Tạp chí Tin học và Điều khiển học, T. 15, S. 3 (1999), 8-17

MỘT VÀI THUẬT TOÁN CÀI ĐẶT CÁC PHÉP TOÁN CỦA ĐẠI SỐ QUAN HỆ TRONG MÔ HÌNH DỮ LIỆU DẠNG KHỐI

NGUYỄN XUÂN HUY, TRỊNH ĐÌNH THẮNG

Abstract. This paper presents some fundamental results about the block form - relational algebra together with algorithms realizing some block form - relational operators. Also, the estimation complexity of these algorithms are claimed and proved here.

1. MỞ ĐẦU

Trong những năm gần đây các mô hình dữ liệu dạng khối được phát triển mạnh mẽ nhằm trợ giúp cho việc khai thác các cơ sở dữ liệu một cách triệt để theo nhiều dấu hiệu trích chọn. Sự phát triển của các hệ trợ giúp quyết định và các hệ thống phát hiện tri thức từ cơ sở dữ liệu đòi hỏi một cách nhìn mới đối với cơ sở dữ liệu. Các quan hệ phẳng nói chung không còn thích hợp đối với các vấn đề nêu trên. Một trong những giải pháp được quan tâm nghiên cứu nhiều là tích hợp các quan hệ phẳng thành các khối dữ liệu. Quan niệm mới đòi hỏi những thao tác mới. Bài viết này trình bày cơ sở của đại số quan hệ dạng khối và đề xuất một số thuật toán cài đặt các phép toán quan hệ. Độ phức tạp tính toán của các thuật toán này cũng được đánh giá.

1.1. Mô hình dữ liệu dạng khối

Định nghĩa 1.1. Gọi $R = (\mathrm{id}; A_1, A_2, ..., A_n)$ là một bộ hữu hạn các phần tử, trong đó id là tập chỉ số hữu hạn khác rỗng, A_i $(i = 1, ..., n)$ là các thuộc tính. Mỗi thuộc tính A_i $(i = 1, ..., n)$ có miền giá trị tương ứng là $\mathrm{dom}(A_i)$. Một khối r trên tập R, kí hiệu $r(R)$ gồm một số hữu hạn phần tử mà mỗi phần tử là một họ các ánh xạ từ tập chỉ số id đến các miền trị của các thuộc tính A_i $(i = 1, ..., n)$. Nói một cách khác:

$$t \in r(R) \Leftrightarrow t = \{t^i : \mathrm{id} \to \mathrm{dom}(A_i)\}_{i=\overline{1,n}}.$$

Ta kí hiệu khối đó là $r(R)$ hoặc $r(\mathrm{id}; A_1, A_2, ..., A_n)$, đôi khi nếu không sợ nhầm lẫn ta kí hiệu đơn giản là r.

Khi đó khối $r(R)$ được gọi là có lược đồ khối R. Như vậy trên cùng một lược đồ khối R ta có thể xây dựng được nhiều khối khác nhau.

Định nghĩa 1.2. Cho $R = (\mathrm{id}; A_1, A_2, ..., A_n)$, $r(R)$ là một khối trên R. Với mỗi $x \in \mathrm{id}$ ta kí hiệu $r(R_x)$ là một khối với $R_x = (\{x\}; A_1, A_2, ..., A_n)$ sao cho:

$$t_x \in r(R_x) \Leftrightarrow t_x = \{t_x^i = t^i\Big|_x\}_{i=\overline{1,n}} \quad \text{với } t \in r(R), \text{ và } t = \{t^i : \mathrm{id} \to \mathrm{dom}(A_i)\}_{i=\overline{1,n}}.$$

Ở đây $t_x^i(x) = t^i(x)$ với $i = 1, ..., n$.

Khi đó $r(R_x)$ được gọi là một lát cắt trên khối $r(R)$ tại điểm x.

1.2. Đại số quan hệ trên khối

Cho r là một khối trên $R = (\mathrm{id}; A_1, A_2, ..., A_n)$, ở đây ta luôn giả thiết rằng r là một khối gồm một tập hữu hạn các phần tử. Cũng tương tự như đại số quan hệ trong mô hình dữ liệu quan hệ, ở

đây các phép toán của đại số quan hệ lại được áp dụng cho các khối; bên cạnh đó còn có thêm một phép toán mới được xây dựng là phép nối dài.

