TẠP CHÍ KHOA HỌC VÀ CÔNG NGHỆ, Trường Đại học Khoa học, ĐH Huế
Số chuyên san Công nghệ thông tin Tập 21, Số 1B (2022)

MỘT SỐ THUẬT TOÁN TỔ HỢP VỀ TOÁN TỬ BAO ĐÓNG THEO TIẾP CẬN SIÊU ĐỒ THỊ

Nguyễn Hoàng Sơn[1*], Nguyễn Ngọc Thủy[2]

[1] Khoa Toán, Trường Đại học Khoa học, Đại học Huế

[2] Khoa CNTT, Trường Đại học Khoa học, Đại học Huế

*Email: nhson@hueuni.edu.vn

Ngày nhận bài: 30/11/2022; ngày hoàn thành phản biện: 19/12/2022; ngày duyệt đăng: 20/12/2022

TÓM TẮT

Trong bài báo này, chúng tôi đề xuất hai thuật toán tổ hợp hiệu quả tìm tất cả khóa tối thiểu và phản khóa của toán tử bao đóng bằng cách sử dụng mô hình siêu đồ thị. Độ phức tạp của các thuật toán tổ hợp này cũng được phân tích và chứng minh.

Từ khóa: Toán tử bao đóng, khóa tối thiểu, phản khóa, siêu đồ thị.

1. MỞ ĐẦU

Khóa tối thiểu và phản khóa là các đối tượng quan trọng có nhiều ứng dụng của toán tử bao đóng. Bài toán xác định tất cả khóa tối thiểu, phản khóa của toán tử bao đóng được biết có độ phức tạp thời gian hàm số mũ. Do đó, đây là các bài toán cực khó của toán tử bao đóng [13-15].

Trong [15] chúng tôi đã dùng mô hình siêu đồ thị để tiếp cận các vấn đề tổ hợp của toán tử bao đóng. Siêu đồ thị là một lý thuyết toán học chuyên giải quyết các vấn đề tổ hợp một cách hiệu quả [7, 8]. Đặc biệt trong khoảng mấy chục năm trở lại đây, công cụ này được các nhà nghiên cứu sử dụng nhiều trong các lĩnh vực có liên quan đến khoa học máy tính, chẳng hạn như cơ sở dữ liệu, các hệ suy diễn, khai phá dữ liệu, lý thuyết quyết định, … [2, 4, 5, 10-13].

Trong [15] chúng tôi sử dụng mô hình siêu đồ thị để đưa bài toán tính tất cả khóa tối thiểu của toán tử bao đóng về bài toán sinh ra siêu đồ thị transversal của một họ. Sau đó thiết lập mối tương quan giữa các khóa tối thiểu và phản khóa của toán tử bao đóng bằng siêu đồ thị. Trên cơ sở các kết quả trên, trong bài báo này chúng tôi tiếp tục tiến hành nghiên cứu các thuật toán tổ hợp hiệu quả tìm tất cả khóa tối thiểu và phản khóa của toán tử bao đóng bằng công cụ siêu đồ thị. Cụ thể chúng tôi đề xuất hai thuật toán tổ hợp tìm tất cả khóa tối thiểu và phản khóa của toán tử bao đóng. Các thuật toán này được chứng minh có độ phức tạp thời gian là hàm số mũ, nhưng cách tiếp cận và tính toán theo kiểu tổ hợp của lý thuyết siêu đồ thị rất dễ thực hiện.

Với mục đích như vậy, cấu trúc bài báo chia làm 4 phần. Sau phần mở đầu, phần thứ 2 trình bày toán tử bao đóng, siêu đồ thị và các kết quả cơ sở liên quan đến phần 3. Phần thứ 3 giới thiệu hai thuật toán tổ hợp tìm tất cả khóa tối thiểu và phản khóa của toán tử bao đóng theo tiếp cận siêu đồ thị. Phân tích độ phức tạp cũng như giới thiệu một số ví dụ minh họa cho các thuật toán này cũng được đề cập trong phần này. Phần cuối cùng của bài báo là kết luận.

