**TẠP CHÍ KHOA HỌC, Đại học Huế, Số 65, 2011**

# MỘT THUẬT TOÁN KHAI PHÁ TẬP MỤC LỢI ÍCH CAO TRONG CƠ SỞ DỮ LIỆU

*Nguyễn Phúc Xuân Quỳnh*
*Trường Đại học Sư Phạm, Đại học Huế*

## TÓM TẮT

*Khai phá tập mục lợi ích cao (high-utility itemset) là một mở rộng của bài toán khai phá tập mục phổ biến, đã được nhiều tác giả quan tâm với mục đích đánh giá ý nghĩa của các tập mục trong khai phá luật kết hợp. Thuật toán hai pha (Two-Phase) là một trong các thuật toán khai phá tập mục lợi ích cao. Bài báo này đề xuất một cải tiến của thuật toán Two-Phase. Việc cải tiến được thực hiện thông qua chiến lược tỉa hiệu quả hơn các tập mục ứng cử, cải tiến bước sinh tập ứng viên, nhờ đó giảm bớt được thời gian thực hiện thuật toán khai phá.*

## 1. Đặt vấn đề

Khai phá tri thức từ dữ liệu là một trong những vấn đề nhận được nhiều sự quan tâm của các nhà nghiên cứu. Trong lĩnh vực này, bài toán khai phá luật kết hợp được nghiên cứu rộng rãi. Một hướng mở rộng bài toán là quan tâm đến các tập mục đem lại lợi ích cao, quan tâm đến mức độ quan trọng khác nhau của các mục dữ liệu.

Mô hình khai phá tập mục lợi ích cao đã được *Yao* và cộng sự đề xuất [7]), từ đó đã có một số thuật toán khai phá tập mục lợi ích cao được đưa ra trong [1, 2, 5, 6].

*Y.Liu, Liao, Choudhary*, 2005 [5] đã đưa ra khái niệm lợi ích của giao tác và lợi ích của tập mục tính theo lợi ích của giao tác chứa nó (lợi ích *twu*), từ đó đề xuất thuật toán *Two-Phase* [5] khai phá tất cả các tập mục lợi ích cao, tuy nhiên mất nhiều thời gian trong việc sinh ứng viên với cơ sở dữ liệu lớn.

Vấn đề của các thuật toán khai phá tập mục lợi ích cao là giảm thiểu kích thước của tập ứng viên và đơn giản hóa quá trình tính toán lợi ích các tập mục. Nhằm giảm số lượng ứng viên cho tập mục lợi ích cao, giảm thời gian khai phá, bài báo đề xuất thuật toán *Im-Two-Phase* trên cơ sở cải tiến bước sinh tập ứng viên và tính giá trị *twu*.

## 2. Các khái niệm và định nghĩa cơ bản

Phần này trình bày các định nghĩa, tính chất cơ bản về tập mục lợi ích cao từ [5, 6, 7].

**Định nghĩa 2.1: Giá trị khách quan của mục tại một giao tác**

Mỗi mục $i_p$ trong giao tác $T_q$, được đặt tương ứng với một giá trị được gọi là giá trị khách quan *(objective value)* của mục $i_p$ tại giao tác $T_q$, ký hiệu $o(i_p, T_q)$. Chẳng hạn,

giá trị khách quan của mục $i_p$ trong giao tác $T_q$ có thể lấy là số đơn vị mục $i_p$ bán được trong giao tác $T_q$ (Giá trị xác định bởi cột chứa mục $i_p$ và hàng $T_q$ trong CSDL giao tác).

**Bảng 1.** *CSDL giao tác*

| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| T1 | 0 | 0 | 0 | 4 | 1 | 0 | 0 |
| T2 | 0 | 5 | 0 | 5 | 1 | 0 | 0 |
| T3 | 1 | 0 | 0 | 6 | 0 | 8 | 0 |
| T4 | 10 | 0 | 5 | 0 | 1 | 0 | 0 |
| T5 | 0 | 4 | 17 | 5 | 1 | 1 | 0 |
| T6 | 0 | 0 | 0 | 0 | 0 | 0 | 72 |

**Định nghĩa 2.2: Giá trị chủ quan của một mục**

Mỗi mục $i_p$ trong CSDL được đặt tương ứng với một giá trị, được gọi là giá trị chủ quan *(subjective value)* của mục đó, ký hiệu $s(i_p)$. Giá trị này được cho trong một bảng kèm theo với CSDL giao tác gọi là bảng lợi ích. Chẳng hạn, giá trị chủ quan của mục $i_p$ dựa trên đánh giá lợi nhuận của mỗi đơn vị mục dữ liệu đem lại.