Đối với các phép hợp, giao và trừ thì hai khối tham gia phải là khả hợp (nghĩa là có cùng một lược đồ).

a. Phép hợp:

Cho hai khối r và s khả hợp, khi đó hợp của r và s, kí hiệu $r \cup s$ là một khối gồm các phần tử thuộc một trong hai khối r và s đã cho.

$$r \cup s = \{t \mid t \in r \text{ hoặc } t \in s\}.$$

b. Phép giao:

Cho hai khối r và s khả hợp, khi đó giao của r và s là một khối, kí hiệu $r \cap s$ mà các phần tử của nó thuộc đồng thời cả hai khối r và s đã cho

$$r \cap s = \{t \mid t \in r \text{ và } t \in s\}.$$

c. Phép trừ:

Cho hai khối r và s khả hợp, khi đó hiệu của r và s là một khối, kí hiệu $r - s$ mà các phần tử của nó thuộc r nhưng không thuộc s.

$$r - s = \{t \mid t \in r \text{ và } t \notin s\}$$

Ta có mối quan hệ giữa phép giao và phép trừ:

$$r \cap s = r - (r - s).$$

d. Phép chiếu:

Cho $R = (\mathrm{id}; A_1, A_2, ..., A_n)$, r là một khối trên R.

Khi đó ta gọi $P = (\mathrm{id}'; A_{i_1}, A_{i_2}, ..., A_{i_h})$ là lược đồ con của lược đồ R nếu $\mathrm{id}' \subseteq \mathrm{id}$, $A_{i_j} \in \{A_1, A_2, ..., A_n\}$, $j = 1, ..., h$.

Một phép chiếu của khối r trên lược đồ con P, kí hiệu $\Pi_p(r)$, là một khối có lược đồ khối P và mỗi phần tử thuộc khối này có dạng:

$$(t^{i_1}, t^{i_2}, ..., t^{i_h})\Big|_{\mathrm{id}'}, \quad \text{trong đó } t^{i_j} \in \{t^1, t^2, ..., t^n\},\ j = 1, ..., h, \text{ và } (t^1, t^2, ..., t^n) \in r.$$

Biểu diễn hình thức của phép chiếu có dạng:

$$\Pi_p(r) = \left\{(t^{i_1}, t^{i_2}, ..., t^{i_h})\Big|_{\mathrm{id}'} \mid t^{i_j} \in \{t^1, t^2, ..., t^n\},\ j = 1, ..., h,\ (t^1, t^2, ..., t^n) \in r\right\}.$$

e. Phép chọn:

Cho $R = (\mathrm{id}; A_1, A_2, ..., A_n)$, và khối $r(R)$.

Cho một phép chọn nghĩa là ta xây dựng một tập con các phần tử của khối đã cho thỏa mãn biểu thức F cho trước. Biểu thức F được diễn tả bằng một tổ hợp logic của các toán hạng, mỗi toán hạng là một phép so sánh đơn giản giữa hai biến là hai giá trị điểm của hai ánh xạ thành phần nào đó, hoặc giữa một biến là giá trị điểm của một ánh xạ thành phần và một hằng.

Các phép so sánh trong F là $<$, $=$, $>$, $\geq$, $\leq$, $\neq$, còn các phép toán logic trong F là: $\vee$, $\wedge$, $\neg$.

Biểu diễn hình thức của phép chọn có dạng:

$$\sigma_F(r) = \{t \in r \mid F(t)\},$$

trong đó $F(t)$ là giá trị của biểu thức logic F tại phần tử $t \in r$.

g. Phép kết nối:

Cho $R = (\mathrm{id}; A_1, A_2, ..., A_n)$ và $s = (\mathrm{id}, B_1, B_2, ..., B_m)$, cùng với hai khối $r(R)$ và $s(S)$ tương ứng.