2. CÁC KHÁI NIỆM CƠ SỞ

Xét U là một tập hữu hạn khác rỗng bất kỳ. Ký hiệu 2^U là tập lũy thừa của U. Ánh xạ $f: 2^U \to 2^U$ thỏa các điều kiện sau:

(i) $X \subseteq f(X)$

(ii) $X \subseteq Y \Rightarrow f(X) \subseteq f(Y)$

(iii) $f(f(X)) = f(X)$

với mọi $X, Y \in 2^U$, được gọi là một *toán tử bao đóng* (TTBĐ) trên U. Ký hiệu $Cl(U)$ là tập tất cả các TTBĐ trên U.

Tập $X \subseteq U$ được gọi là *đóng* của $f \in Cl(U)$ nếu $f(X) = X$. Tập tất cả các tập đóng của f ký hiệu là $Closed(f)$.

Họ $\mathcal{S} \subseteq 2^U$ được gọi là *nửa dàn giao* trên U nếu các điều kiện sau là đúng:

(i) $U \in \mathcal{S}$

(ii) $\forall \mathcal{T} \subseteq 2^U, \varnothing \neq \mathcal{T} \subseteq \mathcal{S} \Rightarrow \bigcap \mathcal{T} \in \mathcal{S}$.

Rõ ràng $Closed(f)$ là một nửa dàn giao. Có thể thấy [5] nếu $\mathcal{S}$ là một nửa dàn giao thì ánh xạ $f_{\mathcal{S}}$ xác định bởi

$$f_{\mathcal{S}}(X) = \bigcap \left\{ Y \in \mathcal{S} : X \subseteq Y \right\}$$

là một TTBĐ. Ngược lại, nếu $f \in Cl(U)$ thì tồn tại duy nhất một nửa dàn giao $\mathcal{S}$ trên U sao cho $f = f_{\mathcal{S}}$, với

$$\mathcal{S} = \left\{ X \subseteq U : f(X) = X \right\}.$$

Xét TTBĐ $f \in Cl(U)$. Một tập con $K \subseteq U$ được gọi là *khóa* của f nếu $f(K) = U$. Trường hợp nếu với mọi $a \in K$ ta có $f(K-a) \neq U$ [1] thì K được gọi là *khóa tối thiểu* của f. Ký hiệu $Key(f)$ là tập tất cả khóa tối thiểu của f.

Tập con $K^{-1} \subseteq U$ được gọi là *phản khóa* của f nếu K^{-1} thỏa các điều kiện sau:

(i) $f(K^{-1}) \neq U$

(ii) $\forall a \in U - K^{-1}, f(K^{-1} \cup a) = U$.

Ký hiệu $Antikey(f)$ là tập tất cả phản khóa của f. Rõ ràng các phản khóa chính là các tập lớn nhất không phải khóa. Mối quan hệ giữa các khóa tối thiểu và phản khóa trên TTBĐ được xác định là [13]:

$$\bigcap Key(f) = U - \bigcap Antikey(f).$$

Biểu diễn này cho thấy sự quan trọng của các phản khóa. Chúng ta có thể xác định các khóa tối thiểu qua các phản khóa. Sau đây là hai bài toán quan trọng của TTBĐ. Các bài toán này đều cực khó, có độ phức tạp hàm mũ.

Bài toán 2.1. *Cho trước một TTBĐ $f \in Cl(U)$. Tìm tập $Key(f)$.*

Bài toán 2.2. *Cho trước một TTBĐ $f \in Cl(U)$. Tìm tập $Antikey(f)$.*

Một *siêu đồ thị* $\mathcal{H}$ là một cặp $(V, \mathcal{E})$ trong đó V là một tập hữu hạn các phần tử và $\mathcal{E} = \{E_1, \ldots, E_m\}$ là một họ các tập con của V. Người ta gọi các phần tử của V là các *đỉnh* và các phần tử của $\mathcal{E}$ là các *siêu cạnh* hay *cạnh*. Lưu ý rằng một số tác giả, chẳng hạn [1], yêu cầu tập các cạnh cũng như mỗi cạnh phải khác rỗng, và hợp tất cả các cạnh phải bằng tập đỉnh. Trong bài báo này chúng ta không yêu cầu như vậy. Như vậy, một đồ thị vô hướng chính là một siêu đồ thị với các cạnh có đúng hai phần tử.

