i) ≥ ms then
8  |  |  I_new ← I_new ∪ {i};
9  |  end
10 end
11 for i ∈ I_new mà RBU(γ ⋄_i i) ≥ mu do
12 |  i_new ← γ ⋄_i i ;
13 |  LocalPruningGHU(i_new, γ, mu);
14 |  β ← anh em của γ ứng với i, có cùng kiểu mở rộng
   |    với γ;
15 |  LocalPruningGHU(i_new, β, mu);
16 |  if γ.type = s-ext, γ.typeOfParent = s-ext,
   |    γ.lastItemOfParent = γ.lastItem và
   |    ∃δ ∈ i- anh em của γ mà δ.lastItem = i then
17 |  |  LocalPruningGHU(i_new, δ, mu);
18 |  end
19 end
20 if γ.do-s-ext hoặc γ.do-ext then
21 |  for i ∈ S do
22 |  |  if LRU(γ ⋄_s i) ≥ mu và supp(γ ⋄_s i) ≥ ms then
23 |  |  |  S_new ← S_new ∪ {i};
24 |  |  end
25 |  end
26 |  for i ∈ S_new mà RBU(γ ⋄_s i) ≥ mu do
27 |  |  s_new ← γ ⋄_s i ;
28 |  |  α ← anh em của γ ứng với i và có kiểu mở
   |  |    rộng s-ext;
29 |  |  LocalPruningGHU(s_new, α, mu);
30 |  end
31 |  for i ∈ S_new do
32 |  |  DfsFGHUS(γ ⋄_s i, S_new, S_new, ms, mu);
33 |  end
34 else
35 |  S_new ← S;
36 end
37 for i ∈ I_new do
38 |  DfsFGHUS(γ ⋄_i i, S_new, I_new, ms, mu);
39 end
```

dòng 2 và 3, *WPS* dựa vào LRU và supp được sử dụng. Ở dòng 5, nó gọi thủ tục *DfsFGHUS* (Thuật toán 2) tìm kiếm các chuỗi FGHU. Thủ tục này nhận vào chuỗi γ, hai tập thuộc tính S và I chứa các thuộc tính cho các s- và i-mở rộng tương ứng của γ. Trong thủ tục, tại các dòng 1–3, hàm *UpdateFGHUS* (Thuật toán 3) kiểm tra xem γ có là GHU không và cập nhật *FGHUS* khi cần. Tại các dòng 13, 15, 17, 20 và 29 trong *DfsFGHUS* và các dòng 4, 7 và 15 trong *UpdateFGHUS*, chiến lược *LPG* được sử dụng thông qua thủ tục *LocalPruningGHU* (Thuật toán 4).

Tính đúng của thuật toán được bảo đảm bởi định lý 3 trong bài báo này và định lý 2 trong [7]. Cụ thể, việc áp dụng hai chiến lược tỉa *WPS* và *DPS* nhằm loại nhanh các

Thuật toán 3: Hàm UpdateFGHUS

Dữ liệu vào: γ, mu
Dữ liệu ra: *true* nếu γ là GHU và *false* trong trường hợp ngược lại và cập nhật FGHUS khi cần

```
1  if RBU(γ) < mu then
2  |   return true;
3  end
4  if γ.do-ext is false then
5  |   return true;
6  end
7  if γ.do-s-ext is false then
8  |   return false;
9  end
10 if u_min(γ) < mu then
11 |   return false;
12 end
13 for α ∈ FGHUS do
14 |   if supp(α) = supp(γ) và γ ⊐ α then
15 |   |   LocalPruningGHU(γ, α);
16 |   |   if γ.do-ext is false then
17 |   |   |   return true;
18 |   |   end
19 |   |   return false;
20 |   end
21 end
22 for α ∈ FGHUS do
23 |   if supp(α) = supp(γ) và α ⊐ γ then
24 |   |   Loại α ra khỏi FGHUS;
25 |   end
26 end
27 FGHUS ← FGHUS ∪ {γ};
28 return false;
```

Thuật toán 4: Thủ tục LocalPruningGHU

Dữ liệu vào: *cha, con, mu*
Dữ liệu ra: *cha*

```
1 if SF(con) ≥ mu và SE(cha) = SE(con) then
2 |   cha.do-s-ext ← false;
3 |   if SLIP(cha) = SLIP(con) then
4 |   |   cha.do-ext ← false;
5 |   end
6 end
```

nhánh chỉ chứa các chuỗi có lợi ích thấp hoặc ít phổ biến, và mục đích của chiến lược *LPG* là tỉa sớm các nhánh chỉ chứa các chuỗi không là chuỗi sinh lợi ích cao. Vì vậy, việc dùng ba chiến lược trên vào thuật toán *FGenHUSM* không làm mất đi bất kỳ chuỗi sinh phổ biến lợi ích cao nào. Chú ý rằng, khác với chiến lược *LPC* trong [13] nhằm tỉa sớm các chuỗi không là chuỗi đóng phổ biến lợi ích cao, chiến lược *LPG* cần sử dụng thêm chặn dưới SF của độ đo $u_{\min}$. Việc áp dụng *LPG* mà không dùng SF có thể dẫn đến việc tỉa nhầm một số nhánh như được minh họa trong mục III.3.

2. Minh họa thuật toán

Trong ví dụ này, chúng tôi minh họa quá trình khai thác các chuỗi sinh phổ biến lợi ích cao trên QSDB cho trong Bảng II bằng thuật toán *FGenHUSM* với $mu = 200$ và $ms = 1$. Để tiện trình bày, với chuỗi α, chúng tôi viết thêm các giá trị $u_{\min}$, RBU, LRU, supp, SE, SLIP và SF($u_{\min}$) của nó ở dạng $\alpha^{u_{\min},\mathrm{RBU},\mathrm{LRU}}_{\mathrm{supp};\mathrm{SE},\mathrm{SLIP};\mathrm{SF}}$. Ký hiệu (_) cũng được dùng để dấu đi một hoặc một dãy các giá trị kề nhau.

Bảng II
QSDB $\mathcal{D}''$ MINH HỌA THUẬT TOÁN *FGenHUSM*

SID	Chuỗi
1	$\Psi'_1 = (a,2)(c,5)(e,6) \rightarrow (a,3)(b,6) \rightarrow (a,5)(d,50)$ $\rightarrow (a,5)(b,9)(c,40) \rightarrow (a,4)(c,10)(d,10)(f,36)$
2	$\Psi'_2 = (b,12) \rightarrow (a,2)(c,20)(e,6) \rightarrow (a,3)(d,20)$ $\rightarrow (a,1)(c,20)(d,10)(f,9) \rightarrow (a,4)(b,9)(c,15)$
3	$\Psi'_3 = (c,20) \rightarrow (a,4)(c,10)(e,4) \rightarrow (a,1)(f,18)$
4	$\Psi'_4 = (d,80) \rightarrow (a,7)(c,50)(e,6) \rightarrow (a,2)(g,2)$ $\rightarrow (a,9)(f,72)$

Tại dòng 2, ta có

$$S = I = \{a^{6,495,607}_{4;_}, b^{15,306,322}_{2;_}, c^{80,504,607}_{4;_}, d^{100,480,550}_{3;10,3;_}, e^{22,395,607}_{4;_}, f^{135,163,607}_{4;_}, g^{2,83,228}_{1;_}\}.$$

Trước hết, dễ thấy rằng, các lần gọi *DfsFGHUS* trên các nhánh có gốc tại f và g sẽ dừng ngay tại dòng 3 (vì $\mathrm{RBU}(f) = 163$ và $\mathrm{RBU}(g) = 83 < mu$ nên *UpdateFGHUS* trả về *true*). Nói cách khác, hai nhánh tại f và g được tỉa bằng *DPS*. Các chuỗi ứng với các nút a, b, c, d và e bị bỏ qua vì các giá trị $u_{\min}$ của chúng đều bé hơn mu, tuy nhiên, tìm kiếm trên chúng vẫn được tiếp tục. Sau khi kết thúc tìm kiếm trên a, b và c, ta có

$$\begin{aligned} FGHUS = \{&c \rightarrow f^{220}_4, ac \rightarrow a \rightarrow f^{211}_3, ce \rightarrow a \rightarrow f^{218}_3; \\ &ac \rightarrow ad \rightarrow c \rightarrow ac^{200}_2, ac \rightarrow ad \rightarrow cdf^{202}_2, \\ &ace \rightarrow d \rightarrow ac \rightarrow c^{202}_2, c \rightarrow ad \rightarrow ac \rightarrow ac^{202}_2, \\ &c \rightarrow ad \rightarrow acdf^{203}_2, ce \rightarrow ad \rightarrow c \rightarrow c^{200}_2, \\ &ec \rightarrow d \rightarrow c \rightarrow ac^{200}_2, ce \rightarrow d \rightarrow cdf^{200}_2\}, \end{aligned}$$

ta chỉ lưu lại các giá trị $u_{\min}$ (ở trên) và supp (ở dưới).

Bây giờ, ta xét quá trình khai thác trên nhánh $d^{100,480,550}_{3;_}$. Sau khi hàm *UpdateFGHUS* trả về giá trị *false*, thực hiện các dòng 6–10, ta có $I_{\mathrm{new}}(d) = \{f\}$. Các i–mở rộng da, db, dc và dd bị tỉa vì a, b, c và d không lớn hơn d. Hai i-mở rộng de và dg bị tỉa vì de, dg không xuất hiện trong QSDB $\mathcal{D}''$. Tương tự, ta có $S_{\mathrm{new}}(d) = \{a,b,c,d,e,f,g\}$. Không có gì xảy ra khi thực hiện các dòng 6–12. Lúc này, ta gọi *DfsFGHUS* cho các s–mở rộng của d với mỗi thuộc tính trong $S_{\mathrm{new}}(d)$ trước và sau đó cho i-mở rộng df của nó. Năm nhánh $d \rightarrow b^{_,193,252}_{_}$, $d \rightarrow d^{_,163,252}_{_}$, $d \rightarrow e^{_,171,228}_{_}$, $d \rightarrow g^{_,163,228}_{_}$ và $df^{_,93,252}_{_}$ bị tỉa vì các giá trị RBU của chúng bé hơn mu. Ta chỉ còn lại ba nhánh $d \rightarrow a^{150,480,480}_{3;7,3;14}$, $d \rightarrow c^{215,458,480}_{3;7,3;25}$ và $d \rightarrow f^{267,295,480}_{3;4,3;29}$ cần xét với $S_{\mathrm{new}}(d)$.

Xét việc thực thi *DfsFGHUS* với $d \rightarrow a^{_}_{_}$. Trước hết, sau các dòng 3–7, ta có: $I_{\mathrm{new}}(d \rightarrow a) = \{b,c,d,e,f,g\}$. Sau

đó, ở cuối dòng 23, ta có $S_{\text{new}}(d \to a) = \{a, c, f, g\}$. Thật vậy, hai nhánh $d \to a \to b$ và $d \to a \to d$ bị tỉa bởi *WPS* vì LRU của chúng lần lượt là 83 và 164 đều bé hơn mu; còn chuỗi $d \to a \to e$ không xuất hiện trong $\mathcal{D}''$. Thủ tục gọi đệ quy trên các nhánh $d \to a \to a$, $d \to a \to c$, $d \to a \to f$ và $d \to a \to g$ với $S_{\text{new}}(d \to a)$ tại dòng 32; và cho các nhánh $d \to ab$, $d \to ac$, $d \to ad$, $d \to ae$, $d \to af$ và $d \to ag$ với $S_{\text{new}}(d \to a)$ và $I_{\text{new}}(d \to a)$ tại dòng 38.

Trên nhánh $d \to a \to a$, ta đưa chuỗi $d \to a \to af^{258}_{_,2,_}$ vào *FGHUS* (ở dòng 28 trong *UpdateFGHUS*). Các nhánh $d \to a \to c$ và $d \to a \to g$ bị tỉa bởi *DPS*. Trên nhánh $d \to a \to f$, ta thay thế $d \to a \to af$ bởi $d \to a \to f$ trong *FGHUS* (các dòng 22–27 trong *UpdateFGHUS*) vì chúng có cùng độ hỗ trợ mà $d \to a \to f \sqsubset d \to a \to af$.

Vì các giá trị RBU của $d \to ad$, $d \to ae$ và $d \to ag$ bé hơn mu, ta xét $d \to ab$, $d \to af$ và $d \to ac$. Với $d \to ab$, không có thay đổi nào trên *FGHUS*. Với $d \to af$ và $d \to ac$, ta đẩy $d \to af_3^{281}$ và $d \to ac_3^{230}$ vào *FGHUS*. Khi đó, ta có $I_{\text{new}}(d \to ac) = \{d, e, f\}$ và $S_{\text{new}}(d \to ac) = \{a, c, f, g\}$.

Không có thay đổi nào trên *FGHUS* khi xét $d \to ac \to c$ và $d \to ac \to f$. Với $d \to ac \to a$, ta tìm thấy trong *FGHUS* chuỗi con $d \to ac_3^{230}$ có cùng độ hỗ trợ tại dòng 14 trong *UpdateFGHUS*. Tại dòng tiếp theo, ta thực hiện thủ tục *LocalPruningGHU* với $d \to ac \to a$ và $d \to ac$. Mặc dù $\text{SE}(\mathcal{D}''_{d \to ac \to a}) = \text{SE}(\mathcal{D}''_{d \to ac}) = 7$, chiến lược tỉa *LPG* không thể được áp dụng vì

$$\text{SF}(u_{\min}(d \to ac)) = 29 < mu.$$

Tiếp tục, ta có

$$I_{\text{new}}(d \to ac \to a) = \{c, f, g\},$$
$$S_{\text{new}}(d \to ac \to a) = \{a, f\}.$$

Với s-mở rộng $d \to ac \to a \to a$, ta có $I_{\text{new}}(d \to ac \to a \to a) = \{f\}$. Vì $\text{RBU}(d \to ac \to a \to af) \geq mu$, tại dòng 13, ta gọi *LocalPruningGHU* (trường hợp (i)) với $i_{\text{new}} = d \to ac \to a \to af$ và $d \to ac \to a \to a$. Trong thủ tục ta không áp dụng được *LPG*, vì

$$\text{SF}(u_{\min}(d \to ac \to a \to a)) = 148 < mu.$$

Tiếp tục các dòng 14–15, trường hợp (ii) được xét giữa i_{new} với $\beta = d \to ac \to a \to f$. Vì

$$\begin{aligned}&\text{SE}(\mathcal{D}''_{d \to ac \to a \to f}) = \text{SLIP}(\mathcal{D}''_{d \to ac \to a \to af})\\ &\quad = \text{SE}(\mathcal{D}''_{d \to ac \to a \to f}) = \text{SLIP}(\mathcal{D}''_{d \to ac \to a \to f}) = 1,\\ &\text{SF}(u_{\min}(d \to ac \to a \to f)) = 211 \geq mu,\end{aligned}$$

ta tỉa toàn bộ nhánh *non-GHU branch*$(d \to ac \to a \to af)$ bởi *LPG*, cụ thể là đặt giá trị *false* cho trường *do-ext* của i_{new}.

Sau đó, ta xét s-mở rộng $d \to ac \to a \to f$ và chèn nó vào *FGHUS*. Với các i-mở rộng $d \to ac \to ac$ cũng như $d \to ac \to af$, không có thay đổi nào diễn ra vì $d \to ac \to af$ là chuỗi cha của chuỗi $d \to a \to f$ trong *FGHUS* với cùng độ hỗ trợ 2. Cuối cùng, với i-mở rộng $d \to ac \to ag$. Ta có, $I_{\text{new}}(d \to ac \to ag) = \varnothing$ và $S_{\text{new}}(d \to ac \to ag) = \{a, f\}$. Ở các dòng 26–30 trong *DfsFGHUS*, trường hợp (b) xảy ra với lời gọi *LocalPruningGHU*$(d \to ac \to ag \to f, d \to ac \to a \to f)$. Khi đó, ta tỉa toàn bộ nhánh $d \to ac \to ag \to f$ không chứa chuỗi sinh phổ biến lợi ích cao.

Quay trở lại nút $d \to ac$ và xét các i-mở rộng của nó, ta không tìm thấy lời giải nào, đồng thời cũng không có nhánh nào bị tỉa. Tuy nhiên, với $d \to ace$, ta tỉa được 7 nhánh không chứa chuỗi sinh và đưa thêm ứng viên $d \to ace \to f_1^{215}$ vào *FGHUS*.

Khi xét hai nhánh bắt đầu tại $d \to f_3^{267}$ và $d \to c_3^{215}$, ta chèn thêm chúng vào *FGHUS* đồng thời loại bỏ $d \to ace \to f_1^{215}$. Tìm kiếm trên $d \to f_{_}$ kết thúc và ta tiếp tục trên $d \to c_{_}$. Trong quá trình này, ta phát hiện ra 12 lần sử dụng chiến lược tỉa *LPG*, và 64 lần sử dụng *WPS* và *DPS*. Năm ứng viên sau được thêm vào *FGHUS*: $d \to ce \to f_1^{208}$, $d \to c \to a \to f_1^{204}$, $d \to c \to f_2^{328}$ và $d \to c \to g \to f_1^{204}$. Các ứng viên $d \to ac_3^{230}$, $d \to af_3^{231}$, $d \to ac \to a \to f_1^{211}$ và $d \to ace \to f_1^{215}$ bị loại vì $d \to c_3^{215}$, $d \to f_3^{267}$, $d \to c \to a \to f_1^{204}$ và $d \to ce \to f_1^{208}$ đã có trong *FGHUS* tương ứng.

Sau đó, quá trình tìm kiếm trên e, f, và g diễn ra. Đáng tiếc là, ta không thu được gì. Cuối cùng, ta có

$$\begin{aligned}FGHUS = \{&c \to f_4^{220}; ac \to a \to f_3^{211}, ce \to a \to f_3^{218},\\ &d \to f_3^{267}, d \to c_3^{215}; ac \to ad \to c \to ac_2^{200},\\ &ac \to ad \to cdf_2^{202}, ace \to d \to ac \to c_2^{202},\\ &c \to ad \to ac \to ac_2^{202}, c \to ad \to acdf_2^{203},\\ &ce \to ad \to c \to c_2^{200}, ce \to d \to c \to ac_2^{200},\\ &ce \to d \to cdf_2^{200}, d \to a \to f_2^{245},\\ &d \to c \to f_2^{328}; d \to ce \to f_1^{208},\\ &d \to c \to a \to f_1^{204}, d \to c \to g \to f_1^{204}\},\end{aligned}$$

với 18 lời giải, chiếm tỉ lệ 22% so với 83 chuỗi phổ biến lợi ích cao.

3. Thử nghiệm

Các thử nghiệm nhằm minh họa tính hiệu quả của *FGenHUSM* được tiến hành trên các cơ sở dữ liệu được mô tả trong Bảng III. Kosarak và Snake là hai QSDB thực tế, trong khi D4C7T5N5S6I4 và D0.5C10T15N2S6I4 là hai QSDB tổng hợp, với các giá trị về số lượng và lợi ích đơn vị của các thuộc tính trong các QSDB tương ứng được tạo ngẫu nhiên bởi chương trình sinh dữ liệu của IBM (IBM Quest data generator) từ thư viện SPMF [21].

Bảng III
QSDB $\mathcal{D}''$ MINH HỌA THUẬT TOÁN *FGenHUSM*

Tên	Số chuỗi	Số thuộc tính	Độ dài trung bình	Loại dữ liệu
Kosarak	10000	10094	8,14	Duyệt web
D4C7T5N5S6I4	4000	5000	28,7	Tổng hợp
D0.5C10T15N2S6I4	500	2000	127,7	Tổng hợp
Snake	163	20	60,6	Chuỗi protein

Với mỗi QSDB Q cố định, chúng tôi xác định một ngưỡng *ms* tương đối (tần suất tính theo % đã được dùng phổ biến trong các thực nghiệm cho các thuật toán về lĩnh vực này, $ms\% \stackrel{\text{def}}{=} ms * 100/|Q|(\%)$, trong đó $|Q|$ là số các q-chuỗi đầu vào của Q). Nếu không gây hiểu nhầm ta cũng ký hiệu *ms* tương đối là *ms*.

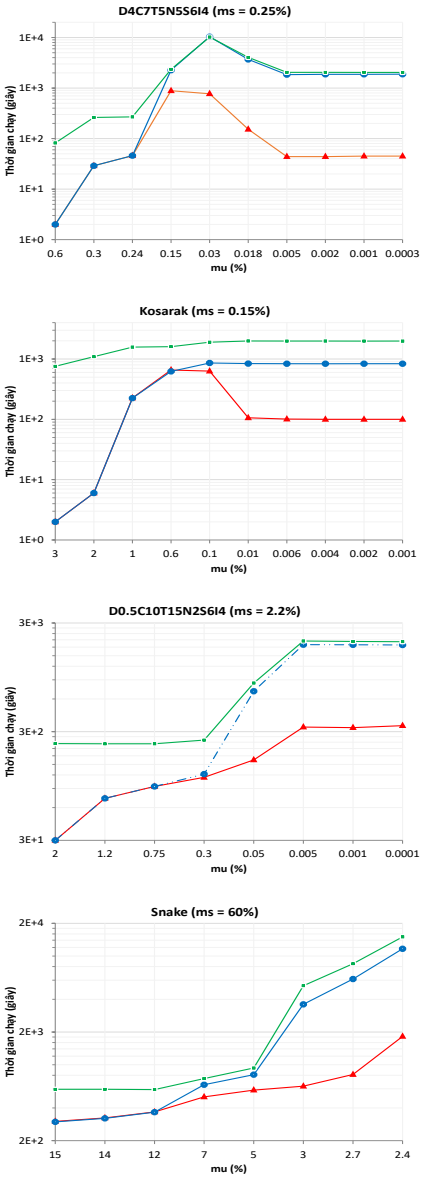
Chúng tôi đã tiến hành khai thác các chuỗi sinh phổ biến lợi ích cao bằng *FGenHUSM* trong ba trường hợp: (i) ***Non***: không dùng chiến lược tỉa nào (tìm các chuỗi phổ biến lợi ích cao trước và sau đó lọc ra các chuỗi sinh); (ii) ***WDPS***: sử dụng *WPS* và *DPS*; (iii) ***All***: dùng cả ba chiến lược *WPS*, *DPS* và *LPG*.

Trước hết, chúng tôi nhận thấy rằng hai tập kết quả của *FGenHUSM* luôn trùng nhau tương ứng với hai trường hợp: khi dùng cả ba chiến lược tỉa (*All*) và khi không dùng bất cứ chiến lược nào (*Non*) mà chỉ đơn thuần dùng định nghĩa của chuỗi sinh lợi ích cao phổ biến (Định nghĩa 5). Do đó, tính đúng của *FGenHUSM* đã được kiểm chứng thêm thông qua thực nghiệm.

Thời gian chạy của *FGenHUSM* được cho trong hình 1. Khi ngưỡng *mu* cao, các điều kiện tỉa $\text{LRU}(\alpha) < mu$ và $\text{RBU}(\alpha) < mu$ (trong hai chiến lược *WPS* và *DPS*, hay *WDPS*) dễ có cơ hội xảy ra hơn, nên số ứng viên (sinh ra và chưa bị tỉa) khi dùng *WDPS* là bé hơn đáng kể so với *Non* (không dùng chiến lược nào). Khi *mu* giảm dần, tác dụng của *WDPS* cũng giảm theo. Ngược lại, với các ngưỡng *mu* thấp, vì điều kiện $\text{SF}(\alpha) \geq mu$ (trong chiến lược *LPG* khi dùng *All*) dễ xảy ra hơn, nên số ứng viên sinh ra khi dùng *All* (áp dụng cả ba chiến lược tỉa nêu trên) là bé hơn đáng kể so với chỉ dùng *WDPS*. Vì vậy, so với *Non*, việc áp dụng đồng thời cả ba chiến lược tỉa sẽ tỉa nhiều hơn các chuỗi ứng viên, do đó thời gian thi hành của thuật toán sẽ nhanh hơn đáng kể.

Ghi nhận số lượng các ứng viên được sinh ra tương ứng trong hình 2 của ba trường hợp lý giải thêm về sự khác nhau về thời gian chạy của chúng.

Hình 3 chỉ ra số lượng các chuỗi FGHU khai thác được bởi *FGenHUSM* (*#fGenHU*) và số lượng các chuỗi FHU (*#fHU*). Có thể thấy rằng, *#fGenHU* bé hơn *#fHU* từ 5 đến 100 lần theo trung bình (đặc biệt khi *mu* bé). Vì vậy, theo

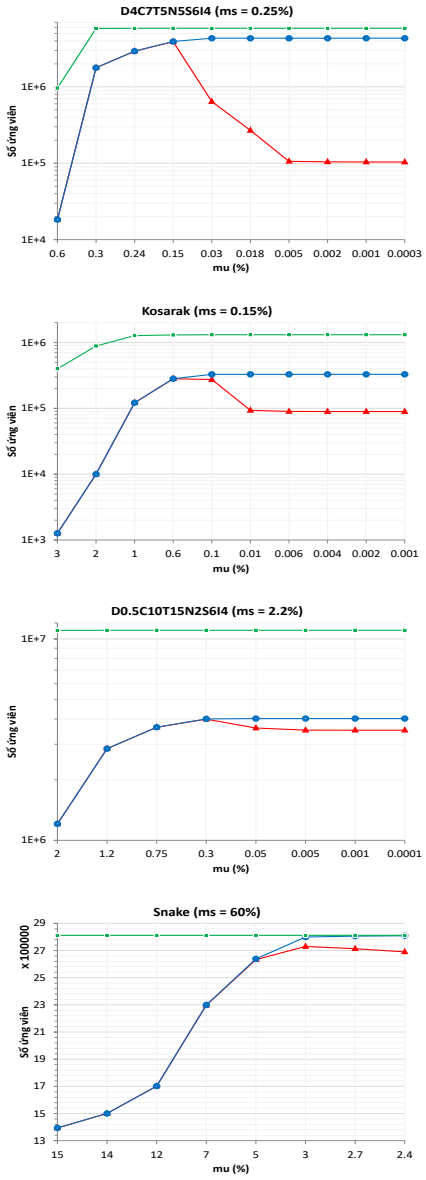


Hình 1. Thời gian chạy của *FGenHUSM*. Ghi chú: *All* (màu đỏ, hình tam giác), *WDPS* (màu xanh đậm, hình tròn), *Non* (màu xanh lá cây, hình vuông).

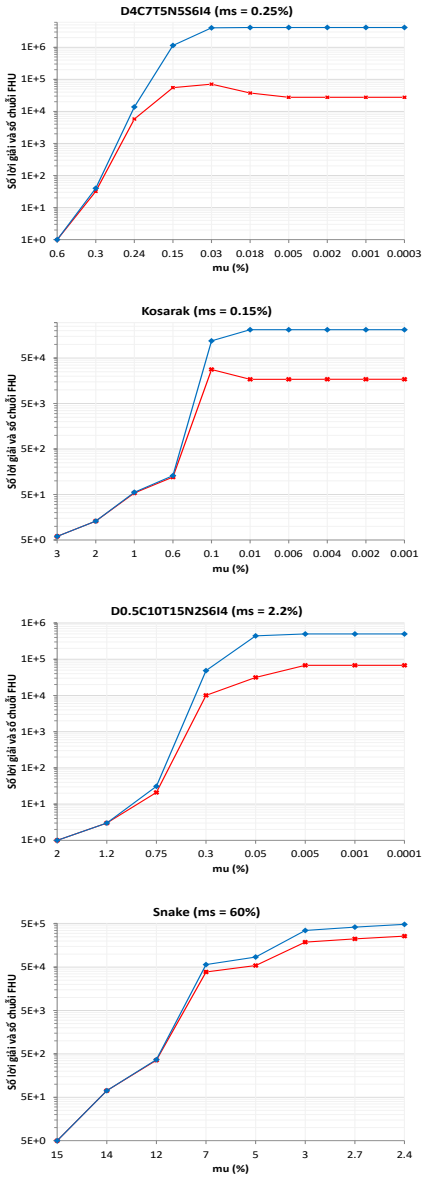
lực lượng, có thể xem *FGHUS* là một biểu diễn súc tích của *FHUS*.

V. KẾT LUẬN

Bài báo này đã đề xuất khái niệm tập *FGHUS* các chuỗi sinh phổ biến lợi ích cao và thuật toán hiệu quả *FGenHUSM* khai thác chúng, thông qua các kỹ thuật tỉa các nhánh ứng viên trên cây tìm kiếm tiền tố. Trước hết, hai chiến lược tỉa theo chiều rộng *WPS* và chiều sâu *DPS* (đã dùng trong các kết quả trước đây cho việc tỉa các chuỗi lợi ích thấp) được điều chỉnh phù hợp để tỉa các chuỗi ít



Hình 2. Số ứng viên sinh bởi *FGenHUSM*. Ghi chú: *All* (màu đỏ, hình tam giác), *WDPS* (màu xanh đậm, hình tròn), *Non* (màu xanh lá cây, hình vuông).



Hình 3. Số lời giải và số chuỗi FHU. Ghi chú: *#fGenHU* (màu xanh đậm, hình thoi), *#fHU* (màu đỏ, hình dấu nhân).

phổ biến hoặc các chuỗi lợi ích thấp. Sau đó, chúng tôi đã chỉ ra một chặn dưới SF của độ đo lợi ích tối thiểu $u_{\min}$. Dựa vào SF và điều kiện tỉa sớm tổng quát [13], chiến lược tỉa *LPG* được đề xuất và dùng để tỉa các ứng viên không là chuỗi sinh phổ biến lợi ích cao. Để ý rằng, *WPS* và *DPS* có tác dụng tỉa mạnh với các ngưỡng lợi ích tối thiểu *mu* cao, trong khi *LPG* có hiệu quả cao với các *mu* thấp. Do đó, chúng tôi đã tích hợp tất cả chúng vào thuật toán *FGenHUSM*. Thực nghiệm trên bốn cơ sở dữ liệu lớn (thực tế lẫn tổng hợp) đã chỉ ra rằng *FGenHUSM* khai thác nhanh *FGHUS* – một biểu diễn súc tích của tập các chuỗi phổ biến lợi ích cao.

LỜI CẢM ƠN

Nghiên cứu này được tài trợ bởi Quỹ Phát triển Khoa học và Công nghệ Quốc gia (NAFOSTED) trong đề tài mã số 102.05-2017.300.

TÀI LIỆU THAM KHẢO

[1] C. F. Ahmed, S. K. Tanbeer, and B.-S. Jeong, "Mining high utility web access sequences in dynamic web log data," in *11th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*, 2010, pp. 76–81.

[2] B.-E. Shie, H.-F. Hsiao, V. S. Tseng, and S. Y. Philip, "Mining high utility mobile sequential patterns in mobile

commerce environments," in *International Conf. Database Systems for Advanced Applications*, 2011, pp. 224–238.
[3] M. Zihayat, H. Davoudi, and A. An, "Top-k utility-based gene regulation sequential pattern discovery," in *IEEE International Conference on Bioinformatics and Biomedicine*, 2016, pp. 266–273.
[4] C. F. Ahmed, S. K. Tanbeer, and B.-S. Jeong, "A novel approach for mining high-utility sequential patterns in sequence databases," *ETRI Journal*, vol. 32, no. 5, pp. 676–686, 2010.
[5] J. Yin, Z. Zheng, and L. Cao, "USpan: An efficient algorithm for mining high utility sequential patterns," in *ACM/SIGKDD Int'l Conf. Knowl. Disc. Data Mining*, 2012, pp. 660–668.
[6] J. C.-W. Lin, J. Zhang, and P. Fournier-Viger, "High-utility sequential pattern mining with multiple minimum utility thresholds," in *Asia-Pacific Web and Web-Age Infor. Management Joint Conf. Web and Big Data*, 2017, pp. 215–229.
[7] T. Truong, A. Tran, H. Duong, B. Le, and P. Fournier-Viger, "EHUSM: Mining high utility sequences with a pessimistic utility model," *1st Int'l Work. Utility-Driven Mining*, 2018.
[8] T. C. Tin, T. N. Anh, D. Van Hai, and L. H. Bac, "HUPSMT: An efficient algorithm for mining high utility-probability sequences in uncertain databases with multiple minimum utility thresholds," *Journal of Computer Science and Cybernetics*, vol. 35, no. 1, pp. 1–20, 2019.
[9] T. Truong-Chi and P. Fournier-Viger, "A survey of high utility sequential pattern mining," in *High-Utility Pattern Mining*, P. Fournier-Viger, J. Lin, R. Nkambou, B. Vo, and V. Tseng, Eds. Springer, 2019, pp. 97–129.
[10] W. Gan, J. C.-W. Lin, J. Zhang, H.-C. Chao, H. Fujita, and S. Y. Philip, "ProUM: Projection-based utility mining on sequence data," *Info. Sciences*, vol. 513, pp. 222–240, 2020.
[11] X. Yan, J. Han, and R. Afshar, "CloSpan: Mining closed sequential patterns in large datasets," in *SIAM International Conference on Data Mining*, 2003, pp. 166–177.
[12] B. Le, H. Duong, T. Truong, and P. Fournier-Viger, "FCloSM, FGenSM: Two efficient algorithms for mining frequent closed and generator sequences using the local pruning strategy," *Knowledge and Information Systems*, vol. 53, no. 1, pp. 71–107, 2017.
[13] T. Truong, H. Duong, B. Le, and P. Fournier-Viger, "FMax-CloHUSM: An efficient algorithm for mining frequent closed and maximal high utility sequences," *Eng. Applications of Artificial Intelligence*, vol. 85, pp. 1–20, 2019.
[14] L. Szathmary, P. Valtchev, A. Napoli, and R. Godin, "Efficient vertical mining of frequent closures and generators," in *Int'l Symp. Intelligent Data Analysis*, 2009, pp. 393–404.
[15] A. Tran, T. Truong, and B. Le, "Simultaneous mining of frequent closed itemsets and their generators: Foundation and algorithm," *Engineering Applications of Artificial Intelligence*, vol. 36, pp. 64–80, 2014.
[16] P. Fournier-Viger, C.-W. Wu, and V. S. Tseng, "Novel concise representations of high utility itemsets using generator patterns," in *International Conference on Advanced Data Mining and Applications*, 2014, pp. 30–43.
[17] P. Fournier-Viger, A. Gomariz, M. Šebek, and M. Hlosta, "VGEN: Fast vertical mining of sequential generator patterns," in *International Conference on Data Warehousing and Knowledge Discovery*, 2014, pp. 476–488.
[18] H. Duong, T. Truong, and B. Le, "Efficient algorithms for simultaneously mining concise representations of sequential patterns based on extended pruning conditions," *Eng. Applications of Artificial Intelligence*, vol. 67, pp. 197–210, 2018.
[19] P. D. Grünwald, I. J. Myung, and M. A. Pitt, *Advances in minimum description length: Theory and applications*. MIT press, 2005.
[20] M.-T. Tran, B. Le, B. Vo, and T.-P. Hong, "Mining non-redundant sequential rules with dynamic bit vectors and pruning techniques," *Applied Intelligence*, vol. 45, no. 2, pp. 333–342, 2016.
[21] P. Fournier-Viger, A. Gomariz, T. Gueniche, A. Soltani, C.-W. Wu, and V. S. Tseng, "SPMF: A Java open-source pattern mining library," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3389–3393, 2014.



Trương Chí Tín là giảng viên tại Khoa Toán – Tin học, Trường Đại học Đà Lạt. Tác giả tốt nghiệp Cử nhân Toán tại Trường Đại học Đà Lạt năm 1983 và nhận bằng Tiến sĩ về Điều khiển tối ưu ngẫu nhiên năm 1990 tại Đại học Quốc gia Hà Nội. Hướng nghiên cứu hiện nay của tác giả là trí tuệ nhân tạo và khai thác dữ liệu.



Trần Ngọc Anh đang giảng dạy và nghiên cứu tại Khoa Toán – Tin học, Trường Đại học Đà Lạt. Tác giả tốt nghiệp Đại học ngành Toán – Tin học vào năm 1999 tại Đại học Đà Lạt, nhận bằng Thạc sĩ và Tiến sĩ về Khoa học máy tính vào các năm 2004 và 2016 tại Trường Đại học Khoa học Tự nhiên, Đại học Quốc gia Tp. Hồ Chí Minh. Tác giả đang nghiên cứu về khai thác dữ liệu và trí tuệ nhân tạo.