Gọi $T = (\mathrm{id}; C_1, C_2, ..., C_p)$, trong đó:

$\{C_1, C_2, ..., C_p\} = \{A_1, A_2, ..., A_n\} \cup \{B_1, B_2, ..., B_m\}$. Như vậy trong $\{C_1, C_2, ..., C_p\}$ tồn tại 2 họ $\{C_{i_k}\}_{k=\overline{1,n}}$ và $\{C_{j_h}\}_{h=\overline{1,m}}$ sao cho: $C_{i_1} = A_1, C_{i_2} = A_2, ..., C_{i_n} = A_n$, $C_{j_1} = B_1, C_{j_2} = B_2, ..., C_{j_m} = B_m$. Khi đó ta ký hiệu:

$t(R) = (t^{i_1}, t^{i_2}, ..., t^{i_n})$, $t^{i_k} \in \{t^1, t^2, ..., t^p\}$, $k = 1, ..., n$,

$t(S) = (t^{j_1}, t^{j_2}, ..., t^{j_m})$, $t^{j_h} \in \{t^1, t^2, ..., t^p\}$, $h = 1, ..., m$,

với $t = (t^1, t^2, ..., t^p)$.

Phép kết nối của 2 khối r và s, kí hiệu $r \bowtie s$ là khối $t(T)$ định nghĩa như sau:

$t(T) = \{t \mid \exists t_r \in r$ và $t_s \in s$ sao cho $t(R) = t_r, t(S) = t_s\}$.

Phép kết nối này cũng gọi là phép kết nối tự nhiên của hai khối $r(R)$ và $s(S)$, đôi khi sử dụng kí hiệu $r * s$.

Đặc biệt, khi các khối $r(R)$ và $s(S)$ có tập chỉ số id trong lược đồ khối của chúng chỉ gồm 1 phần tử thì các khối này trở thành các quan hệ và phép kết nối tự nhiên của hai khối lại trở thành phép kết nối tự nhiên của hai quan hệ trong mô hình cơ sở dữ liệu quan hệ [6].

Nếu hai tập $\{A_1, A_2, ..., A_n\}$ và $\{B_1, B_2, ..., B_m\}$ không giao nhau thì $r * s$ trở thành tích Đề các của hai khối đã cho.

Ta có thể mở rộng khái niệm kết nối như sau:

Giả sử $A_{i_k} \in \{A_1, A_2, ..., A_n\}$, $B_{i_k} \in \{B_1, B_2, ..., B_m\}$, và $\mathrm{dom}(A_{i_k}) = \mathrm{dom}(B_{i_k})$, $1 \le k \le h$, (ở đây A_{i_k} và B_{i_k} không nhất thiết phân biệt).

Khi đó kết nối của r và s theo $A_{i_1}, A_{i_2}, ..., A_{i_h}$ và $B_{i_1}, B_{i_2}, ..., B_{i_h}$ là khối $t(T)$, khối này được định nghĩa như sau:

$$t(T) = \{t \mid \exists t_r \in r \text{ và } t_s \in s \text{ sao cho } t(R) = t_r,\ t(S) = t_s,\ t_r^{i_k} = t_s^{i_k},\ 1 \le k \le h\}$$

ở đây $t_r = (t_r^1, t_r^2, ..., t_r^n)$, $t_s = (t_s^1, t_s^2, ..., t_s^m)$.

Thay cho kí hiệu $r \bowtie s$ ở đây ta kí hiệu rõ hơn:

$$t(T) = r\,[t_r^{i_k} = t_s^{i_k},\ 1 \le k \le h]\ s.$$

h. Phép nối dài:

Cho hai khối $r(\mathrm{id}; A_1, A_2, ..., A_n)$ và $s(\mathrm{id}'; A_1, A_2, ..., A_n)$, ở đó nếu $\mathrm{id} \cap \mathrm{id}' \neq \emptyset$ mà ta có với $t \in r$ và $k \in s$:

$$t^1\big|_{\mathrm{id} \cap \mathrm{id}'} = k^1\big|_{\mathrm{id} \cap \mathrm{id}'}$$

$$t^2\big|_{\mathrm{id} \cap \mathrm{id}'} = k^2\big|_{\mathrm{id} \cap \mathrm{id}'}$$