Nếu không sợ nhầm lẫn, để thuận lợi cho việc ghi chúng ta sẽ mô tả một siêu đồ thị bằng tập cạnh của nó và ngược lại. Do đó, chúng ta sẽ viết $E \in \mathcal{H}$ thay cho $E \in \mathcal{E}$, $\min(\mathcal{H})$ thay cho $(V, \min(\mathcal{E}))$, vân vân.

Ký hiệu $HG(V)$ là tập tất cả siêu đồ thị trên V.

Một siêu đồ thị $\mathcal{H}$ được gọi là *đơn* nếu

$$\forall E_i, E_j \in \mathcal{H}, E_i \subseteq E_j \Rightarrow E_i = E_j.$$

Tập tất cả siêu đồ thị đơn trên V ký hiệu $SH(V)$.

---

[1] Ký hiệu $X-a, X-Y, X \cup a$ tương ứng lần lượt thay cho $X \setminus \{a\}, X \setminus Y, X \cup \{a\}$ với mọi $X, Y \subseteq U; a \in U$.

Một tập $T \subseteq V$ được gọi là một *transversal* của $\mathcal{H} \in HG(V)$ nếu với mọi $E \in \mathcal{H}$ ta có $T \cap E \neq \varnothing$. Ký hiệu $Trs(\mathcal{H})$ là tập tất cả các transversal của $\mathcal{H}$. Một transversal $T \in Trs(\mathcal{H})$ được gọi là *tối thiểu* nếu không tồn tại một tập con thật sự T' của T sao cho $T' \in Trs(\mathcal{H})$. Tập tất cả các transversal tối thiểu của $\mathcal{H}$ còn được gọi *siêu đồ thị transversal* của $\mathcal{H}$, và ký hiệu là $Tr(\mathcal{H})$. Như vậy, có thể thấy $Tr(\mathcal{H}) \in SH(V)$.

Siêu đồ thị transversal có các tính chất quan trọng sau đây.

Mệnh đề 2.1. [1] *Với mọi* $\mathcal{H}_1, \mathcal{H}_2 \in SH(V)$, *ta có*

(1) $\mathcal{H}_1 = Tr(\mathcal{H}_2) \Leftrightarrow \mathcal{H}_2 = Tr(\mathcal{H}_1)$.

(2) $Tr(\mathcal{H}_1) = Tr(\mathcal{H}_2) \Leftrightarrow \mathcal{H}_1 = \mathcal{H}_2$.

(3) $Tr(Tr(\mathcal{H}_1)) = \mathcal{H}_1$.

Chú ý rằng với mọi $\mathcal{H}_1, \mathcal{H}_2 \in SH(V)$, nếu $\mathcal{H}_1 \subseteq Tr(\mathcal{H}_2)$ và $\mathcal{H}_2 \subseteq Tr(\mathcal{H}_1)$ không kéo theo được $\mathcal{H}_1 = Tr(\mathcal{H}_2)$. Siêu đồ thị transversal có rất nhiều ứng dụng trong nhiều lĩnh vực của khoa học máy tính. Tìm một transversal tối thiểu là rất dễ, chỉ thực hiện trong thời gian đa thức. Tìm tất cả các transversal tối thiểu thì có khó hơn, bài toán này được chứng minh có độ phức tạp là hàm mũ.

Tuy vậy, một thuật toán tổ hợp tìm siêu đồ thị transversal rất hiệu quả được đề xuất trong [7]. Độ phức tạp của thuật toán này là hàm mũ, nhưng trong nhiều trường hợp chỉ là đa thức.

Thuật toán 2.1. [7] (*Tìm siêu đồ thị transversal*)

Đầu vào: $\mathcal{H} = \{E_1, E_2, \ldots, E_m\} \in HG(V)$ với $V = \{a_1, \ldots, a_n\}$.

Đầu ra: $Tr(\mathcal{H})$.

Phương pháp:

Bước 1. Đặt $\mathcal{L}_1 = \{\{a\} : a \in E_1\}$.

Bước $k+1$ ($k < m$). Giả sử

$$\mathcal{L}_k = \mathcal{S}_k \cup \{B_1, \ldots, B_{t_k}\}$$

với $B_i \cap E_{k+1} = \varnothing$, $i = 1, \ldots, t_k$ và $\mathcal{S}_k = \{A \in \mathcal{L}_k : A \cap E_{k+1} \neq \varnothing\}$.