Dương Văn Hải tốt nghiệp Trường Đại học Đà Lạt ngành Tin học năm 2004. Tác giả tốt nghiệp Cao học ngành Công nghệ thông tin năm 2009 và đang là Nghiên cứu sinh tại Trường Đại học Khoa học Tự nhiên, Đại học Quốc gia Tp. Hồ Chí Minh. Tác giả hiện đang giảng dạy và nghiên cứu về khai thác dữ liệu tại Khoa Toán – Tin học, Trường Đại học Đà Lạt.



Lê Hoài Bắc nhận bằng Cử nhân Toán năm 1984, Cao học Khoa học máy tính năm 1990 và hoàn thành Luận án Tiến sĩ về Đảm bảo Toán học cho máy tính năm 2000 tại Trường Đại học Khoa học Tự nhiên, Đại học Quốc gia Tp. Hồ Chí Minh. Tác giả hiện là giảng viên tại Khoa Công nghệ Thông tin, Trường Đại học Khoa học Tự nhiên, Đại học Quốc gia Tp. Hồ Chí Minh và đang nghiên cứu về trí tuệ nhân tạo, tính toán mềm, khai thác dữ liệu và khoa học dữ liệu.

Tăng tốc độ phát hiện dị thường trên ảnh đa phổ và siêu phổ ứng dụng trong tìm kiếm cứu nạn

Nguyễn Văn Phương, Đào Khánh Hoài, Tống Minh Đức
Học viện Kỹ thuật Quân sự, Hà Nội

Tác giả liên hệ: Nguyễn Văn Phương, phuongnv@mta.edu.vn
Ngày nhận bài: 14/06/2019, ngày sửa chữa: 27/10/2019, ngày duyệt đăng: 27/10/2019
Định danh DOI: 10.32913/mic-ict-research-vn.v2019.n2.866
Biên tập lĩnh vực điều phối phản biện và quyết định nhận đăng: TS. Phan Anh Huy

Tóm tắt: Máy dò dị thường do Reed và Yu đề xuất được công nhận là máy chuẩn để phát hiện dị thường trên ảnh đa phổ và siêu phổ. Tuy nhiên, máy này có một số hạn chế: dữ liệu ảnh phải tuân theo mô hình Gauss đa biến, tính toán nghịch đảo của ma trận hiệp phương sai rất phức tạp khi ảnh nền có kích thước lớn, hoạt động thiếu ổn định, đôi khi có tỉ lệ báo động giả cao, thiếu mối liên hệ không gian giữa các điểm ảnh. Quy tắc quyết định Neyman-Pearson thường được sử dụng dựa trên việc tính toán hàm mật độ xác suất phi tham số của dữ liệu nền để nâng cao hiệu suất và độ tin cậy, nhưng lại có độ phức tạp tính toán cao. Để giảm độ phức tạp tính toán và thời gian tính toán, nhiều phương pháp đã được sử dụng, như: biến đổi Fourier nhanh, biến đổi Gauss nhanh, lập trình đa luồng trên bộ xử lý trung tâm (CPU), song song trên bộ xử lý đồ họa (GPU). Bài báo này trình bày một phương pháp ước lượng nhanh hàm mật độ xác suất bằng cách phân nhóm các điểm ảnh trên miền giá trị và tổ chức dữ liệu trên cây Kd-tree. Kết quả kiểm nghiệm cho thấy phương pháp đề xuất vượt trội các phương pháp khác và có thể ứng dụng trong thực tế.

Từ khóa: *Tăng tốc độ phát hiện dị thường, Kd-tree, ước lượng mật độ phi tham số.*

Title: **Acceleration of Anomaly Detection in Multispectral and Hyperspectral Images for Search and Rescue Situations**

Abstract: Reed-Yu detector is recognized as a standard algorithm for detecting anomalies on multispectral and hyperspectral images. However, this detector has several limitations: image data must follow the multivariate Gaussian model, calculation of the covariance matrix inverse is complex for large size background images is a complex, lack of robustness, high false alarm rates sometimes, lack of spatial correlation among pixels. The Neyman-Pearson detection criterion is often applied on the nonparametric probability density function of the background data for effectiveness and reliability, at the expense of high computational complexity. To reduce the computational complexity, various methods can be applied, such as: fast Fourier transform, fast Gaussian transform, multi-threaded programming on CPU, parallel on GPU. This paper proposes a method for fast estimation of the density by grouping pixels based on the range of pixels and organizing the data using the Kd-tree. The experimental results show that the proposed method outperforms the state-of-the-art methods and can be applied in practice.

Keywords: *Acceleration of anomaly detection, Kd-tree, non-parametric density estimation.*

I. MỞ ĐẦU

Hoạt động tìm kiếm và cứu nạn bao gồm việc tìm kiếm và giải cứu người, phương tiện bị mắc kẹt trong các tình huống khó khăn hoặc báo nạn. Một cách tiếp cận đang ngày càng được sử dụng nhiều trong tìm kiếm cứu nạn là sử dụng ảnh đa phổ hay siêu phổ có độ phân giải cao được thu từ các bộ cảm biến gắn trên máy bay hoặc vệ tinh. Tuy nhiên, các ảnh hưởng bất lợi gây ra bởi đặc trưng của địa hình, điều kiện thời tiết khắc nghiệt làm cho tọa độ báo nạn có dung sai lớn. Các thiết bị cảm biến thu dữ liệu phải quét trên một diện rộng và dung lượng dữ liệu lớn là một rào cản đối với việc tìm kiếm thủ công bằng mắt thường. Các kỹ thuật tiền xử lý dữ liệu và các thuật toán tìm kiếm tự động là giải pháp phù hợp giúp người quan sát nâng cao hiệu suất và tốc độ tìm kiếm.

Tự động phát hiện mục tiêu dựa trên các đặc trưng hình học có thể được sử dụng để tiếp cận vấn đề này. Tuy nhiên, các đặc trưng hình học của các đối tượng quan tâm không được xác định rõ trong hầu hết các tình huống tìm kiếm cứu nạn. Mặc dù trực tiếp tìm ra người đang gặp nạn sẽ là lý tưởng, nhưng trong một số trường hợp các đồ vật đi kèm như quần áo, vật dụng cá nhân, mảnh vỡ phương tiện, v.v. có thể cung cấp một số thông tin hữu ích. Vì vậy, phát hiện

dị thường sẽ cung cấp một cách tiếp cận phù hợp hơn cho vấn đề này. Dị thường trên ảnh đa phổ và siêu phổ được xác định là những điểm ảnh hoặc cụm điểm ảnh có phổ nổi bật hoặc khác biệt nhiều so với những điểm ảnh lân cận. Những điểm ảnh này thường là thưa thớt và hiếm khi đại diện cho ảnh [1]. Nói chung, các dấu hiệu dị thường là rất nhỏ về mặt không gian và tồn tại với xác suất thấp trong một cảnh ảnh.

Trong hơn 20 năm qua, cộng đồng nghiên cứu trên thế giới đã xây dựng rất nhiều bộ dò dị thường để phát hiện các điểm ảnh dị thường trên ảnh đa phổ, siêu phổ. Dựa trên các kỹ thuật khác nhau của các máy dò, dựa trên bốn nhóm giải pháp chính: thống kê, hạt nhân, không gian đặc trưng và phân đoạn [2]. Máy dò dị thường do Reed và Yu xây dựng vào năm 1990 [3] là một trong những máy dò dị thường dựa trên thống kê và được gọi là máy dò RX (RXD). RXD đã khơi nguồn cho rất nhiều thuật toán được phát triển sau này [2] và nó được coi là máy phát hiện dị thường chuẩn cho hình ảnh đa phổ, siêu phổ [4]. Hiệu quả của RXD trong việc phát hiện dị thường từ các ảnh đa phổ và siêu phổ đã được kiểm chứng [1, 3–9]. Mặc dù vậy, RXD có những hạn chế nhất định. Thứ nhất, việc ước lượng nghịch đảo của ma trận hiệp phương sai của dữ liệu nền với kích thước chiều dữ liệu lớn thường rất phức tạp và hoạt động không ổn định [10, 11] dẫn đến làm suy yếu thuật toán. Thứ hai, đôi khi RXD gây ra tỷ lệ báo động giả cao (ví dụ, một cái cây đơn lẻ trong đồng cỏ được phát hiện là dị thường cục bộ ngay cả khi toàn bộ ảnh có cả một khu rừng) [11–14]. Thứ ba, RXD giả định dữ liệu nền tuân theo mô hình Gauss đa biến, nhưng có nhiều trường hợp giả định này có thể không đầy đủ vì trong thực tế các cảnh ảnh rất đa dạng và chứa nhiều lớp đối tượng khác nhau [11, 14, 15]. Thứ tư, RXD thiếu mối liên hệ về không gian, mỗi điểm ảnh được đánh giá riêng lẻ và không quan tâm đến những điểm ảnh xung quanh nó.

Để giảm những hạn chế của RXD, trong một vài năm gần đây các nhà khoa học đã áp dụng quy tắc ra quyết định dựa trên kiểm nghiệm tỷ lệ khả năng (LRT) dựa trên hàm mật độ xác suất (PDF) của dữ liệu nền để phát hiện các dị thường trên ảnh đa phổ và ảnh siêu phổ. Cụ thể, năm 2011 trong nghiên cứu [16] của Veracini và các cộng sự, phương pháp đề xuất sử dụng Parzen Widnow (PW) để ước tính PDF nền đã cho kết quả đáng tin cậy. Sau khi PDF nền được xấp xỉ thông qua PW, nó được dùng làm đầu vào để phát hiện các dấu hiệu dị thường trên ảnh dựa trên kiểm nghiệm tỷ lệ khả năng. Năm 2012, trong nghiên cứu [1], Bolukbasi và cộng sự đã xây dựng kiểm nghiệm giả thuyết nhị phân cho phát hiện dị thường và sử dụng thuật toán KNN để tìm k láng giềng gần nhất để tính hàm mật độ xác suất phi tham số cho điểm ảnh đang xét. Kết quả thu được đã vượt so với RXD. Năm 2014, trong nghiên cứu [17], Matteoli và nhóm tác giả đã đưa ra chiến lược để quyết định một điểm ảnh có phải là dị thường hay là nền dựa trên định lý Neyman-Pearson sử dụng các hàm PDF. Trong đó các tác giả đã kiểm nghiệm trên ba hàm nhân PDF: hạt nhân Gauss cố định băng thông, hạt nhân Gauss không cố định băng thông (VKDE) và tìm kiếm k láng giềng gần nhất, để ước lượng hàm mật độ giống như trong [1]. Kết quả là cả ba hàm nhân PDF trên đều cho ra hiệu suất phát hiện dị thường cao hơn RXD. Năm 2017, trong nghiên cứu [18] của Zhao và các cộng sự, sự kết hợp của các phương pháp ước lượng mật độ phi tham số và phát hiện dựa trên biểu diễn mối quan hệ tương quan (CRD), cho thấy hiệu suất phát hiện dị thường khá cao và đã vượt RXD.

Tuy nhiên, độ phức tạp tính toán của kỹ thuật phi tham số trong ước lượng hàm mật độ xác suất là $O(kn^2)$, trong đó n là số lượng điểm ảnh và k là số kênh phổ, làm cho việc tính toán tốn kém thời gian (trong phần thực nghiệm của bài báo, một ảnh màu ba kênh RGB, kích thước 3396×3349 pixel tốn gần 21 ngày để tính toán) dẫn đến khả năng ứng dụng vào thực tế rất hạn chế, đặc biệt là ứng dụng trong công tác tìm kiếm cứu nạn. Để tăng tốc độ tính toán, giảm thời gian xử lý, một số kỹ thuật gần đúng đã được đề xuất. Đầu tiên, đó là đề xuất của Silverman trong nghiên cứu [19] sử dụng biến đổi Fourier nhanh (FFT) để ước lượng mật độ. Nó làm giảm đáng kể yêu cầu tính toán của phương pháp ước tính mật độ, đã giảm độ phức tạp tính toán từ $O(N^2)$ xuống còn $O(N \log N)$. Một phương pháp khác là áp dụng biến đổi Gauss nhanh (FGT) được Elgammal và các cộng sự đề xuất trong nghiên cứu [20]. Phương pháp này đã giảm độ phức tạp tính toán từ $O(NM)$ xuống còn $O(N + M)$. Trong đó, $N = kn$ là kích thước dữ liệu, và M là số lượng mục tiêu cần tính PDF. Mặc dù cả hai phương pháp FFT và FGT đã giảm độ phức tạp tính toán PDF nhưng đổi lại, việc tính toán gần đúng giảm hiệu suất phát hiện dị thường của thuật toán.

Ngoài ra, một cách tiếp cận khác để giảm thời gian tính toán là song song hóa quá trình ước tính mật độ hàm hạt nhân trên mạng máy tính, trên CPU hoặc GPU. Trong nghiên cứu [21], Lukasik đã đề xuất sử dụng thư viện giao thức truyền thông điệp (MPI) để song song hóa việc ước lượng hàm mật độ xác suất. Năm 2013, Michailidis và Margaritis đã song song hóa ước lượng mật độ hàm hạt nhân trên các khung lập trình khác nhau như Pthreads, OpenMP, Intel Cilk ++, Intel TBB và SWARM [22]. Cũng trong năm 2013, họ tiếp tục song song hóa ước lượng hàm mật độ hạt nhân trên nền tảng GPU CUDA [23]. Ưu điểm của các phương pháp này là không làm thay đổi hiệu suất phát hiện dị thường của các thuật toán. Tuy nhiên, độ phức tạp tính toán PDF không thay đổi, vẫn là $O(kn^2)$; thời gian tính toán giảm do các phương pháp này đã chia tổng khối lượng công việc làm nhiều phần và tính toán đồng thời.

Qua quá trình nghiên cứu chúng tôi thấy rằng, trong công thức tính mật độ xác suất, việc tìm những điểm ảnh trong phạm vi băng thông để hàm hạt nhân khác 0 tiêu tốn rất nhiều thời gian. Vì vậy, để giảm được thời gian tính toán, chúng tôi phân các điểm ảnh về các nhóm cùng giá trị. Mục đích là làm giảm số lượng dữ liệu cần tính toán, thay vì phải tính toán toàn bộ n điểm ảnh, chúng ta chỉ phải tính toán trên m nhóm các điểm ảnh, với m nhỏ hơn rất nhiều so với n. Trong tự nhiên, lớp phủ thực địa luôn có tính chất phân lớp đối tượng, lớp phủ càng đồng nhất số lượng nhóm càng ít. Bởi vậy, bước phân nhóm các điểm ảnh về cơ bản làm giảm đáng kể số lượng điểm dữ liệu cần xét đến. Bước tiếp theo, chúng tôi tổ chức dữ liệu theo cây Kd-tree đối với dữ liệu chưa phân nhóm và dữ liệu sau phân nhóm để tăng tốc độ tìm kiếm các điểm dữ liệu trong phạm vi băng thông thỏa mãn hàm hạt nhân khác 0.

Phần tiếp theo của bài báo được cấu trúc như sau. Phần II trình bày lý thuyết ước lượng mật độ phi tham số và thuật toán để thực hiện việc ước lượng này. Phần III trình bày phương pháp phân nhóm dữ liệu, xây dựng, tìm kiếm trên cây Kd-tree và thuật toán để tính toán PDF khi dữ liệu đã được nhóm và tổ chức vào cây Kd-tree. Phần IV trình bày kết quả thực nghiệm trên ba loại ảnh (ảnh đa phổ 3 kênh phổ, ảnh đa phổ 8 kênh và ảnh siêu phổ 224 kênh). Cuối cùng là kết luận và tài liệu tham khảo.

II. ƯỚC LƯỢNG MẬT ĐỘ HẠT NHÂN

Phương pháp ước lượng mật độ xác suất phi tham số trong đó công cụ chính là ước lượng mật độ hạt nhân (KDE) đã được Rosenblatt công bố vào năm 1956 [24] và sau đó được Parzen phát triển, công bố vào năm 1962 [25].

Bảng I
MỘT SỐ HÀM NHÂN ĐIỂN HÌNH [27]

Tên hàm nhân	$K(u)$	Điều kiện
Uniform	$\frac{1}{2}$	$\lvert u\rvert \le 1$
Hypercube	1	$\lvert u\rvert \le \frac{1}{2}$
Triangular	$1 - \lvert u\rvert$	$\lvert u\rvert \le 1$
Epanechnikov	$\frac{3}{4\sqrt{5}} - \frac{3u^2}{20\sqrt{5}}$	$\lvert u\rvert \le \sqrt{5}$
Quartic	$\frac{15}{16}(1-u^2)^2$	$\lvert u\rvert \le 1$
Triweight	$\frac{35}{32}(1-u^2)^3$	$\lvert u\rvert \le 1$
Tricube	$\frac{70}{81}(1-\lvert u\rvert^3)^3$	$\lvert u\rvert \le 1$
Gaussian	$\frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}u^2}$	
Cosine	$\frac{\pi}{4} cos\left(\frac{\pi}{2}u\right)$	$\lvert u\rvert \le 1$

Đối với dữ liệu một chiều, xét vector ngẫu nhiên $\mathbf{x} = (x_1, x_2, \ldots, x_n)^T$ của biến ngẫu nhiên $\mathbf{x}$ có n phần tử. Điều này có nghĩa rằng có n quan sát của biến ngẫu nhiên $\mathbf{x}$ và x_i là quan sát thứ i của biến ngẫu nhiên $\mathbf{x}$. Khi đó, mật độ hạt nhân của biến ngẫu nhiên $\mathbf{x}$ được ước lượng như sau:

$$\hat{f}(x_i) = \frac{1}{n}\sum_{j=1}^{n} \frac{1}{h_j} K\left(\frac{x_i - x_j}{h_j}\right), \quad i = 1, 2, \ldots, n, \quad (1)$$

trong đó $\hat{f}(\cdot)$ được gọi là hàm mật độ xác suất (PDF), $K(u)$ được gọi là hàm nhân thỏa mãn điều kiện $\int_{-\infty}^{\infty} K(u)d(u) = 1$ và h_j là hệ số tỷ lệ quyết định "khoảng rộng" của hàm nhân hay còn gọi là băng thông. Thảo luận mở rộng về các thuộc tính thống kê của $\hat{f}(\cdot)$ có thể được tìm thấy trong [26], $K(u)$ có thể là các hàm nhân điển hình do Hardle trình bày trong [27] và được thể hiện trong bảng I.

Trong trường hợp dữ liệu có k chiều, quan sát thứ i của $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)^T$ là $\mathbf{x}_i = (x_i^1, x_i^2, \ldots, x_i^k)^T$, $i = 1, \ldots, n$, và công thức ước tính mật độ hạt nhân của dữ liệu đa biến được định nghĩa trong [27] là:

$$\hat{f}(\mathbf{x}_i) = \frac{1}{n}\sum_{j=1}^{n} \left\{ \prod_{d=1}^{k} \frac{1}{h_d} K\left(\frac{x_i^d - x_j^d}{h_d}\right) \right\}, \quad i = 1, 2, \ldots, n. \quad (2)$$

Đối với ảnh đa phổ và siêu phổ, dữ liệu thuộc dạng đa biến, chúng tôi sử dụng công thức (2) để cài đặt thuật toán. Không làm mất tính tổng quát, chúng tôi cố định băng thông, đặt $h = h_1 = h_2 = \cdots = h_d$ với $d = 1, 2, \ldots, k$. Thuật toán 1 được viết giả lập theo ngôn ngữ lập trình C để ước tính mật độ của dữ liệu đa biến theo phương pháp tuần tự trên CPU, đây là thuật toán do Lukasik [21], Michailidis và Margaritis [22, 23] xây dựng.

Trong thuật toán 1, $\mathbf{X}$ là dữ liệu ảnh đa phổ hoặc siêu phổ được tổ chức thành một ma trận hai chiều từ nhiều vector, chỉ số của chiều thứ nhất tương ứng với vị trí trong không gian của các điểm ảnh, chiều thứ hai chứa dữ liệu của các kênh ảnh tại vị trí đó, n là tổng số điểm ảnh, k là số kênh phổ, h là băng thông của hàm ước lượng mật độ, *pdf* là vector lưu trữ mật độ xác suất của các điểm ảnh. Trong thuật toán 1, hàm Kernel được thiết kế riêng bởi những thuật toán phía sau đều phải sử dụng đến nó. Trong hàm Kernel, $\mathbf{x}_i$ là vector giá trị của điểm ảnh cần tính mật độ, $\mathbf{x}_j$ là vector giá trị của điểm ảnh bất kỳ nằm trong băng thông, $K(u)$ là một trong những hàm đã nêu trong bảng I. Thuật toán 1 có độ phức tạp tính toán là $O(kn^2)$.

III. TĂNG TỐC ĐỘ ƯỚC LƯỢNG HÀM MẬT ĐỘ

Như phân tích trong phần II, thuật toán 1 có độ phức tạp tính toán là $O(kn^2)$. Đây là độ phức tạp tính toán theo hàm số mũ. Trong phần thực nghiệm của nghiên cứu [20], tác giả sử dụng 100.000 điểm dữ liệu để kiểm nghiệm và thời gian tính toán là 4 ngày. Trên thực tế, thời gian chúng tôi tính toán PDF cho một ảnh màu RGB 11.373.204 điểm

Thuật toán 1: Thuật toán ước lượng mật độ hạt nhân [21–23]

input: Ma trận các điểm ảnh x, số điểm ảnh n, số kênh phổ k, băng thông h
output: Mật độ xác suất của các điểm ảnh *pdf*

for $i \leftarrow 0$ **to** $n-1$ **do**
  $sum_ker \leftarrow 0$;
  for $j \leftarrow 0$ **to** $n-1$ **do**
    $sum_ker \leftarrow sum_ker + Kernel(x_i, x_j, k, h)$;
  end
  $pdf[i] \leftarrow \dfrac{sum_ker}{n}$;
end
return *pdf*;

Function Kernel

Input: điểm ảnh đang xét x_i, điểm ảnh kiểm tra x_j, số kênh phổ k, băng thông h
Output: Giá trị của $K(u)$
$mul_ker \leftarrow 1$;
for $d \leftarrow 0$ **to** $k-1$ **do**
  $mul_ker \leftarrow mul_ker \times \dfrac{1}{h} \times K\left(\dfrac{x_i^d - x_j^d}{h}\right)$;
end
return *mul_ker*;

ảnh là 21 ngày; thời gian tính toán PDF cho một ảnh 8 kênh phổ 710.613 điểm ảnh là hơn 3 giờ. Do đó, rất khó áp dụng phương pháp này trong những ứng dụng thực tế, nhất là trong công tác tìm kiếm cứu nạn do nó đòi hỏi cao về thời gian đưa ra quyết định.

Quan sát công thức (2) cho thấy chỉ những điểm ảnh làm cho $K(u) \neq 0$ mới có ý nghĩa, những điểm ảnh còn lại không làm tăng giá trị của $\hat{f}(\mathbf{x}_i)$. Vì vậy, chúng ta chỉ đi tìm tập hợp những điểm ảnh thỏa mãn điều kiện $K(u) \neq 0$. Qua quá trình nghiên cứu, chúng tôi nhận thấy rằng, công đoạn để tìm những điểm ảnh thỏa mãn $K(u) \neq 0$ tiêu tốn nhiều thời gian nhất. Vì vậy, phương pháp đầu tiên chúng tôi nghĩ đến là làm thế nào để giảm bớt dữ liệu tính toán mà không làm thay đổi kết quả đầu ra. Đối với ảnh đa phổ, khả năng các điểm ảnh có vector phổ giống nhau tương đối cao, nhất là ảnh màu RGB. Do đó, chúng tôi chia những điểm ảnh này thành m nhóm có cùng giá trị. Như vậy, thay vì chúng ta phải tính toán PDF cho n điểm ảnh chúng ta chỉ phải tính toán PDF cho m nhóm điểm ảnh, các điểm ảnh trong cùng một nhóm sẽ có giá trị mật độ xác suất giống nhau. Khi đó, m càng nhỏ thì thời gian tính toán càng nhanh.

Tiếp theo, chúng tôi tổ chức dữ liệu theo cấu trúc của cây Kd-tree [28]. Về bản chất cây Kd-tree là một cây nhị phân bởi mỗi nút có nhiều nhất là hai con. Nút lá chứa điểm dữ liệu k chiều, những nút không phải là nút lá tạo ra một siêu phẳng tách (lát cắt) để phân chia không gian thành hai phần, được gọi là nửa không gian. Các điểm ở bên trái của siêu phẳng này được biểu thị bằng cây con bên trái của nút đó và các điểm ở bên phải của siêu phẳng được thể hiện bằng cây con bên phải. Những điểm ảnh $\mathbf{x}_j$ thỏa mãn $K(u) \neq 0$ phải là những điểm ảnh láng giềng gần nhất của $\mathbf{x}_i$. Nói cách khác những điểm ảnh này phải nằm trong hình siêu cầu có bán kính r cho trước, tâm là $\mathbf{x}_i$. Thông thường chúng ta phải duyệt hết toàn bộ dữ liệu mới tìm ra được tập hợp đó. Mục đích tổ chức dữ liệu theo cấu trúc cây Kd-tree là để nhanh chóng tìm được tập hợp các điểm ảnh làm cho $K(u) \neq 0$. Do tính chất của cây Kd-tree, mỗi nút không phải là lá sẽ chia không gian thành hai phần nên bắt đầu xét từ nút gốc, nếu điểm $\mathbf{x}_i$ nhỏ hơn gốc một khoảng r thì rõ ràng những điểm ảnh đáp ứng điều kiện $K(u) \neq 0$ phải nằm nhánh bên trái của nút gốc, chúng ta chỉ phải đi tìm những điểm dữ liệu nằm nhánh bên trái của gốc mà không cần quan tâm đến những những nút dữ liệu nằm nhánh bên phải của nút gốc. Và ngược lại, chúng ta chỉ phải đi tìm những những điểm ảnh nằm nhánh bên phải của gốc mà không cần quan tâm đến những điểm ảnh nằm bên nhánh bên trái của nút gốc. Vì vậy, việc áp dụng cây Kd-tree đã giảm được thời gian tính toán hàm tính tổng trong công thức (2).

Những bước ở trên là quá trình tiền xử lý dữ liệu trước khi dữ liệu được dùng để ước lượng hàm mật độ xác suất. Dưới đây, chúng tôi trình bày chi tiết hai bước tiền xử lý và việc ước lượng hàm mật độ xác suất.

1. Nhóm các điểm ảnh có cùng giá trị phổ

Đối với ảnh đa phổ hoặc ảnh siêu phổ, quá trình tìm kiếm những điểm ảnh có phổ trùng nhau mất rất nhiều thời gian, với độ phức tạp tính toán là $O(kmn)$, trong đó m là số nhóm các điểm ảnh có phổ trùng nhau, n là số điểm ảnh và k là số kênh ảnh. Để giảm độ phức tạp tính toán, ý tưởng nhóm các điểm ảnh cùng giá trị là xây dựng một mảng hai chiều, được gọi là mảng **A**. Kích thước chiều thứ nhất của **A** là số lượng tổ hợp màu của các kênh ảnh. Ví dụ, với ảnh màu RGB 24 bit, chiều thứ nhất của mảng **A** sẽ có kích thước là 16.777.216. Kích thước chiều thứ hai của **A** sẽ được cấp phát linh động để lưu trữ vị trí không gian trên ảnh của các điểm ảnh thuộc về nhóm đó.

Thuật toán nhóm chạy qua lần lượt các điểm ảnh, tính giá trị tổ hợp màu của điểm ảnh đó theo công thức sau:

$$index_i = \sum_{d=0}^{k-1} Max^d \times x_i^d, \quad i = 1, 2, \ldots, n, \tag{3}$$

trong đó Max là giá trị lớn nhất của các kênh ảnh của tất cả các điểm ảnh (thông thường, đối với ảnh lưu trữ 8 bit/kênh ta có $Max = 256$, 10 bit/kênh thì $Max = 1024, \ldots$), $index_i$ tương ứng với chỉ số trên chiều thứ nhất của mảng **A**. Chiều thứ hai của mảng **A** sẽ tự động tăng thêm một ô nhớ để lưu trữ vị trí không gian của điểm ảnh đó.

Thuật toán nhóm được trình bày cụ thể tại thuật toán 2. Độ phức tạp của thuật toán là $O(kn)$. Trong thuật toán, đầu tiên phải xây dựng được một cấu trúc để lưu trữ thông tin của nhóm các điểm ảnh. Trong cấu trúc nhóm điểm ảnh phải lưu trữ được giá trị phổ của điểm ảnh và chỉ số của các điểm ảnh thuộc nhóm. Công việc tiếp theo là đi tìm giá trị Max để có cơ sở tính toán kích thước chiều thứ nhất của mảng **A** (tính toán $size = Max^k$), khởi tạo mảng **A** với chiều thứ nhất có kích thước bằng $size$, chiều thứ hai bằng ∅. Trong vòng lặp chính, tính giá trị tổ hợp màu của điểm ảnh thứ i và gán cho $index$, tại chỉ số $index$ của mảng **A**, điểm ảnh thứ i sẽ được thêm vào nhóm. Cuối cùng, loại bỏ những nhóm điểm ảnh không chứa bất kỳ điểm ảnh nào ta sẽ được các nhóm điểm ảnh.

Để ước lượng mật độ xác suất cho các điểm ảnh, công thức (2) được biến đổi thành:

$$\hat{f}_G(\mathbf{g}_i) = \frac{1}{n}\sum_{j=1}^{m}\left\{\left[\prod_{d=1}^{k}\frac{1}{h_d}K\left(\frac{g_i^d - g_j^d}{h_d}\right)\right] \times |M_j|\right\}, \quad (4)$$

trong đó $\hat{f}_G(g_i)$ là hàm ước tính mật đô xác suất của nhóm thứ i, khi đó, tất cả các điểm ảnh trong nhóm thứ i sẽ có mật độ xác suất bằng nhau, $i = 1, 2, \ldots, m$, m là tổng số nhóm, $\mathbf{g}_i$ là vector chứa giá trị các kênh phổ của nhóm thứ i, M_j là tập hợp các điểm ảnh trong nhóm thứ j, và $|M_j|$ là kích thước của tập hợp M_j.

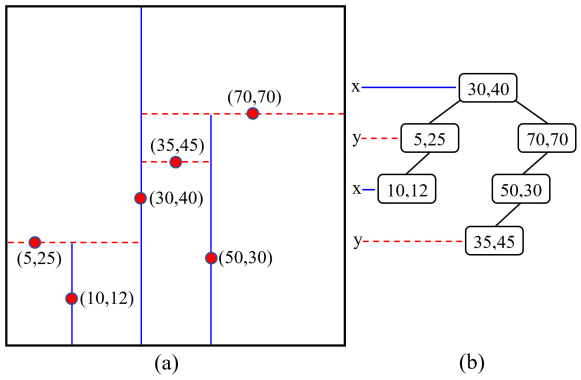
2. Xây dựng và tìm kiếm trên Kd-tree

Cây Kd-tree được phát triển và công bố bởi Bentley [28] vào năm 1975. Về bản chất, nó là một cây nhị phân (do mỗi nút của cây có tối đa 2 nhánh con), trong đó mỗi nút biểu diễn một phân vùng không gian k chiều. Nút gốc đại diện cho toàn bộ không gian, các nút lá đại diện cho không gian con chứa một tập con độc nhất của tập dữ liệu đầu vào. Điểm đặc biệt của cây Kd-tree là các đỉnh của cây là những điểm phân chia không gian thành hai phần. Việc phân chia không gian như vậy sẽ thuận tiện cho tìm kiếm những điểm trong cây gần nhất với một điểm nào đó trong vùng không gian. Điều này có nghĩa rằng, việc tìm những điểm thuộc cây gần với một điểm nào đó trong không gian dựa trên một số phép phân hoạch không gian để loại bỏ các vùng không gian không cần thiết, như vậy sẽ thu hẹp được không gian tìm kiếm.

Thuật toán 2: Thuật toán nhóm các điểm ảnh (CreateGroup)

input: Ma trận điểm ảnh x, số điểm ảnh n, số kênh phổ k
output: vector các nhóm điểm ảnh *groups*

```
Max ← phần tử có giá trị lớn nhất của ma trận x;
size ← Max^k;
Khởi tạo ma trận A, chiều thứ nhất có kích thước bằng size;
for i ← 0 to size − 1 do
    A[i]← {∅}
end
for i ← 0 to n − 1 do
    index ← 0;
    for d ← 0 to k − 1 do
        index ← index + Max^d × x_i^d;
    end
    A[index]← A[index] ∪ {i}
end
groups ← {∅};
for i ← 0 to size − 1 do
    if A[i]≠ ∅ then
        groups ← groups ∪ {A[i]}
    end
end
return groups;
```



Hình 1. a) Minh họa phân chia miền không gian, b) Minh họa cây Kd-tree đã được xây dựng từ dữ liệu đã cho.

Để hiểu rõ hơn về cây Kd-tree, chúng ta xét một ví dụ xây dựng cây Kd-tree từ bộ dữ liệu 2 chiều $(30, 40)$, $(5, 25)$, $(10, 12)$, $(70, 70)$, $(50, 30)$, $(35, 45)$, chi tiết được thể hiện trong hình 1. Trong đó, hình 1(a) thể hiện các vùng không gian đã được chia, những đường thẳng liền nét là những đường chia không gian dữ liệu theo chiều thứ nhất (chiều x), những đường thẳng nét đứt chia không gian dữ liệu theo chiều thứ hai (chiều y). Hình 1(b) thể hiện cây Kd-tree được xây dựng từ bộ dữ liệu trên.

Quy tắc xây dựng cây và phân chia không gian như sau. Chọn một điểm dữ liệu làm gốc, tại gốc chia toàn bộ không gian dữ liệu theo chiều thứ nhất làm hai phần (trong ví dụ, điểm gốc được chọn là $(30, 40)$, điểm này chia toàn bộ không gian dữ liệu theo chiều x thành 2 phần, phần bên trái là những điểm dữ liệu có chiều x nhỏ hơn hoặc bằng 30, phần bên phải là những điểm dữ liệu có chiều x lớn hơn hoặc bằng 30). Tiếp theo, xét lần lượt từng điểm dữ liệu, so sánh điểm dữ liệu này với các nút trên cây, bắt đầu từ gốc. Quy tắc so sánh như sau. Giả sử dữ liệu của chúng ta có k chiều (quy định bắt đầu từ 0 đến $k-1$), lấy bậc của nút cần so sánh chia cho k được phần dư, phần dư này chính là chiều dữ liệu được dùng để so sánh. Nếu điểm dữ liệu nhỏ hơn hoặc bằng với nút so sánh thì điểm dữ liệu sẽ nằm bên trái nút so sánh, ngược lại nằm bên phải. Tiếp tục so sánh đến khi gặp lá thì thêm điểm dữ liệu vào cây.

Thuật toán 3 xây dựng cây Kd-tree từ các điểm ảnh của ảnh đầu vào. Trong thuật toán này, đầu tiên phải xây dựng cấu trúc một nút của cây Kd-tree để lưu trữ dữ liệu và một số thuộc tính khác nữa của node, sau đó xây dựng hàm InsertNode để chèn một nút vào cây, cuối cùng xây dựng hàm CreateKDTree để xây dựng thành một cây hoàn chỉnh. Độ phức tạp tính toán xây dựng cây Kd-tree là $O(n \log n)$ [29].

Sau khi xây dựng cây Kd-tree (theo thuật toán 3), mục tiêu của chúng ta là sử dụng cây này để tìm tất cả những điểm ảnh đáp ứng điều kiện $K(u) \neq 0$. Trong bảng I, chúng ta thấy rằng ngoại trừ hàm nhân Gauss là không có điều kiện, các nhân còn lại đều có điều kiện

$$|u| = \left| \frac{x_i^d - x_j^d}{h} \right| \leq \epsilon,$$

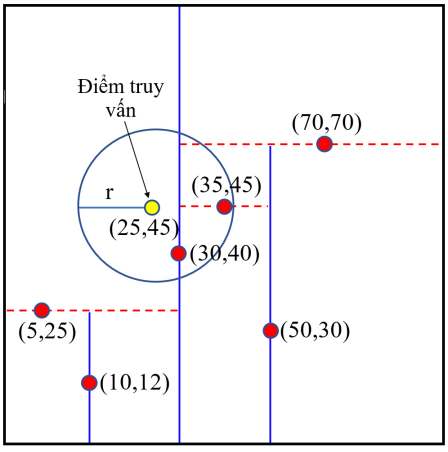
với $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, n$ và $d = 1, 2, \ldots, k$ thì $K(u)$ mới có giá trị, ngược lại $K(u) = 0$. Tùy thuộc vào hạt nhân cụ thể mà ϵ sẽ nhận các giá trị khác nhau. Đặt $r = h \times \epsilon$, khi đó để hạt nhân $K(u) \neq 0$ thì $|x_i^d - x_j^d| \leq r$. Điều này có ý nghĩa rằng với điểm x_i đang được xem xét, những điểm x_j nằm trong hình siêu cầu bán kính r (sử dụng thước đo khoảng cách Chebyshev [30] để đo khoảng cách từ điểm x_i tới điểm x_j) sẽ là những điểm được chọn để tính $K(u)$. Xem minh họa trong hình 2, trong đó điểm dữ liệu cần tính toán PDF là (25,45) với $r = 10$ thì những điểm dữ liệu (35,45) và $(30, 40)$ nằm trong hình tròn bán kính r sẽ thỏa mãn điệu kiện để $K(u) \neq 0$. Vì vậy, những điểm dữ liệu này được tham gia tính toán PDF cho điểm dữ liệu đang xét.