**Bảng 2.** *Bảng lợi ích*

| **Mục** | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| **Lợi nhuận ($/đơn vị)** | 1 | 3 | 1 | 4 | 7 | 2 | 1 |

Lợi ích của một tập mục được đánh giá qua hàm 2 biến như sau:

**Định nghĩa 2.3: Hàm lợi ích**

Gọi $x$ là giá trị khách quan của một mục trong một giao tác và $y$ là giá trị chủ quan của một mục. Một hàm 2 biến $f(x, y)$ = R x R → R đơn điệu tăng theo $x$ và $y$ gọi là hàm lợi ích, thông thường hàm lợi ích được xác định $f(x, y) = x \times y$.

**Định nghĩa 2.4: Lợi ích của một mục tại một giao tác**

Cho hàm lợi ích $f(x, y)$. Lợi ích của mục $i_p$ tại giao tác $T_q$, ký hiệu $u(i_p, T_q)$ là giá trị của hàm $f(x, y)$ tại $o(i_p, T_q)$ và $s(i_p)$, tức là: $u(i_p, T_q) = f(o(i_p, T_q), s(i_p))$.

**Định nghĩa 2.5: Lợi ích của một tập mục tại giao tác**

Cho tập mục X $\subseteq T_q$. Lợi ích của tập mục $X$ tại giao tác $T_q$, ký hiệu $u(X, T_q)$, là tổng lợi ích của tất cả các mục $i_p$ thuộc $X$ tại giao tác $T_q$: $u(X, T_q) = \sum_{i_p \in X} u(i_p, T_q)$ với X $\subseteq T_q$.

Ký hiệu $db_X = \{T_q \mid X \subseteq T_q, T_q \in DB\}$ là tập các giao tác chứa tập mục $X$ trong CSDL $DB$.

**Định nghĩa 2.6: Lợi ích của một tập mục trong CSDL**

Lợi ích (hay còn gọi là lợi ích thực sự) của tập mục $X$ trong CSDL $DB$, ký hiệu $u(X)$, là tổng lợi ích của tập mục $X$ tại các giao tác thuộc $db_x$:

$$u(X) = \sum_{T_q \in db_X} u(X, T_q) = \sum_{T_q \in db_X} \sum_{i_p \in X} u(i_p, T_q)$$

**Định nghĩa 2.7: Lợi ích của một giao tác**

Lợi ích của giao tác $T_q$, ký hiệu $tu(T_q)$, là tổng lợi ích của tất cả các mục dữ liệu trong giao tác: $tu(T_q) = \sum_{i_p \in T_q} u(i_p, T_q)$.

**Định nghĩa 2.8: Giá trị lợi ích tối thiểu**

Giá trị lợi ích tối thiểu *(minutil)* là tích của ngưỡng lợi ích tối thiểu $\delta$ với tổng lợi ích của toàn bộ CSDL.

**Định nghĩa 2.9: Tập mục lợi ích cao**

Tập mục $X$ là tập mục lợi ích cao nếu $u(X) \geq minutil$ *(minutil>0).*

**Định nghĩa 2.10: Bài toán khai phá tập mục lợi ích cao**

Bài toán khai phá tập mục lợi ích cao là bài toán tìm tập tất cả các tập mục lợi ích cao $HU = \{X \mid X \subseteq I, u(X) \geq \min util\}$ với CSDL giao tác $DB$ và ràng buộc *minutil* cho trước.

**Định nghĩa 2.11: Lợi ích kéo theo của tập mục**

*(Transaction Weighted Utility – TWU)*

Cho tập mục $X$ và $db_X$ là tập tất cả các giao tác chứa $X$. Ta gọi tổng lợi ích của tất cả các giao tác trong $db_X$ là lợi ích kéo theo *(lợi ích twu)* của $X$.

Ký hiệu lợi ích kéo theo của $X$ là $twu(X)$, ta có:

$$twu(X) = tu(db_X) = \sum_{T_q \in db_X} tu(T_q) = \sum_{T_q \in db_X} \sum_{i_p \in T_q} u(i_p, T_q)$$

*Ví dụ:* Trong ví dụ ở bảng 2.1 và bảng 2.2, $X=\{B, D, E\}$. Có 2 giao tác chứa $X$ là $T_2$ và $T_5$.