Với mỗi i $(i = 1, \ldots, t_k)$ sinh ra tập $\{B_i \cup a : a \in E_{k+1}\}$, và ký hiệu là $A_1^i, \ldots, A_{r_i}^i$. Đặt

$$\mathcal{L}_{k+1} = \mathcal{S}_k \cup \{A_l^i : A \in \mathcal{S}_k \Rightarrow A \not\subset A_l^i, 1 \le i \le t_k, 1 \le l \le r_i\}$$

Bước m+1. Đặt $Tr(\mathcal{H}) = \mathcal{L}_m$.

Thuật toán 2.1 có độ phức tạp thời gian là hàm mũ theo n. Tuy nhiên, trong nhiều trường hợp độ phức tạp thời gian chỉ là $\mathcal{O}(n^2 m |Tr(\mathcal{H})|^2)$. Do đó, khi m nhỏ thì thuật toán rất hiệu quả.

Trong các vấn đề tổ hợp của nhiều lĩnh vực trong khoa học máy tính có thể tiếp cận được bằng siêu đồ thị, thì thuật toán này thường được sử dụng với tính hiệu quả rất cao.

3. KẾT QUẢ

Khóa tối thiểu là khái niệm quan trọng và có nhiều ứng dụng của TTBĐ. Bài toán 2.1 xác định tập $Key(f)$ với TTBĐ $f \in Cl(U)$ cho trước là bài toán khó, có độ phức tạp thời gian hàm mũ theo số phần tử của U (chẳng hạn, xem [13-15]). Tìm tập $Key(f)$ bằng cách dựa vào định nghĩa rõ ràng không hiệu quả. Sinh ra tập $Key(f)$ từ các đối tượng cho trước khác, chẳng hạn như $Antikey(f)$ cũng rất phức tạp và không thật sự hiệu quả (chẳng hạn, xem [13]). Bắt đầu từ TTBĐ cho trước $f \in Cl(U)$, xác định ngay $Key(f)$ cho đến nay chưa có thuật toán hiệu quả nào, ngoại trừ trong các thể hiện của TTBĐ trên các đối tượng cụ thể chẳng hạn như quan hệ, lược đồ quan hệ trong lý thuyết cơ sở dữ liệu quan hệ.

Bằng nhận xét $Key(f) \in SH(U)$, tức là $Key(f)$ là một siêu đồ thị đơn. Chúng tôi đã tiếp đã cận vấn đề này bằng mô hình siêu đồ thị. Chúng tôi dùng siêu đồ thị biểu diễn được tập $Key(f)$ [15]. Khi đó, có thể vận dụng Thuật toán 2.1 rất hiệu quả để sinh ra tất cả khóa tối thiểu của TTBĐ.

Cụ thể, chúng ta xét họ

$$C(f) = \{C(X) : X \subseteq U\},$$

với $C(X) = U - f(X)$.

Họ $C(f)$ này có tính chất sau.

Mệnh đề 3.1. [15]

(1) $\varnothing \in C(f)$.

(2) *X là một khóa của f nếu và chỉ nếu $C(X) = \varnothing$.*

Từ họ $C(f)$, chúng ta xây dựng:

$$Min(f) = \{Y \in C(f) : Y \neq \varnothing, (\forall Z \in C(f) \Rightarrow Y \not\subset Z)\}.$$

Có thể thấy ngay $Min(f) \in SH(U)$, và số phần tử của $Min(f)$ khá nhỏ. Trong [15] chúng tôi đã biểu diễn được tập $Key(f)$ qua siêu đồ thị transversal của họ $Min(f)$ như sau:

Định lý 3.2. *Với mọi* $f \in Cl(U)$, *ta có*

$$Key(f) = Tr(Min(f)) .$$

Điều này có nghĩa chúng ta có thể dùng mô hình siêu đồ thị để tiếp cận các khóa tối thiểu của TTBĐ. Cụ thể, dựa vào Định lý 3.2, chúng tôi đề xuất một thuật toán tổ hợp hiệu quả tìm tất cả các khóa tối thiểu của một TTBĐ. Sau đó, chúng tôi phân tích độ phức tạp thời gian của thuật toán cũng như cho ví dụ minh họa.

Thuật toán 3.1. TRKEY (*Tìm các khóa tối thiểu*)

Đầu vào: $f \in Cl(U)$ với $U = \{a_1, a_2, \ldots, a_n\}$.