Thuật toán 4 tìm kiếm những điểm ảnh nằm trong hình siêu cầu có bán kính r, có tâm là x_i. Thuật toán sử dụng phương pháp đệ quy để tìm kiếm danh sách các điểm dữ liệu nằm trong hình siêu cầu bán kính r có tâm là điểm đang xét. Những điểm dữ liệu thỏa mãn yêu cầu được lưu trong

Thuật toán 3: Tạo cây Kd-tree (Create Kd-tree)

Function `InsertNode()`
- **input:** nút gốc *root*, điểm dữ liệu *point*, số chiều dữ liệu k, mức của cây *level*;
- **output:** điểm dữ liệu được thêm vào cây;
- Tìm chiều của không gian dữ liệu: $axis \leftarrow level \,\%k$;
- **if** $root = \emptyset$ **then**
  - - Khởi tạo một nút mới;
  - - Gán nút mới khởi tạo cho *root*;
- **else**
  - **if** *point[axis]<root->data[axis]* **then**
    - $InsertNode(root \rightarrow left, point, k, level+1)$
  - **else**
    - $InsertNode(root \rightarrow right, point, k, level+1)$
  - **end**
- **end**

Function `CreateKDTree()`
- **input:** Ma trận điểm ảnh $\mathbf{X}$, số điểm ảnh n, số kênh phổ k;
- **output:** Cây Kd-tree hoàn chỉnh;
- KDNode*$root \leftarrow \emptyset$;
- **for** $i \leftarrow 0$ **to** $n-1$ **do**
  - $InsertNode(root, \mathbf{x}_i, k, 0)$;
- **end**
- **return** *root*;



Hình 2. Minh họa những điểm được chọn để tính $K(u)$.

danh sách *list*. Đầu tiên, thuật toán kiểm tra nút gốc *root*, nếu nó rỗng thì thoát khỏi thuật toán. Tiếp đến tính toán khoảng cách từ điểm đang xét đến *root* (sử dụng phương pháp tính khoảng cách Chebyshev), tìm chiều dữ liệu để so sánh. Kiểm tra nếu *root* nằm trong hình siêu cầu đó thì thêm *root* vào *list*. So sánh điểm đang xét với *root*, nếu

Thuật toán 4: Search [27]

input: Node gốc *root*, điểm cần tính PDF *query*, số chiều dữ liệu k, bán kính siêu cầu r;
output: Danh sách các điểm được tìm thấy *list*;

if *root*= $\emptyset$ **then**
  return
end
Tính khoảng cách từ *query* đến *root*, gán vào d;
Tìm chiều dữ liệu so sánh: $axis \leftarrow root.level\%k$;
if $d \leq r$ **then**
  $list \leftarrow list \cup \{root\}$;
end
if $query[axis] < root.data[axis]$ **then**
  if $root.left \neq \emptyset$ **then**
    $Search(root.left, query, r, list)$;
  end
  if $root.right \neq \emptyset$ **then**
    if $|query[axis] - root.data[axis]| \leq r$ **then**
      $Search(root.right, query, r, list)$;
    end
  end
else
  if $root.right \neq \emptyset$ **then**
    $Search(root.right, query, r, list)$;
  end
  if $root.left \neq \emptyset$ **then**
    if $|query[axis] - root.data[axis]| \leq r$ **then**
      $Search(root.left, query, r, list)$;
    end
  end
end

nhỏ hơn *root* tìm nhánh bên trái của *root*, ngược lại tìm nhánh bên phải của *root*. Chỉ xét riêng trên một chiều dữ liệu so sánh, nếu khoảng cách từ điểm đang xét đến *root* trên chiều dữ liệu đó mà nhỏ hơn hoặc bằng r bắt buộc phải tìm cả nhánh bên trái và nhánh bên phải của *root*. Ví dụ trên hình 2, rõ ràng điểm truy vấn nằm bên phần không gian bên trái của nút gốc là $(30, 40)$, thông thường chúng ta chỉ tìm những điểm nằm trên nhánh trái của *root* sẽ bỏ qua những điểm dữ liệu thỏa mãn yêu cầu của hàm nhân $K(u) \neq 0$.

Trong nghiên cứu của Kakde [29] tác giả đã chứng minh rằng, độ phức tạp của thuật toán xây dựng cây là $O(n \log n)$, độ phức tạp thuật toán tìm kiếm một vùng không gian trên Kd-tree là $O(\sqrt{n} + c)$, trong đó c là số điểm ảnh được tìm thấy trong thuật toán 4. Khi đó, để ước lượng mật độ xác suất cho các điểm ảnh, công thức (2) trở thành:

$$\hat{f}(\mathbf{x}_i) = \frac{1}{n} \sum_{j=1}^{c_i} \left\{ \prod_{d=1}^{k} \frac{1}{h_d} K\left(\frac{x_i^d - p_{ij}^d}{h_d} \right) \right\}, \quad i = 1, 2, \ldots, n, \quad (5)$$

Thuật toán 5: Thuật toán tính PDF khi dữ liệu đầu vào là GRP

input: Ma trận điểm ảnh x, số điểm ảnh n, số kênh phổ k, băng thông h;
output: Mật độ xác suất của các điểm ảnh *pdf*;

$groups \leftarrow CreateGroup(\mathbf{X}, n, k)$;
$m \leftarrow |groups|$;
for $i \leftarrow 0$ **to** $m - 1$ **do**
  $sum_ker \leftarrow 0$;
  for $j \leftarrow 0$ **to** $m - 1$ **do**
    $sum_ker \leftarrow sum_ker + Kernel(groups_i, groups_j, k, h) \times |groups_j|$;
  end
  for $j \leftarrow 0$ **to** $|groups_i| - 1$ **do**
    $pdf[groups[i].indexes[j]] \leftarrow \frac{sum_ker}{n}$;
  end
end
return *pdf*;

trong đó c_i là tổng số điểm ảnh nằm trong hình siêu cầu bán kính r có tâm x_i đã tìm được trong thuật toán 4, p_{ij} là vector chứa giá trị của các kênh phổ của điểm ảnh thứ j trong danh sách các điểm ảnh được tìm thấy trong thuật toán này.

Với việc sắp xếp các nhóm điểm ảnh trên Kd-tree công thức ước lượng mật độ xác suất của các điểm ảnh được viết lại như sau:

$$\hat{f}_G(\mathbf{g}_i) = \frac{1}{n} \sum_{j=1}^{mc_i} \left\{ \left[\prod_{d=1}^{k} \frac{1}{h_d} K\left(\frac{g_i^d - p_{ij}^d}{h_d} \right) \right] \times |C_j^i| \right\}, \quad (6)$$

với $i = 1, 2, \ldots, m$, trong đó m là số nhóm các điểm ảnh có các giá trị phổ trùng nhau, mc_i là tổng số nhóm điểm ảnh nằm trong hình siêu cầu bán kính r và tâm g_i tìm được trong thuật toán 4 (gọi là danh sách thứ i), C_j^i là tập hợp các điểm ảnh trong nhóm thứ j của danh sách thứ i.

3. Tính toán PDF

Các phần III.1 và III.2 giải quyết khâu tiền xử lý dữ liệu, việc tiếp theo là tính toán PDF. Chúng ta có ba kiểu cấu trúc dữ liệu, bao gồm: cấu trúc dữ liệu mà từ dữ liệu ảnh gốc sau khi được nhóm, gọi tắt là GRP, cấu trúc dữ liệu sau khi dữ liệu ảnh gốc đã được tổ chức vào Kd-tree, gọi tắt là KDT, cấu trúc dữ liệu mà dữ liệu ảnh gốc sau khi được nhóm sẽ tiếp tục được tổ chức lại vào Kd-tree, gọi tắt là GGP-KDT.

Trong phần này, chúng tôi xây dựng các thuật toán khác nhau để giải bài toán ước lượng mật độ xác suất của các điểm ảnh khi dữ liệu đầu vào là các cấu trúc dữ liệu kể trên. Thuật toán đầu tiên (Thuật toán 5) tính toán PDF theo công thức (4) khi dữ liệu đầu vào là GRP. Độ phức tạp tính

Thuật toán 6: Thuật toán tính PDF khi dữ liệu đầu vào là KDT

input: Ma trận điểm ảnh x, số điểm ảnh n, số kênh phổ k, băng thông h
output: Mật độ xác suất của các điểm ảnh *pdf*

$root \leftarrow CreateKDTree(\mathbf{X}, n, k)$;
$r \leftarrow h \times \epsilon$;
for $i \leftarrow 0$ **to** $n-1$ **do**
  $sum_ker \leftarrow 0$;
  $list \leftarrow \varnothing$;
  $Search(root, x_i, r, list)$;
  for $j \leftarrow 0$ **to** $|list|-1$ **do**
    $sum_ker \leftarrow sum_ker + Kernel(x_i, list_j, h)$;
  end
  $pdf[i] \leftarrow \frac{sum_ker}{n}$;
end
return *pdf*;

toán của thuật toán 5 là $O(km^2)$, trong đó m là số nhóm các điểm ảnh. Hàm *Kernel* đã được trình bày trong phần II.

Thuật toán thứ hai (Thuật toán 6) tính toán PDF theo công thức (5) khi dữ liệu đầu vào là KDT. Cây Kd-tree đóng vai trò là cây tìm kiếm những nút thỏa mãn điều kiện để nhân $K(u) \neq 0$. Thuật toán này có độ phức tạp là $O(kn(\sqrt{n}+c_i))$, trong đó c_i là số điểm ảnh được tìm thấy trong Thuật toán 4, với $i = 1, 2, \ldots, n$.

Thuật toán thứ ba (Thuật toán 7) tính toán PDF theo công thức (6) với dữ liệu đầu vào là GRP-KDT, Kd-tree đóng vai trò là cây tìm kiếm để tìm các nhóm điểm ảnh thỏa mãn điều kiện của hạt nhân $K(u) \neq 0$. Độ phức tạp của thuật toán là $O(km(\sqrt{m}+cm_i))$, m là số nhóm của các điểm ảnh, mc_i là số nhóm điểm ảnh được tìm thấy trong thuật toán 4, với $i = 1, 2, \ldots, m$.

Thuật toán 7: Thuật toán tính PDF khi dữ liệu đầu vào là GRP-KDT

input: Ma trận điểm ảnh x, số điểm ảnh n, số kênh phổ k, băng thông h;
output: Mật độ xác suất của các điểm ảnh *pdf*;

$groups \leftarrow CreateGroup(x, n, k)$;
$m \leftarrow |groups|$;
$root \leftarrow CreateKDTree(groups, m, k)$;
$r \leftarrow h \times \epsilon$;
for $i \leftarrow 0$ **to** $m-1$ **do**
  $sum_ker \leftarrow 0$;
  $list \leftarrow \varnothing$;
  $Search(root, group_i, r, list)$;
  for $j \leftarrow 0$ **to** $|list|-1$ **do**
    $sum_ker \leftarrow sum_ker + Kernel(group_i, list_j, h)) \times |group[list[j].index]|$;
  end
  for $j \leftarrow 0$ **to** $|group_i|-1$ **do**
    $pdf\,[group[i].indexes[j]] \leftarrow \frac{sum_ker}{n}$;
  end
end
return *pdf*;

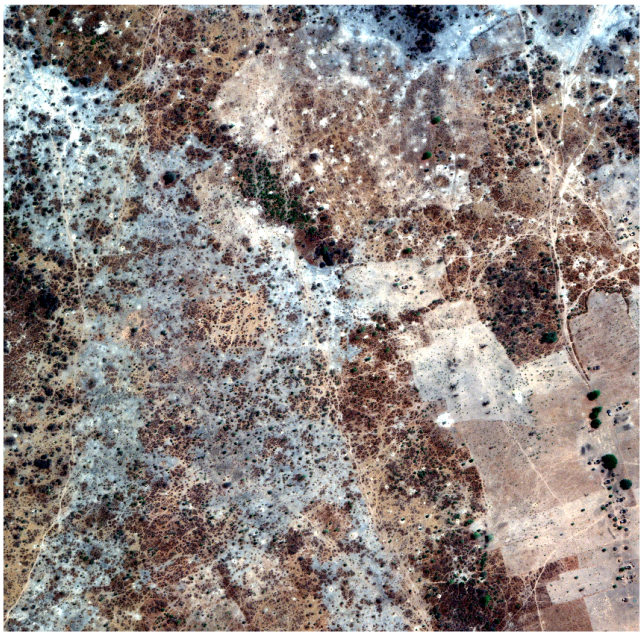
IV. THỰC NGHIỆM

1. Kịch bản thử nghiệm

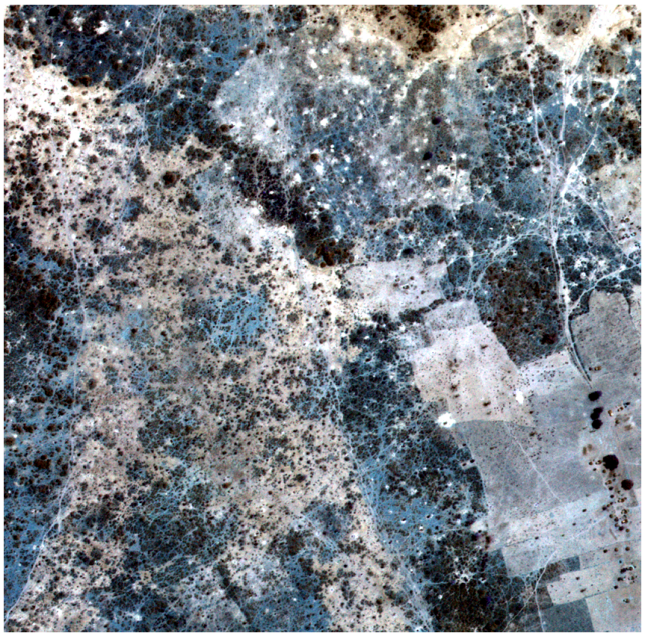
Chúng tôi thử nghiệm trên 3 loại ảnh khác nhau: ảnh màu có 3 kênh phổ (RGB), ảnh đa phổ có 8 kênh phổ, và ảnh siêu phổ có 224 kênh phổ. Tương ứng với mỗi một ảnh như vậy, chúng tôi chạy với thuật toán khi dữ liệu chưa qua giai đoạn tiền xử lý (Thuật toán 1). Sau đó, tương ứng với mỗi loại ảnh, chúng tôi chuyển dữ liệu ảnh đó qua giai đoạn tiền xử lý để có được dữ liệu theo ba cấu trúc GRP, KDT và GRP-KDT. Dữ liệu này sẽ chạy với các thuật toán 5, 6 và 7 tương ứng.

Ngoài ra, chúng tôi còn chạy thử nghiệm 3 ảnh trên với hai phương pháp: song song hóa việc ước lượng hàm mật độ trên khung lập trình Intel TBB (gọi tắt là Intel TBB), tốt nhất theo kết luận trong [22], và song song hóa tính toán PDF trên nên tảng GPU CUDA (gọi tắt là GPU CUDA), đề xuất trong công bố [23]. Việc này nhằm mục đích so sánh thời gian chạy và có cái nhìn khách quan hơn về phương pháp chúng tôi đề xuất. Lưu ý rằng phương pháp của chúng tôi đề xuất là giảm độ phức tạp tính toán dẫn đến giảm thời gian tính toán, trong khi Intel TBB và GPU CUDA không giảm độ phức tạp tính toán mà giảm thời gian tính toán bằng việc phân nhỏ tổng khối lượng công việc và tính toán đồng thời. Điểm giống nhau là các phương pháp này đều giảm thời gian tính toán mà không làm thay đổi kết quả tính toán PDF.

Trong thực tế, dữ liệu ảnh đa phổ hoặc siêu phổ thu chụp trong các tình huống tìm kiếm cứu nạn còn khan hiếm và cơ bản là không được phát hành công khai. Vì vậy, chúng tôi lựa chọn cách tiếp cận theo cách sử dụng các thư viện ảnh đã được công bố phù hợp với mục đích bài toán phát hiện dị thường trên ảnh. Đầu tiên, chúng tôi sử dụng ảnh 3 kênh phổ và ảnh 8 kênh phổ tại [32]. Đây là những ảnh được dùng trong cuộc thi "Dstl Satellite Imagery Feature Detection" do Phòng thí nghiệm khoa học và công nghệ quốc phòng (Dstl)- Vương quốc Anh cung cấp. Ảnh 3 kênh phổ có mã là 6010_1_2 (gọi tắt là ảnh 3 kênh phổ) và ảnh 8 kênh phổ có mã là 6010_1_2_M (gọi tắt là ảnh 8 kênh phổ). Hai ảnh này được thu từ bộ cảm biến WorldView 3 tại cùng một địa điểm trong phạm vi 1 km^2, kích thước (1 km $\times$ 1 km).



Hình 3. Ảnh 3 kênh phổ có mã là 6010_1_2.



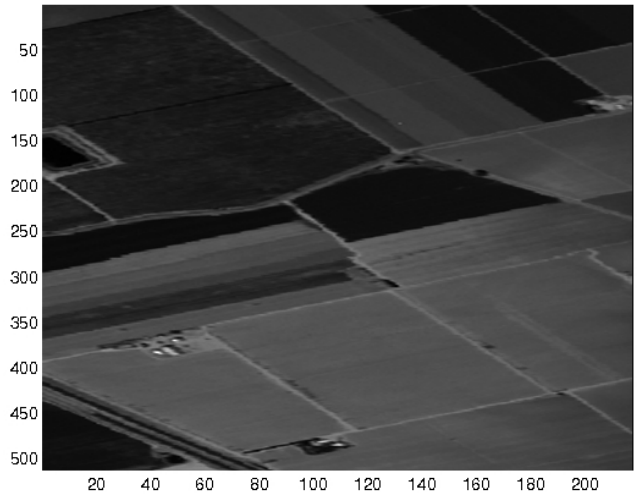
Hình 4. Ảnh hiển thị tổ hợp 3 kênh phổ (kênh 1, 2 và 3) từ ảnh 8 kênh phổ có mã là 6010_1_2_M.

Ảnh 3 kênh phổ (Hình 3) là ảnh màu RGB với độ phân giải mặt đất là $0,31$ m, kích thước ảnh là 3396×3349 điểm ảnh. Ảnh 8 kênh phổ (Hình 4) có độ phân giải mặt đất là $1,24$ m, kích thước ảnh là 849×837 điểm ảnh.

Hình 5 hiển thị những điểm ảnh dị thường (màu trắng là những công trình nhân tạo hỗn hợp trên thực địa) và những điểm ảnh không dị thường (màu đen) của hai ảnh 6010_1_2 và 6010_1_2_M.



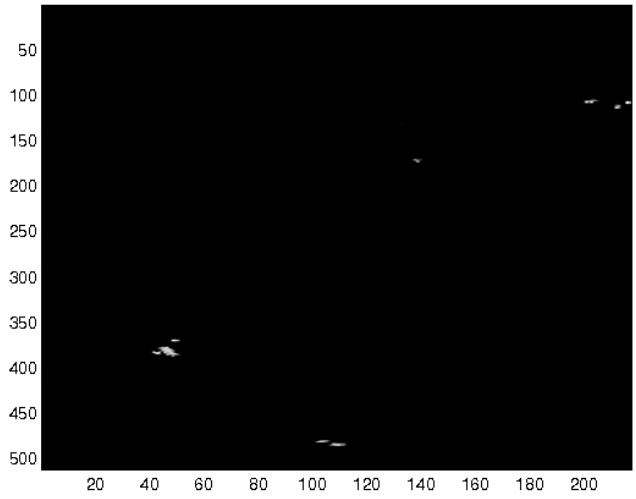
Hình 5. Những điểm ảnh dị thường (màu trắng).



Hình 6. Kênh thứ 220 của ảnh siêu phổ Salinas 224 kênh phổ.

Ảnh siêu phổ 224 kênh phổ là ảnh miễn phí được cung cấp tại [33] (Hình 6). Cảnh ảnh này được thu tại thung lũng Salinas, California bởi bộ cảm biến AVIRIS với 224 kênh phổ, độ phân giải không gian là $3,7$ m, kích thước ảnh là 512×217 điểm ảnh (gọi tắt là ảnh Salinas). Hình 7 thể hiện những điểm ảnh dị thường (màu trắng là những công trình nhân tạo được bao quanh bởi cánh đồng trồng các loại thực vật) và những điểm ảnh không dị thường (màu đen).

Bước tiếp theo, chúng tôi sử dụng hàm nhân Hypercube (Bảng I) với băng thông cố định $h = 10$ để kiểm nghiệm các thuật toán trên ba ảnh này. Cấu hình máy tính dùng để chạy các thuật toán tính PDF là như sau.



Hình 7. Những điểm ảnh dị thương (màu trắng).

Bảng II
THỜI GIAN CHẠY CỦA CÁC THUẬT TOÁN TRÊN BA ẢNH (TÍNH BẰNG GIÂY)

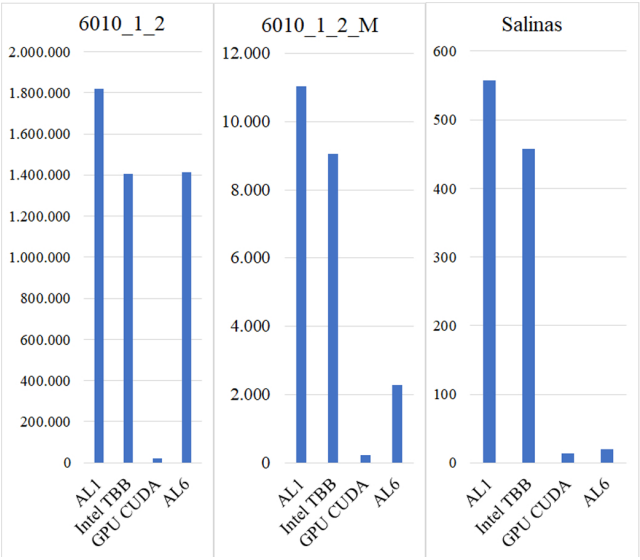
Thuật toán	6010_1_2	6010_1_2_M	Salinas
Thuật toán 1 (AL1)	1.819.712	11.038	557
Intel TBB	1.404.590	9.052	458
GPU CUDA	21.609	**253**	**13,36**
Thuật toán 6 (AL6)	1.414.826	2.269	20
Thuật toán 5 (AL5)	230	n/a	n/a
Thuật toán 7 (AL7)	**21,19**	n/a	n/a

- **CPU:** Intel Core i5-7400 3.00 GHz (4 core, 8 thread);
- **Mainboard:** MSI B150M MORTAR ARCTIC;
- **RAM:** DDR4 16 GB;
- **HDD:** SDD BIOSTAR S100 - 240 GB;
- **Graphic:** NVIDIA GeForce GTX 1070 Ti (2432 core, 1683 MHz, 8 GB RAM).

2. Đánh giá thời gian chạy của các thuật toán

Thời gian thực thi của các thuật toán trên cả ba ảnh như đã mô tả trong Phần IV.1 được thể hiện trên Bảng II. Thuật toán Intel TBB thực thi trên cả 4 nhân và 8 luồng của CPU, CPU chạy hết công suất 100%. Thuật toán GPU CUDA thực thi trên card màn hình NVIDIA GeForce GTX 1070 Ti, 2432 nhân CUDA, 8GB RAM, tốc độ xử lý 1683 MHz. Các thuật toán AL1, AL5, AL6 và AL7 chạy trên một nhân và một luồng của CPU.

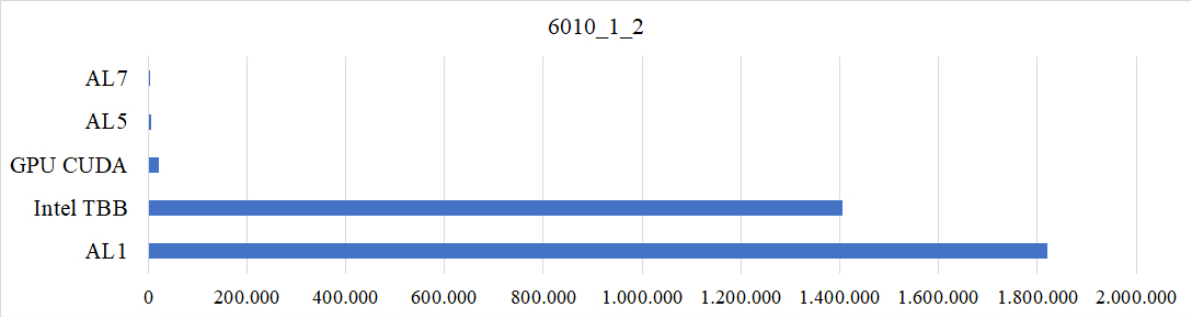
Trong trường hợp dữ liệu đầu vào của các thuật toán là KDT, kết quả được thể hiện trên hình 8. Rõ ràng rằng, khi dữ liệu được tổ chức vào cây Kd-tree, thời gian tính toán PDF đã giảm đi đáng kể so với trường hợp tính toán PDF khi dữ liệu chưa qua giai đoạn tiền xử lý (AL1). Cụ thể: trên ảnh 3 kênh phổ thời gian đã giảm đi 404.886s tương đương với việc giảm đi 22,25% thời gian tính toán; trên



Hình 8. Biểu đồ hiển thị thời gian chạy của thuật toán tính toán PDF khi dữ liệu chưa qua giai đoạn tiền xử lý (AL1), tính toán song song trên CPU (Intel TBB), tính toán song song trên GPU (GPU CUDA) và dữ liệu đã được tổ chức vào cây Kd-tree (AL6).

ảnh 8 kênh phổ, thời gian đã giảm đi 8.769s tương đương với việc giảm đi 79,44% thời gian tính toán; trên ảnh 224 kênh phổ, thời gian tính toán đã giảm đi 537s tương đương với việc giảm đi 96,41% thời gian tính toán. Trên ảnh ba kênh phổ, thuật toán AL6 có thời gian tính toán lớn hơn thuật toán Intel TB 10.236s, tương đương với thời gian tính toán của LA6 nhiều hơn Intel TBB 0,73%. Trên hai ảnh 8 kênh phổ và 224 kênh phổ thời gian tính toán của AL6 đã nhanh hơn Intel TBB lần lượt là: 6.783s và 438s, tương đương với việc giảm đi 74,93% và 95,63% thời gian tính toán. Trong trường hợp so sánh về thời gian tính toán giữa AL6 và GPU CUDA thì AL6 đều có thời gian tính chậm hơn GPU CUDA trên cả ba ảnh. Tuy nhiên có sự khác biệt, AL6 tính toán chậm hơn GPU CUDA rõ rệt trên ảnh ba kênh phổ, trên ảnh 8 kênh phổ khoảng cách đã được thu hẹp, trên ảnh 224 kênh phổ chênh lệch không nhiều (AL6 chỉ chậm hơn GPU CUDA 6,64s). Như vậy, áp dụng cây Kd-tree để quản lý dữ liệu trong giai đoạn tiền xử lý trước khi tính toán PDF sẽ hiệu quả hơn đối với những ảnh có số kênh phổ lớn.

Đối với trường hợp nhóm các điểm ảnh có phổ trùng nhau, chúng tôi chỉ áp dụng cho ảnh 3 kênh phổ. Không áp dụng cho loại ảnh 8 kênh phổ và 224 kênh phổ bởi tổ hợp màu của loại ảnh 8 hoặc 224 kênh phổ là một số rất lớn vượt khỏi khả năng quản lý của máy tính và không thể áp dụng được cho thuật toán 2. Nếu sử dụng thuật toán nhóm thông thường sẽ tốn rất nhiều thời gian do đó không khả thi. Nhìn vào bảng II và hình 9 ta thấy, thời gian tính toán PDF của ảnh 3 kênh phổ khi dữ liệu đã được



Hình 9. Biểu đồ hiển thị thời gian chạy của thuật toán tính toán PDF khi dữ liệu chưa qua giai đoạn tiền xử lý (AL1), tính toán song song trên CPU (Intel TBB), tính toán song song trên GPU (GPU CUDA), dữ liệu đã được nhóm (AL5) và dữ liệu đã được nhóm sau đó tổ chức trên cây Kd-tree (AL7).

nhóm đã giảm tới 1.819.482s tương đương với việc giảm đi 99,98% thời gian tính toán so với trường hợp tính toán PDF khi dữ liệu chưa qua giai đoạn tiền xử lý (AL1). Lý do giảm thời gian tính toán PDF là quá trình nhóm các điểm ảnh có phổ trùng nhau đã làm giảm dữ liệu cần tính toán. Cụ thể, việc này đã giảm số lượng dữ liệu cần tính toán từ $3396 \times 3349 = 11.373.204$ điểm ảnh xuống còn 65.607 nhóm điểm ảnh, dẫn đến giảm được 99,4% khối lượng dữ liệu cần tính toán. Thời gian tính toán của thuật toán AL5 nhanh hơn thuật toán Intel TBB 6.107 lần, nhanh hơn thuật toán GPU CUDA 94 lần.

Dữ liệu sau khi nhóm tiếp tục đưa vào cây Kd-tree để quản lý, thời gian tính toán PDF trên ảnh 3 kênh phổ chỉ còn là 21,19s. So sánh với trường hợp tính toán PDF khi dữ liệu chưa qua giai đoạn tiền xử lý (thời gian tính toán là 1.819.712s, tương đương với hơn 21 ngày tính toán) dữ liệu GRP-KDT đưa vào tính toán PDF này đã giảm đi 99,999% thời gian tính toán. Nhìn vào bảng II và hình 9 ta thấy AL7 hoàn toàn vượt trội so với thuật toán Intel TBB và GPU CUDA, cụ thể, AL7 đã nhanh hơn Intel TBB 66.285 lần và nhanh hơn GPU CUDA 1.020 lần.

3. Đánh giá về độ phức tạp tính toán

Độ phức tạp tính toán của thuật toán AL1 là $O(kn^2)$, của thuật toán AL5 là $O(km^2)$, của thuật toán AL6 là $O(kn(\sqrt{n}+c_i))$ và của thuật toán AL7 là $O(km(\sqrt{m}+mc_i))$. Rõ ràng độ phức tạp tính toán của thuật toán AL6 và AL7 luôn luôn nhỏ hơn thuật toán AL1. Đối với thuật toán AL5, trong trường hợp xấu nhất $m = n$ (không có bất kỳ điểm ảnh nào có phổ trùng nhau) thì hai thuật toán này có độ phức tạp tính toán tương đương nhau. Tuy nhiên, điều này rất khó xảy ra ngoài thực tế bởi các ảnh chụp trong tự nhiên các lớp phủ thực địa luôn có tính chất phân lớp đối tượng, lớp phủ càng đồng nhất (chụp trên biển, trên rừng, hoang mạc, sa mạc, v.v.) thì m càng nhỏ dẫn đến độ phức tạp tính toán của AL5 nhỏ hơn AL1.

Đối với phương pháp biến đổi Fourier nhanh do Silverman để xuất [19], độ phức tạp tính toán là $O(N \log N)$, phương pháp biến đổi Gauss nhanh do Elgamall và các cộng sự đề xuất [20] có độ phức tạp tính toán là $O(N+M)$, trong đó $N = kn$ là kích thước dữ liệu và M là số lượng điểm dữ liệu cần tính PDF. Ứng dụng trong trường hợp tính toán PDF cho các điểm ảnh thì $M = N$. Rõ ràng thuật toán AL6 có độ phức tạp tính toán lớn hơn cả hai phương pháp biến đổi nhanh này do $O(kn(\sqrt{n} + c_i)) > O(N \log N) > O(N + M)$. Cả hai phương pháp biến đổi Fourier nhanh và biến đổi Gauss nhanh có độ phức tạp tính toán phụ thuộc hoàn toàn vào kích thước dữ liệu, thuật toán AL5 và AL7 có độ phức tạp tính toán phụ thuộc vào cấu trúc, nội dung của ảnh. Trong phần thực nghiệm với ảnh ba kênh phổ ở trên cho ta thấy, trong khi $n = 11.373.204$ thì m chỉ bằng 65.607, có nghĩa là $m \approx 0.006n$, dẫn đến độ phức tạp tính toán của AL5 và AL7 nhỏ hơn rất nhiều so với phương pháp biến đổi Fourier nhanh và biến đổi Gauss nhanh.

V. KẾT LUẬN

Trong công tác tìm kiếm cứu nạn, thời gian phản ứng mang ý nghĩa hết sức quan trọng. Việc rút ngắn thời gian xử lý dữ liệu và ra quyết định đồng nghĩa với việc giảm phí tổn tài chính, sức lực, tinh thần và nâng cao khả năng sống sót của nạn nhân. Trong nghiên cứu này, chúng tôi đề xuất phương pháp để làm giảm thời gian phát hiện các điểm dị thường trên ảnh siêu phổ và đa phổ (những điểm ảnh dị thường này có thể là mục tiêu cần tìm kiếm hoặc những dấu hiệu phục vụ cho công tác tìm kiếm cứu nạn). Đầu tiên là giai đoạn tiền xử lý dữ liệu với mục đích làm giảm số lượng dữ liệu cần tính toán (sử dụng nhóm các điểm ảnh có các kênh phổ trùng nhau) và tổ chức lại dữ liệu một cách hợp lý để phục vụ cho quá trình tính toán PDF (tổ chức dữ liệu vào cây Kd-tree). Sau đó, 3 cấu trúc dữ liệu trong giai đoạn tiền xử lý GRP, KDT và GRP-KDT được sử dụng để tính toán PDF.

Độ phức tạp về tính toán trong giai đoạn tiền xử lý dữ liệu khi nhóm những điểm ảnh có các kênh phổ trùng nhau là $O(kn)$, xây dựng Kd-tree là $O(n \log n)$. Trong giai đoạn tính toán PDF, độ phức tạp tính toán của các hàm PDF khi dữ liệu đã được nhóm là $O(km^2)$, trong đó m là số nhóm các điểm ảnh có các kênh phổ trùng nhau. Trong rất nhiều trường hợp, m nhỏ hơn rất nhiều so với n nên thời gian tính toán PDF cũng giảm đi tương ứng. Độ phức tạp tính toán khi dữ liệu đầu vào đã được quản lý bởi Kd-tree là $O(kn(\sqrt{n} + c_i))$, do c_i nhỏ hơn n rất nhiều nên thời gian tính các hàm PDF cũng giảm đi tương ứng. Độ phức tạp tính toán các hàm PDF khi dữ liệu đã được nhóm và tổ chức trên cây Kd-tree là $O(km(\sqrt{m} + mc_i))$.

Kết quả kiểm nghiệm trên ba loại ảnh (ảnh đa phổ 3 kênh, ảnh đa phổ 8 kênh và ảnh siêu phổ 224 kênh) đã vượt ngoài mong đợi của nhóm tác giả. Đặc biệt là đối với ảnh màu, đây là những ảnh thường được thu chụp từ thiết bị bay không người lái hoặc có người lái và được ứng dụng rộng rãi trong công tác tìm kiếm cứu nạn.

LỜI CẢM ƠN

Nghiên cứu này được tài trợ kinh phí bởi đề tài nghiên cứu khoa học cấp quốc gia mã số VT-UD.04/16-20 thuộc Chương trình KHCN vũ trụ của Bộ khoa học và công nghệ Việt Nam. Nhóm tác giả trân trọng cảm ơn sự ủng hộ và đồng hành của Ban chủ nhiệm Chương trình KHCN vũ trụ.

TÀI LIỆU THAM KHẢO

[1] T. Bolukbasi, P. Tran, "Outline Color Identification For Search And Rescue," Technical Report of Department of Electrical and Computer Engineering, Boston University, no. ECE-2012-07, 2012.