$twu(BDE) = tu(T_2)+tu(T_5)=$

$(o(B,T_2)*s(B,T_2)+o(D,T_2)*s(D,T_2)+o(E,T_2)*s(E,T_2))+$

$(o(B,T_5)*s(B,T_5)+o(C,T_5)*s(C,T_5)+o(D,T_5)*s(D,T_5)+o(E,T_5)*s(E,T_5))+o(F,T_5)*s$

*(F,$T_5$))* = (5.3+5.4+1.7)+(4.3+17.1+5.4+1.7+1.2)=42+58=100.

**Định nghĩa 2.12: Tập mục có lợi ích kéo theo cao**

Cho giá trị lợi ích tối thiểu *minutil*>0, tập mục *X* là tập mục có lợi ích kéo theo cao *(hay còn gọi là kéo theo cao)* nếu $twu(X) \geq minutil$.

**Định lý 2.1: Tính chất phản đơn điệu của lợi ích kéo theo**

Cho $X^k$ là một k-tập mục, $X^{k-1}$ là một (k-1)-tập mục con của $X^k$ ($X^{k-1} \subset X^k$). Nếu $X^k$ có lợi ích kéo theo cao thì $X^{k-1}$ cũng có lợi ích kéo theo cao.

*Chứng minh:*

Vì $X^{k-1} \subset X^k$, nên $db_X^k \subseteq db_X^{k-1}$. Theo công thức tính *twu* ở định nghĩa 2.11:

$$twu(X^{k-1}) = \sum_{T_q \in db_X^{k-1}} tu(T_q) \geq \sum_{T_q \in db_X^k} tu(T_q) = twu(X^k)$$

Do đó nếu $twu(X^k) \geq minutil$ thì $twu(X^{k-1}) \geq minutil$.

*Nhận xét:* Tính chất phản đơn điệu của lợi ích kéo theo có nghĩa là nếu một k-tập mục $X^k$ có chứa tập mục con $X^{k-1}$ mà $X^{k-1}$ là tập mục có lợi ích kéo theo thấp thì $X^k$ cũng là tập mục có lợi ích kéo theo thấp. Các ứng viên k-tập mục lợi ích kéo theo cao chỉ có thể có được từ các kết nối của các (k-1)-tập mục có lợi ích kéo theo cao. Dựa vào nhận xét này, có thể sử dụng các phương pháp khai phá tập mục phổ biến để tìm các tập mục lợi ích *twu* cao.

**Định lý 2.2:** Nếu *X* là tập mục lợi ích cao thì *X* cũng là tập mục có lợi ích kéo theo cao.

*Chứng minh:*

Vì $u(X, T_q) \leq tu(T_q)$ nên $u(X) = \sum_{T_q \in db_X} u(X, T_q) \leq \sum_{T_q \in db_X} tu(T_q) = twu(X)$

Vậy, nếu $u(X) \geq minutil$ thì $twu(X) \geq minutil$.

## 3. Thuật toán Im-Two-Phase

### *3.1. Cơ sở lý thuyết*

Trong thuật toán *Two-Phase* [5], giá trị *twu* được so với *minutil* để sinh tập ứng viên cho tập mục lợi ích cao. Tuy nhiên, trong bước tìm ra các 1-tập mục có lợi ích *twu* cao, nhận xét rằng các 1-tập mục có lợi ích *twu* thấp không tham gia vào quá trình sinh tập ứng viên cho tập mục lợi ích cao *(theo định lý 2.1 và 2.2)* nên có thể bỏ đi các 1-tập mục này trong từng giao tác. Từ đó, giá trị *tu* sẽ trừ đi các giá trị lợi ích của 1-tập mục lợi ích thấp, làm giá trị *twu* giảm đi so với giá trị *twu* ban đầu, thu gọn các ứng viên hơn khi so với *minutil*.

Cụ thể, sau khi đã có tập $WHU_1$ như trong thuật toán *Two-Phase*, sau khi đã có

tập $WHU_1$, duyệt CSDL lần nữa để bỏ đi các 1-tập mục lợi ích thấp trong từng giao tác và cập nhật lợi ích tu của từng giao tác:

**for** mỗi giao tác $T \in DB$

Bỏ đi các mục $X \in T \setminus WHU_1$;

Cập nhật lợi ích $tu(T):=tu(T) - \sum_{X \in T \setminus WHU_1} u(X,T)$;

Thuật toán giữ lại các câu lệnh còn lại như của thuật toán *Two-Phase*, tuy nhiên cải tiến bước nối trong quá trình sinh ứng viên cho tập $C_k$ từ tập $WHU_{k-1}$: Thay vì nối hai (k-1)-tập mục trong $WHU_{k-1}$ với nhau để tạo ứng viên cho tập $C_k$ như trong thuật toán *Two-Phase,* thì thuật toán *Im-Two-Phase* sẽ nối một (k-1)-tập mục trong $WHU_{k-1}$ với 1-tập mục trong $WHU_1$ giúp thời gian thực hiện của thuật bước nối được giảm xuống.