Đầu ra: $Key(f)$.

Phương pháp:

Bước 1. Xây dựng họ:

$$C(f) = \{C(X) : X \subseteq U\}$$

với $C(X) = U - f(X)$.

Bước 2. Từ họ $C(f)$, tính họ:

$$Min(f) = \{Y \in C(f) : Y \neq \varnothing, (\forall Z \in C(f) \Rightarrow Y \not\subset Z)\}.$$

Bước 3. Sử dụng Thuật toán 2.1, tính siêu đồ thị transversal $Tr(Max(f))$.

Bước 4. Đặt $Key(f) = Tr(Min(f))$.

Theo Định lý 3.2, rõ ràng Thuật toán **TRKEY** tính đúng tập $Key(f)$. Biết rằng $Min(f)$ là rất nhỏ, nên Bước 3 tính siêu đồ thị transversal của $Min(f)$ theo Thuật toán 2.1 chỉ yêu cầu thời gian đa thức theo *n, m* và $|Min(f)|$. Điều này suy ra độ phức tạp thời gian của Thuật toán **TRKEY** chính là Bước 1 và Bước 2, tức là hàm mũ theo *n*.

Chúng ta biết các phản khóa có vai trò quan trọng trong các bài toán tổ hợp extremal của TTBĐ [3, 6]. Bài toán 2.2 xác định tập $Antikey(f)$ cũng là bài toán cực khó trong TTBĐ, có độ phức tạp thời gian là hàm mũ theo số phần tử của U , và cách tính cũng rất phức tạp (chẳng hạn, xem [13]). Tương tự như khóa tối thiểu, chúng ta thấy $Antikey(f) \in SH(U)$, tức là $Antikey(f)$ là một siêu đồ thị đơn trên U . Do đó, dẫn đến trong [15] chúng tôi thiết lập được mối quan hệ giữa các phản khóa và khóa tối thiểu qua siêu đồ thị như sau.

Mệnh đề 3.3. *Với mọi* $f \in Cl(U)$, *ta có*

(1) $Antikey(f) = \overline{Min(f)}$.

(2) $Antikey(f) = \overline{Tr(Key(f))}$.

Dựa vào kết quả này, chúng tôi đề xuất thuật toán tổ hợp hiệu quả tìm các phản khóa của TTBĐ như sau.

Thuật toán 3.2. TRANTIKEY (*Tìm các phản khóa*)

Đầu vào: $f \in Cl(U)$ với $U = \{a_1, a_2, \dots, a_n\}$.

Đầu ra: $Antikey(f)$.

Phương pháp:

Bước 1. Xây dựng họ:

$$C(f) = \{C(X) : X \subseteq U\}$$

với $C(X) = U - f(X)$.

Bước 2. Từ họ $C(f)$, tính họ:

$$Min(f) = \{Y \in C(f) : Y \neq \varnothing, (\forall Z \in C(f) \Rightarrow Z \not\subset Y)\}.$$

Bước 3. Tính họ bù $\overline{Min(f)}$.

Bước 4. Đặt $Antikey(f) = \overline{Min(f)}$.

Theo Mệnh đề 3.3 (1), rõ ràng Thuật toán **TRANTIKEY** tính đúng tập $Antikey(f)$. Bởi vì họ $Min(f)$ rất nhỏ, nên Bước 3 chỉ yêu cầu thời gian đa thức theo n. Do đó, có thể thấy độ phức tạp thời gian của Thuật toán **TRANTIKEY** là độ phức tạp thời gian của Bước 1 và Bước 2, tức là hàm mũ theo n.

Trường hợp vận dụng Mệnh đề 3.3 (2) và Thuật toán **TRKEY**, chúng ta cũng có thể tính được tập $Antikey(f)$ một cách hiệu quả. Độ phức tạp của cách này cũng hàm mũ theo n.

Ví dụ 3.1. Xét ánh xạ $f_a : 2^U \to 2^U$ với $a \in U$ và

$$f_a(X) = \begin{cases} U, \ neu\ a \in X \\ X, \ nguoc\, lai. \end{cases}$$

Dễ kiểm chứng được $f_a \in Cl(U)$. Khi đó

$$C(f_a) = \{U - f_a(X) : X \subseteq U\}$$

và do đó

$$Min(f_a) = \{\{a\}\},$$

$$\overline{Min(f_a)} = \{U - a\},$$

$$Tr(Min(f_a)) = \{\{a\}\}.$$

Như vậy $Key(f_a) = \{\{a\}\}$ và $Antikey(f_a) = \{U - a\}$.