[2] M. B. Salem, K. S. Ettabaa, M. A. Hamdi, "Anomaly detection in hyperspectral imagery: An overview," in *International Image Processing, Applications and Systems Conference*, pp. 1–6, 2015.

[3] I. S. Reed and X. Yu, "Adaptive Multiple-Band CFAR Detection of an Optical Pattern with Unknown Spectral Distribution," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 38, no. 10, pp. 1760–1770, 1990.

[4] T. E. Smetek, K. W. Bauer, "Finding hyperspectral anomalies using multivariate outlier detection," *IEEE Aerospace Conference*, pp. 1–24, 2007.

[5] D. Manolakis, D. Marden, G. A. Shaw, "Hyperspectral image processing for automatic target detection applications," *Lincoln Laboratory Jour.*, vol. 14, no. 1, pp. 79–116, 2003.

[6] D. C. Borghys, V. Achard, S. R. Rotman, N. Gorelik, C. Perneel, E. Scwheicher, "Hyperspectral anomaly detection: a comparative evaluation of methods," *XXXth URSI General Assembly and Scientific Symp.*, pp. 1–4, 2011.

[7] T. Marshall, L. N. Perkins, "Color Outline Detection For Search And Rescue," Technical Report of Department of Electrical and Computer Engineering, Boston University, no. ECE-2015-01, 2015.

[8] M. Ramachandran, W. Moik, *Outline Color Identification For Search And Rescue*, Technical Report of Department of Electrical and Computer Engineering, Boston University, No. ECE-2013-03, 2013.

[9] D. K. Hoai, N. V. Phuong, "Anomaly Color Detection on UAV Images for Search and Rescue works," in *2017 9th International Conference on Knowledge and Systems Engineering*, pp. 287–291, 2017.

[10] S. Khazai, S. Homayouni, A. Safari, and B. Mojaradi, "Anomaly Detection in Hyperspectral Images Based on an Adaptive Support Vector Method," *IEEE Geoscience and Remote Sensing Letters*, vol. 8, no. 4, pp. 646–650, 2011.

[11] A. Banerjee, P. Burlina, and C. Diehl, "A support vector method for anomaly detection in hyperspectral imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 44, no. 8, pp. 2282–2291, Aug. 2006.

[12] D. W. J. Stein, S. G. Beaven, L. E. Ho, E. M. Winter, A. P. Schaum, and A. D. Stocker, "Anomaly detection from hyperspectral imagery," *IEEE Signal Process. Mag.*, vol. 19, no. 1, pp. 58–69, 2002.

[13] S. Matteoli, T. Veracini, M. Diani, and G. Corsini, "Models and Methods for Automated Background Density Estimation in Hyperspectral Anomaly Detection," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 51, no. 5, pp. 2837–2852, 2013.

[14] P. Gurram and H. Kwon, "Support-Vector-Based Hyperspectral Anomaly Detection Using Optimized Kernel Parameters," *IEEE Geoscience and Remote Sensing Letters*, vol. 8, pp. 1060–1064, 2011.

[15] C.-I. Chang and S.-S. Chiang, "Anomaly detection and classification for hyperspectral imagery," *IEEE Trans. Geosci. Remote Sensing*, vol. 40, no. 6, pp. 1314–1325, 2002.

[16] T. Veracini, S. Matteoli, M. Diani, and G. Corsini, "Nonparametric Framework for Detecting Spectral Anomalies in Hyperspectral Images," *IEEE Geoscience and Remote Sensing Letters*, vol. 8, no. 4, pp. 666–670, 2011.

[17] S. Matteoli, T. Veracini, M. Diani and G. Corsini, "Background Density Nonparametric Estimation With Data-Adaptive Bandwidths for the Detection of Anomalies in Multi-Hyperspectral Imagery," *IEEE Geoscience and Remote Sensing Letters*, vol. 11, pp. 163–167, 2014.

[18] C. Zhao, X. Wang, and G. Zhao, "Detection of hyperspectral anomalies using density estimation and collaborative representation," *Remote Sensing Letters*, vol. 8, no. 11, pp. 1025–1033, 2017.

[19] B. Silverman, "Algorithm AS 176: Kernel density estimation using the fast Fourier transform," *Applied Statistics*, vol. 31, no. 1, pp. 93–99, 1982.

[20] A. Elgammal, R. Duraiswami and L.S. Davis, "Efficient Kernel density estimation using the Fast Gauss Transform with applications to color modeling and tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, pp. 1499–1504, 2003.

[21] S. Lukasik, "Parallel computing of kernel density estimates with MPI," in *7th International Conference on Computational Science*, pp. 726–733, 2007.

[22] P. D. Michailidis, and K. G. Margaritis, "Parallel Computing of Kernel Density Estimation with Different Multi-core Programming Models," in *21st Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*, pp. 77–85, 2013.

[23] P. D. Michailidis, K. G. Margaritis, "Accelerating Kernel Density Estimation on the GPU Using the CUDA Framework," *Applied Mathematical Sciences*, vol. 7, no. 30, pp. 1447–1476, 2013.

[24] M. Rosenblatt, "Remarks on Some Nonparametric Estimates of a Density Function," *Annals of Mathematical Statistics*, vol. 27, no. 3, pp. 832–837, 1956.

[25] E. Parzen, "On Estimation of a Probability Density Function and Mode," *Annals of Mathematical Statistics*, vol. 33, pp. 1065–1076, 1962.

[26] L. Devroye and L. Gyorfi, *Nonparametric Density Estimation: The L1 View*, Wiley, New York, 1985.
[27] W. Hardle, A. Werwatz, M. Muller and S. Sperlich, *Nonparametric Density Estimation, In: Nonparametric and Semiparametric Models*, Springer Series in Statistics, pp. 39-83, 2004.
[28] J. L. Bentley, "Multidimensional Binary Search Trees Used for Associative Searching," *Communications of the ACM*, vol. 18, no. 9, pp. 509-517, 1975.
[29] H. M. Kakde, "Range Searching using Kd Tree," 2005. References, Aug. 12, 2019. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.122.5818.
[30] P. Trebuňa, J. Halčinová, "Experimental Modelling of the Cluster Analysis Processes," *Procedia Engineering 48*, pp. 673–678, 2012.
[31] M. Harris, "Optimizing parallel reduction in CUDA," *Nvidia developer technology 2*, no. 4, p. 70, 2007. [Online]. Available: https://developer.download.nvidia.com/assets/cuda/files/reduction.pdf.
[32] Dstl Satellite Imagery Feature Detection. [Online]. Available: https://www.kaggle.com/c/dstl-satellite-imagery-feature-detection. [Accessed: Oct 25, 2019].
[33] Hyperspectral Remote Sensing Scenes. [Online]. Available: http://www.ehu.eus/ccwintco/index.php?title=Hyperspectral_Remote_Sensing_Scenes. [Accessed: Oct 25, 2019].



Nguyễn Văn Phương tốt nghiệp Đại học và Thạc sĩ tại Học viện Kỹ thuật Quân sự năm 2003 và 2009. Hiện tại là nghiên cứu sinh tại Khoa Công nghệ Thông tin cũng tại Học viện Kỹ thuật Quân sự. Lĩnh vực nghiên cứu bao gồm: GIS, xử lý ảnh viễn thám quang học.



Đào Khánh Hoài nhận học vị Tiến sĩ năm 2005. Hiện công tác tại Học viện Kỹ thuật Quân sự. Lĩnh vực nghiên cứu bao gồm: GIS, xử lý ảnh vệ tinh, UAV, đo ảnh và thị giác máy tính.



Tống Minh Đức tốt nghiệp Đại học tại Học viện Kỹ thuật Quân sự năm 2000, nhận học vị Tiến sĩ tại Trường Đại học Tổng hợp Kỹ thuật Điện (LETI), Nga năm 2007. Hiện là giảng viên tại Khoa Công nghệ Thông tin của Học viện Kỹ thuật Quân sự. Lĩnh vực nghiên cứu bao gồm: xử lý ảnh, nhận dạng đối tượng, an toàn bảo mật thông tin.

Song song hóa thuật toán Lamport trong loại trừ tương hỗ phân tán

Đặng Hùng Vĩ[1], Lê Văn Sơn[1], Nguyễn Xuân Huy[2]
[1] Trường Đại học Sư phạm, Đại học Đà Nẵng
[2] Viện Công nghệ Thông tin, Viện Hàn lâm Khoa học và Công nghệ Việt Nam

Tác giả liên hệ: Đặng Hùng Vĩ, dhungvi@ued.udn.vn
Ngày nhận bài: 05/04/2019, ngày sửa chữa: 04/12/2019, ngày duyệt đăng: 04/12/2019
Định danh DOI: 10.32913/mic-ict-research-vn.v2019.n2.848
Biên tập lĩnh vực điều phối phản biện và quyết định nhận đăng: PGS.TS. Trần Minh Quang

Tóm tắt: Hệ phân tán là hệ thống cung cấp tài nguyên dùng chung với quy mô lớn. Hệ phân tán sử dụng cơ chế truyền thông điệp để hợp lực trong môi trường truyền thông. Trong hợp lực, nhiều tiến trình cùng tương tranh tài nguyên dùng chung dễ dẫn đến bế tắc trong cung cấp tài nguyên. Loại trừ tương hỗ phân tán cho phép chỉ có một tiến trình duy nhất được thực thi trong miền găng tại một thời điểm đối với một tài nguyên để giải quyết bế tắc. Để đạt được loại trừ tương hỗ phân tán, các tiến trình phải được gắn dấu đồng hồ lô-gic để xác lập trật tự và loại trừ các tiến trình gây ra bế tắc. Bài báo trình bày giải pháp song song hóa thuật toán Lamport trong loại trừ tương hỗ phân tán. Kết quả giải pháp là xác lập giá trị đồng hồ lô-gic dựa trên song song hóa thuật toán Lamport và xác định các tiến trình thực thi trong đảm bảo tính nhất quán và gắn bó trong hệ phân tán.

Từ khóa: *Hệ phân tán, đồng hồ lô-gic, thuật toán Lamport, loại trừ tương hỗ phân tán, truyền thông điệp.*

Title: **A Parallelization of the Lamport Algorithm for Distributed Mutual Exclusion**

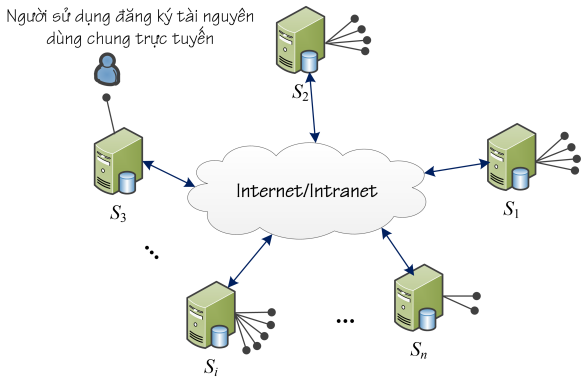
Abstract: A distributed system is a complex system in which the shared resources are allocated at a large scale. Such a system uses the message passing mechanism over the communication environment to coordinate the system's entities. During coordination, multiple concurrent processes might request the same resources, leading to deadlock in resource allocation. In order to resolve this deadlock, distributed mutual exclusion allows only one process to be executed in the critical section at a time for each shared resource. To this end, each process is assigned a timestamp to establish an order, and the processes that cause deadlock are eliminated. There are several proposed distributed mutual exclusion algorithms such as by Lamport, Ricart–Agrawala, Raymond, and Suzuki–Kasami. In this paper, we develop a parallelization of Lamport algorithm for distributed mutual exclusion. Our solution establishes a global state and determines the implementation process in the critical section to ensure consistency and coherence in discrete systems.

Keywords: *Distributed system, logical clock, Lamport algorithm, mutual exclusion distributed, message passing.*

I. GIỚI THIỆU

Hiện nay, các ứng dụng lớn trên môi trường điện toán đám mây được triển khai trong hệ phân tán để đáp ứng số lượng người dùng cực đại. Theo nghiên cứu trong [1], đám mây là một hệ thống song song và hệ phân tán bao gồm một tập hợp các máy chủ được kết nối và ảo hóa, được cung cấp động và được xử lý dưới dạng một hoặc nhiều tài nguyên tính toán hợp nhất dựa trên các thỏa thuận cấp dịch vụ được thiết lập thông qua thỏa thuận giữa nhà cung cấp dịch vụ và người dùng. Điện toán đám mây là một giải pháp toàn diện cung cấp hạ tầng, dịch vụ công nghệ thông tin. Đây là một giải pháp điện toán dựa trên internet cung cấp tài nguyên dùng chung thông qua hệ phân tán.

Hệ phân tán, được biểu diễn trên hình 1, là một tập hợp các máy chủ kết nối qua môi trường truyền thông trong cung cấp tài nguyên dùng chung. Nếu xét hoạt động mỗi máy chủ một cách độc lập, không có sự phối hợp để chia sẻ tài nguyên dùng chung thì đây là hệ tập trung. Nếu xét các máy chủ hợp lực để chia sẻ tài nguyên dùng chung thì đây là hệ phân tán. Sự hợp lực các máy chủ là sự phối hợp giữa các máy chủ với nhau thông qua môi trường truyền thông để cung cấp tài nguyên dùng chung cho người sử dụng. Khác biệt giữa hệ tập trung và hệ phân tán là các đặc tính như: tính gắn bó, khả năng chịu lỗi, sự mở rộng, cân bằng tải, v.v. Các nghiên cứu, triển khai hệ phân tán để cung cấp tài nguyên dùng chung tập trung vào giải pháp đảm bảo gắn bó dữ liệu [2]. Giải pháp gắn bó dựa trên



Hình 1. Mô hình kết nối trong hệ phân tán cung cấp tài nguyên dùng chung.

sự hợp lực của các máy chủ trong hệ phân tán thông qua cơ chế truyền thông điệp [3]. Hệ phân tán không có đồng hồ dùng chung, do đó, giải pháp của bài báo cải tiến thuật toán Lamport để giải quyết loại trừ tương hỗ phân tán trong cung cấp tài nguyên dùng chung.

Nội dung chính của bài báo được tổ chức như sau. Phần II trình bày các nghiên cứu liên quan. Phần III đề xuất giải pháp song song hóa thuật toán Lamport trong loại trừ tương hỗ phân tán. Phần IV trình bày hiệu năng thực thi song song hóa thuật toán Lamport. Phần V đưa ra kết luận.

Trong toàn bộ bài báo, chúng tôi ký hiệu N là số máy chủ, T là độ trễ của quá trình đồng bộ hóa và E là thời gian thực thi miền găng.

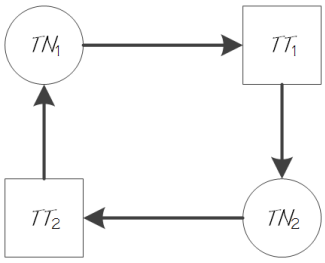
II. CÁC NGHIÊN CỨU LIÊN QUAN

Các nghiên cứu trong [4, 5] trình bày về truyền thông trong hệ phân tán đề cập đến cơ chế truyền multicast. Trong cơ chế này, gói tin vào và ra một máy chủ không tuân thủ nguyên tắc về lưu lượng như truyền unicast. Truyền multicast là sự kết hợp đặc biệt từ các máy chủ kết nối với máy chủ phát thông tin truyền.

Cơ chế hợp lực sử dụng truyền multicast cho phép các máy chủ kết nối với nhau thông qua môi trường truyền thông truyền thông điệp qua lại nhằm xác định các tiến trình di chuyển trong hệ phân tán [3]. Trong quá trình hợp lực, nhiều tiến trình cùng tương tranh tài nguyên dùng chung dễ dẫn đến bế tắc trong cung cấp tài nguyên [6–9].

Theo nghiên cứu của Singhal trong [10], quá trình bế tắc diễn ra khi hai hay nhiều tiến trình chiếm giữ tài nguyên dùng chung được giới hạn và đồng thời tiếp tục phát đi yêu cầu tài nguyên khác đang bị chiếm giữ. Các quá trình này tạo ra một vòng tròn khép kín làm cho các tiến trình kẹt chéo lẫn nhau dẫn đến bế tắc trong cung cấp tài nguyên theo mô tả trong hình 2.

Nhiều giải pháp xử lý phân tán liên quan đến việc chia sẻ tài nguyên dùng chung giữa các tiến trình khác nhau



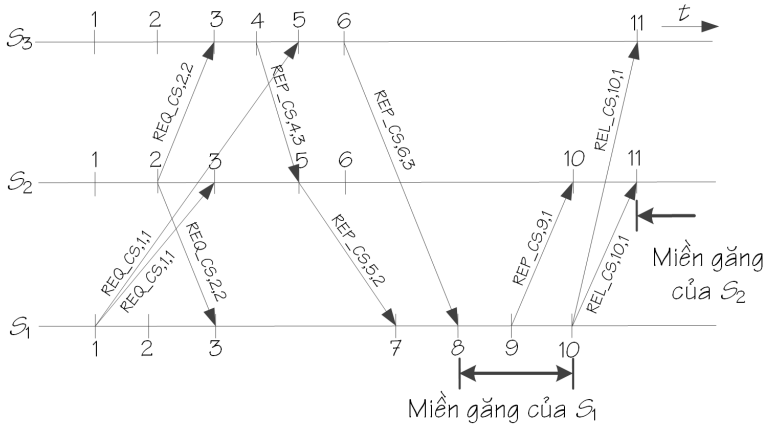
Hình 2. Đồ thị cung cấp tài nguyên.

đòi hỏi tài nguyên phải được cung cấp duy nhất cho một tiến trình tại một thời điểm. Do đó, loại trừ tương hỗ là giải pháp trong các hệ phân tán cung cấp tài nguyên dùng chung [3, 4, 11]. Giải pháp loại trừ tương hỗ được giải quyết dựa trên đồng bộ hóa tiến trình truy cập vào các tài nguyên dùng chung để đảm bảo tính nhất quán và gắn bó trong hệ phân tán. Quá trình đồng bộ hóa bằng cách truyền thông điệp giữa các máy chủ dựa vào môi trường truyền thông. Loại trừ tương hỗ phân tán tuân thủ các yêu cầu sau:

1) Cho phép một tiến trình duy nhất được thực thi trong miền găng tại một thời điểm đối với một tài nguyên;
2) Nếu không có tiến trình nào trong miền găng, tiến trình yêu cầu vào miền găng phải được phép vào và thực thi trong khoảng thời gian cho phép;
3) Khi có nhiều tiến trình yêu cầu vào miền găng, việc cho phép có thể bị trì hoãn đến khi được cấp phép;
4) Tiến trình xử lý trong miền găng trong không bị chặn bởi các tiến trình khác.

Theo thuật toán loại trừ tương hỗ phân tán, Kshemkalyani và Singhal trình bày quá trình một máy chủ vào và ra khỏi miền găng, được mô tả như trong hình 3 [4, Mục 9.3, 9.4]. Thông điệp có cấu trúc và chứa một trong ba giá trị: *REQ-CS* (yêu cầu vào miền găng), *REP-CS* (phản hồi chấp nhận của máy chủ cho phép vào miền găng), và *REL-CS* (giải phóng khỏi miền găng). Một máy chủ được phép vào miền găng khi máy chủ đó tiếp nhận đủ thông điệp *REP-CS* sau thông điệp *REQ-CS*. Máy chủ S_1 phát thông điệp yêu cầu *REQ-CS* vào miền găng tại thời điểm 1. Đến thời điểm 8, S_1 nhận đầy đủ thông điệp phản hồi *REP-CS* chấp nhận và vào miền găng cho đến thời điểm 10 phát thông điệp *REL-CS* rời khỏi miền găng sau khi xử lý xong. Trật tự từng phần trên các máy chủ thể hiện qua bảng I.

Xét truyền thông điệp theo hình 3, nếu một máy chủ bị sự cố trong quá trình truyền thông điệp, ví dụ đối với trường hợp truyền thông điệp *REP-CS* từ máy chủ S_2 đến máy chủ S_1 tại thời điểm 6, S_1 phải chờ đợi vào miền găng với khoảng thời gian không xác định. Ngoài ra, thông điệp trong quá trình truyền có thể bị phân mảnh, thất lạc, nghẽn, v.v. Các vấn đề này dẫn đến hiệu năng của hệ phân tán giảm trong quá trình hợp lực. Cụ thể, [4] trình bày hiệu



Hình 3. Quá trình máy chủ vào và ra miền găng [4, Mục 9.3, 9.4].

Bảng I
HOẠT ĐỘNG DIỄN RA TRÊN CÁC MÁY CHỦ TRONG TRẬT TỰ TỪNG PHẦN

Đồng hồ	Máy chủ 1	Máy chủ 2	Máy chủ 3
1	$S_1 \rightarrow S_2$: REQ-CS,1,1 $S_1 \rightarrow S_3$: REQ-CS,1,1		
2		$S_2 \rightarrow S_1$: REQ-CS,2,2 $S_2 \rightarrow S_3$: REQ-CS,2,2	
3	$S_2 \rightarrow S_1$: REQ-CS,2,2	$S_1 \rightarrow S_2$: REQ-CS,1,1	$S_2 \rightarrow S_3$: REQ-CS,2,2
4			$S_3 \rightarrow S_2$: REP-CS,4,3
5		$S_3 \rightarrow S_2$: REP-CS,4,3	$S_1 \rightarrow S_3$: REQ-CS,1,1
6			$S_3 \rightarrow S_1$: REP-CS,6,3
7	$S_2 \rightarrow S_1$: REP-CS,5,2		
8	**Máy chủ S_1 vào miền găng**		
9	$S_1 \rightarrow S_2$: REP-CS,9,1		
10	$S_1 \rightarrow S_2$: REL-CS,10,1 $S_1 \rightarrow S_3$: REL-CS,10,1	$S_1 \rightarrow S_2$: REP-CS,9,1	
11	**Máy chủ S_2 vào miền găng**		

năng loại trừ tương hỗ được xác định dựa trên các tham số: độ phức tạp thông điệp, độ trễ quá trình đồng bộ hóa, thời gian hồi đáp và thông lượng hệ thống. Bên cạnh đó, hiệu năng loại trừ tương hỗ dựa trên điều kiện của tải trong hệ thống. Trong đó tải được xác định bởi tỷ lệ thông điệp đến yêu cầu thực thi miền găng. Đối với tải thấp, số lượng tiến trình phải đợi chờ để vào miền găng là rất thấp. Đối với tải cao, luôn có tiến trình yêu cầu thực thi miền găng phải chờ trong hàng đợi.

Có hai nhóm giải pháp chính trong loại trừ tương hỗ. Nhóm thứ nhất là phương pháp tiếp cận dựa trên token: Ricart-Agrawala [12], Suzuki-Kasami [13], Mizuno-Neilsen-Rao [14], Neilsen-Mizuno [14], Helary-Plouzeau-Raynal [15], Raymond [16], Singhal [17], Naimi-Trehel [18], Mishra-Srimani [19], và Nishio [20]. Nhóm thứ hai là phương pháp tiếp cận dựa trên quyền: Lamport [21], Ricart-Agrawala [22], Carvalho-Roucairol [23], Raynal [24], Maekawa [25], Sanders [26], Agrawal-El Abbadi [27], và Singhal [28]. Hiệu năng của các thuật toán trong nhóm thứ hai, tính theo số thông điệp cần được truyền, được khảo sát bởi Velazquez [29] và tóm tắt trong bảng II.

Một so sánh khác được trình bày bởi Yadav và cộng sự trong [30]. Trong đó, hai giải pháp tiếp cận cho loại trừ tương hỗ phân tán là giải pháp dựa trên nội dung và giải pháp dựa trên điều khiển. Giải pháp dựa trên nội dung sử dụng thuật toán trật tự nhãn thời gian lô-gic và thuật toán bầu chọn. Giải pháp dựa trên điều khiển sử dụng cấu trúc cây, cấu trúc truyền broadcast và cấu trúc mạng vòng [31]. Yadav và cộng sự phân tích, so sánh hiệu năng của các thuật toán loại trừ tương hỗ phân tán. Kết quả được thể hiện trong bảng III.

Bảng II
HIỆU NĂNG CỦA THUẬT TOÁN DỰA TRÊN QUYỀN [29]

Thuật toán	**Tổng số thông điệp**
Lamport [21]	$3(N-1)$
Ricart-Agrawala [22]	$2(N-1)$
Carvalho-Roucairol [23]	0 đến $2(N-1)$
Raynal [24]	$2(N-1)^2$
Maekawa [25]	$3\sqrt{N}$ đến $5\sqrt{N}$
Sanders [26]	$\|Ii-\{i\}\|+2\|Ri-\{i\}\|$
Agrawal-El Abbadi [27]	$O(\log N)$
Singhal [28]	$(N-1)$ đến $3(N-1)/2$

Bảng III
PHÂN TÍCH, SO SÁNH HIỆU NĂNG CỦA CÁC THUẬT TOÁN LOẠI TRỪ TƯƠNG HỖ [30]

Thuật toán	**Thời gian hồi đáp**	**Độ trễ đồng bộ**	**Th. điệp tải thấp**	**Th. điệp tải cao**
[21]	$2T+E$	T	$3(N-1)$	$3(N-1)$
[22]	$2T+E$	T	$2(N-1)$	$2(N-1)$
[13]	$2T+E$	T	N	N
[16]	$T\log N+E$	$(T\log N)/2$	$\log N$	4

Hướng nghiên cứu của bài báo là song song hóa thuật toán Lamport trong loại trừ tương hỗ phân tán để giảm độ phức tạp thông điệp, thời gian hồi đáp và độ trễ đồng bộ. Cụ thể là đồng bộ hóa các tiến trình dựa trên song song hóa thuật toán Lamport đạt được trật tự tổng quát chặt chẽ với độ phức tạp thông điệp $3(N-1)$. Sau đó, các máy chủ hợp lực để vào miền găng thời điểm sớm hơn trong thuật toán loại trừ tương hỗ phân tán với thời gian hồi đáp là $N-1+E$, độ phức tạp thông điệp là $N-1$ và độ trễ đồng bộ $T=0$.

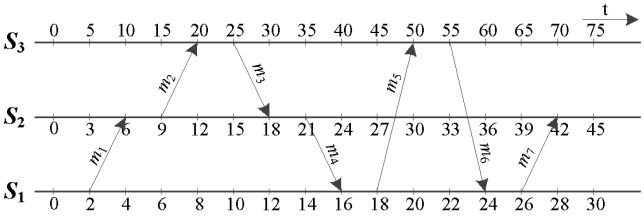
III. SONG SONG HÓA THUẬT TOÁN LAMPORT

1. Trật tự tổng quát chặt chẽ dựa trên song song hóa thuật toán Lamport

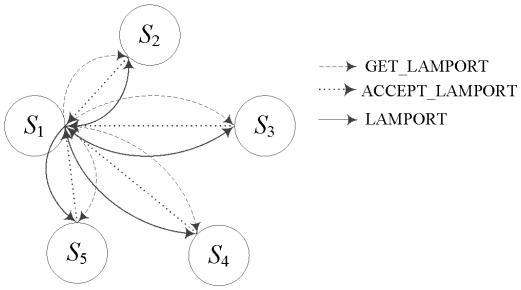
Nhãn thời gian lô-gíc được xây dựng dựa trên thuật toán Lamport trình bày trong [32]. Thuật toán Lamport cho phép ghi lại các sự kiện của hệ phân tán. Thuật toán tập trung vào nguyên lý sau: mỗi máy chủ S đều có trang bị công tơ với các giá trị nguyên gọi là H_{S_i}. Đó chính là đồng hồ lô-gíc tăng lên giữa hai sự kiện kế tiếp. Máy chủ e phát thông điệp ghi dấu của mình dựa trên giá trị hiện hành của H_{S_e}. Khi nhận được thông điệp, máy chủ nhận r cập nhật đồng hồ H_{S_r} riêng của mình bằng giải thuật rút gọn [32]:

$$\textbf{If } H_{S_r} \le E \textbf{ then}$$
$$H_{S_r} := E+1; \qquad (1)$$
$$\textbf{EndIf}.$$

Một sự kiện a (sk_a) sinh ra trong máy chủ i (S_i) và được đánh dấu bởi đồng hồ cục bộ gọi là $H_{S_i}(a)$. Nếu sk_a và



Hình 4. Nhãn thời gian của thông điệp không theo trật tự.



Hình 5. Hoạt động cấp giá trị đồng hồ lô-gic theo thuật toán 1.

sk_b là hai sự kiện được gửi từ cùng máy chủ S_i đến S_j, ta luôn luôn có quan hệ xác định như sau:

$$sk_a \rightarrow sk_b \Leftrightarrow H_{S_i}(a) < H_{S_j}(b), \qquad (2)$$

trong đó $sk_a \rightarrow sk_b$ thể hiện sự kiện a gửi cho sự kiện b, $H_{S_i}(a) < H_{S_j}(b)$ thể hiện giá trị đồng hồ máy chủ a nhỏ hơn giá trị đồng hồ cục bộ máy chủ b, ký hiệu $\rightarrow$ biểu thị phép kéo theo và ký hiệu $\Leftrightarrow$ biểu thị phép tương đương.

Tuy nhiên, các nhãn đồng hồ này phải được cập nhật và nhất quán trên tất cả các máy chủ. Nếu giá trị không được cập nhật thì việc xử lý thông điệp sẽ bị sai và hoạt động của hệ sai trật tự theo lý thuyết trật tự như hình 4.

Khi các thông điệp di chuyển qua các máy chủ, giá trị gửi và nhận có giá trị khác nhau. Chính vì giá trị đồng hồ sai lệch, khi một máy chủ phát lệnh xử lý đồng thời trên các máy chủ sẽ dẫn đến sai lệch về các tiến trình được triệu gọi để xử lý. Do đó, dữ liệu không nhất quán trên tất cả các máy chủ.

Các máy chủ hoạt động nhận và gửi thông điệp dựa trên đồng hồ cục bộ của mình theo cơ chế truyền unicast. Do đó, các máy chủ chỉ biết được *trật tự từng phần* trên máy chủ của mình và không nhận biết được các hoạt động trên máy chủ khác. *Trật tự từng phần* ảnh hưởng đến hoạt động tổng quát trong hệ phân tán. Hai vấn đề cơ bản là: (i) giá trị đồng hồ lô-gic trên các máy chủ không nhất quán; (ii) tiến trình yêu cầu vào đoạn găng phải chờ đợi cho đến khi nhận đủ thông điệp có thể gây ảnh hưởng đến các máy chủ khác hoặc sai lệch khi tiến hành cập nhật dữ liệu. Để giải quyết bài toán trật tự từng phần, nghiên cứu của bài báo là song song hóa thuật toán Lamport để xây dựng *trật tự tổng quát chặt chẽ* trên các máy chủ theo thuật toán 1.

Thuật toán 1: Song song hóa thuật toán Lamport

Dữ liệu vào:
- Máy chủ S_i;
- Giá trị đồng hồ lô-gic H_{S_i};
- Hành động *act*; và
- Sự kiện *sk*.

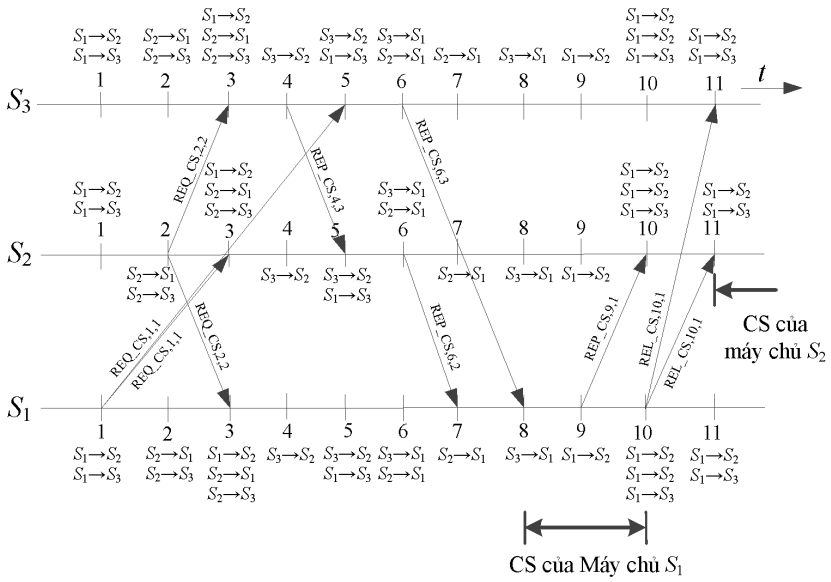
Dữ liệu ra: Giá trị đồng hồ lô-gic H_{S_i} đã cấp.

```
1  Khởi tạo hoạt động H_{S_local} = 0;
2  Biến đếm count = 0;
3  countCS[S_i, sk] = 0;
4  Số lượng máy chủ S = N;
5  Lắng nghe sự kiện act;
6  if act = REQ-LAMPORT then
7      H_{S_i} = H_{S_local} + 1;
8      act = REQ;
9      Thiết lập thông điệp yêu cầu cung cấp giá trị đồng hồ lô-
         gic: GetLamport(S_local, H_{S_local}, act, sk);
10     return multicast(GetLamport(S_i, H_{S_i}, act, sk));
11 else if act = LAMPORT then
12     return H_{S_i};
13 end
14 GetLamport (S_i, H_{S_i}, act, sk)
15 if act = REQ then
16     if H_{S_i} ≤ H_{S_local} then
17         Xác định H_{S_i} bị sai, H_{S_i} đã được gán cho sự kiện
             khác;
18         act = REP;
19         return multicast(AcceptLamport(S_local, S_i, H_{S_i}, act,
             sk, false));
20     else if H_{S_i} = H_{S_local} + 1 then
21         Xác định H_{S_i} đúng;
22         act = REP;
23         return multicast(AcceptLamport(S_local,
             S_i, H_{S_i}, act, sk, true));
24     end
25 end
26 AcceptLamport (S_local, S_i, H_{S_i}, act, sk, boolean)
27 if act = REP then
28     if S_i = S_local&&true then
29         count = count + 1;
30         if count = N − 1 then
31             Xác nhận đủ số lượng thông điệp phản hồi REP;
32             act = ACC;
33             count = 0;
34             return multicast(UpdateLamport(S_local,
                 H_{S_local}, act, sk));
35         end
36     else if S_i = S_local&&false then
37         H_{S_local} = max(H_{S_i} + 1);
38         act = REQ;
39         Thiết lập lại thông điệp yêu cầu cung cấp với giá trị
             đồng hồ lô-gic mới với sự kiện đang yêu cầu;
40         GetLamport(S_local, H_{S_local}, act, sk);
41     end
42 end
43 UpdateLamport (S_i, H_{S_i}, act, sk)
44 if act = ACC then
45     H_{S_local} = H_{S_i};
46 end
47 if sk = REP-CS then
48     count[S_i, sk] = count[S_i, sk] + 1;
49     if count[S_i, sk] = N − 1 then
50         CriticalSection(sk, S_i);
51         count[S_i, sk] = 0;
52     end
53 end
54 multicast ((S_local, H_{S_local}, act, sk))
55 S_n = tập tất cả máy chủ ngoại trừ S_local;
56 for j = 1 to S_n do
57     kn = connect(IP(S_n), port(S_n));
58     if kn then
59         sendUDP(GetLamport((S_local, H_{S_local}, act, sk),
             IP(S_n), port(S_n));
60     else if kn then
61         log-err(kn);
62     end
63 end
64 return H_{S_i};
```

Để đạt được trật tự tổng quát chặt chẽ, thuật toán Lamport được song song hóa nhằm đồng bộ hóa các tiến trình di chuyển trong hệ phân tán. Mỗi sự kiện diễn ra trên bất kỳ máy chủ nào đều phải yêu cầu giá trị đồng hồ và giá trị này được nhận biết và nhất quán trên tất cả các máy chủ.