$$\cdots$$

$$t^n\big|_{\mathrm{id} \cap \mathrm{id}'} = k^n\big|_{\mathrm{id} \cap \mathrm{id}'}$$

thì khi đó ta xây dựng được một phần tử mới có dạng:

$$u = (u^1, u^2, ..., u^n) \text{ với } u^h : \mathrm{id} \cup \mathrm{id}' \to \mathrm{dom}(A^h)$$

sao cho: $u^h\big|_{\mathrm{id}} = t^h$, $u^h\big|_{\mathrm{id}'} = k^h$, $\forall h = 1, ..., n$, và kí hiệu: $u^h = t^h *_{\mathrm{id}} k^h$, $\forall h = 1, ..., n$.

Những phần tử $u = (u^1, u^2, ..., u^n)$ này tạo ra một khối mới, được kí hiệu: $r *_{\mathrm{id}} s$, gọi là khối nối dài của hai khối r và s.

Phép toán được xây dựng ở trên gọi là phép nối dài của hai khối r và s đã cho.

Biểu diễn hình thức của phép nối dài có dạng:

$$r *_{\mathrm{id}} s = \left\{u = (u^i)_{i=\overline{1,n}} \mid (u^i\big|_{\mathrm{id}})_{i=\overline{1,n}} \in r \text{ và } (u^i\big|_{\mathrm{id}'})_{i=\overline{1,n}} \in s\right\}$$

k. Phép chia:

Cho khối $r(\text{id}; A_1, A_2, ..., A_h)$ và khối $s(\text{id}; A_{i_1}, A_{i_2}, ..., A_{i_k})$, trong đó $A_{i_j} \in \{A_1, A_2, ..., A_n\}$, $\forall j = 1, ..., k$, khi đó phép chia của khối r cho khối s, kí hiệu $r \div s$, là một khối gồm các phần tử $t = (t^1, t^2, ..., t^{n-h})$ sao cho $\forall u = \{u^1, u^2, ..., u^h)$, $u \in s$ thì phần tử $tu \in r$, ở đây phần tử tu có dạng: $tu = (t^1, t^2, ..., t^{n-h}, u^1, u^2, ..., u^h)$.

Biểu diễn hình thức của phép chia có dạng: $r \div s = \{t \mid \forall u \in s, tu \in r\}$.

Trong trường hợp các khối r và s có tập chỉ số id chỉ gồm 1 phần tử thì khi đó các khối này trở thành các quan hệ và phép toán chia của hai khối trở thành phép toán chia của hai quan hệ trong mô hình dữ liệu quan hệ [6].

2. CÁC TOÁN TỬ THAO TÁC CƠ SỞ

Để xây dựng các thuật toán cho các phép toán của đại số quan hệ và đánh giá độ phức tạp của nó, trước hết ta xem xét các toán tử thao tác cơ sở trên các khối (các toán tử thao tác này tương tự như các toán tử thao tác trên các quan hệ đã được trình bày trong [1], [4]):