Ví dụ 3.2. Xét ánh xạ $f : 2^U \to 2^U$, with $U = \{a_1, a_2, a_3, a_4\}$, xác định bởi:

X	$f(X)$	X	$f(X)$	X	$f(X)$	X	$f(X)$
$\varnothing$	$\varnothing$	$\{a_4\}$	$\{a_1, a_2, a_4\}$	$\{a_2, a_3\}$	$\{a_2, a_3\}$	$\{a_1, a_2, a_4\}$	$\{a_1, a_2, a_4\}$
$\{a_1\}$	$\{a_1\}$	$\{a_1, a_2\}$	$\{a_1, a_2, a_4\}$	$\{a_2, a_4\}$	$\{a_1, a_2, a_4\}$	$\{a_1, a_3, a_4\}$	U
$\{a_2\}$	$\{a_2\}$	$\{a_1, a_3\}$	$\{a_1, a_3\}$	$\{a_3, a_4\}$	U	$\{a_2, a_3, a_4\}$	U
$\{a_3\}$	$\{a_3\}$	$\{a_1, a_4\}$	$\{a_1, a_2, a_4\}$	$\{a_1, a_2, a_3\}$	U	U	U

Ta có:

$$C(f) = \{\{a_2, a_3, a_4\}, \{a_1, a_3, a_4\}, \{a_1, a_2, a_4\}, \{a_3\}, \{a_2, a_4\}, \{a_1, a_4\}, \varnothing\}$$

$$Min(f) = \{\{a_3\}, \{a_1, a_4\}, \{a_2, a_4\}\}$$

$$\overline{Min(f)} = \{\{a_1, a_3\}, \{a_2, a_3\}, \{a_1, a_2, a_4\}\}.$$

Suy ra

$$Tr(Min(f)) = \{\{a_1, a_2, a_3\}, \{a_3, a_4\}\}.$$

Do đó

$$Key(f) = \{\{a_1, a_2, a_3\}, \{a_3, a_4\}\},$$

$$Antikey(f) = \{\{a_1, a_3\}, \{a_2, a_3\}, \{a_1, a_2, a_4\}\}.$$

4. KẾT LUẬN

Bằng cách sử dụng mô hình siêu đồ thị, bài báo đề xuất hai thuật toán tổ hợp tìm tất cả khóa tối thiểu và phản khóa của một TTBĐ cho trước. Các thuật toán đều có độ phức tạp hàm số mũ. Tuy nhiên, các thuật toán này thực hiện dựa vào tính toán các cấu trúc tổ hợp như họ $Min(f)$ và $\overline{Min(f)}$ cùng với công cụ chuyên giải quyết các vấn đề tổ hợp là lý thuyết siêu đồ thị. Do đó, phương pháp này rõ ràng thú vị và hiệu quả.

Cách tiếp cận này cũng được chúng tôi đang triển khai ứng dụng để giải quyết những vấn đề liên quan đến bài toán rút gọn thuộc tính trong các hệ thống thông tin [9].

TÀI LIỆU THAM KHẢO

[1]. C. Berge (1989). *Hypergraphs: combinatorics of finite sets*, North – Holland, Amsterdam.

[2]. K. Bertet, C. Demko, J. Viaud, C. Guéri (2018). Lattices, closures systems and implication bases: a survey of structural aspects and algorithms, *Theoretical computer science*, Vol. 743, pp. 93–109.

[3]. G. Burosch, J. Demetrovics, G.O.H. Katona, D. J. Kleitman, A.A. Sapozhenko (1993). *On the number of closure operations*, Combinatorics. Paul Erdos is Eighty, Vol. 1, pp. 91–105.

[4]. N. Caspard, B. Monjardet (2003). *The lattices of closure systems, closure operators, and implicational systems on a finite set: a survey*, Discrete Applied Mathematics, Vol. 127, No 2, pp. 241–269.