Hình 5 mô tả hoạt động theo thuật toán 1, máy chủ S_1 yêu cầu cung cấp giá trị đồng hồ lô-gic, thông điệp mang giá trị *GetLamport* được truyền multicast đến các máy chủ trong hệ thống (dòng 10). Các thủ tục trong thuật toán 1 được giải thích như sau. Thủ tục *multicast* (dòng 54 đến 64) là quá trình xử lý truyền song song các thông điệp đến tập máy chủ trong hệ thống. Thủ tục *GetLamport* (dòng 14 đến 25) thực hiện tính toán và xử lý giá trị đồng hồ so với máy chủ cục bộ nhận được thông điệp, nếu giá trị đồng hồ đúng trả về giá trị *true* (dòng 23), nếu sai trả về giá trị *false* (dòng 19). Thủ tục *AcceptLamport* (dòng 26 đến 42) cho phép giá trị đồng hồ được xác lập dựa trên giá trị true của tất cả các máy chủ. Ngược lại nếu một máy chủ bất kỳ trả về giá trị *false*, thủ tục này xác lập lại giá trị đồng hồ lô-gic dựa trên lấy giá trị cực đại và phát đi thông điệp *GetLamport* (dòng 37). Thủ tục *UpdateLamport* (dòng 43 đến 53) nhằm khẳng định cho máy chủ có sự kiện được yêu cầu gán giá trị đồng hồ lô-gic và giá trị này cập nhật trên tất cả các máy chủ.

Hình 6 mô tả kết quả thực hiện song song hóa thuật toán Lamport so với thuật toán nguyên thủy của Lamport theo hình 3. Theo thuật toán nguyên thủy của Lamport, giá trị đồng hồ lô-gic sinh ra khi nó lấy giá trị cực đại của thông điệp nhận và tăng lên mỗi khi có sự kiện mới phát sinh bên trong máy chủ. Thuật toán này tuân thủ luật *happened-*



Hình 6. Hoạt động song song hóa Thuật toán Lamport.

before. Tuy nhiên, đây là giải pháp thụ động. Thông qua hình 6, mỗi một giá trị đồng hồ lô-gic được gán cho sự kiện đều gửi đến tất cả các trạm. Vì vậy, mặc dù không có hoạt động trên trạm nhưng trạm nhận biết được hoạt động trên các trạm còn lại. Đây là giải pháp chủ động trong việc giám sát và điều khiển sự kiện. Giải pháp này tạo điều kiện thuận lợi cho các sự kiện yêu cầu vào miền găng để giải quyết loại trừ tương hỗ phân tán.

2. Áp dụng song song hóa thuật toán Lamport để giải quyết loại trừ tương hỗ phân tán

Mô hình hệ phân tán theo hình 1 được mô tả là hệ thống bao gồm N máy chủ S_i $(i = 1, \ldots, N)$ kết nối với nhau qua môi trường truyền thông. Một tiến trình p_j $(j = 1, \ldots, M)$ thực thi trên một máy chủ S_i.

Nghiên cứu trong [33] trình bày giải pháp gắn bó dựa trên thuật toán 4PCoDT. Giải pháp thực hiện cơ chế truyền thông điệp trong vòng tròn ảo, mỗi pha di chuyển qua từng máy chủ theo vòng tròn định trước. Pha thứ nhất thực hiện xử lý tiến trình yêu cầu đi vào miền găng để tránh tương tranh tài nguyên. Yêu cầu thông điệp trong tiến trình thực thi miền găng trên máy chủ có chứa một trong ba giá trị: *REQ-CS*, *REP-CS* và *REL-CS*. Khi tiến trình yêu cầu *REQ-CS* đã nhận đầy đủ phản hồi *REP-CS* mới được phép vào miền găng. Tiến trình vào miền găng tuân thủ nguyên tắc loại trừ tương hỗ phân tán trình bày trong phần II. Sau khi tiến trình xử lý xong và rời khỏi miền găng, máy chủ phát đi thông điệp có chứa giá trị *REL-CS* để tiến trình khác được phép vào miền găng.

Trên cơ sở song song hóa thuật toán Lamport trình bày trong mục III.1, thuật toán loại trừ tương hỗ phân tán thực hiện dựa trên cải tiến thuật toán Lamport được trình bày trong thuật toán 2. Hàm *RequestCriticalSection* (dòng 10 đến 24) thể hiện các tiến trình yêu cầu vào miền găng. Hàm *RequestCriticalSection* xác định giá trị đồng hồ lô-gic đã gán cho tiến trình, nếu tiến trình nào có giá trị nhỏ hơn có quyền ưu tiên cao hơn để vào miền găng. Hàm *CriticalSection* (dòng 25 đến 33) thực hiện xử lý tiến trình trong miền găng. Quá trình xử lý tiến trình đảm bảo các trường nội dung tác động lên cơ sở dữ liệu là duy nhất trên tập máy chủ, đảm bảo tính gắn bó trong hệ phân tán. Sau khi kết thúc quá trình xử lý trong miền găng lập tức yêu cầu xóa tiến trình khỏi hàng đợi, phát thông điệp giải phóng miền găng để tiến trình tiếp theo được phép vòa miền găng. Hàm *NextCriticalSection* (dòng 34 đến 38) xử lý tiến trình tiếp theo trong hàng đợi. Nếu tiến trình tiếp theo này đã tiếp nhận đầy đủ phản hồi theo thuật toán 1 thì lập tức vào miền găng mà không cần phải đợi bất kỳ xác nhận nào.

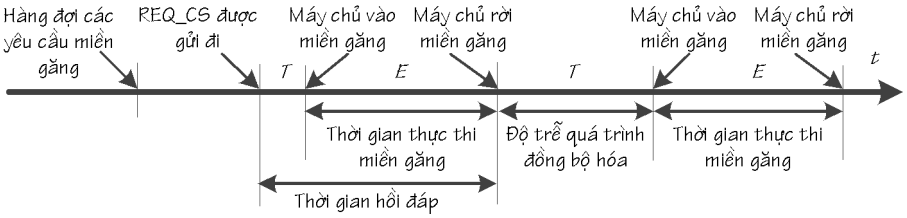
Theo thuật toán Lamport nguyên thủy trong bảng I, máy chủ S_1 vào miền găng tại thời điểm 8 khi đã nhận đủ thông điệp phản hồi *REP-CS*. Sau khi áp dụng song song hóa, thuật toán Lamport đạt được một trật tự tổng quát chặt chẽ trên các máy chủ theo bảng IV. Kết quả bảng IV cho thấy máy chủ vào miền găng thời điểm 6, sớm hơn so với trật tự từng phần ở bảng I.

IV. HIỆU NĂNG THỰC THI SONG SONG HÓA THUẬT TOÁN LAMPORT

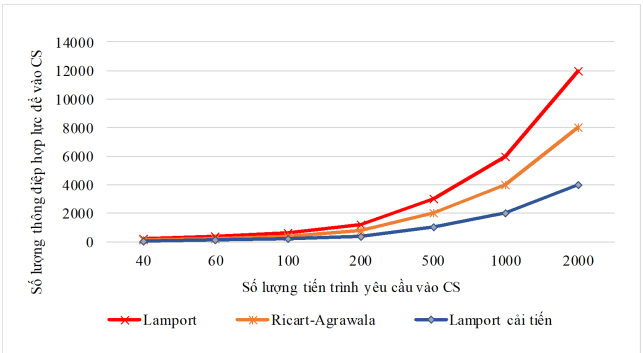
Dựa vào mô tả các tiến trình hoạt động trong miền găng và các tham số trình bày trong phần I, chúng tôi đánh giá hiệu năng loại trừ tương hỗ phân tán theo Hình 7. Tham số độ phức tạp thông điệp của song song hóa thuật toán

Bảng IV
HOẠT ĐỘNG DIỄN RA TRÊN CÁC MÁY CHỦ TRONG TRẬT TỰ TỔNG QUÁT CHẶT CHẼ

Đồng hồ	**Máy chủ 1**	**Máy chủ 2**	**Máy chủ 3**
1	$S_1 \rightarrow S_2$: REQ-CS,1,1 $S_1 \rightarrow S_3$: REQ-CS,1,1	$S_1 \rightarrow S_2$: REQ-CS,1,1 $S_1 \rightarrow S_3$: REQ-CS,1,1	$S_1 \rightarrow S_2$: REQ-CS,1,1 $S_1 \rightarrow S_3$: REQ-CS,1,1
2	$S_2 \rightarrow S_1$: REQ-CS,2,2 $S_2 \rightarrow S_3$: REQ-CS,2,2	$S_2 \rightarrow S_1$: REQ-CS,2,2 $S_2 \rightarrow S_3$: REQ-CS,2,2	$S_2 \rightarrow S_1$: REQ-CS,2,2 $S_2 \rightarrow S_3$: REQ-CS,2,2
3	$S_2 \rightarrow S_1$: REQ-CS,2,2 $S_1 \rightarrow S_2$: REQ-CS,1,1 $S_2 \rightarrow S_3$: REQ-CS,2,2	$S_1 \rightarrow S_2$: REQ-CS,1,1 $S_2 \rightarrow S_3$: REQ-CS,2,2 $S_2 \rightarrow S_1$: REQ-CS,2,2	$S_2 \rightarrow S_3$: REQ-CS,2,2 $S_2 \rightarrow S_1$: REQ-CS,2,2 $S_1 \rightarrow S_2$: REQ-CS,1,1
4	$S_3 \rightarrow S_2$: REP-CS,4,3	$S_3 \rightarrow S_2$: REP-CS,4,3	$S_3 \rightarrow S_2$: REP-CS,4,3
5	$S_3 \rightarrow S_2$: REP-CS,4,3 $S_1 \rightarrow S_3$: REQ-CS,1,1	$S_3 \rightarrow S_2$: REP-CS,4,3 $S_1 \rightarrow S_3$: REQ-CS,1,1	$S_1 \rightarrow S_3$: REQ-CS,1,1 $S_3 \rightarrow S_2$: REP-CS,4,3
6	$S_3 \rightarrow S_1$: REP-CS,6,3	$S_3 \rightarrow S_1$: REP-CS,6,3 **S_1 vào miền găng**	$S_3 \rightarrow S_1$: REP-CS,6,3
7	$S_1 \rightarrow S_2$: REP-CS,7,1 **S_2 đã nhận đủ REP-CS, chờ lượt tiếp theo vào miền găng**	$S_1 \rightarrow S_2$: REP-CS,7,1	$S_1 \rightarrow S_2$: REP-CS,7,1
8	$S_1 \rightarrow S_2$: REL-CS,8,1 $S_1 \rightarrow S_3$: REL-CS,8,1 $S_1 \rightarrow S_2$: REP-CS,7,1	$S_1 \rightarrow S_2$: REP-CS,7,1 $S_1 \rightarrow S_2$: REL-CS,8,1 $S_1 \rightarrow S_3$: REL-CS,8,1 **S_2 vào miền găng**	$S_1 \rightarrow S_2$: REL-CS,8,1 $S_1 \rightarrow S_3$: REL-CS,8,1 $S_1 \rightarrow S_2$: REP-CS,7,1
9	$S_1 \rightarrow S_2$: REL-CS,8,1 $S_1 \rightarrow S_3$: REL-CS,8,1	$S_1 \rightarrow S_2$: REL-CS,8,1 $S_1 \rightarrow S_3$: REL-CS,8,1	$S_1 \rightarrow S_2$: REL-CS,8,1 $S_1 \rightarrow S_3$: REL-CS,8,1



Hình 7. Mô tả các tiến trình hoạt động trong miền găng.



Hình 8. Số lượng thông điệp hợp lực đáp ứng các tiến trình để vào miền găng.

Lamport được xác định dựa trên số lượng thông điệp yêu cầu trên một máy chủ cho mỗi thực thi miền găng. Đối với song song hóa thuật toán Lamport yêu cầu $N-1$ thông điệp *REQ*, $N-1$ thông điệp *REP*, $N-1$ thông điệp *ACC*, do đó, thuật toán yêu cầu $3(N-1)$ thông điệp. Đối với thuật toán cải tiến loại trừ tương hỗ của bài báo yêu cầu $N-1$ thông điệp *REQ-CS* và không xét thông điệp *REP-CS* và *REL-CS* trong quá trình nhận, do đó, thuật toán yêu cầu $N-1$ thông điệp khi vào miền găng.

Nguyên nhân độ phức tạp thông điệp thuật toán loại trừ tương hỗ thấp là khi áp dụng song song hóa thuật toán Lamport, các thông điệp *REP-CS* và *REL-CS* đã được nhận biết và đánh dấu khi yêu cầu giá trị đồng hồ lô-gic trên các máy chủ. Do đó, máy chủ yêu cầu vào miền găng không

cần phải đợi tiếp nhận đủ thông điệp *REP-CS* và *REL-CS*. Ngoài ra, trong quá trình truyền thông điệp trong hệ thống, thông điệp *REP-CS* và *REL-CS* bị phân mảnh hoặc thất lạc không ảnh hưởng đến quá trình vào miền găng của máy chủ. Theo kết quả hình 8, đối với số lượng tiến trình yêu cầu vào miền găng lớn thì số lượng thông điệp hợp lực để tiến trình vào miền găng càng lớn. Do đó, nếu giảm được số lượng thông điệp hợp lực thì hiệu năng tiến trình vào miền găng tăng.

Tham số độ trễ quá trình đồng bộ hóa T được xác định dựa trên khoảng thời gian yêu cầu sau khi máy chủ bắt đầu phát thông điệp yêu cầu *REQ-CS* cho đến khi vào miền găng hoặc máy chủ rời miền găng để nhường cho máy chủ tiếp theo vào miền găng. Trong trường hợp tải cao, nghĩa là các máy chủ yêu cầu vào miền găng đã nhận đủ thông điệp phản hồi theo song song hóa thuật toán Lamport (dòng 52 của thuật toán 1) và trên hàng đợi của các máy chủ luôn có tiến trình sẵn sàng vào miền găng. Máy chủ tiếp theo được thực hiện tức thì trong miền găng trong trường hợp tải cao sau khi máy chủ trước vừa rời khỏi miền găng, tham số độ trễ quá trình đồng bộ hóa được xác định là $T = 0$.

Thuật toán 2: Cải tiến thuật toán loại trừ tương hỗ Lamport

Dữ liệu vào: Tiến trình *tt(start,jeton, lamport$_1$,lamport$_2$, S_act,type,action,circle,content)*
Dữ liệu ra: Trật tự tổng quát chặt chẽ tiến trình, tiến trình vào miền găng và loại trừ tương hỗ nhờ dấu

1 *action* = *tt.action*;
2 **if** *action* = 1 **then**
3 *sk* = *REQ-CS*;
4 *act* = *REQ-LAMPORT*;
5 $H_{S_{local}} = LAMPORT(S_{local}, act, sk)$;
6 *tt(start,jeton,*$H_{S_{local}}$*,*$lamport_2$*,*S_{local}*,type,action, circle,content)*;
7 *req_queue(tt)*;
8 *multicast(RequestCriticalSection(sk,tt))*;
9 **end**
10 `RequestCriticalSection(`*sk*, *tt*`)`
11 $H_{S_{local}} = tt.lamport_1$;
12 $S_i = tt.S_{act}$;
13 **if** $sk = REQ\text{-}CS, H_{S_i} < H_{S_{local}}$ **then**
14 *sk* = *REP-CS*;
15 *act* = *REQ-LAMPORT*;
16 $H_{S_{local}} = LAMPORT(S_{local}, act, sk)$;
17 *tt(start,jeton,*H_{S_i}*,*$H_{S_{local}}$*,*S_i*,type,action,circle,content)*;
18 **else if** $sk = REP\text{-}CS, H_{S_i} < H_{S_{local}}$ **then**
19 *sk* = *REP-CS*;
20 *act* = *REQ-LAMPORT*;
21 $H_{S_{local}} = LAMPORT(S_{local}, act, sk)$;
22 $writelog(tt(start, jeton, H_{S_i}, H_{S_{local}}, S_{local}, type, action, circle, content))$;
23 **end**
24 **return** *RequestCriticalSection(sk,tt)*;
25 `CriticalSection(`*sk*, *tt*`)`
26 *content* = *tt.content*;
27 *process(content)*;
28 *remove_req_queue(tt)*;
29 *sk* = *REL-CS*;
30 *act* = *REQ-LAMPORT*;
31 $H_{S_{local}} = LAMPORT(S_{local}, act, sk)$;
32 $tt(start, jeton, lamport_1, H_{S_{local}}, S_i, type, action, circle, content)$;
33 **return** *multicast(NextCriticalSection())*;
34 `NextCriticalSection()`
35 **if** *req_queue* $\neq \emptyset$ **then**
36 *pop_req_queue(tt)*;
37 **return** *CriticalSection(sk, tt)*;
38 **end**

Tham số thời gian hồi đáp H được xác định là khoảng thời gian từ khi gửi yêu cầu vào miền găng cho đến khi ra khỏi miền găng. Theo mô tả trong hình 7, H được tính bằng công thức sau [4]:

$$H = T + E, \tag{3}$$

trong đó T là độ trễ quá trình đồng bộ hóa và E là thời gian thực thi miền găng. Đối với trường hợp áp dụng song song hóa thuật toán Lamport, máy chủ được phép vào miền găng ngay tại thời điểm máy chủ cuối cùng bắt đầu phản hồi thông điệp *REP-CS* sau thông điệp *REQ-CS*. Như vậy, thời gian hồi đáp được xác định như sau. Đối với trường hợp chờ tiếp nhận máy chủ cuối cùng bắt đầu phát thông điệp *REP-CS*, chúng ta có $T = N - 1$, do đó $H = N - 1 + E$. Đối với trường hợp đã tiếp nhận đủ thông điệp *REP-CS* và đang nằm trong hàng đợi vào miền găng trong lượt tiếp theo, chúng ta có $T = 0$, do đó $H = E$.

Tham số thông lượng hệ thống ký hiệu là A được xác định dựa trên tỷ lệ mà hệ thống thực thi các yêu cầu trong miền găng. A được tính bằng công thức [4]:

$$A = \frac{1}{H}, \tag{4}$$

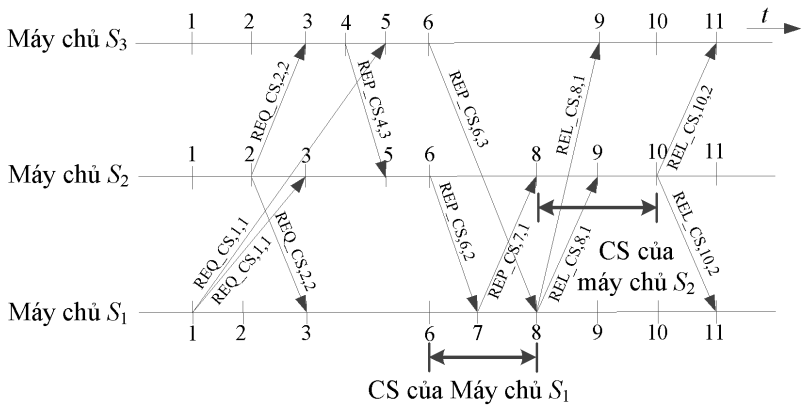
trong đó H là thời gian hồi đáp. Đối với trường hợp tải thấp $T = N - 1$ thì $A = 1/(N-1) + E$. Đối với trường hợp tải cao $T = 0$ thì $A = 1/E$.

Ký hiệu $\overline{X}$ là số lượng trung bình thông điệp vào miền găng trên các máy chủ. Đối với trường hợp tải cao, trên mỗi máy chủ đều có ít nhất một tiến trình yêu cầu vào miền găng. Một máy chủ cần phải phát $N - 1$ thông điệp để yêu cầu vào miền găng. Vì vậy, $\overline{X}$ được xác định như sau:

$$\overline{X} = \frac{N-1}{N} + \frac{N + (N-1)}{N} = \frac{3N-2}{N}. \tag{5}$$

Theo công thức (5), nếu số lượng các các chủ lớn ($N \to \infty$), số lượng trung bình thông điệp trên các máy chủ xấp xỉ 3. Trong khi đó thông số này cho thuật toán Lamport nguyên thủy là xấp xỉ 7 và cho thuật toán Ricart–Agrawala là xấp xỉ 5 với số lượng máy chủ tương tự.

Đối với việc áp dụng song song hóa thuật toán Lamport trong thực thi miền găng, thời gian hồi đáp của tiến trình yêu cầu miền găng và độ trễ quá trình đồng bộ hóa được rút ngắn. Kết quả so sánh các thuật toán thể hiện trong



Hình 9. Hiệu năng thực thi song song hóa thuật toán.

Bảng V
SO SÁNH HIỆU NĂNG CỦA THUẬT TOÁN LAMPORT CẢI TIẾN TRONG LOẠI TRỪ TƯƠNG HỖ PHÂN TÁN

Thuật toán	Thời gian hồi đáp	Độ trễ đồng bộ	Th. điệp tải thấp	Th. điệp tải cao
[21]	$2T + E$	T	$3(N-1)$	$3(N-1)$
[22]	$2T + E$	T	$2(N-1)$	$2(N-1)$
Cải tiến	$T + E$	T	$N-1$	$N-1$

bảng V. Thuật toán Lamport cải tiến khi áp dụng song song hóa thuật toán Lamport đạt được hiệu năng loại trừ tương hỗ cao so với thuật toán Lamport và Ricart-Agrawala.

Theo kết quả thực hiện trong hình 9, áp dụng song song hóa thuật toán Lamport, tiến trình trên máy chủ S_1 vào miền găng tại thời điểm 6 và trên máy chủ S_2 vào miền găng tại thời điểm 8.

Tiến trình yêu cầu miền găng trên máy chủ S_1 mô tả cho trường hợp tải thấp: thời gian hồi đáp $D = 8$ được xác định từ thời điểm giá trị đồng hồ là 1 cho đến lúc phát thông điệp rời khỏi miền găng tại thời điểm 8. So với hình 3, S_1 phải đợi đủ thông điệp phản hồi từ các máy chủ còn lại tại thời điểm 8 mới bắt đầu vào miền găng, do đó S_1 có $D = 10$.

Tiến trình yêu cầu miền găng trên máy chủ S_2 mô tả cho trường hợp tải cao: thời gian hồi đáp $D = 8$ được xác định từ thời điểm giá trị đồng hồ là 2 cho đến lúc phát thông điệp rời khỏi miền găng tại thời điểm 10. Trong trường hợp tải cao, tiến trình trên S_2 đã nhận đủ thông điệp phản hồi và chờ vào miền găng thì tại thời điểm 8 S_2 lập tức vào miền găng. Đối với trường hợp này, S_2 không phải chờ đợi tiếp nhận thông điệp giải phóng khỏi miền găng từ S_1 như mô tả trong hình 3. Bên cạnh đó, theo mô tả trong hình 3 nếu thông điệp giải phóng từ S_1 bị thất lạc hoặc phân mảnh trong quá trình truyền thì S_2 phải chờ đợi cho đến khi tiếp nhận đầy đủ, điều này dẫn đến hiệu năng thực hiện bị giảm.

V. KẾT LUẬN

Trong bài báo này, nghiên cứu giải pháp song song hóa thuật toán Lamport nhằm cải tiến thuật toán loại trừ tương hỗ phân tán. Song song hóa thuật toán Lamport cho phép thiết lập một trật tự tổng quát chặt chẽ và ghi dấu các sự kiện diễn ra trên các máy chủ. Thuật toán cải tiến gán dấu cho sự kiện yêu cầu $3(N-1)$ thông điệp. Khi áp dụng song song hóa thuật toán Lamport trong thuật toán loại trừ tương hỗ, tiến trình đi vào miền găng yêu cầu $N-1$ thông điệp. Do đó, giải pháp cải tiến của bài báo đạt hiệu năng cao trong cải tiến thuật toán loại trừ tương hỗ phân tán thể hiện trong bảng V. Tuy nhiên, để đạt được hiệu năng cao, độ phức tạp thông điệp song song hóa thuật toán Lamport để gắn dấu cho tiến trình yêu cầu $3(N-1)$ thông điệp và độ dài thông điệp lớn hơn so với các thuật toán gắn dấu khác. Do đó, đối với hệ thống tải cao thì song song hóa thuật toán Lamport phải xử lý với số lượng lớn các yêu cầu giá trị đồng hồ lô-gic trong quá trình hợp lực. Vì vậy, giải pháp để giải quyết tối ưu hệ thống trong hợp lực và truyền thông cần được tiếp tục nghiên cứu.

TÀI LIỆU THAM KHẢO

[1] R. Buyya, "Market-oriented cloud computing: Vision, hype, and reality of delivering computing as the 5th utility," in *2009 Fourth ChinaGrid Annual Conf.*, 2009, pp. xii–xv.
[2] G. F. Coulouris and J. Dollimore, *Distributed systems: Concepts and design*, 4th ed. Addison-Wesley, 2005.
[3] M. Raynal, *Distributed Algorithms for Message-Passing Systems*. Springer, 2013.
[4] A. D. Kshemkalyani and M. Singhal, *Distributed Computing: Principles, Algorithms, and Systems*. Cambridge University Press, 2008.
[5] M. Raynal, A. Schiper, and S. Toueg, "The causal ordering abstraction and a simple way to implement it," *Information Processing Letters*, vol. 39, no. 6, pp. 343–350, 1991.
[6] N. Sharma and A. Parikh, "Deadlock detection and removal in distributed systems," *International Jour. Engineering and Computer Science*, vol. 2, no. 10, pp. 2900–2902, Oct. 2013.
[7] H. H. C. Nguyen, H. V. Dang, N. M. N. Pham, V. S. Le, and T. T. Nguyen, "Deadlock detection for resource

allocation in heterogeneous distributed platforms," in *Recent Advances in Information and Communication Technology 2015*. Springer, 2015, pp. 285–295.
[8] H. H. C. Nguyen, V. S. Le, and T. T. Nguyen, "Algorithmic approach to deadlock detection for resource allocation in heterogeneous platforms," in *International Conference on Smart Computing*, 2014, pp. 97–103.
[9] H. H. C. Nguyen, H. D. Tran, V. T. Doan, and V. T. P. Anh, "Deadlock avoidance for resource allocation model V VM-out-of-N PM," in *Context-Aware Systems and Applications*. Springer, 2017, pp. 172–182.
[10] M. Singhal, "Deadlock detection in distributed systems," *Computer*, vol. 22, no. 11, pp. 37–48, 1989.
[11] K. Erciyes and V. Adve, *Distributed Graph Algorithms for Computer Networks*. Springer, 2013.
[12] G. Ricart and A. K. Agrawala, "Author's response to 'On mutual exclusion in computer networks' by Carvalho and Roucairol," *Commun. of the ACM*, pp. 147–148, 1983.
[13] I. Suzuki and T. Kasami, "A distributed mutual exclusion algorithm," *ACM Transactions on Computer Systems*, vol. 3, no. 4, pp. 344–349, Nov. 1985.
[14] M. Mizuno, M. L. Neilsen, and R. Rao, "A token based distributed mutual exclusion algorithm based on quorum agreements," in *11th International Conference on Distributed Computing Systems*, 1991, pp. 361–368.
[15] J.-M. Helary, N. Plouzeau, and M. Raynal, "A distributed algorithm for mutual exclusion in an arbitrary network," *The Computer Journal*, vol. 31, no. 4, pp. 289–295, 1988.
[16] K. Raymond, "A tree-based algorithm for distributed mutual exclusion," *ACM Transactions on Computer Systems*, vol. 7, no. 1, pp. 61–77, 1989.
[17] M. Singhal, "A heuristically-aided algorithm for mutual exclusion in distributed systems," *IEEE Transactions on Computers*, vol. 38, no. 5, pp. 651–662, 1989.
[18] N. Mohamed and T. Michel, "How to detect a failure and regenerate the token in the Log(n) distributed algorithm for mutual exclusion," in *Distributed Algorithms*. Springer Berlin Heidelberg, 1988, pp. 155–166.
[19] S. Mishra and P. K. Srimani, "Fault-tolerant mutual exclusion algorithms," *Journal of Systems and Software*, vol. 11, no. 2, pp. 111–129, 1990.
[20] S. Nishio, K. F. Li, and E. G. Manning, "A resilient mutual exclusion algorithm for computer networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 1, no. 3, pp. 344–356, 1990.
[21] L. Lamport, "Time, clocks, and the ordering of events in a distributed system," *Communications of the ACM*, vol. 21, no. 7, pp. 558–565, 1978.
[22] G. Ricart and A. K. Agrawala, "An optimal algorithm for mutual exclusion in computer networks," *Communications of the ACM*, vol. 24, no. 1, pp. 9–17, 1981.
[23] O. Carvalho and G. Roucairol, "On mutual exclusion in computer networks," *Communications of the ACM*, vol. 26, no. 2, pp. 146–147, Feb. 1983.
[24] M. Raynal, "Prime numbers as a tool to design distributed algorithms," *Information Processing Letters*, vol. 33, no. 1, pp. 53–58, 1989.
[25] M. Maekawa, "A Sqrt(N) algorithm for mutual exclusion in decentralized systems," *ACM Transactions on Computer Systems*, vol. 3, no. 2, pp. 145–159, 1985.
[26] B. A. Sanders, "The information structure of distributed mutual exclusion algorithms," *ACM Transactions on Computer Systems*, vol. 5, no. 3, pp. 284–299, 1987.
[27] D. Agrawal and A. E. Abbadi, "An efficient and fault-tolerant solution for distributed mutual exclusion," *ACM Transactions on Computer Systems*, vol. 9, no. 1, pp. 1–20, 1991.
[28] M. Singhal, "A dynamic information-structure mutual exclusion algorithm for distributed systems," in *9th International Conference on Distributed Computing Systems*, 1989, pp. 70–78.
[29] M. G. Velazquez, "A survey of distributed mutual exclusion algorithms," Colorado State University, Tech. Rep., 1993.
[30] N. Yadav, S. Yadav, and S. Mandiratta, "A review of various mutual exclusion algorithms in distributed environment," *International Journal of Computer Applications*, vol. 129, no. 14, pp. 11–16, 2015.
[31] S. Naseera, "A distributed ring algorithm for coordinator election in distributed systems," *ICTACT Journal on Communication Technology*, vol. 7, no. 3, pp. 1341–1344, 2016.
[32] L. V. Sơn, *Hệ tin học phân tán*. Nhà xuất bản Đại học Quốc gia Tp. Hồ Chí Minh, 2002.
[33] D. H. Vĩ and L. V. Sơn, "Cung cấp tài nguyên truyền thông cho hệ phân tán trong máy ảo," *Journal of Science and Technology, The University of Danang*, vol. 11, no. 108, pp. 90–94, 2016.



Đặng Hùng Vĩ sinh năm 1980 tại Đà Nẵng. Tốt nghiệp đại học tại Trường Đại học Bách khoa, Đại học Đà Nẵng năm 2003 chuyên ngành Công nghệ Thông tin. Nhận bằng Thạc sỹ Khoa học Máy tính năm 2008 tại Đại học Đà Nẵng. Hiện đang là nghiên cứu sinh chuyên ngành Khoa học máy tính tại Đại học Đà Nẵng. Lĩnh vực nghiên cứu: Mạng máy tính, mã mạng, hệ phân tán, điện toán đám mây.



Lê Văn Sơn sinh năm 1948 tại Điện Bàn, Quảng Nam. Tốt nghiệp Đại học năm 1978, bảo vệ luận án Tiến sĩ năm 1997 tại Trường Đại học Tổng hợp Donesk, Ukraina. Công nhận Phó Giáo sư năm 2004. Hiện công tác tại Hội tin học Đà Nẵng. Lĩnh vực nghiên cứu: Hệ điều hành, mạng máy tính, hệ phân tán, tính toán đám mây.



Nguyễn Xuân Huy sinh năm 1944 tại Hải Phòng. Bảo vệ luận án Tiến sĩ năm 1978 chuyên ngành Toán-Tin tại Liên Xô. Bảo vệ Tiến sĩ Khoa học năm 1988 thuộc chuyên ngành Công nghệ Thông tin tại Liên Xô. Công nhận Phó Giáo sư năm 1992. Nguyên Chủ tịch Hội đồng Tư vấn Giáo dục Microsoft Việt Nam, Nghiên cứu viên cao cấp Viện Công nghệ thông tin, Viện Khoa học và Công nghệ Việt Nam. Lĩnh vực nghiên cứu: Công nghệ phần mềm, cơ sở dữ liệu.

Toán tử lân cận mới cho thuật toán Tabu Search và PSO giải bài toán lập lịch luồng công việc trong môi trường điện toán đám mây

Phan Thanh Toàn[1], Đặng Quốc Hữu[2], Nguyễn Thế Lộc[3]

[1] Khoa Sư phạm Kỹ thuật, Trường Đại học Sư phạm Hà Nội

[2] Trung tâm Công nghệ Thông tin, Trường Đại học Thương mại, Hà Nội

[3] Khoa Công nghệ Thông tin, Trường Đại học Sư phạm Hà Nội

Tác giả liên hệ: Phan Thanh Toàn, pttoan@hnue.edu.vn

Ngày nhận bài: 11/06/2019, ngày sửa chữa: 27/10/2019, ngày duyệt đăng: 27/10/2019

Định danh DOI: 10.32913/mic-ict-research-vn.v2019.n2.865

Biên tập lĩnh vực điều phối phản biện và quyết định nhận đăng: PGS.TS. Huỳnh Thị Thanh Bình

Tóm tắt: Điện toán đám mây là xu thế mới của công nghệ thông tin và truyền thông. Trong mô hình điện toán đám mây mọi khả năng liên quan đến công nghệ thông tin đều được cung cấp dưới dạng dịch vụ, cho phép người sử dụng truy cập đến các dịch vụ công nghệ (phần cứng và phần mềm) từ các nhà cung cấp dịch vụ. Điện toán đám mây là sự tập hợp của nhiều máy chủ vật lý và máy chủ ảo, được cấu hình để làm việc với nhau trên môi trường mạng Internet. Một trong số các vấn đề lớn nhất trong môi trường điện toán đám mây là bài toán lập lịch luồng công việc. Hiệu năng của các hệ thống điện toán đám mây phụ thuộc rất nhiều vào việc sắp xếp các tác vụ trong luồng thực thi trên các máy tính trong môi trường đám mây để hoàn thành luồng công việc một cách tối ưu. Trong bài báo này chúng tôi đề xuất một thuật toán lập lịch luồng công việc mới dựa trên chiến lược tối ưu bày đàn và tìm kiếm Tabu.

Từ khóa: *Lập lịch luồng công việc, tìm kiếm Tabu, tối ưu bày đàn, điện toán đám mây.*

Title: **New Effective Neighborhoods for Tabu Search and Particle Swarm Optimization to Schedule Workflow in Cloud Computing**

Abstract: Cloud computing is a new trend of information and communication technology that enables resource distribution and sharing at a large scale. The cloud consists of a collection of virtual machines that promises to provision on-demand computational and storage resources when needed. End-users can access these resources via the Internet and have to pay only for their usage. Workflow scheduling is a big issue in cloud computing. Basically the issue relates to discovering resources and allocating tasks on suitable resources. Workflow scheduling plays a vital role in the system management. In this work, we propose a new algorithm for workflow scheduling that is derived from particle swarm optimization and Tabu search.

Keywords: *Workflow scheduling, Tabu search, particle swarm optimization, cloud computing.*

I. GIỚI THIỆU

Với sự phát triển của công nghệ thông tin và truyền thông, điện toán đám mây được ứng dụng rộng rãi trong nghiên cứu khoa học và thực tiễn. Mọi tài nguyên trong môi trường điện toán đám mây đều được cung cấp cho người dùng dưới dạng dịch vụ, như: dịch vụ về phần mềm (SaaS: Software as a Service), dịch vụ cơ sở hạ tầng (IaaS: Infrastructure as a Service), dịch vụ nền tảng hạ tầng (PaaS: Platform as a Service). Nhiều ứng dụng được mô hình hóa dưới dạng luồng công việc (workflow) bao gồm tập các tác vụ (task) và các phụ thuộc giữa chúng theo kiểu cha–con. Tác vụ con chỉ được bắt đầu sau khi tác vụ cha đã hoàn thành. Ứng dụng dạng luồng công việc được sử dụng rộng rãi trong nhiều lĩnh vực: thiên văn học, tin sinh, dự báo động đất, v.v. Hơn nữa, ngày nay các ứng dụng ngày càng phức tạp và đòi hỏi phải xử lí một khối lượng lớn dữ liệu, chính vì vậy các ứng dụng này cần phải được thực hiện trên các hệ thống siêu máy tính, hệ thống tính toán lưới, hay điện toán đám mây. Lập lịch luồng công việc (workflow scheduling) là tìm phương án để gán các tác vụ của luồng công việc vào thực hiện trên các máy ảo (VM: Virtual Machine) của môi trường điện toán đám mây nhằm giảm thiểu thời gian và chi phí thực hiện.