***Mệnh đề 2.1:*** Độ phức tạp của bước nối trong bước sinh ứng viên $C_k$ trong thuật toán *Two-Phase* là $O(k(C_m^{k-1})^2)$.

*Chứng minh:*

Trong thuật toán *Two-Phase*, nối hai *(k-1)*-tập mục trong $WHU_{k-1}$: số tập mục trong $WHU_{k-1}$ tối đa là $C_m^{k-1}$, với $m$ là số mục. Số khả năng chọn 2 tập mục ra từ $WHU_{k-1}$ là $C^2_{C_m^{k-1}}$. Khi xét hai *(k-1)*-tập mục này cần tối đa *(k-1)* phép so sánh, do đó tổng số phép tính là: (k-1) $C^2_{C_m^{k-1}}$. Ta có:

$$\text{(k-1)}.\ C^2_{C_m^{k-1}} = \text{(k-1)}.\frac{C_m^{k-1}!}{2!(C_m^{k-1}-2)!} = \text{(k-1)}.\frac{C_m^{k-1}(C_m^{k-1}-1)}{2} \approx k.(C_m^{k-1})^2$$

***Mệnh đề 2.2:*** Độ phức tạp của bước nối của hàm *Im_Gen_Ck* trong thuật toán *Im-Two-Phase* là $O(m.C_m^{k-1})$.

*Chứng minh*:

Trong thuật toán *Im-Two-Phase*, nối một *(k-1)*-tập mục trong $WHU_{k-1}$ với 1-tập mục trong $WHU_1$: $WHU_1$ có tối đa $m$ tập mục, $WHU_{k-1}$ có tối đa $C_m^{k-1}$ phần tử, thuật toán chọn một *(k-1)*-tập mục trong $WHU_{k-1}$ với 1-tập mục trong $WHU_1$, khi nối cần 1 phép so sánh, nên tổng số phép tính: 1.m. $C_m^{k-1}$, do đó độ phức tạp của thuật toán này là: $O(m.C_m^{k-1})$.

Như vậy, thuật toán *Im-Two-Phase* đã giảm thời gian bước nối sinh tập ứng viên

$C_k$ từ $O(k.(C_m^{k-1})^2)$ trong thuật toán *Two-Phase* xuống còn $O(m.C_m^{k-1})$.

### *3.2. Nội dung thuật toán*

**Input:** CSDL giao tác, giá trị lợi ích tối thiểu *minutil*.

**Output:** Tập *HU* gồm các tập mục lợi ích cao.

**Method:**

***Các ký hiệu:***

$C_k$: Tập các ứng viên k-tập mục có lợi ích *twu* cao.

$WHU_k$: Tập các k-tập mục có lợi ích *twu* cao.

*WHU:* Tập tất cả các tập mục có lợi ích *twu* cao.

$HU_k$: Tập các k-tập mục có lợi ích cao.

*HU:* Tập tất cả các tập mục có lợi ích cao.

***Nội dung thuật toán:***

// **Pha 1**: Phát hiện các tập mục có lợi ích *twu* cao

1. **for** mỗi giao tác $T \in DB$
2. Tính lợi ích $tu(T_q)$;
3. k=1;
4. $WHU=\phi$;
5. $WHU_1=\{i / i \in I, twu(i) \geq minutil\}$;
6. **for** mỗi giao tác $T \in DB$
7. Bỏ đi các mục $X \in T \setminus WHU_1$;
8. Cập nhật lợi ích $tu(T):=tu(T) - \sum_{X \in T \setminus WHU_1} u(X,T)$;
9. $WHU_1=\{i / i \in I, twu(i) \geq minutil\}$; //Cập nhật lại các phần tử cho $WHU_1$
10. $WHU=WHU_1$;
11. **for** (k=2; $C_{k-1} \neq \phi$; k++)
12. $C_k$ = Im_Gen_Ck($WHU_{k-1}$); // Tạo các tập mục ứng viên ở bước k
13. **for** mỗi giao tác $T \in DB$
14. **for** mỗi ứng viên $c \in C_k$
15. *If* $c \subseteq T$ *then*
16. $twu(c)=twu(c)+tu(T)$;
17. $WHU_k=\{c \in C_k \mid twu(c) \geq minutil\}$; //Lọc các k-tập mục có lợi ích *twu* cao
18. $WHU=WHU \cup WHU_k$;