[5]. J. Demetrovics, G. Hencsey, L. Libkin, I. Muchnik (1992). *On the interaction between closure operations and choice functions with applications to relational databases*, Acta Cybernetica, Vol. 10, No 3, pp. 129–139.

[6]. J. Demetrovics, G.O.H. Katona (1988). *Extremal combinatorial problems of database models*, MFDBS 87 (Dresden, 1987), 99–127. Lecture Notes in Comput. Sci., 305, Springer, Berlin, 1988.

[7]. J. Demetrovics, V.D. Thi (1999). *Describing candidate keys by hypergraphs*, Computers and Artificial Intelligence, Vol. 18, No 2, pp. 191–207.

[8]. T. Eiter, G. Gottlob (1995). *Identifying the minimal transversals of a hypergraph and related problems*, SIAM Journal on Computing, Vol. 24, No 6, pp. 1278–1304.

[9]. N.L. Giang, J. Demetrovics, V.D. Thi, P.D. Khoa (2021). *Some properties related to reduct of consistent decision systems*. Cybernetics and Information Technologies, Vol. 21, No 2, pp. 3–9.

[10]. H. Mao, S. Liu (2012). *Posets and closure operators relative to matroids*, Matematika, Vol. 28, No 1, pp. 77-85.

[11]. V.D. Nghia (2004). *Relationships between closure operations and choice functions equivalent descriptions of a family of functional dependencies*, Acta Cybernetica, Vol. 16, No 3, pp. 485–506.

[12]. S. Rudolph (2017). *Succinctness and tractability of closure operator representations*, Theoretical Computer Science, Vol. 658, pp. 327–345.

[13]. N.H. Son, V.D. Thi (2019). *Some the combinatorial characteristics of closure operations*, Algebra and Discrete Mathematics, Vol. 28, No 1, pp. 144–156.

[14]. Nguyễn Hoàng Sơn, Trần Việt Khoa (2021). *Bài toán NP-đầy đủ của toán tử bao đóng*, Tạp chí Khoa học và Công nghệ: Khoa học tự nhiên, Tập 18, số 1, tr. 1–5.

[15]. Nguyễn Hoàng Sơn (2022). *Toán tử bao đóng theo tiếp cận siêu đồ thị*, Tạp chí Khoa học Đại học Huế: Kỹ thuật và Công nghệ, (đã chấp nhận đăng).

SOME COMBINATORIAL ALGORITHMS ON CLOSURE OPERATIONS BY HYPERGRAPH APPROACH

Nguyen Hoang Son[1*], Nguyen Ngoc Thuy[2]

[1]Faculty of Mathematisc, University of Sciences, Hue University

[2]Faculty of Information Technology, University of Sciences, Hue University

Email: nhson@hueuni.edu.vn

ABSTRACT

In this paper, we present two efficient combinatorial algorithms that find all minimal keys and antikeys of the closure operation using a hypergraph. The time complexity of these combinatorial algorithms is proven.

Keywords: closure operation, minimal key, antikey, hypergraph.

Nguyễn Hoàng Sơn sinh ngày 28/06/1973 tại Thừa Thiên Huế. Năm 1995, ông tốt nghiệp đại học ngành Toán - Tin tại Trường Đại học Tổng hợp Huế. Ông nhận bằng thạc sỹ Tin học tại Trường Đại học Bách khoa Hà Nội năm 1998, và nhận học vị Tiến sĩ chuyên ngành Đảm bảo toán học cho máy tính và hệ thống tính toán tại Viện Công nghệ Thông Tin, Viện Hàn lâm Khoa học và Công nghệ Việt Nam năm 2006. Hiện ông công tác tại Trường Đại học Khoa học, Đại học Huế.

Lĩnh vực nghiên cứu: Khai phá dữ liệu, toán rời rạc, lý thuyết độ phức tạp tính toán.

Nguyễn Ngọc Thuỷ tốt nghiệp đại học và cao học ngành Tin học vào năm 2012 và 2014 tại Trường Đại học Khoa học, Đại học Huế; ông nhận bằng Tiến sĩ Khoa học máy tính năm 2020 tại Trường Đại học Khon Kaen, Thái Lan. Ông hiện đang công tác tại Trường Đại học Khoa học, Đại học Huế.

Lĩnh vực nghiên cứu: Lý thuyết tập thô, Khai phá dữ liệu và học máy