Luồng công việc là một chuỗi có thứ tự các tác vụ có thể được thực hiện đồng thời hay tuần tự nếu dữ liệu đầu ra của tác vụ này là đầu vào của tác vụ kế tiếp. Vấn đề lập lịch luồng công việc trong môi trường điện toán đám mây về bản chất là tìm phương án ánh xạ những tác vụ của luồng công việc tới các máy chủ của đám mây sao cho thời gian xử lý toàn bộ luồng công việc là nhỏ nhất, biết rằng khối lượng tính toán và yêu cầu dữ liệu của các tác vụ, tốc độ tính toán và truyền thông của các máy chủ là khác nhau.

Bài toán lập lịch luồng công việc đã được nghiên cứu từ những năm 1950 và đã được chứng minh là thuộc lớp NP-Khó (NP-Hard) [1]. Trong những năm gần đây đã có rất nhiều ứng dụng khoa học được mô hình hóa bởi dạng đồ thị luồng công việc như ứng dụng Montage [2], CyberShake [3], Epigenomics [4], LIGO [5].

Phần tiếp theo của bài báo có cấu trúc như sau. Phần II giới thiệu một số công trình nghiên cứu liên quan đến bài toán lập lịch luồng công việc. Trong phần III chúng tôi trình bày mô hình lý thuyết biểu diễn năng lực tính toán và truyền thông của đám mây dựa trên mô hình lý thuyết này. Phần IV đề xuất: (i) Phương thức mới để cập nhật vị trí của cá thể; (ii) Toán tử lân cận mới cho thuật toán tìm kiếm Tabu nhằm thoát khỏi cực trị địa phương trong phương pháp PSO; và (iii) Thuật toán lập lịch mới tên là TSPSO. Phần V mô tả các thực nghiệm được tiến hành dựa trên công cụ mô phỏng Cloudsim [6] và phân tích những số liệu thực nghiệm thu được. Phần VI tóm tắt những kết quả chính của bài báo và hướng nghiên cứu trong tương lai.

II. NHỮNG CÔNG TRÌNH LIÊN QUAN

Bài toán lập lịch luồng công việc thuộc lớp NP-Khó nên việc tìm lời giải đúng cho các luồng công việc có số lượng tác vụ lớn là không khả thi. Có nhiều công trình nghiên cứu nhằm tìm ra lời giải gần đúng cho bài toán này. Trong [7], Dubey và các cộng sự đã đề xuất thuật toán lập lịch điều phối các tác vụ của luồng công việc trong môi trường điện toán đám mây dựa trên việc cải tiến thuật toán HEFT nhằm cực tiểu hóa thời gian hoàn thành luồng công việc, trong công trình nhóm tác giả đã trình bày tóm tắt các công trình nghiên cứu liên quan đến bài toán lập lịch luồng công việc và đề xuất một thuật toán lập lịch mới dựa trên thuật toán HEFT, kết quả thực nghiệm đã chỉ ra thuật toán mới có thời gian hoàn thành luồng công việc tốt hơn thuật toán CPOP và HEFT.

Manasrah và Ba Ali [8] đã đề xuất thuật toán lập lịch luồng công việc trong môi trường điện toán đám mây dựa trên kết hợp giữa thuật toán di truyền và thuật toán tối ưu bày đàn, trong công trình nhóm tác giả đã thực hiện khởi tạo quần thể ban đầu một cách ngẫu nhiên, sau đó thực hiện các toán tử cơ bản của thuật toán di truyền cho quần thể, tiếp theo sẽ áp dụng thuật toán tối ưu bày đàn dựa trên quần thể đã được tiến hóa bởi thuật toán di truyền. Kết quả thực nghiệm đã chỉ ra thuật toán đề xuất làm việc tốt hơn thuật toán GA và PSO.

Grigoreva [9] đã đề xuất thuật toán lập lịch điều phối các tác vụ của luồng công việc vào thực hiện trên một hệ thống đa bộ vi xử lý nhằm cực tiểu hóa thời gian hoàn thành luồng công việc. Tác giả đã sử dụng kết hợp phương pháp nhánh cận và kỹ thuật tìm kiếm nhị phân để tìm ra phương án xếp lịch có thời gian hoàn thành luồng công việc là nhỏ nhất.

Rajavel và Mala [10] đã đề xuất thuật toán lập lịch luồng công việc dựa trên nhu cầu của khách hàng như thời gian hoàn thành, chi phí thực thi, v.v. qua đó sẽ điều phối các tác vụ vào thực hiện trên các máy chủ nhằm thỏa mãn tốt nhất nhu cầu của khách hàng. Các tác giả trong bài báo [11] đã đề xuất thuật toán EGA (Enhanced Genetic Algorithm) lập lịch bằng phương pháp di truyền. Trong công trình các tác giả sử dụng thuật toán Enhanced Max Min trong bước khởi tạo quần thể nhằm tìm ra các cá thể tốt cho quá trình tiến hóa.

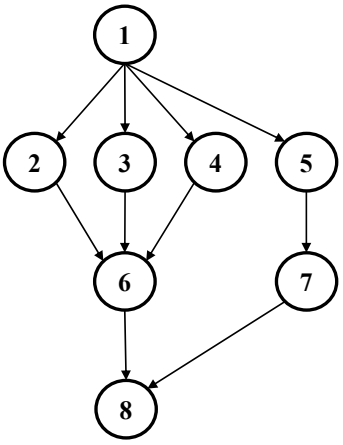
Pandey và cộng sự [12] đã đề xuất thuật toán lập lịch luồng công việc PSO Heuristic (PSO_H: Particle Swarm Optimization Heuristic) trong môi trường điện toán đám mây dựa trên chiến lược tối ưu bày đàn. Cụ thể, họ đã đề xuất một thuật toán lập lịch phân cấp và đưa vào các tham số dịch vụ khác nhau, chẳng hạn như thời gian đáp ứng. Thuật toán sử dụng tham số này như một quyền ưu tiên để lựa chọn các tác vụ lập lịch. Cao và cộng sự đã trình bày thuật toán lập lịch dựa trên giải thuật ABC (Activity Based Costing) [13]. Thuật toán này gán mức ưu tiên cho mỗi tác vụ trong luồng công việc theo các tham số về thời gian, không gian, các tài nguyên và chi phí, quá trình lập lịch sẽ sử dụng mức ưu tiên này để quyết định các tác vụ được chọn trong quá trình lập lịch.

III. MÔ HÌNH LÝ THUYẾT

Trong bài báo này, chúng tôi sử dụng một số ký hiệu sau:

- $\mathbf{T} = \{T_1, T_2, \ldots, T_M\}$ là tập các tác vụ.
- $\mathbf{S} = \{S_1, S_2, \ldots, S_N\}$ là tập N máy chủ trong môi trường điện toán đám mây; mỗi máy chủ S_i có một năng lực tính toán xác định bởi $P(S_i)$, đơn vị tính là MI/s (million instructions/second).
- W_i là khối lượng tính toán của tác vụ T_i, đơn vị tính là flop (floating point operations).
- Mỗi cặp máy chủ đều được kết nối với nhau bởi một đường truyền riêng có băng thông là $B(S_i, S_j)$, với $B(\cdot,\cdot)$ là hàm xác định băng thông $B : S \times S \rightarrow R^+$. Do tính chất của băng thông, hiển nhiên ta có:

$$\forall i, j : B(S_i, S_i) = \infty \text{ và } B(S_i, S_j) = B(S_j, S_i).$$



Hình 1. Đồ thị biểu diễn một luồng công việc với 8 tác vụ.

- Xuyên suốt bài báo, chúng tôi sử dụng hai tập chỉ số $\mathcal{M} = \{1, 2, \ldots, M\}$ và $\mathcal{N} = \{1, 2, \ldots, N\}$.

Đồ thị luồng công việc được biểu diễn bởi đồ thị có hướng không có chu trình $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, ví dụ như ở hình 1, trong đó $\mathbf{V}$ là tập đỉnh, mỗi đỉnh tương ứng với một tác vụ trong đồ thị luồng công việc và $\mathbf{E}$ là tập cạnh, biểu diễn mối quan hệ giữa các tác vụ. Nếu $e = (T_i, T_k)$ là một cạnh của đồ thị $\mathbf{G}$ thì T_i là tác vụ cha của tác vụ T_k, và tác vụ T_i sẽ gửi tới tác vụ T_k một khối lượng dữ liệu là $tdata^k$.

Định nghĩa 1 (Khái niệm lịch biểu): Mỗi lịch biểu được biểu diễn bởi hàm

$$f : \mathbf{T} \rightarrow \mathbf{S},$$

trong đó $f(T_i) \in \mathbf{S}$ là máy chủ thực hiện tác vụ T_i.

Dưới đây là một số tham số của mô hình:

- x_j^k là biến logic, với $x_j^k = 1$ nếu tác vụ T_k được gán vào thực hiện trên máy chủ S_j, nếu không thì $x_j^k = 0$.
- $d_{i,j}^k$ là khối lượng dữ liệu được truyền từ máy chủ S_i tới máy chủ S_j cho tác vụ T_k nếu $x_j^k = 1$.
- $tftime_{i,j}$ là thời gian truyền dữ liệu từ máy chủ S_i tới máy chủ S_j cho tác vụ T_k nếu $d_{i,j}^k > 0$ và $x_j^k = 1$:

$$tftime_{i,j} = \frac{d_{i,j}^k}{B(S_i, S_j)}.$$

- $extime_j^k$ là thời gian thực thi tác vụ T_k trên máy chủ S_j nếu $x_j^k = 1$:

$$extime_j^k = \frac{W_k}{P(S_j)}.$$

- ET^k là thời gian thực hiện tác vụ T_k:

$$ET^k = \sum_{i=1}^{N} \sum_{j=1}^{N} d_{i,j}^k \times tftime_{i,j} \times x_j^k + \sum_{j=1}^{N} extime_j^k \times x_j^k.$$

- $W_i / P(f(T_i))$ là thời gian tính toán tác vụ T_i với $i \in \mathcal{M}$.
- $D_{ij} / B(f(T_i), f(T_j))$ là thời gian truyền dữ liệu giữa tác vụ T_i và tác vụ con T_j.
- CT là thời gian hoàn thành luồng công việc (Makespan):

$$CT = \max_{k \in \mathcal{M}} \{ET^k\}.$$

Tiếp theo là các điều kiện ràng buộc của mô hình:

a) $x_j^k \geq 0$ với mọi $k \in \mathcal{M}$ và $j \in \mathcal{N}$;
b) $d_{i,j}^k \geq 0$ với mọi $i, j \in \mathcal{N}$ và $k \in \mathcal{M}$;
c) $tdata_j^k \geq 0$ với mọi $k \in \mathcal{M}$;
d) $tftime_{i,j} \geq 0$ với mọi $i, j \in \mathcal{N}$;
e) $extime_j^k \geq 0$ với mọi $k \in \mathcal{M}$ và $j \in \mathcal{N}$;
f) $\sum_{j=1}^{N} x_j^k = 1$;
g) $\sum_{i=1}^{N} \sum_{j=1}^{N} x_j^k \times d_{i,j}^k = tdata^k$; và
h) $\sum_{k=1}^{M} \sum_{i=1}^{N} \sum_{j=1}^{N} x_j^k \times d_{i,j}^k = \sum_{k=1}^{M} tdata^k$.

Định nghĩa 2 (Hàm mục tiêu): Hàm mục tiêu được xác định bằng cách tối thiểu hóa Makespan như sau:

$$\max_{k \in \mathcal{M}} \left\{ \sum_{i=1}^{N} \sum_{j=1}^{N} d_{i,j}^k \times tftime_{i,j} \times x_j^k + \sum_{j=1}^{N} extime_j^k \times x_j^k \right\} \rightarrow min$$

trong đó Makespan là thời gian hoàn thành luồng công việc, được tính từ khi tác vụ gốc được khởi động cho tới thời điểm tác vụ cuối cùng được thực hiện xong.

IV. GIẢI PHÁP ĐỀ XUẤT

1. Mã hóa cá thể

Theo phương pháp PSO, tại bước lặp thứ k, cá thể thứ i trong đàn được xác định bởi vector vị trí $\mathbf{x}_i^k$ (cho biết vị trí hiện tại) và vector dịch chuyển $\mathbf{v}_i^k$ (cho biết hướng dịch chuyển hiện tại). Trong bài toán xếp lịch đang xét, hai vector đó đều có số chiều bằng số tác vụ trong luồng công việc, ký hiệu là M. Cả vector vị trí và vector dịch chuyển đều được biểu diễn bằng cấu trúc dữ liệu bảng băm trong ngôn ngữ lập trình Java. Kí hiệu vector vị trí $\mathbf{x}_i = (\pi_{i1}, \pi_{i1}, \ldots, \pi_{iM})$ với $\pi_{ij} \in \mathbf{S}$ và $j \in \mathcal{M}$.

Ví dụ 1: Giả sử luồng công việc gồm tập tác vụ $\mathbf{T} = \{T_1, T_2, T_3, T_4, T_5, T_6, T_7\}$ và đám mây có tập máy chủ $\mathbf{S} = \{S_1, S_2, S_3\}$. Khi đó, cá thể $\mathbf{x}_i$ được biểu diễn bằng vector vị trí $(1; 2; 1; 3; 2; 3; 1)$ chính là phương án xếp lịch mà theo đó các tác vụ T_1, T_3, và T_7 được bố trí thực hiện bởi máy chủ S_1, tác vụ T_2 và T_5 được thực hiện trên S_2, còn tác vụ T_4 và T_6 được thực hiện bởi S_3 (Hình 2).

2. Phương pháp cập nhật vị trí cá thể

Xét công thức cập nhật vị trí cá thể theo công thức gốc của PSO:

$$\mathbf{v}_i^{k+1} = \omega \mathbf{v}_i^k + c_1 rand_1 \times (pbest_i - \mathbf{x}_i^k) + c_2 rand_2 \times (gbest - \mathbf{x}_i^k), \quad (1)$$

$$\mathbf{x}_i^{k+1} = \mathbf{x}_i^k + \mathbf{v}_i^k. \quad (2)$$

Hai công thức (1) và (2) cho thấy, giống như đa số các metaheuristic khác, PSO vốn được thiết kế cho dữ liệu liên tục, các thành phần của vector dịch chuyển $\mathbf{v}_i^k$ là số thực do công thức (1) tính vector dịch chuyển có những tham số là số thực như $rand_1$, $rand_2$, c_1, và c_2. Nhưng vì tập máy chủ $\mathbf{S}$ là hữu hạn và đếm được nên các thành phần của vector vị trí $\mathbf{x}_i$ phải là số nguyên để có thể ánh xạ tới một máy chủ nào đó nơi mà tác vụ tương ứng sẽ được thực hiện, chẳng hạn vector vị trí $\mathbf{x}_i$ trong ví dụ 1 có các thành phần là $\mathbf{x}_i[1] = 1$, $\mathbf{x}_i[2] = 2$, $\mathbf{x}_i[3] = 1$, $\mathbf{x}_i[4] = 3$ và $\mathbf{x}_i[5] = 2$. Hậu quả là hai vế của phép gán (2) khác kiểu nhau, vế trái, $\mathbf{x}_i^{k+1}[t]$, thuộc kiểu số nguyên còn vế phải, $\mathbf{x}_i^k[t] + \mathbf{v}_i^k[t]$, thuộc kiểu số thực.

Để giải quyết mâu thuẫn này, một số nghiên cứu trước đây như [12] đã làm tròn giá trị số thực ở vế phải rồi gán cho biến vị trí $\mathbf{x}_i^{k+1}t$ ở vế trái. Kết quả là nếu giá trị của vế phải là 3,2 thì phân phối tác vụ tới thực thi tại máy chủ có số thứ tự là 3, còn nếu vế phải là 3,8 thì tác vụ sẽ được phân cho máy chủ có số thứ tự là 4. Cách làm có vẻ tự nhiên này thực chất là gán một vị trí được tính toán cẩn thận theo chiến lược PSO cho máy chủ mà số thứ tự của nó tình cờ đúng bằng giá trị nguyên sau khi làm tròn. Cách làm như vậy đã phá hỏng quá trình tiến hóa từng bước của phương pháp PSO.

Để giải quyết vấn đề trên, bài báo này đề xuất cách giải quyết như sau: Giá trị thực của vế phải, $\mathbf{x}_i^k[t] + \mathbf{v}_i^k[t]$, sẽ được để nguyên không làm tròn, còn vế trái, $\mathbf{x}_i^{k+1}[t]$, sẽ được gán bởi định danh của máy chủ có tốc độ tính toán gần với giá trị của vế phải nhất so với các máy chủ còn lại. Làm như vậy tác vụ sẽ được gán cho máy chủ có năng lực phù hợp với giá trị được tính toán theo PSO. Như vậy,

$$\mathbf{x}_i^{k+1}[t] \leftarrow j$$

nếu, với mọi $S_r \in \mathbf{S}$ và $t \in \mathcal{M}$, ta có

$$|P(S_j) - (\mathbf{x}_i^k[t] + \mathbf{v}_i^k[t])| \leq |P(S_r) - (\mathbf{x}_i^k[t] + \mathbf{v}_i^k[t])|. \quad (3)$$

Ví dụ 2: Giả thiết tập máy chủ $\mathbf{S}$ trong ví dụ 1 có tốc độ tính toán được liệt kê trong bảng I. Giả sử ở bước thứ k+1 tổng $\mathbf{x}_i^k + \mathbf{v}_i^k = (4{,}4; 2{,}1; 6{,}7; 5{,}6; 10{,}2)$ thì vector vị trí $\mathbf{x}_i^{k+1}$ sẽ được gán bằng $(3; 1; 2; 2; 2)$. Nghĩa là cá thể đó tương ứng với phương án xếp lịch như mô tả trong hình 3. Thật vậy, thành phần thứ nhất của vector vị trí $\mathbf{x}_i^{k+1}[1]$ sẽ nhận

T_1	T_2	T_3	T_4	T_5	T_6	T_7
1	2	1	3	2	3	1

Hình 2. Mô tả luồng công việc trong ví dụ 1.

T_1	T_2	T_3	T_4	T_5
S_3	S_1	S_2	S_2	S_2

Hình 3. Mô tả phương án xếp lịch trong ví dụ 2.

Bảng I
TỐC ĐỘ TÍNH TOÁN CỦA CÁC MÁY CHỦ

Máy chủ S_i	**Tốc độ xử lý $P(S_i)$**
S_1	3,1
S_2	5,2
S_3	4,1

giá trị 3 ($\mathbf{x}_i^{k+1}[1] \leftarrow 3$), nghĩa là tác vụ T_1 sẽ được gán cho máy chủ S_3 bởi vì

$$|P(S_3) - 4{,}4| \leq |P(S_r) - 4{,}4| \ \forall S_r \in \mathbf{S}.$$

Nghĩa là, trong 3 máy chủ thì máy S_3 có tốc độ tính toán gần với giá trị 4,4 nhất so với 2 máy chủ còn lại, theo bảng I, do đó tác vụ T_1 được gán cho máy chủ S_3 để thực hiện, tức là $f(T_1) = S_3$. Phép gán tương tự cũng được thực hiện với bốn tác vụ còn lại là T_2, T_3, T_4 và T_5.

Vấn đề tương tự cũng xảy ra với phép trừ hai vector vị trí trong công thức (1): $(pbest_i - \mathbf{x}_i^k)$ và $(gbest - \mathbf{x}_i^k)$. Một số công trình hiện có như [10] chỉ đơn giản thực hiện phép trừ các thành phần số nguyên rồi gán cho máy chủ có số thứ tự tương ứng. Ví dụ nếu $pbest_i = (2; 4; 3; 3; 5)$ và $\mathbf{x}_i^k = (1; 3; 2; 1; 2)$ thì

$$pbest_i - \mathbf{x}_i^k = (2-1; 4-3; 3-2; 3-1; 5-2) = (1; 1; 1; 2; 3).$$

Như đã giải thích ở trên, cách làm này thực chất là gán các tác vụ cho những máy chủ mà số thứ tự của nó tình cờ đúng bằng kết quả phép trừ. Cách làm mang tính ngẫu nhiên như vậy đã phá hỏng quá trình từng bước tiếp cận tới vị trí cực trị của phương pháp PSO. Bài báo này đề xuất một "phép trừ vector" áp dụng riêng cho công thức (1) như sau. Giả sử

$$pbest_i = (x_{i1}, x_{i2}, \ldots, x_{iM}) \text{ với } x_{ik} \in \mathbf{S} \ \forall k,$$
$$\mathbf{x}_j = (x_{j1}, x_{j2}, \ldots, x_{jM}) \text{ với } x_{jk} \in \mathbf{S} \ \forall k.$$

Khi đó kết quả phép trừ $pbest_i - \mathbf{x}_j$ được tính như sau:

$$pbest_i - \mathbf{x}_j = (y_1, y_2, \ldots, y_M),$$

Thuật toán 1: Toán tử lân cận BE($\mathbf{x}_i, p_1, p_2$)

Dữ liệu vào: Vector $(\pi_1, \ldots, \pi_{p1}, \ldots, \pi_{p2}, \ldots, \pi_n)$.
Dữ liệu ra: Vector $(\pi_1, \ldots, \pi_{p2}, \ldots, \pi_{p1}, \ldots, \pi_n)$.

$S_1 \leftarrow remove(\pi, p1)$;
if $P(\pi_{p2}) > P(\pi_{p1})$ **then**
  $insert(\mathbf{x}_i, \pi, p_2 - 1, S_1)$;
else
  $insert(\mathbf{x}_i, \pi, p_2, S_i)$;
end
if $P(\pi_{p2}) > P(\pi_{p1})$ **then**
  $S_2 \leftarrow remove(\mathbf{x}_i, \pi, p_2)$;
else
  $S_2 \leftarrow remove(\mathbf{x}_i, \pi, p_{2+1})$;
end
$insert(\mathbf{x}_i, \pi, p_1 - 1, S_2)$;

với các thành phần y_k ($k \in \mathcal{M}$) là các số thực, được cho bởi

$$y_k = \left\{P(x_{ik}) + \frac{\sum_{q \in \mathbf{S}} B(x_{ik}, x_q)}{N-1}\right\} - \left\{P(x_{jk}) + \frac{\sum_{q \in \mathbf{S}} B(x_{jk}, x_q)}{N-1}\right\}.$$

Theo cách tính này, các máy chủ được xếp thứ tự theo tốc độ tính toán và băng thông của những đường truyền kết nối tới nó. Ví dụ 3 sau đây sẽ minh họa cụ thể hơn.

Ví dụ 3: Ta tiếp tục sử dụng tập máy chủ trong ví dụ 2. Giả sử $lbest_j = (2;1;2;1;1)$ và $\mathbf{x}_j = (3;2;1;2;1)$. Vậy ta có $lbest_j - \mathbf{x}_j = (y_1, y_2, y_3, y_4, y_5)$ với y_1 được tính như sau:

$$y_1 = \left\{P(S_2) + \frac{B(S_2, S_1) + B(S_2, S_3)}{3-1}\right\} - \left\{P(S_3) + \frac{B(S_3, S_1) + B(S_3, S_2)}{3-1}\right\}.$$

Cách tính tương tự được áp dụng cho $y_2, \ldots, y_5$.

3. Biện pháp thoát khỏi cực trị địa phương

Phương pháp PSO nói riêng và các phương pháp tìm kiếm tiến hóa nói chung đôi khi bị mắc kẹt tại các lời giải cực trị địa phương mà không thể thoát ra để đi tới lời giải tốt hơn. Bài báo này đề xuất các toán tử lân cận mới và sử dụng phương pháp tìm kiếm lân cận Tabu Search cho cá thể *gbest* để tìm kiếm phần tử *gbest* mới tốt hơn và qua đó định hướng quần thể dịch chuyển sang vùng tìm kiếm mới.

Thủ tục tìm kiếm lân cận: Tabu Search là phương pháp tìm kiếm lân cận được để xuất bởi Glove năm 1986 [14, 15]. Phương pháp này đã được ứng dụng vào tìm lời giải gần đúng cho nhiều bài toán tối ưu tổ hợp. Phương pháp Tabu Search bắt đầu từ một lời giải ngẫu nhiên của bài toán, sau đó sẽ áp dụng một số các toán tử lân cận để sinh ra lời giải mới, quá trình này được lặp đi lặp lại cho đến khi tìm ra lời giải tối ưu hoặc thỏa mãn điều kiện của bài toán. Bài báo này để xuất các toán tử lân cận mới sử dụng cho thuật toán Tabu Search.

Thuật toán 2: Toán tử BS($\mathbf{x}_i$)

Dữ liệu vào: Vector $(\pi_1, \pi_2, \ldots, \pi_n)$.
Dữ liệu ra: Vector $(\pi''_1, \pi''_2, \ldots, \pi''_n)$.

$S_i \leftarrow \max\{P(S_1), P(S_2), \ldots, P(S_n)\}$;
for $(i = 1;\ i < M;\ i++)$ **do**
  $M_i = \text{makespan}((\pi_1, \ldots, \pi_{i-1}, S_i, \pi_{i+1}, \ldots, \pi_n))$;
end
$k \leftarrow \min\{M_i\}$;
$j \leftarrow remove(\mathbf{x}_i, \pi, k)$;
$insert(\mathbf{x}_i, \pi, k+1, j)$;

Thuật toán 3: Tabu_Search($\mathbf{x}_i$)

Dữ liệu vào: Vector vị trí $\mathbf{x}_i$.
Dữ liệu ra: Vector vị trí $\mathbf{x}_k$ có $f(\mathbf{x}_k) < f(\mathbf{x}_i)$.

Khởi tạo bước lặp $t \leftarrow 0$;
while điều kiện lặp **do**
  Khởi tạo ngẫu nhiên r_1 và r_2 trong đoạn $[1, M]$;
  $\mathbf{x}_i \leftarrow \text{BE}(\mathbf{x}_i, r_1, r_2)$;
  $x_k \leftarrow \text{BS}(\mathbf{x}_i)$;
  if $f(\mathbf{x}_k) < f(\mathbf{x}_i)$ **then**
    return $\mathbf{x}_k$;
  else
    return $\mathbf{x}_i$;
  end
  $t \leftarrow t + 1$;
end

Gọi $remove(\mathbf{x}_i, \pi, p)$ là toán tử loại bỏ phần tử π_p trong vector vị trí của cá thể $\mathbf{x}_i = (\pi_1, \pi_2, \ldots, \pi_n)$. Sau khi thực hiện toán tử này ta có vector mới là $(\pi_1, \pi_2, \ldots, \pi_{p-1}, \pi_{p+1}, \ldots, \pi_n)$. Gọi $insert(\mathbf{x}_i, \pi, p, S_i)$ là toán tử chèn vào vị trí p trong vector vị trí của cá thể máy chủ mới là S_i. Sau khi thực hiện toán tử này ta có vector mới là $(\pi_1, \pi_2, \ldots, \pi_{p-1}, S_i, \pi_p, \ldots, \pi_n)$. Một toán tử lân cận Best Exchange (BE) hoán đổi vị trí của hai máy chủ tại vị trí p_1 và p_2 tùy theo năng lực của máy chủ được mô tả trong thuật toán 1.

Toán tử Best Swap (BS) tìm ra vị trí tốt nhất p trong một cá thể $\mathbf{x}_i = (\pi_1, \pi_2, \ldots, \pi_n)$ và thực hiện hoán chuyển hai vị trí p và $p+1$. Vị trí tốt nhất được tính toán dựa trên giá trị Makespan của phương án hiện thời. Toán tử BS được mô tả trong thuật toán 2.

4. Thuật toán đề xuất TSPSO

Tổng hợp những cải tiến nói trên, thuật toán đề xuất với tên gọi TSPSO được mô tả như trong thuật toán 4. Thuật toán hoạt động theo phương pháp PSO theo đó tại mỗi bước lặp các cá thể cập nhật vị trí của mình hướng tới vị trí tốt nhất của quần thể (*gbest*), đồng thời có dựa trên kinh nghiệm cá nhân ($pbest_i$). Nếu sau K thế hệ liên tiếp mà cả quần thể không cải thiện được một cách đáng kể giá trị *gbest* (mức chênh không vượt quá ω) thì chứng tỏ quần thể đang hội tụ tại một cực trị địa phương. Khi đó thủ tục Tabu_Search (Thuật toán 3) được gọi để tìm ra cá thể *gbest*

mới và cá thể này sẽ di cư cả quần thể tới một vùng không gian mới, tại đó quá trình tìm kiếm được tái khởi động.

Tiếp theo ta tìm hiểu độ phức tạp của thuật toán TSPSO. Trước khi thực hiện thuật toán chính TSPSO ta cần phải sắp xếp các máy chủ thực thi theo thứ tự tăng dần của tốc độ thực hiện, giải thuật sắp xếp có độ phức tập về thời gian là $O(n\log(n))$. Thủ tục tính ma trận thời gian thực thi của mỗi tác vụ trên các máy chủ có độ phức tạp thời gian tính toán là $O(M \times N)$, trong đó M là số tác vụ, N là số máy chủ. Thủ tục tính ma trận thời gian truyền dữ liệu giữa các máy chủ có độ phức tạp tính toán là $O(N^2)$. Trong thuật toán TSPSO thì thủ tục khởi tạo sẽ khởi tạo các cá thể của quần thể một cách ngẫu nhiên, mỗi cá thể được mã hóa bởi một véc tơ độ dài M, do vậy độ phức tạp của thủ tục khởi tạo là $O(M \times SCT)$, trong đó SCT là số cá thể trong quần thể; trong thực nghiệm chúng tôi sử dụng $SCT = 100$. Hàm tính thời gian thực hiện (makespan) của mỗi phương án xếp lịch là $O(M^2)$. Thuật toán Tabu_Search có độ phức tạp là $O(M^2)$. Trong bài toán lập lịch luồng công việc, thường số tác vụ lớn hơn số máy chủ ($M > N$), do vậy độ phức tạp của thuật toán TSPSO là (Số thế hệ) $\times O(M^2)$.

Thuật toán 4: Thuật toán TSPSO

Dữ liệu vào: Tập **T**, tập **S**, mảng $W[1\times M]$, mảng $P[1\times N]$, mảng $B[N\times N]$, mảng $D[M\times M]$, hằng số K, độ lệch ω, và số cá thể SCT
Dữ liệu ra: Lời giải tốt nhất $gbest$

```
Khởi tạo ngẫu nhiên vector vị trí và vector dịch chuyển của cá thể i;
Khởi tạo bước lặp t ← 0;
while điều kiện lặp do
    for i = 1 to SCT do
        Tính vector vị trí x_i theo (3);
        M_i = Makespan của cá thể x_i;
    end
    for i = 1 to SCT do
        if pbest_i < M_i then
            pbest_i = M_i;
        end
    end
    gbest = min{M_i};
    for i = 1 to SCT do
        Cập nhật vector v_i^k theo (1) và (3);
        Tính x_i theo (2);
    end
    t ← t + 1;
    if sau K thế hệ mà độ lệch giữa các gbest không vượt quá ω then
        gbest = Tabu_Search(gbest);
    end
end
return gbest;
```

V. KẾT QUẢ THỰC NGHIỆM

1. Phân nhóm dữ liệu thực nghiệm

Dữ liệu về tốc độ tính toán của các máy chủ và băng thông giữa các máy chủ được lấy từ các công ty cung cấp dịch vụ cloud [16] và địa chỉ website http://aws.amazon.com/ec2/pricing.

Dữ liệu luồng công việc được lấy từ các bộ dữ liệu thử nghiệm được xây dựng theo độ trù mật khác nhau và các luồng công việc từ các ứng dụng thực tế như ứng dụng Montage [17].

Những dữ liệu đó được tổng hợp lại và chia thành hai nhóm như sau. Nhóm 1 là các luồng công việc ngẫu nhiên với sự khác nhau về hệ số α và nhóm 2 là các luồng công việc từ ứng dụng Montage. Hệ số α được xác định như sau:

$$\alpha = \frac{|E|}{M \times (M-1)/2}.$$

Tham số α cho biết đồ thị **G** phân thành bao nhiêu cấp, mỗi cấp có nhiều hay ít tác vụ, nói cách khác α phản ánh độ trù mật của đồ thị **G**. Khi làm thực nghiệm với mỗi nhóm, số máy chủ và số tác vụ được giữ cố định còn tỷ lệ α lần lượt thay đổi như trong hình 4.

2. Tham số cấu hình hệ thống

Các tham số cấu hình của đám mây được thiết lập trong miền giá trị như sau: tốc độ tính toán của các máy chủ $P_i \in [1; 250]$ (MI/s); khối lượng dữ liệu giữa các tác vụ $D_{ij} \in [1; 10.000]$ (Mb); băng thông giữa các máy chủ $B(S_i, S_j) \in [10; 100]$ (Mb/s); hệ số quán tính $\omega = 0,729$; hệ số gia tốc $c_1 = c_2 = 1,49445$; hằng số $K = 30$; số cá thể $SCT = 25$; độ lệch $\epsilon = 0,21$; hệ số $\alpha \in [0,2; 0,7]$.

3. Quá trình tiến hành thực nghiệm

Để kiểm chứng thuật toán đề xuất TSPSO chúng tôi đã sử dụng công cụ mô phỏng Cloudsim [6] để tạo lập môi trường đám mây kết hợp với dữ liệu luồng công việc của ứng dụng Montage [17]. Các hàm của gói thư viện Jswarm [6] được sử dụng để thực hiện các phương thức tối ưu bày đàn. Đối tượng so sánh là các thuật toán PSO_H [12], EGA [18] và Round Robin [19].

Các chương trình mô phỏng được viết bằng ngôn ngữ Java và chạy trên máy tính cá nhân với bộ vi xử lý Intel Core i5, 2, 2 GHz, RAM 4 GB, và hệ điều hành Windows 7 Ultimate. Thực nghiệm được tiến hành một cách độc lập 30 lần trên mỗi bộ dữ liệu thực nghiệm.

4. Kết quả thực nghiệm

Hình 4 cho thấy sự chênh lệch về thời gian xử lý (makespan) của lời giải tốt nhất mà thuật toán đề xuất TSPSO và các thuật toán đối chứng (PSO_H, EGA, và Round Robin) tìm được khi chạy trên các bộ dữ liệu khác nhau thuộc cả 2 nhóm luồng công việc ngẫu nhiên và luồng

Bảng II
KẾT QUẢ THỰC HIỆN THUẬT TOÁN VỚI CÁC BỘ DỮ LIỆU NGẪU NHIÊN

Ký hiệu	RRTSM			PSO_H			TSPSO			EGA		
	STD	Mean	Best	STD	Mean	Best	STD	Mean	Best	STD	Mean	Best
T532	-	18,7	18,7	4,7	9,9	7,1	1,0	7,2	7,0	3,4	9,8	7,0
T1031	-	33,1	33,1	2,4	20,4	17,4	1,1	18,3	16,5	1,2	20,4	16,8
T1032	-	16	16	2,0	8,2	3,5	1,2	5,1	3,5	1,8	8,1	3,5
T1035	-	18,9	18,9	2,5	9,7	5,2	1,8	6,8	3,9	2,3	9,7	4,9
T1051	-	23,7	23,7	1,5	21,5	16,4	1,1	17,6	14,8	1,4	19,1	16,1
T1054	-	25,2	25,2	2,0	20,7	18,9	0,9	19,0	17,3	1,3	20,5	17,6
T2081	-	72,7	72,7	5,2	44,2	34,1	2,7	37,8	32,4	3,7	41,9	35,1
T2083	-	20,3	20,3	2,2	21,2	19,8	0,5	19,4	18,2	1,4	20,3	19,6
T2084	-	72,0	72,0	6,1	44,7	37,4	2,7	39,3	34,1	3,2	42,5	36,5
T2086	-	32,5	32,5	1,5	26,2	22,8	1,2	21,4	17,4	1,4	25,4	22,6

Bảng III
KẾT QUẢ THỰC HIỆN THUẬT TOÁN VỚI CÁC BỘ DỮ LIỆU TỪ ỨNG DỤNG MONTAGE

Ký hiệu	RRTSM			PSO_H			TSPSO			EGA		
	STD	Mean	Best	STD	Mean	Best	STD	Mean	Best	STD	Mean	Best
M2032	-	162,7	162,7	4,9	142,7	131,6	3,6	123,4	129,0	5,3	135,5	130,1
M2051	-	146,6	146,6	5,4	132	121,8	3,3	123,4	115,7	4,5	131,7	123,3
M2531	-	465,0	465,0	18,7	373,4	345,5	13,0	346,4	336,1	7,0	352,7	339,4
M2532	-	183,8	183,8	3,9	110,7	101,5	1,5	104,7	101,2	1,4	112,1	104,5
M2533	-	322,9	322,9	4,0	311,4	311,7	0,5	312,5	311,6	0,5	315,2	311,8
M2581	-	300,6	300,6	15	261,3	232,1	6,1	236,1	223,4	6,5	246,2	231,9
M2582	-	133,9	133,9	5,1	84,8	77,9	4,7	81,4	72,3	2,9	85,1	77,6
M2583	-	236,5	236,5	8,3	239	224	5,3	221,3	215,7	4,7	233,5	221,4
M5081	-	155,8	155,8	6,3	108,0	95,0	5,5	101,7	91,5	3,2	107,8	96,6
M5082	-	82,1	82,1	0,9	14,0	18,1	0,8	12,6	13,1	0,5	14,0	14,8
M5083	-	101,7	101,7	4,3	98,3	89,8	4,3	90,0	80,2	1,8	95,3	88,5

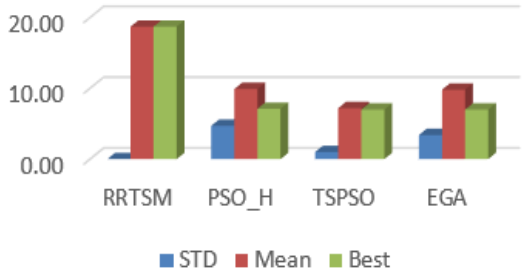
công việc từ ứng dụng Montage. Kết quả thực nghiệm được trình bày chi tiết trong các bảng II và III và hình 4. Kết quả so sánh giữa giá trị trung bình tính được bởi thuật toán TSPSO với các thuật toán đối sánh, trong hầu hết các trường hợp thuật toán TSPSO đều cho kết quả tốt hơn các thuật toán đối sánh, giá trị trung bình tìm được bởi TSPSO nhỏ hơn giá trị trung bình tìm được bởi PSO_H từ 4%–11% và nhỏ hơn giá trị trung bình tìm được bởi thuật toán EGA từ 2%–7%.