// **Pha 2**: Phát hiện các tập mục có lợi ích cao

19. $HU=\varnothing$;

20. **for** mỗi giao tác $T \in DB$
21. **for** mỗi ứng viên $w \in WHU$
22. *If* $w \subseteq T$ *then*
23. $u(w)=u(w)+u(w,T)$;
24. $HU=\{w \in WHU \mid u(w) \geq minutil\}$; //Tuyển chọn các tập mục lợi ích cao

***Hàm Im_Gen_Ck:***

**Input:** Tập các (k-1)-tập mục có lợi ích kéo theo cao $WHU_{k-1}$ *(Các mục trong từng phần tử được sắp xếp theo thứ tự từ điển).*

**Output:** Tập các ứng viên k-tập mục có lợi ích kéo theo cao $C_k$.

**Method:**

*//Bước kết nối*

1. $C_k=\varnothing$;
2. **for** mỗi (k-1)-tập mục $X \in WHU_{k-1}$
3. **for** mỗi 1-tập mục $Y \in WHU_1$
4. **if** X[k-1]<Y **then**
5. $C_k=C_k \cup$ {X[1], X[2], …, X[k-2], X[k-1], Y};

*//Bước cắt tỉa*

6. **for** mỗi tập mục $c \in C_k$
7. **for** mỗi (k-1)-tập mục $s \in c$
8. **if** ($s \notin WHU_{k-1}$) **then**
9. $C_k=C_k-\{c\}$;
10. **return** $C_k$;

***3.3. Ví dụ minh họa***

Với CSDL ở bảng 2.1 và 2.2, *minutil*=27%*tổng lợi ích=45%*253=68,31.

**** Kết quả thực hiện thuật toán Im-Two-Phase***:

- Câu lệnh 1-2:

**Bảng 3.** *Kết quả thực hiện câu lệnh 1-2*

| | **A** | **B** | **C** | **D** | **E** | **F** | **G** | **tu** |
|---|---|---|---|---|---|---|---|---|
| T1 | 0 | 0 | 0 | 4 | 1 | 0 | 0 | 23 |
| T2 | 0 | 5 | 0 | 5 | 1 | 0 | 0 | 42 |
| T3 | 1 | 0 | 0 | 6 | 0 | 8 | 0 | 41 |
| T4 | 10 | 0 | 5 | 0 | 1 | 0 | 0 | 17 |
| T5 | 0 | 4 | 17 | 5 | 1 | 1 | 0 | 58 |
| T6 | 0 | 0 | 0 | 0 | 0 | 0 | 72 | 72 |

- Câu lệnh 3-5: $WHU=\varnothing$ . Nhận được tập $WHU_1$ với các giá trị *twu* tương ứng:

$WHU_1$={B:100, C:75, D:164, E:123, F:116, G:72}

- Câu lệnh 6-8:

**Bảng 4.** *Kết quả thực hiện câu lệnh 6 - 8*

| | B | C | D | E | F | G | tu |
|---|---|---|---|---|---|---|---|
| T1 | 0 | 0 | 4 | 1 | 0 | 0 | 23 |
| T2 | 5 | 0 | 5 | 1 | 0 | 0 | 42 |
| T3 | 0 | 0 | 6 | 0 | 8 | 0 | 40 |
| T4 | 0 | 5 | 0 | 1 | 0 | 0 | 7 |
| T5 | 4 | 17 | 5 | 1 | 1 | 0 | 58 |
| T6 | 0 | 0 | 0 | 0 | 0 | 72 | 72 |

- Câu lệnh 9-10: $WHU=WHU_1$={B:100, D:163, E:123, F:105, G:72}

- Câu lệnh 11-18: Nhận được các tập với các giá trị *twu* tương ứng

+ *Bước k=2*

$C_2$={BD:100, BE:100, BF: 58, BG:0, DE:123, DF:98, DG:0, EF: 58, EG:0, FG:0}

$WHU_2$={BD:100, BE:100, DE:123, DF:98}

$WHU$={B:100, D:163, E:123, F:105, G:72, BD:100, BE:100, DE:123, DF:98}

+ *Bước k=3*

$C_3$= {BDE:100}

$WHU_3$={BDE:100}

$WHU$={B:100, D:163, E:123, F:105, G:72, BD:100, BE:100, DE:123, DF:98, BDE:100}.