1. for each t in r do M;
   Ngữ nghĩa: với mỗi bộ t thuộc khối r thực hiện toán tử M.
2. t in r;
   Đây là một hàm logic nhận giá trị đúng (true) nếu bộ t có mặt trong khối r, ngược lại hàm nhận giá trị sai (false).
3. add t to r;
   Ngữ nghĩa: đưa thêm bộ t vào khối r, không kiểm tra trùng lặp.
4. card(r): hàm xác định lực lượng (số bộ) của khối r.
5. construct1 $t = \langle t^1, t^2, ..., t^n \rangle$;
   Ngữ nghĩa: tạo một bộ t từ các ánh xạ $t^1, t^2, ..., t^n$ cho trước.
6. construct2 $t = \langle x^1, x^2, ..., x^n, y^1, y^2, ..., y^n \rangle$;
   Ngữ nghĩa: tạo một bộ $t = (z^1, z^2, ..., z^n)$, ở đó ánh xạ $z^i : (\text{id} \cup \text{id}') \to \text{dom}(A_i)$, $i = 1, ..., n$ được xây dựng từ các ánh xạ $x^i : \text{id} \to \text{dom}(A_i)$, $y^i : \text{id}' \to \text{dom}(A_i)$, $i = 1, ..., n$ với $\text{id} \cap \text{id}' = \emptyset$, bằng cách như sau:
   Nếu card(id) = h, card(id') = k thì ánh xạ x^i được biểu diễn là bộ h giá trị, còn ánh xạ y^i được biểu diễn là bộ k giá trị:
   $x^i = (x^i_1, x^i_2, ..., x^i_h)$, $y^i = (y^i_1, y^i_2, ..., y^i_k)$ khi đó $z^i = (x^i_1, x^i_2, ..., x^i_h, y^i_1, y^i_2, ..., y^i_k)$.
7. label t in r;
   Ngữ nghĩa: đánh nhãn cho bộ t trong khối r. Bộ được đánh nhãn sẽ không tham gia vào quá trình xét duyệt khối r, ta nói rằng bộ đó bị loại về mặt logic khỏi khối r. Việc này được thực hiện thông qua việc lập chỉ dẫn trên các phần tử của khối r chẳng hạn. Mỗi phần tử của chỉ dẫn bao gồm hai thành phần: khóa tuyển chọn và địa chỉ (con trỏ) tới phần tử của khối. Việc đánh nhãn cho mỗi phần tử tương đương với việc loại bỏ tạm thời một phần tử của chỉ dẫn trỏ tới phần tử được đánh nhãn đó của khối.
8. test(t, b): hàm logic nhận giá trị đúng nếu bộ t thỏa mãn biểu thức logic b, ngược lại hàm nhận giá trị sai.
9. create(R);
   Ngữ nghĩa: tạo một khối rỗng từ lược đồ khối R cho trước.

3. CÁC KẾT QUẢ

Phần này trình bày một số thuật toán cài đặt và đánh giá độ phức tạp thời gian của chúng đối với các phép toán của đại số quan hệ nói trên.

Các thuật toán 1-6 được xây dựng trên cơ sở các thuật toán tương ứng trong mô hình dữ liệu quan hệ với toán tử đánh nhãn đã nêu trong [4].

1. Phép hợp

```
algorithm hop
input: hai khối khả hợp p(R) và q(R).
output: khối r(R) = {x : x ∈ p ∨ x ∈ q}.
format: hop (p, q)
begin
    r := create (R);
    for each x in p do
        add x to r;
    for each y in q do
        if y in p then label y in p
                  else add y to r.
    return (r)
    end;
```

2. Phép giao

```
algorithm giao
input: hai khối khả hợp p(R) và q(R).
output: khối r(R) = {x : x ∈ p ∧ x ∈ q}.
format: giao (p, q)
begin
    r := create (R);
    for each x in p do
        if x in q then
            begin
                add x to r;
                label x in q;
            end;
    return(r)
end;
```

3. Phép trừ

```
Algorithm tru
input: hai khối khả hợp p(R) và q(R).
output: khối r(R) = {x : x ∈ p ∧ x ∉ q}.
format: tru (p, q)
begin
    r := create (R);
    for each x in p do
        if x in q then label x in q
                  else add x to r;
    return(r)
end;
```

4. Phép chiếu

algorithm chieu

input; khối $r(R)$ với $R = (\text{id}; A_1, A_2, ..., A_n)$ và $P = (\text{id}'; A_{i_1}, A_{i_2}, ..., A_{i_h})$,
$A_{i_j} \in \{A_1, A_2, ..., A_n\}$, $\forall j = 1, ..., h$.

output: khối $s(P) = \{x.P : x \in r\}$.

format: chieu(r, P)

```
begin
    s :=create(P);
    for each x in r do
        if not (x.P in s) then add x.P to s;
    return(s)
end;
```

ở đây phần tử $x.P$ có dạng: $x.P = (x^{i_1}|_{\text{id}'}, x^{i_2}|_{\text{id}'}, ..., x^{i_h}|_{\text{id}'})$

trong đó $x^{i_j} \in \{x^1, x^2, ..., x^n\}$, $j = 1, ..., h$, và $(x^1, x^2, ..., x^n) \in r$.