Các hình này cũng so sánh giữa giá trị tốt nhất tìm được bởi thuật toán TSPSO với các thuật toán đối sánh, qua đó ta thấy giá trị tốt nhất tìm được bởi TSPSO nhỏ hơn giá trị tốt nhất tìm được bởi PSO_H từ 2%–9%, và nhỏ hơn giá trị tốt nhất tìm được bởi *rand*om từ 20%–40%; giá trị tốt nhất tìm được bởi thuật toán TSPSO nhỏ hơn giá trị tốt nhất tìm được bởi thuật toán EGA từ 1%–8%. Kết quả so sánh giữa độ lệch chuẩn tìm được bởi thuật toán TSPSO với các thuật toán đối sánh, giá trị độ lệch chuẩn của TSPSO đều nhỏ hơn độ lệch chuẩn của các thuật toán RRTSM, PSO_H và EGA, điều đó chứng tỏ thuật toán TSPSO có chất lượng lời giải tốt hơn các thuật toán đối sánh và độ ổn định trong các lần chạy cũng tốt hơn.

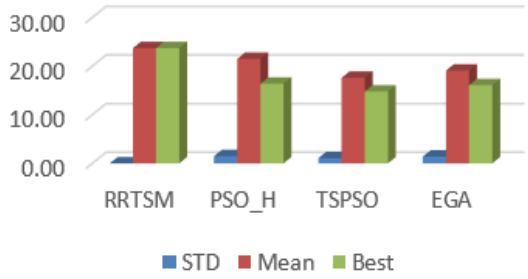
VI. KẾT LUẬN

Bài báo này đã trình bày một kiến trúc lân cận mới cho thuật toán Tabu Search và thuật toán Tối ưu bày đàn để tìm lời giải gần đúng cho bài toán lập lịch thực thi luồng công việc trong môi trường điện toán đám mây. Những kết quả chính gồm có:

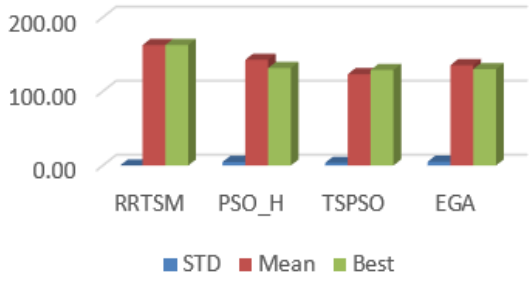
- Đề xuất một phương thức mới để cập nhật vị trí của cá thể bằng cách ánh xạ một giá trị thực tới máy chủ có tốc độ tính toán và băng thông gần với giá trị đó nhất.
- Đề xuất công thức tính vector dịch chuyển của cá thể thứ i theo giá trị *gbest* và $pbest_i$.
- Đề xuất hai toán tử lân cận mới cho thuật toán tìm kiếm lân cận Tabu Search để chương trình thoát ra khỏi cực trị địa phương bằng cách dịch chuyển các cá thể tới một miền không gian tìm kiếm mới.



(a) T532



(b) T1051



(c) M2032

Hình 4. So sánh các thuật toán với các bộ dữ liệu khác nhau: T532, T1051 và M2032.

- Đề xuất thuật toán TSPSO sử dụng phương thức cập nhật vị trí cá thể và thủ tục Tabu Search để tìm kiếm lời giải cho bài toán lập lịch thực thi luồng công việc trong môi trường đám mây.

Những kết quả thực nghiệm được tiến hành với nhiều bộ dữ liệu thực nghiệm khác nhau đã chứng tỏ chất lượng lời giải tìm được bởi thuật toán đề xuất tốt hơn so với các thuật toán đối chứng là thuật toán PSO_H, thuật toán EGA và thuật toán Round Robin. Về hướng công việc tiếp theo chúng tôi dự định đề xuất một kiến trúc lân cận mới phù hợp với bài toán nhằm nâng cao khả năng tìm kiếm tổng thể qua đó nhằm đạt được lời giải có chất lượng tốt hơn.

TÀI LIỆU THAM KHẢO

[1] J. D. Ullman, "NP-complete scheduling problems," *Journal of Computer and System sciences*, vol. 10, no. 3, pp. 384–393, 1975.

[2] G. B. Berriman, E. Deelman, J. C. Good, J. C. Jacob, D. S. Katz, C. Kesselman *et al.*, "Montage: a grid-enabled engine for delivering custom science-grade mosaics on demand," in *Optimizing Scientific Return for Astronomy through Information Technologies*, vol. 5493, 2004, pp. 221–232.

[3] P. Maechling, E. Deelman, L. Zhao, R. Graves, G. Mehta, N. Gupta *et al.*, "SCEC CyberShake workflows—automating probabilistic seismic hazard analysis calculations," in *Workflows for e-Science*. Springer, 2007, pp. 143–163.

[4] USC Epigenome Center. [Online]. Available: http://epigenome.usc.edu.

[5] LIGO - Laser Interferometer Gravitational Wave Observatory. [Online]. Available: http://www.ligo.caltech.edu.

[6] R. Buyya, R. Ranjan, and R. N. Calheiros, "Modeling and simulation of scalable cloud computing environments and the CloudSim toolkit: Challenges and opportunities," in *IEEE International Conference on High Performance Computing & Simulation*, 2009, pp. 1–11.

[7] K. Dubey, M. Kumar, and S. Sharma, "Modified HEFT algorithm for task scheduling in cloud environment," *Procedia Computer Science*, vol. 125, pp. 725–732, 2018.

[8] A. M. Manasrah and H. Ba Ali, "Workflow scheduling using hybrid GA-PSO algorithm in cloud computing," *Wireless Communications and Mobile Computing*, vol. 2018, no. 1934784, pp. 1–16, 2018.

[9] N. Grigoreva, "Branch and bound method for scheduling precedence constrained tasks on parallel identical processors," in *Proceedings of The World Congress on Engineering*, 2014, pp. 832–836.

[10] R. Rajavel and T. Mala, "Achieving service level agreement in cloud environment using job prioritization in hierarchical scheduling," in *International Conf. on Information Systems Design and Intelligent Applications*, 2012, pp. 547–554.

[11] S. Singh and M. Kalra, "Task scheduling optimization of independent tasks in cloud computing using enhanced genetic algorithm," *Int'l J. Application or Innovation in Engineering & Management*, vol. 3, no. 7, pp. 2319–4847, 2014.

[12] S. Pandey, L. Wu, S. M. Guru, and R. Buyya, "A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments," in *IEEE International Conference on Advanced Information Networking and Applications*, 2010, pp. 400–407.

[13] Q. Cao, Z.-B. Wei, and W.-M. Gong, "An optimized algorithm for task scheduling based on activity based costing in cloud computing," in *IEEE International Conference on Bioinformatics and Biomedical Engineering*, 2009, pp. 1–3.

[14] F. Glover, "Tabu search– part I," *ORSA Journal on Computing*, vol. 1, no. 3, pp. 190–206, 1989.

[15] D. de Werra and A. Hertz, "Tabu search techniques: A tutorial and an application to neural networks," *Operations Research Spektrum*, vol. 11, no. 3, pp. 131–141, 1989.

[16] J. Vliet and F. Paganelli, *Programming Amazon EC2*. O'Reilly Media, Inc, 2011.

[17] Q. Jiang, Y. C. Lee, M. Arenaz, L. M. Leslie, and A. Y. Zomaya, "Optimizing scientific workflows in the cloud: A Montage example," in *IEEE International Conference on Utility and Cloud Computing*, 2014, pp. 517–522.

[18] R. Kaur and S. Kinger, "Enhanced genetic algorithm based task scheduling in cloud computing," *International Journal of Computer Applications*, vol. 101, no. 14, pp. 1–6, 2014.

[19] E. Upfal, *Probability and computing: Randomized algorithms and probabilistic analysis*. Cambridge University Press, 2005.



Phan Thanh Toàn sinh năm 1974 tại Thái Nguyên, tốt nghiệp Đại học và Thạc sĩ tại Trường Đại học Bách khoa Hà Nội và nhận bằng Tiến sĩ năm 2018 tại Học viện Khoa học Công nghệ Quân sự. Tác giả hiện đang công tác tại trường Đại học Sư phạm Hà Nội. Lĩnh vực nghiên cứu bao gồm phương pháp tiến hóa, tối ưu, xử lý song song và phân tán.



Đặng Quốc Hữu tốt nghiệp Đại học và Thạc sĩ tại Khoa Công nghệ Thông tin, Đại học Quốc gia Hà Nội năm 2006 và 2015. Tác giả hiện đang công tác tại Trường Đại học Thương mại, đồng thời là nghiên cứu sinh tại Viện Khoa học và Công nghệ Quân sự từ năm 2017. Lĩnh vực nghiên cứu bao gồm mạng máy tính và truyền thông, xử lý song song và phân tán.



Nguyễn Thế Lộc tốt nghiệp Đại học tại Khoa Toán–Tin, Trường Đại học Sư phạm Hà Nội năm 1993, Thạc sĩ Công nghệ Thông tin tại Trường Đại học Bách khoa Hà Nội, nhận bằng Tiến sĩ tại Viện Nghiên cứu Khoa học Công nghệ Nhật Bản (JAIST) năm 2007. Tác giả hiện đang công tác tại Trường Đại học Sư phạm Hà Nội. Lĩnh vực nghiên cứu bao gồm mạng máy tính và truyền thông, xử lý song song và phân tán.

Một phương pháp sinh luật mờ dựa trên cây quyết định và đại số gia tử xây dựng hệ luật mờ giải bài toán hồi quy

Nguyễn Đức Dư, Hoàng Văn Thông

Khoa Công nghệ Thông tin, Trường Đại học Giao thông Vận tải, Hà Nội

Tác giả liên hệ: Nguyễn Đức Dư, nducdu@utc.edu.vn

Ngày nhận bài: 12/11/2019, ngày sửa chữa: 24/12/2019, ngày duyệt đăng: 25/12/2019

Định danh DOI: 10.32913/mic-ict-research-vn.v2019.n2.901

Biên tập lĩnh vực điều phối phản biện và quyết định nhận đăng: PGS.TS. Võ Đình Bảy

Tóm tắt: Bài báo này đề xuất một phương pháp sinh luật mờ dựa trên cây quyết định và đại số gia tử để xây dựng hệ luật mờ giải bài toán hồi quy. Thuật toán được thử nghiệm trên 9 bài toán mẫu và đối sánh với các phương pháp đã có PAES_KB và HA-MG-PAES-Kmax trên các mục tiêu độ chính xác và độ phức tạp của hệ luật. Kết quả đối sánh cho thấy thuật toán đề xuất cho kết quả tốt hơn.

Từ khóa: *Hệ luật mờ, đại số gia tử, cây quyết định, bài toán hồi quy.*

Title: **A Method to Generate Fuzzy Rules based on Decision Tree and Hedge Algebras for Building Fuzzy Rule based Systems for Regression**

Abstract: This paper proposes a method to generate fuzzy rules based on decision tree and hedge algebras for building fuzzy rule-based systems for regression problems. The proposed method was experimented on nine regression problems and was compared to two existing methods PAES_KB and HA-MG-PAES-Kmax on two objectives including the accuracy and complexity of rule-based systems. Comparative results show that our proposed method outperforms the existing ones.

Keywords: *Fuzzy rule-based system, hedge algebra, decision tree, regression.*

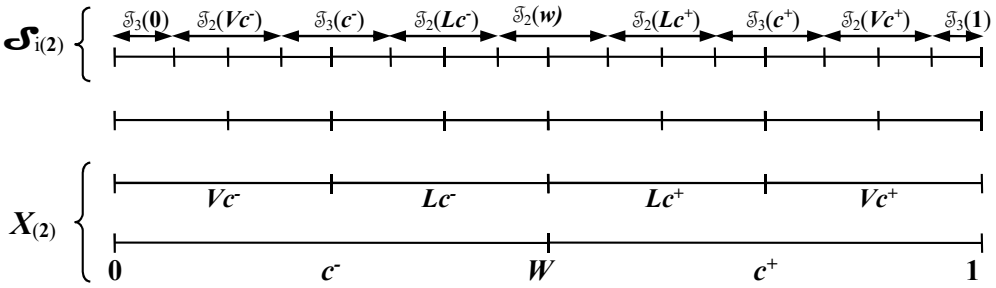
I. MỞ ĐẦU

Cách giải quyết các bài toán phân lớp và hồi quy bằng hệ luật mờ (FRBS: Fuzzy Rule based System) nhận được nhiều sự quan tâm của các nhóm nghiên cứu, như: Acalá và cộng sự [1], Antonelli và cộng sự [2], Ishibuchi và Nojima [3, 4], Pulkkinen và Koivisto [5], Cordón và cộng sự [6], Nguyễn Cát Hồ và cộng sự [7–9], Aghaeipoor và Javidi [10]. Các nghiên cứu đã đề xuất chủ yếu tập trung tìm kiếm các phương pháp xây dựng FRBS cho bài toán phân lớp, còn bài toán hồi quy chưa có nhiều nghiên cứu đề cập tới [1]. Khi xây dựng FRBS giải các bài toán này chúng ta chủ yếu giải quyết ba vấn đề chính: thiết kế phân hoạch mờ (ngữ nghĩa tính toán của từ), sinh tập các luật mờ ứng cử, tìm kiếm hệ luật mờ tối ưu.

Về vấn đề sinh luật ứng cử, các phương pháp tiếp cận dựa trên lý thuyết tập mờ thường sinh luật bằng cách tổ hợp tất cả các giá trị ngôn ngữ sử dụng cho các biến [1–4, 11]. Nhược điểm của hướng tiếp cận này là khi tập dữ liệu có nhiều thuộc tính thì không gian tìm kiếm luật sẽ rất lớn. Một số đề xuất sinh luật từ cây quyết định (decision tree) cho bài toán phân lớp [5, 12–14]. Phương pháp này đã làm giảm đáng kể số luật phải xem xét nhờ vào các kỹ thuật như là hạn chế chiều cao và cắt tỉa cây, tuy nhiên lại gặp khó khăn trong quá trình tối ưu tham số tập mờ.

Với hướng tiếp cận theo lý thuyết đại số gia tử (ĐSGT), trong [7–9] Nguyễn Cát Hồ và cộng sự đề xuất một phương pháp sinh luật từ mẫu dữ liệu. Theo đó, mỗi mẫu dữ liệu sinh ra một luật có độ dài n bằng số thuộc tính của tập mẫu dữ liệu, từ các luật này sinh các luật có độ dài $lmax$ nhỏ hơn cho trước ($lmax < n$). Với phương pháp sinh luật ứng cử này thì số luật tối đa phải xem xét giảm đi đáng kể so với phương pháp sinh luật tổ hợp. Tuy nhiên theo hướng tiếp cận này chúng ta vẫn phải xem xét một số lượng luật khá lớn.

Trong bài báo này chúng tôi đề xuất một phương pháp xây dựng FRBS giải bài toán hồi quy với các luật được sinh ra bằng cây quyết định và ĐSGT. Thuật toán giải quyết cả hai vấn đề sinh luật và tối ưu tham số của các tập mờ.



Hình 1. Hệ khoảng tương tự của các từ có độ dài không quá 2.

Thuật toán đề xuất gồm hai pha. Pha thứ nhất tối ưu tham số của ĐSGT sử dụng cho mỗi biến của bài toán. Ở pha này chúng tôi sử dụng thuật giải di truyền để tìm kiếm tham số tối ưu. Pha thứ hai, với bộ tham số tối ưu của ĐSGT tìm được của pha thứ nhất, chúng tôi xây dựng các ĐSGT và sử dụng chúng để chuyển đổi tập dữ liệu giá trị số của bài toán thành tập dữ liệu giá trị ngôn ngữ tương ứng. Từ tập dữ liệu ngôn ngữ cây quyết định được xây dựng, từ cây quyết định sinh ra tập luật ứng cử và sử dụng thuật toán cải tiến (2+2)M-PAES được sử dụng để tìm FRBS tối ưu.

Thuật toán đề xuất được thử nghiệm trên 9 bài toán hồi quy và đối sánh kết quả thu được trên các chỉ số độ phức tạp của hệ luật và giá trị sai số trung bình phương với các thuật toán cùng hướng tiếp cận PAES_{KB} [1] và HA-MG-PAES-Kmax [8]. Kết quả đối sánh cho thấy thuật toán đề xuất cho kết quả có độ chính xác tốt hơn.

Phần còn lại của bài báo được bố cục như sau. Phần II trình bày bài toán hồi quy và phương pháp giải bài toán hồi quy bằng hệ mờ dựa trên đại số gia tử. Phần III trình bày phương pháp sinh luật mờ dựa trên cây quyết định và ĐSGT. Phần IV trình bày thuật toán xây dựng hệ luật. Phần V trình bày kết quả thử nghiệm. Cuối cùng, Phần VI rút ra một số kết luận.

II. BÀI TOÁN HỒI QUY VÀ HỆ LUẬT MỜ MAMDANI DỰA TRÊN ĐSGT

1. Bài toán hồi quy

Cho một tập mẫu dữ liệu $\mathbf{D} = \{(x_i, y_i),\ i = 1, 2, \ldots, N\}$, trong đó x_i là một véc-tơ n chiều có dạng $(x_{i1}, x_{i2}, \ldots, x_{in})$ với $x_{ij} \in U_j \subset \mathbb{R}$, U_j là miền xác định của các biến độc lập X_j (thuộc tính đầu vào) của bài toán ($j = 1, 2, \ldots, n$), $y_i \in U_{n+1} \subset \mathbb{R}$, U_{n+1} là miền xác định của biến phụ thuộc Y (thuộc tính đầu ra), N là số mẫu dữ liệu.

Từ tập dữ liệu mẫu $\mathbf{D}$ xây dựng một hệ mờ dựa trên luật cho phép tính giá trị $\hat{y}$ ứng với mỗi giá trị đầu vào $x \in U = U_1 \times \ldots \times U_n$.

2. Giải bài toán hồi quy bằng hệ luật mờ Mamdani dựa trên ĐSGT

Giải bài toán hồi quy bằng hệ luật mờ Mamdani là xây dựng một hệ các luật mờ Mamdani có dạng

$$R_m: \quad \text{if } X_1 \text{ is } A_{1,j_m}, \ldots, X_n \text{ is } A_{n,j_m} \quad \text{then } Y \text{ is } A_{n+1,j_m}, \tag{1}$$

để dự đoán giá trị đầu ra $\hat{y}$ ứng với giá trị đầu vào x có n chiều. Trong (1),

$$A_{i,j_m} \in L_i = \left\{\{A_{i,0}\} \cup X_{(k_i)} = \{A_{i,1}, A_{i,2}, \ldots, A_{i,|X_{(k_i)}|}\}\right\},$$

với $i = 1, 2, \ldots, n+1$, trong đó $X_{(k_i)}$ là tập các từ ngôn ngữ có độ dài không quá k_i được sinh ra bằng ĐSGT $\mathcal{AX}_i$ tương ứng, và nó được sử dụng để xây dựng phân hoạch thuộc tính thứ i, $A_{f,0}$ có giá trị là *Don't care* với hàm thuộc đồng nhất bằng một. Chú ý, L_{n+1} không chứa giá trị *Don't care* và $m = 1, 2, \ldots, M$, với M là số luật của hệ.

Tương tự như các đề xuất trong [1, 2, 8, 9], chúng tôi sử dụng phương pháp trung bình trọng số để suy diễn giá trị $\hat{y}$ từ hệ luật khi biết véc-tơ đầu vào x_i theo công thức sau:

$$\hat{y}_i = \frac{\sum_{m=1}^{M} \mu_m(x_i) \bar{A}_{n+1,j_m}}{\sum_{m=1}^{M} \mu_m(x_i)}, \tag{2}$$

với $i = 1, 2, \ldots, N$, trong đó

$$\mu_m(x_i) = \prod_{f=1}^{n} \mu_{A_{f,j_m}}(x_{if})$$

là độ đốt cháy luật thứ m của mẫu dữ liệu x_i, $\bar{A}_{n+1,j_m}$ là giá trị định lượng của hạng từ ngôn ngữ A_{n+1,j_m} và $\mu_{A_{f,j_m}}(\cdot)$ là hàm thuộc của từ ngôn ngữ A_{f,j_m}. Trong (2), nếu $\sum_{m=1}^{M} \mu_m(x_i) = 0$, có nghĩa là hệ luật *không phủ* mẫu dữ liệu x_i, khi đó $\hat{y}$ được suy diễn theo phương pháp đề xuất trong [1].

Để đánh giá độ chính xác của hệ luật đã xây dựng, chúng tôi dựa vào giá trị sai số trung bình phương (MSE: Mean Squared Error):

$$\text{MSE} = \frac{1}{2N} \sum_{i=1}^{N} (\hat{y}_i - y_i)^2. \tag{3}$$

Thuật toán 1: BuildDecisionTree($\mathbf{D}, \{\pi_i\}_{i=1}^{n+1}, k, lmax$)

Dữ liệu vào: Cơ sở dữ liệu của bài toán **D**; bộ tham số của các ĐSGT $\{\pi_i\}_{i=1}^{n+1}$; chiều dài tối đa của các hạng từ sinh ra từ ĐSGT k; chiều cao tối đa của cây $lmax$.
Dữ liệu ra: Cây quyết định **T**.
for $i = 1$ **to** $n+1$ **do**
  Xây dựng ĐSGT $\mathcal{AX}_i$ tương ứng với bộ tham số π_i;
  Sinh tập các từ $X_{(k_i)}$;
  Sinh hệ khoảng tính mờ tương tự $S_{(k_i)}$;
end
D_1 = Chuẩn hóa **D** về đoạn $[0, 1]$;
D_2 = Chuyển đổi D_1 thành cơ sở dữ liệu ngôn ngữ dựa trên các hệ khoảng tính mờ $\{S_{(k_i)}\}_{i=1}^{n+1}$;
Xây dựng cây quyết định **T** có chiều cao tối đa $lmax$ từ cơ sở dữ liệu D_2 bằng thuật toán C4.5;
return T;

Trong (3), $\hat{y}_i$ và y_i lần lượt là giá trị suy diễn từ hệ luật và giá trị quan sát tương ứng với giá trị đầu vào x_i. Giá trị MSE càng nhỏ thì hệ luật mờ càng chính xác.

III. ĐỀ XUẤT PHƯƠNG PHÁP SINH HỆ LUẬT MỜ DỰA TRÊN CÂY QUYẾT ĐỊNH VÀ ĐSGT

1. Hệ khoảng tính mờ tương tự của từ ngôn ngữ

ĐSGT được Nguyễn Cát Hồ và cộng sự đề xuất lần đầu tiên trong [15], xây dựng một phương pháp hình thức sinh các từ ngôn ngữ và ánh xạ từ ngôn ngữ tới các giá trị định lượng tương ứng trên đoạn $[0; 1]$. ĐSGT giúp chúng ta có thể sinh ra các dạng ngữ nghĩa khác nhau của từ ngôn ngữ từ ngữ nghĩa vốn có của từ bao gồm: giá trị định lượng $v_x(x)$, khoảng tính mờ $fm(x)$, khoảng tương tự [7]. Gọi $X_{(k_i)}$ là tập các từ có độ dài không quá k_i của biến X_i. Tập $S_{(k_i)}$ các khoảng tương tự của các từ trong $X_{(k_i)}$ hình thành một phân hoạch trên U và giá trị định lượng ngữ nghĩa của mọi từ $x \in X_{(k_i)}$ là $v_x(x) \in T(x)$, ở đó $T(x)$ là *khoảng tương tự* của từ x. Các giá trị của khoảng tương tự $T(x)$ được coi như là tương tự nhau và với giá trị định lượng ngữ nghĩa $v_x(x)$ của x với một cấp độ k_i, k_i càng lớn mức độ tương tự của các giá trị trong mỗi khoảng tương tự càng cao. Hệ khoảng tượng tự là một công cụ hữu dụng để phân hoạch miền tham chiếu của các biến, được sử dụng trong các thuật toán sinh luật của các phương pháp tiếp cận dựa trên ĐSGT.

2. Thuật toán xây dựng cây quyết định

Thuật toán xây dựng cây quyết định được trình bày trong thuật toán 1. Để xây dựng cây quyết định từ tập dữ liệu **D** của bài toán hồi quy gồm các véc-tơ đầu vào $x_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, với $x_{ij} \in U_j \subset \mathbb{R}$ và các giá trị đầu ra $y_i \in U_{n+1} \subset \mathbb{R}$. Bước đầu tiên chúng ta cần chuẩn hóa

Thuật toán 2: GenFRBS(**T**)

Dữ liệu vào: Cây quyết định **T**.
Dữ liệu ra: Hệ luật mờ **S**.
S = ∅;
Leafs = Tập các nút lá của cây **T**;
foreach $lf \in Leafs$ **do**
  Với mỗi lá lf xây dựng danh sách $lsNode$ các Node từ lá lf đến gốc cây;
  Tạo luật r có n điều kiện tiền đề, tất cả các tiền đề đều có giá trị *Don't care*;
  for $j = lsNode.Count - 1$ ***down to*** 1 **do**
    Thay thế giá trị *Don't care* của luật r ứng với thuộc tính của nút $lsNode[j]$ bằng giá trị phân chia của nút $lsNode[j-1]$ (nút cha của nút $lsNode[j]$);
  end
  Gán kết luận của luật r là giá trị của nút $lsNode[0]$;
  $\boldsymbol{S} = \boldsymbol{S} \cup \{r\}$;
end
return S;

tập dữ liệu **D** về đoạn $[0, 1]$ bằng chuyển đổi tuyến tính ta được tập D_1.

Bước thứ hai, với mỗi thuộc tính đầu vào, ra ta xác định một bộ tham số của ĐSGT tương ứng, giả sử là π_i $(i = 1, 2, \ldots, n+1)$. Với bộ tham số π_i ta xây dựng ĐSGT $\mathcal{AX}_i$ sinh ra tập các từ có độ dài không quá k_i ký hiệu $X_{(k_i)}$, tính giá trị định lượng ngữ nghĩa của các từ trong $X_{(k_i)}$ và xây dựng hệ khoảng tương tự $S_{(k_i)}$ tương ứng theo thuật toán trong [7].

Bước thứ ba chuyển đổi cơ sở dữ liệu D_1 thành cơ sở dữ liệu từ ngôn ngữ D_2 theo nguyên tắc sau: với mỗi véc-tơ $x_i = (x_{i1}, x_{i2}, \ldots, x_{in})$ chuyển đổi thành véc-tơ từ ngôn ngữ $x'_i = (A_{i1}, A_{i2}, \ldots, A_{in})$, trong đó $x_{ij} \in T(A_{ij})$ với $j = 1, 2, \ldots, n$; giá trị đầu ra Y cũng chuyển đổi tương tự. Từ cơ sở dữ liệu ngôn ngữ D_2 chúng ta áp dụng thuật toán C4.5 [16] xây dựng cây quyết định có chiều cao tối đa $lmax$, việc thiết lập chiều cao tối đa của cây nhằm hạn chế chiều dài của luật được sinh ra. Mỗi nút của cây quyết định chứa hai giá trị là nhãn thuộc tính và giá trị phân chia của nút cha.

3. Thuật toán sinh luật từ cây quyết định

Thuật toán sinh luật từ cây quyết định được trình bày trong thuật toán 2. Đầu vào của thuật toán là cây quyết định **T** được xây dựng bằng thuật toán 1. Gọi **S** là tập luật được sinh ra, khởi đầu **S** = ∅, *Leafs* là tập các nút lá của cây **T**. Với mỗi nút lá sẽ sinh ra một luật có phần kết luận là nhãn của nút đó và phần tiền đề của luật là các nhãn của các nút nằm trên đường đi từ gốc đến lá đó. Để xác định đường đi, chúng ta xuất phát từ nút lá đi ngược về nút gốc. Giả sử để sinh luật r từ nút lá lf, từ nút này ta dễ dàng xác định được danh sách ($lsNode$)

là các nút mô tả đường đi từ nút lá *lf* đến nút gốc. Tiếp theo ta tạo ra luật r có độ dài bằng số chiều của bài toán và có tất cả tiền điều kiện đều bằng *Don't care*. Với mỗi nút $lsNode[j](j = lsNode.Count - 1, \ldots, 1)$ thay thế giá trị *Don't care* của tiền điều kiện tương ứng với thuộc tính là nhãn của nút $lsNode[j]$ bằng giá trị phân chia của nút $lsNode[j-1]$ (nút cha của nút $lsNode[j]$), gán kết luận của r bằng giá trị của nút $lsNode[0]$. Thêm luật r vào tập luật **S**.

IV. PHƯƠNG PHÁP XÂY DỰNG FRBS GIẢI BÀI TOÁN HỒI QUY

Trong phần này chúng tôi áp dụng phương pháp sinh luật đề xuất trong phần III để xây dựng hệ mờ giải bài toán hồi quy. Phương pháp xây dựng FRBS được thực hiện với hai pha. Pha thứ nhất chúng tôi phát triển thuật toán *OptHAParams* sử dụng thuật giải di truyền để tìm bộ tham số mờ của các ĐSGT của các thuộc tính của bài toán. Pha thứ hai sử dụng bộ tham số mờ của ĐSGT đã tìm ở pha thứ nhất xây dựng cây quyết định của bài toán, từ cây quyết định sinh ra tập các luật ứng cử, sau đó áp dụng thuật toán HA-De-PAES để tìm kiếm hệ luật tối ưu. Ở đây HA-De-PAES được phát triển dựa trên thuật toán (2+2)M-PAES [17]. Thuật toán (2+2)M-PAES tối ưu đồng thời hệ luật và tham số mờ trong khi thuật toán HA-De-PAES chỉ tối ưu hệ luật và các luật được chọn từ tập luật ứng cử được sinh ra từ cây quyết định, với hai mục tiêu là MSE và Comp (tổng chiều dài của luật).

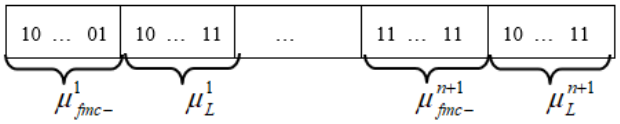
1. Thuật toán tìm tham số tối ưu của ĐSGT bằng thuật toán GA

Để tìm kiếm tham số mờ của ĐSGT, trong bài báo này chúng tôi thực hiện thiết kế thuật giải di truyền dựa trên sơ đồ mã hóa nhị phân với hàm thích nghi là giá trị sai số trung bình phương MSE. Bài toán có n thuộc tính, khi đó với mỗi thuộc tính ta cần xác định hai tham số là μ_{fmC^-} và μ_L (ở đây chúng tôi sử dụng ĐSGT có hai gia tử V và L). Như vậy số tham số cần xác định cho bài toán $\{\pi_i\}_{i=1}^{n+1}$ là $2(n+1)$. Chi tiết được trình bày trong thuật toán 3. Hình 2 mô tả sơ đồ mã hóa một cá thể, trong đó mỗi biến mục tiêu được mã hóa bằng một chuỗi bít có l bít.

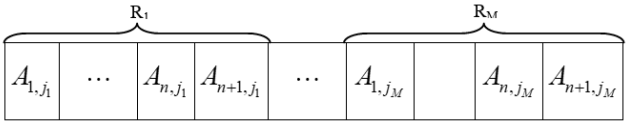
2. Thuật toán HA-De-PAES xây dựng hệ mờ tối ưu

Mỗi cá thể của quần thể được mã hóa gồm M luật (M có thể khác nhau giữa các cá thể), với mỗi luật R_i được lấy ra từ tập luật ứng cử **S**. Nhằm đạt được sự cân bằng giữa tính dễ hiểu và độ chính xác của hệ luật, chúng tôi giới hạn số luật trong mỗi FRBS nằm trong đoạn $[M_{\min}, M_{\max}]$.

Với mỗi cá thể cần tối thiểu hai mục tiêu là MSE và Comp, trong đó MSE được xác định theo (3) và Comp là tổng độ dài của các luật trong cơ sở luật, biểu thị độ phức tạp của FRBS.



Hình 2. Mã hóa cá thể tối ưu tham số của ĐSGT.



Hình 3. Cấu trúc mã hóa một cá thể biểu diễn một hệ luật.

a) *Các toán tử di truyền*

Toán tử lai ghép: Với hai cá thể bố mẹ p_1 và p_2 sử dụng phương pháp lai ghép một điểm (one-point crossover), điểm lai ghép được chọn ngẫu nhiên trong đoạn $[1, \rho_{\min} - 1]$, trong đó $\rho_{\min}$ là số luật nhỏ nhất trong p_1 và p_2. Lưu ý rằng nếu toán tử lai ghép không được thực hiện thì đột biến luôn xảy (có nghĩa là nếu không xảy ra lai ghép thì xác suất đột biến sẽ bằng 1).

Toán tử đột biến: Nếu đột biến xảy ra thì chọn ngẫu nhiên thực hiện một trong hai toán tử đột biến sau. *Toán tử đột biến thêm luật:* Thêm γ luật vào với γ được chọn ngẫu nhiên trong đoạn $[1, \gamma_{\max}]$, nếu $\gamma + M > M_{\max}$ thì $\gamma = M_{\max} - M$, các luật được chọn từ tập **S**. *Toán tử đột biến thay đổi luật:* Thay đổi ngẫu nhiên δ giá trị ngôn ngữ của một số luật của hệ luật, với δ được chọn ngẫu nhiên trong đoạn $[1, \delta_{\max}]$. Thực hiện δ lần quá trình sau: chọn ngẫu nhiên một luật R, tiếp theo chọn ngẫu nhiên một gen $j \in [1, n+1]$ của R; nếu $j \le n$ thì chọn ngẫu nhiên một từ trong $\{Don'tcare\} \cup X_{(k_j)}$; nếu $j = n+1$ thì chọn ngẫu nhiên một từ trong $X_{(k_{n+1})}$; thay thế từ của gen thứ j của R bằng từ vừa chọn; nếu luật R sau khi đột biến có độ dài lớn hơn *lmax* thì đột biến sẽ bị bỏ qua.

Lưu ý rằng sau khi lai ghép, đột biến nếu có nhiều luật trùng nhau thì chỉ giữ lại một, loại bỏ các luật có độ dài bằng 0.

b) *Thuật toán tiến hóa HA-De-PAES*

Phần này trình bày những bước cơ bản của thuật toán HA-De-PAES được phát triển dựa trên lược đồ tiến hóa (2+2)M-PAES đề xuất trong [17]. Tại mỗi thời điểm thuật toán lưu trữ một quần thể mà một cá thể bất kỳ trong đó đều không bị trội bởi các cá thể còn lại theo các mục tiêu MSE và Comp. Một quần thể như vậy sẽ xác định một mặt Pareto. Thuật toán thực hiện tìm kiếm một mặt Pareto xấp xỉ tối ưu theo hai mục tiêu MSE và Comp.