- Câu lệnh 19-24: Có được tập *HU* với giá trị lợi ích thực sự tương ứng:

$HU$={D:80, G:72, DE:77, BDE:81}

**** Kết quả thực hiện thuật toán Two-Phase***:

$WHU_1$={B:100, C:75, D:164, E:123, F:116, G:72}

$C_2$= {BC:58, BD:100, BE:100, BF: 58, BG:0, CD:58, CE:75, CF:58 , CG:0 , DE: 123, DF:99, DG:0 , EF:58, EG:0 , FG:0}

$WHU_2$= {BD:100, BE:100, CF:75, DE:123, DF:99}

$C_3$= {BDE:100}

$WHU_3$= {BDE:100}

*WHU*={B:100, C:75, D:164, E:123, F:116, G:72, BD:100, BE:100, CF:75, DF:99, DE:123, DF:100, BDE:100}

*HU*={D:80, G:72, DE:77, BDE:81}

**** So sánh số lượng phần tử của các tập ứng viên:***

**Bảng 5.** *So sánh số lượng phần tử của các tập của hai thuật toán*

| | $WHU_1$ | $C_2$ | $WHU_2$ | $C_3$ | $WHU_3$ | WHU |
|---|---|---|---|---|---|---|
| **Two-Phase** | 6 | 15 | 5 | 1 | 1 | 12 |
| **Im-Two-Phase** | 5 | 10 | 4 | 1 | 1 | 10 |

Như vậy, với thuật toán *Im-Two-Phase*, số lượng ứng viên được thu gọn ở bước sinh tập $C_k$ và tập $WHU_k$ của từng bước, nếu *bước k-1* sinh càng ít ứng viên thì thời gian thực hiện bước *k* sẽ nhanh hơn và ứng viên cho bước tiếp theo sẽ ít hơn, giúp thuật toán thực hiện nhanh hơn. Số phần tử trong tập *WHU* càng ít thì sẽ tiết kiệm thời gian tính toán lợi ích thực sự của các tập mục hơn do số lượng các tập mục cần tính lợi ích thực sự sẽ ít hơn.

### *3.4. Nhận xét thuật toán*

Thuật toán sẽ phải duyệt CSDL thêm một lần so với thuật toán *Two-Phase* để tính lại giá trị *tu* và bỏ đi các 1-tập mục có lợi ích *twu* thấp. Tuy nhiên thời gian duyệt CSDL thêm một lần không ảnh hưởng đến hiệu năng của thuật toán trong các CSDL lớn do vấn đề sinh tập mục ứng viên và tính toán lợi ích của các tập mục mới thực sự ảnh hưởng đến thời gian thực hiện của các thuật toán. Mặc dù thêm một lần duyệt trước khi sinh ứng viên, tuy nhiên điều này làm giảm số lượng ứng viên ở các bước sau, nên sẽ giảm số lần duyệt CSDL sau này để tính lợi ích của các tập mục. Nếu không thêm lần duyệt trước khi sinh ứng viên thì số lượng ứng viên các bước sau sẽ nhiều hơn và số lần duyệt CSDL ở các bước sau sẽ nhiều, ảnh hưởng đến thời gian thực hiện của thuật toán.

Nếu CSDL càng chứa nhiều 1-tập mục có lợi ích *twu* thấp thì khi cập nhật giá trị *tu* bằng cách trừ đi lợi ích của các 1-tập mục có lợi ích thấp sẽ thu gọn đáng kể tập ứng viên, giảm các bước sinh ứng viên hơn.

Thuật toán đã giảm thời gian sinh tập ứng viên $C_k$ từ $O(k(C_m^{k-1})^2)$ xuống còn $O(m.C_m^{k-1})$.

### *3.5. Thực nghiệm thuật toán*

Chương trình được cài đặt bằng ngôn ngữ Visual C++ 6.0 trên hệ điều hành Windows XP, chạy trên máy tính PC với Pentium dual core 2.0 GHz CPU, 1GB RAM. Kết quả thực nghiệm được thử trên CSDL thực (*Retail*) và CSDL nhân tạo (*T5I200D50K*, *T5I500D100K)*. Vì tất cả các CSDL này dùng cho việc khai phá tập mục

phổ biến, nên chúng tôi thêm vào số lượng các mục dữ liệu và giá trị lợi ích cho mỗi mục dữ liệu.