5. Phép chọn

algorithm chon

input: khối $r(R)$ và tiêu chuẩn F.

output: khối $s(R) = \{x \in r(R) \mid \text{test } (x, F)\}$.

format: chon (r, F)

```
begin
    s :=create(R);
    for each x in r do
        if test(x, F) and add x to s;
    return(s)
end;
```

6. Phép kết nối tự nhiên

algorithm ketnoi

input: hai khối $r(R)$ và $s(S)$, $R = (\text{id}; A_1, A_2, ..., A_n)$, $S = (\text{id}; B_1, B_2, ..., B_m)$ với $A_1 = B_1, ..., A_h = B_h$.

output: khối $t(T) = \{z \mid z.R \in r(R) \wedge z.S \in s(S)\}$.

format: ketnoi (r, s)

```
begin
    T := (id; A1, ..., Ah, Ah+1, ..., An, Bh+1, ..., Bm);
    P := (id; A1, A2, ..., Ah); t := create(T);
    for each x in r do
        for each y in s do
            if x.P = y.P then
                begin
                    construct1 z = ⟨x, y.(S \ P)⟩;
                    add z to t
                end;
    return(t)
end;
```

với $x.P = (x^1|_{\mathrm{id}}, x^2|_{\mathrm{id}}, ..., x^h|_{\mathrm{id}})$ và $y.P = (y^1|_{\mathrm{id}}, y^2|_{\mathrm{id}}, ..., y^h|_{\mathrm{id}})$

trong đó $(x^1, x^2, ..., x^h, x^{h+1}, ..., x^n) \in r$, $(y^1, y^2, ..., y^h, y^{h+1}, ..., y^m) \in s$.

$$S \setminus P = (\mathrm{id}; B_1, B_2, ..., B_m) \setminus (\mathrm{id}; A_1, A_2, ..., A_h)$$
$$= (\mathrm{id}; B_1, B_2, ..., B_m) \setminus (\mathrm{id}; B_1, B_2, ..., B_h)$$
$$= (\mathrm{id}; B_h, B_{h+1}, ..., B_m).$$

7. Phép nối dài

algorithm noidai
input: hai khối $r(R)$ và $s(S)$, $R = (\mathrm{id}; A_1, A_2, ..., A_n)$, $S = (\mathrm{id}'; A_1, A_2, ..., A_n)$, $\mathrm{id} \cap \mathrm{id}' \neq \emptyset$, $T = (\mathrm{id} \cup \mathrm{id}'; A_1, A_2, ..., A_n)$.
output: khối $t(T) = \{z \mid z.R \in r(R) \wedge z.S \in s(S)\}$.
format: noidai (r, s)

```
begin
    P := (id ∩ id'; A1, A2, ..., An);
    t := create(T);
    for each x in r do
        for each y in s do
            if x.P = y.P then
                begin
                    construct2 z = ⟨x, y.(S \ P)⟩;
                    add z to t
                end;
    return (t)
end;
```

trong đó: $S \setminus P = (\mathrm{id}; A_1, A_2, ..., A_n) \setminus (\mathrm{id} \cap \mathrm{id}'; A_1, A_2, ..., A_h) = (\mathrm{id}' \setminus (\mathrm{id} \cap \mathrm{id}'); A_1, A_2, ..., A_h)$.

8. Phép chia

algorithm chia
input: hai khối $r(R)$ và $s(S)$, $R = (\mathrm{id}; A_1, A_2, ..., A_g)$, $S = (\mathrm{id}; A_1, A_2, ..., A_k)$, $k \leq g$.
output: khối $t(T) = \{x.T \mid x \in r, \forall y \in s$ ta có $\langle x.T, y\rangle \in r\}$, $T = (\mathrm{id}; A_{k+1}, A_{k+2}, ..., A_g)$.
format: chia (r, s)

```
begin
    n := card(s);  t := create(T);
    if n = 0 then t := {x.T|x ∈ r} else
        begin
            đếm := 0;
        {đếm dùng để chỉ số phần tử hiện có của mảng left}
            for each x in r do
                begin
                    construct1 x1 := ⟨x.T⟩;
                    construct1 x2 := ⟨x.S⟩;
                    k := find(x1);
```

```
                if k = 0 then {điền thêm đại diện vào cuối của left}
                    begin
                        đếm := đếm+1; left(đếm) := x1;
                        k :=đếm; add x2 to right(k);
                        count(k) := 1;
                    end;
                if k > 0 then {thêm phần tử vào cuối của right}
                    begin
                        add x2 to right(k);
                        count(k) := cound(k) + 1;
                    end;
            end;
        k := 0;
        while k < đếm do
            begin
               k := k + 1; {xét các lớp tương đương có số phần tử >= n}
               if count(k) >= n then
                begin
                    đếm 1 := 0;
                    for each y in right(k) do
                        if đếm1 < n then
                            if y in s then đếm1 := đếm1 + 1;
                        if đếm1 = n then add left(k) to t
                end;
            end;
    end;
  return(t)
end;
```