Xét bài toán tối ưu n mục tiêu, cực tiểu hàm $\mathbf{F}(x) = \big(f_1(x), f_2(x), \ldots, f_n(x)\big)$ với $x \in X \subseteq \mathbb{R}^m$, trong đó $\mathbf{F}(x)$

Thuật toán 3: OptHAParams($\mathbf{D}, k, lmax, GApars$)

Dữ liệu vào: Cơ sở dữ liệu của bài toán $\mathbf{D}$; Chiều dài tối đa của các hạng từ sinh ra từ ĐSGT k; Chiều cao tối đa của cây $lmax$; Bộ tham số của thuật giải di truyền $GApars$ gồm có chiều dài nhiễm sắc thể $lchrom$, kích thước quần thể $Popsize$, số thế hệ G, xác suất lai ghép $Pcross$, và xác suất đột biến Pmu.
Dữ liệu ra: Bộ tham số tối ưu π_{opt}.
Bước 1: Khởi tạo.
Gán $j \leftarrow 0$;
Khởi tạo quần thể ban đầu: Initial(Pop_j);
foreach $p \in Pop_j$ **do**
  Giải mã p để được bộ tham số $\{\pi_i\}_{i=1}^{n+1}$ của các ĐSGT;
  $\mathbf{T} \leftarrow$ BuildDecisionTree($\mathbf{D}, \{\pi_i\}_{i=1}^{n+1}, k, lmax$);
  $\mathbf{S} \leftarrow$ GenFRBS($\mathbf{T}$);
  Suy diễn trên tập dữ liệu $\mathbf{D}$ bằng hệ luật $\mathbf{S}$ và tính giá trị hàm mục tiêu MSE của p;
end
$X_{best} \leftarrow$ GetIndividualHasBestObj(Pop_j); //Chọn cá thể có giá trị mục tiêu tốt nhất.
Bước 2: Tiến hóa.
ComputeFitnessMeasure(Pop_j); //Tính độ đo thích nghi của các cá thể.
$Pop_{parent} \leftarrow$ Select(Pop_j); //Chọn cá thể cha mẹ.
$Pop_{child} \leftarrow$ Mute(Crossover(Pop_{parent})); //Lai ghép và đột biến tạo quần thể con.
$j \leftarrow j + 1$;
$Pop_j \leftarrow Pop_{child}$; //Thay thế quần thể hiện tại bằng quần thể con vừa tạo sinh.
foreach $p \in Pop_j$ **do**
  Giải mã p để được bộ tham số $\{\pi_i\}_{i=1}^{n+1}$ của các ĐSGT;
  $\boldsymbol{T} \leftarrow$ BuildDecisionTree($\boldsymbol{D}, \{\pi_i\}_{i=1}^{n+1}, k, l$);
  $\boldsymbol{S} \leftarrow$ GenFRBS($\boldsymbol{T}$);
  Suy diễn trên tập dữ liệu $\mathbf{D}$ bằng hệ luật $\mathbf{S}$ và tính giá trị hàm mục tiêu MSE của p;
end
$X \leftarrow$ GetIndividualHasBestObj(Pop_j);
if $obj(X) > obj(X_{best})$ **then**
  $X_{best} \leftarrow X$; //$obj(X)$ là giá trị mục tiêu của cá thể X.
end
Bước 3: Lặp lại Bước 2 cho đến khi $j > G$.
Bước 4: Trả lại bộ tham số tối ưu.
Giải mã X_{best} để được bộ tham số $\{\pi_i\}_{i=1}^{n+1}$ của các ĐSGT;
$\pi_{opt} \leftarrow \{\pi_i\}_{i=1}^{n+1}$;
return π_{opt};

Thuật toán 4: HA-De-PAES ($\mathbf{D}, k, lmax, \pi_{\text{opt}}, paespars$)

Dữ liệu vào: Cơ sở dữ liệu của bài toán $\mathbf{D}$, chiều dài tối đa của các hạng từ sinh ra từ ĐSGT k, chiều cao tối đa của cây $lmax$, bộ tham số mờ của ĐSGT đã được tối ưu, và bộ tham số của thuật toán PAES $paespars$.
Dữ liệu ra: P_A (mặt xấp xỉ tối ưu Pareto với hai mục tiêu MSE và Comp của các hệ luật).
Bước 1: Tạo tập luật ứng cử.
$\mathbf{T}$ = BuildDecisionTree($\mathbf{D}, \pi_{opt}, k, lmax$);
$\mathbf{S}$ = GenFRBS($\mathbf{T}$);
Bước 2: Sinh ngẫu nhiên 2 cá thể c_1, c_2. Mỗi cá thể gồm M luật được chọn từ tập luật ứng cử $\mathbf{S}$. M được chọn ngẫu nhiên trong đoạn $[M_{\min}, M_{\max}]$.
Bước 3: Bổ sung c_1, c_2 vào P_A.
Bước 4: Lặp $i = 1, \ldots, MaxGen$ (số thế hệ tối đa).
Chọn ngẫu nhiên hai cá thể bố mẹ p_1, p_2 trong P_A (p_1, p_2 có thể trùng nhau);
Thực hiện lai ghép hai cá thể p_1, p_2 để sinh ra hai cá thể con o_1, o_2;
Thực hiện đột biến trên o_1, o_2;
Tính giá trị mục tiêu (MSE, Comp) của o_1, o_2;
Lần lượt bổ sung o_1, o_2 vào P_A nếu có thể;
Lặp lại bước 4 với thế hệ kế tiếp;
return P_A;

là véc-tơ mục tiêu, $f_i(x)$ là mục tiêu thứ i cần cực tiểu, x là véc-tơ lời giải trong không gian m chiều, và X là không gian lời giải của bài toán. Một mặt $P \subseteq X$ được gọi là mặt Pareto nếu mỗi điểm bất kỳ của nó đều không bị trội bởi các điểm còn lại trong P. Một lời giải $x \in P$ được gọi là trội hơn lời giải $y \in P$, ký hiệu $x \succ y$, nếu $f_i(x) \leq f_i(y)$ với mọi $i \in \{1, \ldots, m\}$ và tồn tại j thỏa mãn $f_j(x) < f_j(y)$.

Kí hiệu P_A là quần thể hiện tại, thuật toán gồm các bước được trình bày trong thuật toán 4. Một cá thể con o nếu không bị trội bởi bất kỳ cá thể nào trong P_A thì o được bổ sung vào P_A, đồng thời loại bỏ tất cả những cá thể trong P_A bị trội bởi o. Nếu số cá thể trong P_A lớn hơn số lượng tối đa ($MaxArchive$) được phép lưu trữ trong P_A thì loại bỏ ngẫu nhiên một cá thể trong vùng có mật độ cao nhất ra khỏi P_A. Xác định vùng có mật độ cao nhất theo thuật toán trong [17].

V. NGHIÊN CỨU THỬ NGHIỆM

Chúng tôi tiến hành thử nghiệm thuật toán xây dựng FRBS được đề xuất trong bài báo và đối sánh kết quả với các thuật toán PAES_{KB} trong [1] và HA-PAES-MG-Kmax trong [8]. PAES_{KB} tiếp cận dựa trên lý tuyết tập mờ, các tập mờ được biểu diễn bằng bộ 2 (two-tuples), các luật mờ được sinh ra bằng tổ hợp ngẫu nhiên các từ ngôn ngữ sử dụng trên mỗi biến, quá trình tối ưu hóa tham số tập mờ và hệ luật cũng bằng thuật toán (2+2)M-PAES trong [17]. HA-PAES-MG-Kmax tiếp cận dựa trên lý thuyết ĐSGT, các tham số tập mờ được xác định dựa trên tham số mờ của ĐSGT, luật mờ được sinh ra dựa trên mẫu dữ liệu, quá trình tối ưu hóa tham số tập mờ và hệ luật cũng bằng thuật toán (2+2)M-PAES trong [17]. Chúng tôi chọn hai thuật toán này để đối sánh do chúng cùng sử dụng thuật toán tiến hóa (2+2)M-PAES và nhằm chứng tỏ tính hiệu quả của phương pháp sinh luật dựa trên cây quyết định.

Bảng I
CÁC BÀI TOÁN SỬ DỤNG THỬ NGHIỆM [1, 8]

TT	Bài toán	$\#NoP$	$\#NoA$
1	Electrical Length 1 (ELE1)	495	2
2	Electrical Maintainance 2 (ELE2)	1056	4
3	Weather Ankara (WA)	1609	9
4	Weather Izmir (WI)	1461	9
5	Treasury (TR)	1049	15
6	Abalone (AB)	4177	8
7	Mortgage (MTG)	1049	15
8	Computer Activity (CA)	8192	21
9	Pole Telecommunication (PT)	15000	26

Bảng II
CÁC THAM SỐ THỬ NGHIỆM PHA THỨ NHẤT, TÌM THAM SỐ TỐI ƯU

$\mu_{\min} = 0{,}3$	$fmC_{\min} = 0{,}3$	$fmC_{\max} = 0{,}7$
$\mu_{\max} = 0{,}7$	$kmax = 3$	$lmax = 2$
$Lchrom = 8$	$Popsize = 100$	$G = 100$
$Pcross = 0{,}7$	Xác suất lai ghép	
$Pmu = 0{,}1$	Xác suất đột biến	

Bảng III
CÁC THAM SỐ THỬ NGHIỆM PHA THỨ HAI, TÌM KIẾM HỆ LUẬT TỐI ƯU

$MaxArchive = 64$	$MaxGen = 300.000$	
$\gamma_{\max} = 5$	$\delta_{\max} = 5$	$lmax = \min(\#NoA, 5)$
$M_{\min} = 5$	$M_{\max} = 30$	
$PcRB = 0{,}3$	Xác suất lai ghép trên C_{RB}	
$PmRB = 0{,}1$	Xác suất đột biến trên C_{RB}	
$P_{Add} = 0{,}75$	Xác suất đột biến thêm luật trên C_{RB}	

Để công bằng trong so sánh hiệu quả của các phương pháp, chúng tôi sử dụng dạng phân hoạch mờ và các tham số thử nghiệm tương tự như các phương pháp được so sánh. Phân hoạch mờ được sử dụng có dạng đa thể hạt, các tập mờ có dạng tam giác, độ dài tối đa của các hạng từ được sinh ra bằng ĐSGT là $k = 3$ cho tất cả các thuộc tính đầu vào và đầu ra.

Chúng tôi tiến hành thử nghiệm trên 09 bài toán, trong đó 08 bài toán thử nghiệm được lấy từ http://archive.ics.uci.edu/ml/datasets.php, riêng bài toán Abalone lấy từ https://sci2s.ugr.es/keel/dataset.php?Cod=96, với $\#NoP$ là số mẫu và $\#NoA$ là số thuộc tính. Các tham số thử nghiệm cho pha thứ nhất được trình bày trong bảng II và pha thứ hai trình bày trong bảng III. Phương pháp thử nghiệm kiểm tra chéo 5-fold, với 4 fold học và 1 fold kiểm tra. Mỗi fold được thử nghiệm 6 lần ($6 \times 5 = 30$ lần). Mỗi lần thử nghiệm pha thứ nhất sẽ gọi thuật toán OptHAParams để tìm bộ tham số mờ của ĐSGT cho tất cả các thuộc tính, bộ tham số tìm được là đầu vào của thuật toán HA-De-PAES. Để giảm thời gian xây dựng cây quyết định ở pha thứ nhất, chúng tôi giới hạn chiều cao tối đa của cây được sinh ra $lmax = 2$, và để giới hạn chiều dài của luật được sinh ra trong pha thứ hai chúng tôi thiết lập $lmax$ bằng giá trị nhỏ nhất của $\#NoA$ và 5.

Mỗi lần thử nghiệm, kết quả thu được là một mặt Pareto theo hai mục tiêu MSE và Comp. Chúng tôi tính toán mặt Pareto trung bình của 30 lần thử nghiệm, tương tự như trong [1, 2, 8]. Thực hiện đối sánh kết quả thu được của thuật toán đề xuất với các thuật toán HA-PAES-MG-Kmax và PAES_{KB} tại điểm FIRST của mặt Pareto. Điểm FIRST là điểm tương ứng với hệ luật có $\overline{\text{MSE}}_{\text{Tr}}$ nhỏ nhất. Ký hiệu $\overline{\text{MSE}}_{\text{Tr}}$ và $\overline{\text{MSE}}_{\text{Ts}}$ là các giá trị MSE trung bình lần lượt trên tập dữ liệu huấn luyện (Tr) và tập dữ liệu kiểm tra (Ts), $\overline{\sigma_{\text{Ts}}}$ là phương sai trung bình trên tập kiểm tra, còn $\overline{\text{Comp}}$ và $\overline{\#R}$ lần lượt là độ phức tạp trung bình và số luật trung bình của hệ luật.

Chúng tôi tiến hành phân tích sử dụng phương pháp thống kê phi tham số Wilcoxon theo hai mục tiêu là độ phức tạp Comp và độ chính xác (dựa trên MSE), với mức ý nghĩa $\alpha = 0{,}05$. Kết quả thống kê được trình bày trong các bảng V, VI, và VII. Từ bảng V ta thấy giá trị Exact P-value lớn hơn $\alpha = 0{,}05$, do đó giả thiết H_0 là "*độ phức tạp của các hệ luật được tạo ra bởi hai thuật toán là như nhau*" được chấp nhận. Như vậy độ phức tạp của hệ luật được xây dựng bởi thuật toán đề xuất trong bài báo này không có sự khác biệt với các thuật toán được so sánh.

Kết quả phân tích bảng VI cho thấy giá trị Exact P-value nhỏ hơn $\alpha = 0{,}05$, do đó giả thiết H_0 là "*độ chính xác của các hệ luật trên tập huấn luyện của thuật toán là như nhau*" bị loại bỏ. Như vậy có sự khác biệt giữa giá trị MSE của các hệ luật được sinh ra từ thuật toán được đề xuất trong bài báo với giá trị MSE của các hệ luật được sinh ra từ các thuật toán đối sánh. Từ bảng IV ta thấy giá trị MSE của các hệ luật được sinh ra từ thuật toán HA-De-PAES tốt hơn trên hầu hết các bài toán trừ bài toán AB. Kết phân tích trong bảng VII cho thấy giá trị Exact P-value lớn hơn $\alpha = 0{,}05$, do đó giả thiết H_0 là "*độ chính xác của các hệ luật trên tập kiểm tra của thuật toán là như nhau*" được chấp nhận. Mặc dù không có sự khác biệt giữa độ chính xác trên tập kiểm tra của các hệ luật được sinh ra bởi thuật toán đề xuất trong bài báo này nhưng từ bảng IV chúng ta thấy độ chính xác của thuật toán đề xuất chỉ kém các thuật toán được đối sánh trên một bài toán, tốt hơn trên 8 bài toán. Chúng ta có thể kết luận rằng thuật toán được đề xuất tốt hơn các thuật toán đối sánh trên mục tiêu độ chính xác.

Bảng IV
SO SÁNH KẾT QUẢ THỬ NGHIỆM THUẬT TOÁN HA-DE-PAES (HADE) VỚI CÁC THUẬT TOÁN HA-PAES-MG-KMAX (HATG), $PAES_{KB}$ TẠI ĐIỂM **FIRST**

	$\overline{\#R}$			$\overline{\text{Comp}}$			$\overline{\text{MSE}}_{\text{Tr}}$			$\overline{\text{MSE}}_{\text{Ts}}$			$\overline{\sigma_{\text{Ts}}}$		
Bài toán	$PAES_{KB}$	HATg	HADe	$PAES_{KB}$	HATg	HADe	$PAES_{KB}$	HATg	HADe	$PAES_{KB}$	HATg	HADe	$PAES_{KB}$	HATg	HADe
ELE1	27	27,3	27,4	46,0	46,1	52,7	145.995	141.666	**141.321**	**194.028**	202.591	201.836	**24.745**	35.321	30.0234
ELE2	30	29,9	30,0	65,0	67,0	65,1	11,043	8.813	**8.504**	12.606	10.686	**10.372**	3.105	3.114	**1.771**
WA	28	25,0	25,0	103	60,0	71,6	1,64	1,03	**1,01**	3,92	1,25	**1,22**	9,27	**0,17**	**0,17**
WI	25	24,9	25,0	91,0	61,3	64,2	1,30	0,79	**0,77**	1,49	0,96	**0,95**	0,26	**0,13**	0,14
TR	11	15,0	15,0	40,0	29,4	33,9	0,080	0,031	**0,026**	0,140	0,045	**0,039**	0,15	0,02	**0,01**
AB	29	19,8	22,6	107	59,6	49,1	2,32	**2,31**	2,43	2,48	**2,41**	2,68	0,18	**0,17**	0,20
MTG	12	15,0	13,0	49,0	28,1	28,3	0,050	0,016	**0,014**	0,090	0,022	**0,019**	0,10	**0,01**	**0,01**
CA	10	13,8	14,5	30,0	44,7	45,6	11,99	4,58	**4,09**	13,43	4,86	**4,81**	4,66	0,63	**0,55**
PT	14	13,3	14,4	53,0	38,3	36,3	87,00	71,89	**65,07**	89,00	73,47	**68,97**	25,00	17,02	**10,44**

Bảng V
SO SÁNH ĐỘ PHỨC TẠP CỦA HỆ LUẬT SỬ DỤNG WILCOXON TEST VỚI MỨC $\alpha = 0{,}05$

So sánh với	**R+**	**R-**	**Exact P-value**	**Confidence-interval**	**Giả thuyết**
PAES-KB	37	8	0,09766	[-28,9; -0,55]	Loại bỏ giả thuyết H_0
HA-Tg	15	30	$\geq$ 0,2	[-1,95; 4,85]	Chấp nhận giả thuyết H_0

Bảng VI
SO SÁNH SAI SỐ MSE TRÊN TẬP HUẤN LUYỆN SỬ DỤNG WILCOXON TEST VỚI MỨC $\alpha = 0{,}05$

So sánh với	**R+**	**R-**	**Exact P-value**	**Confidence-interval**	**Giả thuyết**
PAES-KB	42	3	0,019532	[-2.337,315; -0,054]	Loại bỏ giả thuyết H_0
HA-Tg	40	5	0,03906	[-172,51; -0,002]	Chấp nhận giả thuyết H_0

Bảng VII
SO SÁNH SAI SỐ MSE TRÊN TẬP KIỂM TRA SỬ DỤNG WILCOXON-TEST VỚI MỨC $\alpha = 0{,}05$

So sánh với	**R+**	**R-**	**Exact P-value**	**Confidence-interval**	**Giả thuyết**
PAES-KB	33	12	$\geq$ 0,2	[-1.117,0505; 3.902,65]	Loại bỏ giả thuyết H_0
HA-Tg	39	6	0,05468	[-377,505; 0,11]	Loại bỏ giả thuyết H_0

VI. KẾT LUẬN

Bài báo đề xuất một hướng tiếp cận sinh luật mới giải bài toán hồi quy bằng hệ luật mờ. Các phương pháp truyền thống dựa trên lý thuyết tập mờ thường sự dụng phương pháp sinh luật bằng cách tổ hợp các từ ngôn ngữ sử dụng cho mỗi thuộc tính. Với cách tiếp cận này số luật phải xem xét rất lớn. Tiếp cận dựa trên lý thuyết ĐSGT sử dụng phương pháp sinh luật dựa trên mẫu dữ liệu, phương pháp tiếp cận này làm giảm không gian luật cần phải xem xét, tuy nhiên nó lại không tận dụng những thông tin và quan hệ của dữ liệu. Bài báo này chúng tôi đã đề xuất một phương pháp sinh luật tiếp cận theo lý thuyết ĐSGT và cây quyết định. Chúng tôi đã phát triển một thuật toán xây dựng FRBS gồm hai pha, kết quả thử nghiệm thuật toán cho thấy các mục tiêu độ phức tạp và độ chính xác của hệ luật có thể so sánh được với các thuật toán đã đề xuất.

LỜI CẢM ƠN

Nghiên cứu này nằm trong khuôn khổ đề tài "Nghiên cứu và phát triển các phương pháp thao tác trực tiếp trên các từ ngôn ngữ dựa trên đại số gia tử để giải quyết một số vấn đề trong các lĩnh vực trích rút tri thức, tăng cường chất lượng ảnh và cơ sở dữ liệu mờ", mã số 102.01-2017.06, được tài trợ bởi Quỹ phát triển khoa học và công nghệ quốc gia (NAFOSTED).

TÀI LIỆU THAM KHẢO

[1] R. Alcalá, P. Ducange, F. Herrera, B. Lazzerini, and F. Marcelloni, "A multiobjective evolutionary approach to concurrently learn rule and data bases of linguistic fuzzy-rule-based systems," *IEEE Transactions on Fuzzy Systems*, vol. 17, no. 5, pp. 1106–1122, Oct. 2009.

[2] M. Antonelli, P. Ducange, B. Lazzerini, and F. Marcelloni, "Learning concurrently data and rule bases of Mamdani fuzzy rule-based systems by exploiting a novel interpretability index," *Soft Computing*, vol. 15, pp. 1981–1998, 2011.

[3] H. Ishibuchi and Y. Nojima, "Analysis of interpretability-accuracy tradeoff of fuzzy systems by multiobjective fuzzy genetics-based machine learning," *International Journal of Approximate Reasoning*, vol. 44, no. 1, pp. 4–31, 2007.

[4] ——, "Repeated double cross-validation for choosing a single solution in evolutionary multi-objective fuzzy classifier design," *Knowledge-Based Systems*, vol. 54, pp. 22–31, 2013.

[5] P. Pulkkinen and H. Koivisto, "Fuzzy classifier identification using decision tree and multiobjective evolutionary algorithms," *International Journal of Approximate Reasoning*, vol. 48, no. 2, pp. 526–543, 2008.

[6] O. Cordón, M. J. Del Jesus, and F. Herrera, "A proposal on reasoning methods in fuzzy rule-based classification systems," *International Journal of Approximate Reasoning*, vol. 20, no. 1, pp. 21–45, 1999.

[7] C. H. Nguyen, W. Pedrycz, T. L. Duong, and T. S. Tran, "A genetic design of linguistic terms for fuzzy rule based classifiers," *International Journal of Approximate Reasoning*, vol. 54, no. 1, pp. 1–21, 2013.

[8] C. H. Nguyen, V. T. Hoang, and V. L. Nguyen, "A discussion on interpretability of linguistic rule based systems and its application to solve regression problems," *Knowledge-Based Systems*, vol. 88, pp. 107–133, 2015.

[9] C. H. Nguyen, V. Hoang, T. Tran, and V. Nguyen, "LFoC-Interpretability of linguistic rule based systems and its applications to solve regression problems," *International Journal of Computer Technology & Applications*, vol. 8, no. 2, pp. 94–117, 2017.

[10] F. Aghaeipoor and M. M. Javidi, "On the influence of using fuzzy extensions in linguistic fuzzy rule-based regression systems," *Applied Soft Computing*, vol. 79, pp. 283–299, 2019.

[11] C. Mencar and A. M. Fanelli, "Interpretability constraints for fuzzy information granulation," *Information Sciences*, vol. 178, no. 24, pp. 4585–4618, 2008.

[12] L. A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, no. 3, pp. 338–353, 1965.

[13] N. M. Han, N. C. Hao *et al.*, "An algorithm to building a fuzzy decision tree for data classification problem based on the fuzziness intervals matching," *Journal of Computer Science and Cybernetics*, vol. 32, no. 4, pp. 367–380, 2016.

[14] X. Liu, X. Feng, and W. Pedrycz, "Extraction of fuzzy rules from fuzzy decision trees: An axiomatic fuzzy sets (AFS) approach," *Data & Knowledge Engineering*, vol. 84, pp. 1–25, 2013.

[15] N. C. Ho and W. Wechler, "Hedge algebras: An algebraic approach to structure of sets of linguistic truth values," *Fuzzy Sets and Systems*, vol. 35, no. 3, pp. 281–293, 1990.

[16] J. Han, J. Pei, and M. Kamber, *Data Mining: Concepts and Techniques*. Elsevier, 2011.

[17] J. D. Knowles and D. W. Corne, "Approximating the nondominated front using the Pareto archived evolution strategy," *Evolutionary Computation*, vol. 8, pp. 149–172, 2000.



Nguyễn Đức Dư nhận bằng Cử nhân Toán tin ứng dụng và Thạc sĩ Toán ứng dụng tại Trường Đại học Khoa học Tự nhiên, Đại học Quốc gia Hà Nội lần lượt các năm 2001 và 2005. Tác giả hiên đang là nghiên cứu sinh tại Viện Khoa học và Công nghệ Quân sự từ năm 2015, đồng thời là giảng viên Khoa Công nghệ Thông tin, Trường Đại học Giao thông Vận tải. Các lĩnh vực nghiên cứu bao gồm khai phá dữ liệu, lô-gic mờ, hệ mờ, tính toán mềm, tính toán với từ, và học máy.



Hoàng Văn Thông nhận bằng Cử nhân Toán tin ứng dụng và Thạc sĩ Công nghệ Thông tin lần lượt tại Trường Đại học Khoa học Tự nhiên và Trường Đại học Công nghệ, Đại học Quốc gia Hà Nội năm 2001 và 2005. Năm 2016, tác giả nhận bằng Tiến sĩ tại Học viện Khoa học và Công nghệ, Viện Hàn lâm Khoa học Việt Nam. Tác giả hiện đang là giảng viên Khoa Công nghệ Thông tin, Trường Đại học Giao thông Vận tải. Các lĩnh vực nghiên cứu bao gồm khai phá dữ liệu, lô-gic mờ, hệ mờ, tính toán mềm, tính toán với từ, học máy, và trí tuệ nhân tạo.

DANH SÁCH PHẢN BIỆN NĂM 2019

Trần Hải Anh	Trường Đại học Bách khoa Hà Nội
Trịnh Quốc Anh	Trường Đại học Khoa học Tự nhiên, Đại học Quốc gia Hà Nội
Võ Đình Bảy	Trường Đại học Công nghệ TP. Hồ Chí Minh
Huỳnh Thị Thanh Bình	Trường Đại học Bách khoa Hà Nội
Lâm Sinh Công	Trường Đại học Công nghệ, Đại học Quốc gia Hà Nội
Nguyễn Việt Dũng	Trường Công nghệ Tiên tiến Quốc gia Bretagne, Pháp
Nguyễn Long Giang	Viện Công nghệ Thông tin, Viện Hàn lâm Khoa học Công nghệ Việt Nam
Lê Vũ Quỳnh Giang	Học viện Quản lý Giáo dục
Nguyễn Trung Hiếu	Học viện Công nghệ Bưu chính Viễn thông
Ngô Khắc Hoàng	Trường CentraleSupelec, Pháp
Nguyễn Nam Hoàng	Trường Đại học Công nghệ, Đại học Quốc gia Hà Nội
Nguyễn Quang Hùng	Trường Đại học Bách khoa, Đại học Quốc gia TP. Hồ Chí Minh
Phan Quốc Huy	Trường Đại học Kent, Anh
Trần Thiện Khải	Trường Đại học Bách khoa, Đại học Quốc gia TP. Hồ Chí Minh
Đinh Quốc Khánh	Trường Đại học Sungkyunkwan, Hàn Quốc
Trương Trung Kiên	Học viện Công nghệ Bưu chính Viễn thông
Nguyễn Lê Duy Lai	Trường Đại học Bách khoa, Đại học Quốc gia TP. Hồ Chí Minh
Nguyễn Phi Lê	Trường Đại học Bách khoa Hà Nội
Nguyễn Thị Thúy Loan	Trường Đại học Quốc tế, Đại học Quốc gia TP. Hồ Chí Minh
Nguyễn Bình Minh	Trường Đại học Bách khoa Hà Nội
Trần Thị Ngân	Trường Đại học Thủy lợi
Nguyễn Hữu Phát	Trường Đại học Bách khoa Hà Nội
Nguyễn Khánh Phương	Trường Đại học Bách khoa Hà Nội
Trần Minh Quang	Trường Đại học Bách khoa, Đại học Quốc gia TP. Hồ Chí Minh
Lê Hoàng Sơn	Viện Công nghệ Thông tin, Đại học Quốc gia Hà Nội
Lê Lam Sơn	Trường Đại học Bách khoa, Đại học Quốc gia TP. Hồ Chí Minh
Đinh Duy Tài	Viện tiên tiến về Khoa học và Công nghệ, Nhật Bản
Trần Đức Tân	Trường Đại học Phenikaa
Trần Minh Thái	Trường Đại học Ngoại ngữ - Tin học TP. Hồ Chí Minh
Tạ Minh Thanh	Học viện Kỹ thuật Quân sự
Nguyễn Thành	Viện Khoa học và Công nghệ Quốc gia Ulsan, Hàn Quốc
Nguyễn Trí Thành	Trường Đại học Công nghệ, Đại học Quốc gia Hà Nội
Lê Trung Thành	Trường Đại học Công nghệ, Đại học Quốc gia Hà Nội
Lê Thị Ngọc Thơ	Trường Đại học Công nghệ TP. Hồ Chí Minh
Quản Thành Thơ	Trường Đại học Bách khoa, Đại học Quốc gia TP. Hồ Chí Minh
Nguyễn Đình Thuận	Trường Đại học Công nghệ Thông tin, Đại học Quốc gia TP. Hồ Chí Minh
Nguyễn Đức Thuận	Trường Đại học Nha Trang
Nguyễn Cảnh Thưởng	Trường Đại học Osaka, Nhật Bản
Lê Hồng Trang	Trường Đại học Bách khoa, Đại học Quốc gia TP. Hồ Chí Minh
Nguyễn Linh Trung	Trường Đại học Công nghệ, Đại học Quốc gia Hà Nội
Lê Cung Tưởng	Trường Đại học Tôn Đức Thắng

THÔNG TIN DÀNH CHO TÁC GIẢ

Chuyên san Các công trình nghiên cứu và phát triển Công nghệ Thông tin và Truyền thông – gọi tắt là *Chuyên san CNTT-TT*, hay tiếng Anh là *ICT Research* – là ấn phẩm khoa học của Tạp chí Thông tin và Truyền thông, do Bộ Thông tin và Truyền thông xuất bản từ năm 1999. ICT Research có phản biện, mở trực tuyến (open access) và được phát hành 4 số mỗi năm, trong đó *Ấn phẩm tiếng Việt* (ISSN: 1859-3526) có 2 số và *Ấn phẩm tiếng Anh* (ISSN: 1859-3534) – Research and Development on Information and Communication Technology – có 2 số.

ICT Research là một diễn đàn dành cho các nhà nghiên cứu và các chuyên gia phổ biến các ý tưởng mới và sáng tạo trong các lĩnh vực của Công nghệ Thông tin, Truyền thông và Điện tử ở Việt Nam và trên toàn thế giới. Ban biên tập của ICT Research bao gồm các nhà khoa học uy tín đến từ các đơn vị nghiên cứu ở Việt Nam và các quốc gia khác.

HƯỚNG DẪN DÀNH CHO CÁC TÁC GIẢ

Các tác giả nên tìm hiểu cách viết bài báo kỹ thuật, chẳng hạn như các hướng dẫn trong tài liệu "How to write for technical periodicals & conferences" của Viện Kỹ sư Điện, Điện tử Hoa Kỳ (IEEE).

ICT Research chỉ nhận các bản thảo được nộp trực tuyến trên trang web của ICT Research tại http://ictmag.vn. Tác giả cần chế bản bản thảo theo mẫu mà ICT Research hướng dẫn.

Các tác giả phải chịu trách nhiệm trong việc đảm bảo rằng bản thảo này chưa từng được công bố trước đó, cũng như hiện đang không trong quá trình phản biện hoặc chuẩn bị xuất bản ở nơi khác. Tác giả liên hệ cũng phải chịu trách nhiệm đảm bảo sự đồng ý của tất cả các đồng tác giả khi nộp bài cho ICT Research. Bản thảo cần được nộp dưới dạng PDF. Để hỗ trợ phản biện kín hai chiều, tác giả cần nộp hai phiên bản của bản thảo, một phiên bản có đầy đủ thông tin của tất cả các tác giả, một không có thông tin gì.

Các bản thảo nộp lại sau khi chỉnh sửa cũng cần tuân theo các chỉ dẫn như của bản thảo nộp lần đầu, nhưng cần kèm theo một văn bản giải trình các ý kiến của phản biện, trong đó giải thích các nội dung được chỉnh sửa và/hoặc trả lời các ý kiến của các phản biện. Sau khi một bản thảo được chính thức chấp nhận xuất bản, các tác giả cần phải chuẩn bị bản thảo cuối cùng bằng LATEX theo mẫu trình bày cho IEEE Transactions.

ĐẠO ĐỨC TRONG XUẤT BẢN

ICT Research am kết tuân thủ các quy định, tiêu chuẩn và hướng dẫn được thiết lập bởi COPE (Committee on Publication Ethics) trong việc đảm bảo tính liêm chính của tạp chí cũng như xử lý các trường hợp vi phạm. Tất cả các bên tham gia (Biên tập viên, Tác giả, Phản biện và Nhà xuất bản) cần phải hiểu và tuân thủ các quy định, tiêu chuẩn và hướng dẫn được công bố trên trang web của COPE.

Ban biên tập của ICT Research sẽ sàng lọc tất cả các bản thảo trước khi công bố. Tất cả các bản thảo sẽ được kiểm tra về mức độ tương tự với các công trình khác sử dụng các công cụ trực tuyến có được. Mọi hình thức đạo văn không được chấp nhận và bản thảo sẽ bị từ chối.

QUÁ TRÌNH PHẢN BIỆN VÀ XUẤT BẢN

Một Biên tập viên lĩnh vực sẽ được Trưởng Ban biên tập giao phụ trách điều phối phản biện của mỗi bản thảo. Biên tập viên này thực hiện đánh giá nhanh bản thảo và quyết định liệu bản thảo này có phù hợp đối với độc giả của ICT Research hay không. Nếu phù hợp, Biên tập viên sẽ khởi động quá trình phản biện.

Mỗi bản thảo được phản biện bởi ít nhất hai phản biện. Quá trình phản biện là kín hai chiều, theo đó danh tính của các tác giả và phản biện không được tiết lộ cho nhau. Các kết luận phản biện gồm có: *(A) Chấp nhận*, *(B1) Chấp nhận với yêu cầu chỉnh sửa nhỏ*, *(B2) Chỉnh sửa và nộp lại* (bản thảo cần phải viết lại với các thay đổi lớn và sau đó cần qua vòng phản biện lần thứ hai), và *(C) Từ chối*.

Mỗi vòng phản biện có thể cần khoảng hai tháng. Trong trường hợp các tác giả được Biên tập viên yêu cầu chỉnh sửa bản thảo, phiên bản được chỉnh sửa cần được nộp trong vòng hai tháng đối với yêu cầu chỉnh sửa lớn và một tháng với yêu cầu chỉnh sửa nhỏ. Tác giả nào cần nhiều thời gian hơn xin vui lòng liên hệ với Biên tập viên.

Mỗi bài báo được chấp nhận sẽ được xuất bản trực tuyến ngay sau đó, với số DOI, sau khi được biên tập và chế bản theo các quy tắc biên tập của ICT Research. Đồng thời, khi có đủ một số lượng bài phù hợp, ICT Research sẽ tập hợp lại và xuất bản một số in, kèm với mục lục các bài báo.

THÔNG TIN VỀ BẢN QUYỀN

Khi bài báo được chấp nhận, bản quyền của bài báo sẽ tự động được chuyển giao cho Tạp chí Công nghệ Thông tin và Truyền thông, là tổ chức chịu trách nhiệm xuất bản ICT Research, với đảm bảo rằng mọi nội dung của bài báo được chấp nhận xuất bản sẽ được công bố miễn phí trên trang web của ICT Research. Việc chuyển giao này cho phép bên xuất bản có toàn quyền giải quyết bất kỳ khiếu nại nào về việc sử dụng trái phép. Các tác giả có quyền đăng tải lại hay sử dụng lại bài báo được xuất bản dưới bất kỳ hình thức nào cho bất kỳ mục đích nào, miễn là có trích dẫn đầy đủ tới bài báo gốc xuất bản trên ICT Research. Việc nộp bài báo được xem như là các tác giả đã đọc và đồng ý với chính sách bản quyền này.