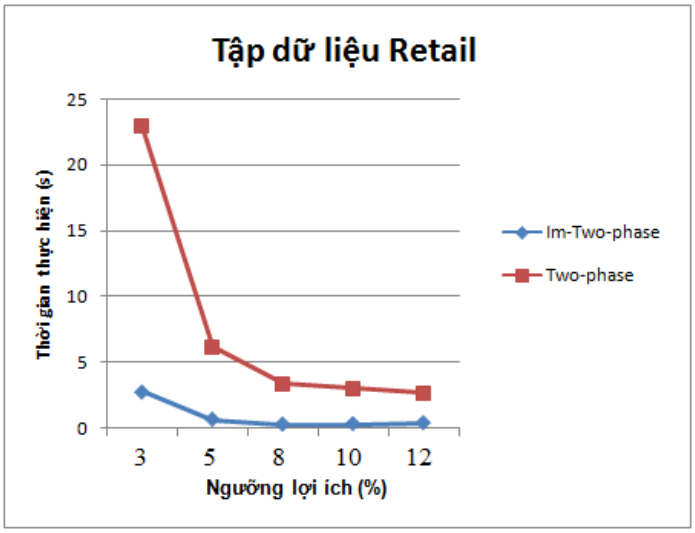
**Bảng 6.** *Đặc điểm các tập dữ liệu thử nghiệm*

| Tập dữ liệu | Số giao tác | Số mục dữ liệu | Độ dài trung bình giao tác |
|---|---|---|---|
| Retail | 88.162 | 16.470 | 9,79 |
| T5I500D100K | 100.000 | 500 | 7,62 |
| T51000D100K | 100.000 | 1000 | 10,1 |

Dựa vào kết quả so sánh thời gian thực hiện của hai thuật toán với sự thay đổi ngưỡng lợi ích trên ba CSDL trên, nhận thấy rằng thuật toán *Im-Two-Phase* thực hiện nhanh hơn thuật toán *Two-Phase*, đặc biệt khi ngưỡng lợi ích càng nhỏ.

**Bảng 7.** *Thời gian thực hiện (giây) của hai thuật toán với CSDL Retail*

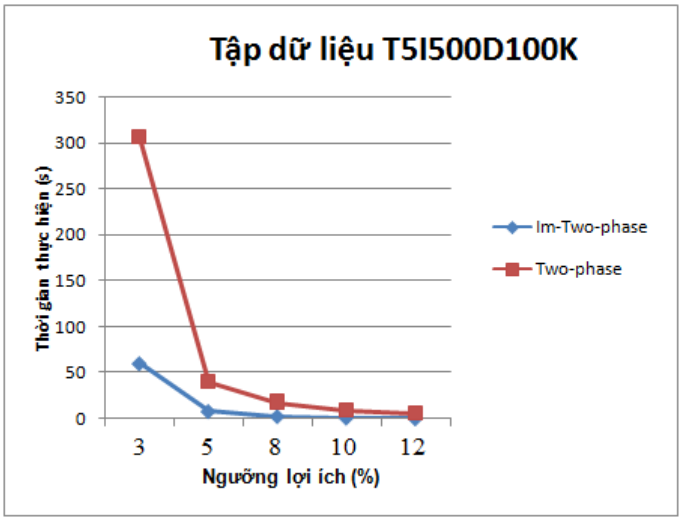
| Ngưỡng lợi ích (%) | 3 | 5 | 8 | 10 | 12 |
|---|---|---|---|---|---|
| **Two-phase** | 22,9 | 6,17 | 3,38 | 3,02 | 2,72 |
| **Im-Two-phase** | 2,8 | 0,69 | 0,32 | 0,35 | 0,43 |




**Hình 1.** *So sánh thời gian thực hiện (giây) của hai thuật toán với CSDL Retail*

**Bảng 8.** *Thời gian thực hiện (giây) của hai thuật toán với CSDL T5I500D100K*

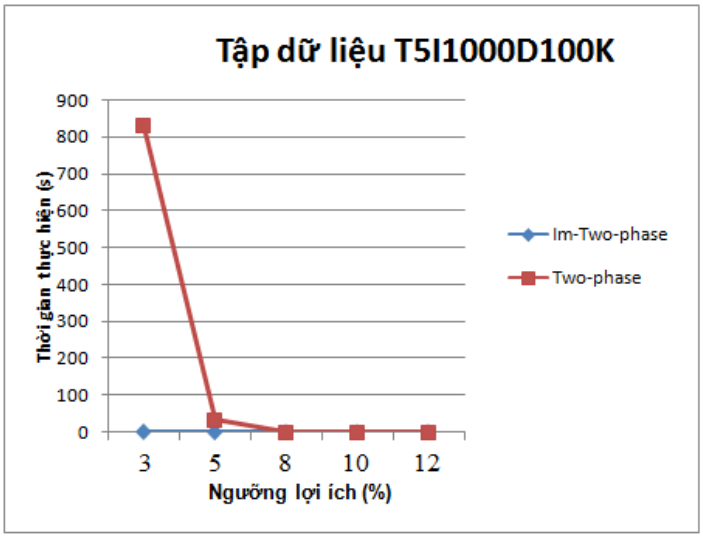
| Ngưỡng lợi ích (%) | 3 | 5 | 8 | 10 | 12 |
|---|---|---|---|---|---|
| **Two-phase** | 305,76 | 39,49 | 17,72 | 8,5 | 6,02 |
| **Im-Two-phase** | 60,1 | 8,34 | 3 | 1,37 | 0,59 |




**Hình 2.** *So sánh thời gian thực hiện (giây) của hai thuật toán với CSDL T5I500D100K*

**Bảng 9.** *Thời gian thực hiện (giây) của hai thuật toán với CSDL T5I1000D100K*

| Ngưỡng lợi ích (%) | 3 | 5 | 8 | 10 | 12 |
|---|---|---|---|---|---|
| **Two-phase** | 828,85 | 33,26 | 0,41 | 0,17 | 0,39 |
| **Im-Two-phase** | 1,3 | 0,35 | 0,34 | 0,33 | 0,33 |




**Hình 3.** *So sánh thời gian thực hiện (giây) của hai thuật toán với CSDL T51000D100K*

## 4. Kết luận

Trên cơ sở thuật toán *Two-Phase*, bài báo đã đề xuất một cải tiến thông qua chiến lược tỉa hiệu quả hơn các tập mục ứng cử, cải tiến bước sinh tập ứng viên, nhờ đó giảm bớt được thời gian thực hiện thuật toán khai phá. Chúng tôi đã cài đặt thử nghiệm với một số CSDL lớn, cả CSDL thực (*Retail*) và CSDL nhân tạo (*T5I200D50K*,

*T5I500D100K)* và so sánh với thuật toán *Two-Phase*, cho thấy thuật toán *Im-Two-Phase* mang lại hiệu quả.

Hướng nghiên cứu tiếp của chúng tôi là tìm hiểu, cài đặt một số thuật toán khai phá tập mục lợi ích cao trên cấu trúc dữ liệu dạng cây, cải tiến, so sánh hiệu quả giữa các thuật toán.

## TÀI LIỆU THAM KHẢO


[1]. Vũ Đức Thi, Nguyễn Huy Đức, *Khai phá hiệu quả tập mục lợi ích cao trong cơ sở dữ liệu lớn,* Tạp chí Tin học và Điều khiển học, 2008.

[2]. Nguyễn Thanh Tùng, *Khám phá tập mục lợi ích cao trong cơ sở dữ liệu,* Hội thảo Một số vấn đề chọn lọc của Công nghệ thông tin và truyền thông, Đại Lải, (2007), 181-197.

[3]. FIMI*, Frequent ItemSet Mining Implementations Repository*, 2003, http://fimi.cs.helsinki.fi/data.

[4]. *IBM Almaden Research Center Intelligent Information Systems*, Quest software, 2004.

[5]. *http://www.almaden.ibm.com/software/quest/Resources/index.shtml*.

[6]. Ying Liu, Wei-keng Liao, Alok Choudhary, *A Fast High Utility Itemsets Mining Algorithm,* Proceedings of the 1st international workshop on Utility-based data mining, Chicago, Illinois, (2005), 90-99.

[7]. Hong Yao, Howard J, Hamilton, Liqiang Geng, *A Unified Framework for Utility Based Measures for Mining Itemsets,* Second International Workshop on Utility-Based Data Mining, Philadelphia, PA, (2006), 28-37.

[8]. Hong Yao, Howard J, Hamilton, Cory J, Butz, *A Foundational Approach to Mining Itemset Utilities from Databases,* Proceedings of the Fourth SIAM International Conference on Data Mining, Orlando, Florida, USA, (2004), 482-486.


## AN ALGORITHM FOR MINING HIGH ULTILITY ITEMSETS IN DATABASE


*Nguyen Phuc Xuan Quynh*
*College of Pedagogy, Hue University*


### SUMMARY


*Developing from the mining associate rules problem, the mining high-utility itemsets problem has been a research topic of interest of many scientists with the aim to evaluate the role of itemsets in databases. Some data mining algorithms for high-utility itemsets have been developed. This article presents an algorithm which improves the Two-Phase algorithm with a strategy of inducing and pruning candidates to induce the execution time.*