Mệnh đề 3.1. *Các thuật toán 1-8 cho ta kết qud đúng đối với các phép toán tương ứng trên các khối đã cho.*

Sau đây ta đánh giá độ phức tạp thời gian của một vài thuật toán.

Giả thiết rằng nếu một khối có m phần tử thì việc xác định xem một phần tử x cho trước có mặt hay không trong khối đó đòi hỏi $f(m)$ phép so sánh phần tử x với các phần tử khác của khối.

Với $m = \text{card}(p)$, $n = \text{card}(q)$, $h = \text{card}(p \cap q)$, ở thuật toán hợp ta thấy nếu y in q thì việc kiểm tra $(y \in p)$ cần $f(k)$ phép so sánh với $k \leq m$. Nếu trước đó trong p đã có một số bộ được đánh nhãn thì $k < m$. Trường hợp xấu nhất là h phần tử của $p \cap q$ lại ở cuối của khối q, khi đó việc kiểm tra $(y \in p)$ đối với $n - h$ phần tử đầu tiên của khối q đòi hỏi $(n - h)f(m)$ phép so sánh. Còn với h phần tử cuối của q, vì chúng cũng có mặt trong p nên sau mỗi bước ta đánh nhãn được một phần tử trong p, do vậy số các phép so sánh cho h phần tử này là

$$\sum_{i=1}^{h} f(m - i + 1).$$

Như vậy trong trường hợp xấu nhất thuật toán hợp đòi hỏi

$$\sum_{i=1}^{h} f(m - i + 1) + (n - h)f(m) \quad \textbf{phép so sánh.}$$

Mệnh đề 3.2. *Thuật toán hợp có độ phức tạp thời gian là:*

$$\sum_{i=1}^{h} f(m-i+1) + (n-h)f(m).$$

Tương tự như trên ta có mệnh đề sau:

Mệnh đề 3.3. *Các thuật toán giao và hiệu có độ phức tạp thời gian là:*

$$\sum_{i=1}^{h} f(n-i+1) + (m-h)f(n).$$

Bây giờ ta xét thuật toán chia. Với lược đồ con $T = (\mathrm{id}; A_{k+1}, A_{k+2}, \ldots, A_g)$, ta thực hiện một phân hoạch khối $r(R)$ theo quan hệ α sau:

$$\forall x, y \in r : x \alpha y \Leftrightarrow x.T = y.T,$$

trong đó $x.T$ và $y.T$ được hiểu như đã nói ở phần trên là

$$x.T = (x^{k+1}\Big|_{\mathrm{id}}, x^{k+2}\Big|_{\mathrm{id}}, \ldots, x^{g}\Big|_{\mathrm{id}}),$$

$$y.T = (y^{k+1}\Big|_{\mathrm{id}}, y^{k+2}\Big|_{\mathrm{id}}, \ldots, y^{g}\Big|_{\mathrm{id}}).$$

Dễ thấy rằng α là một quan hệ tương đương. Gọi u là số lớp tương đương trong phân hoạch r/α. Kí hiệu m_i là số phần tử trong lớp thứ i $(i = 1, 2, \ldots, u)$ thì theo tính chất của phân hoạch ta có:

$$m = \sum_{i=1}^{u} m_i .$$

Đối với mỗi lớp tương đương của quan hệ α ta thành lập hai mảng left và right, mảng left dùng để chứa các phần tử dạng $x.T$, $(x \in r)$, còn mảng right dùng để chứa các phần tử tương ứng dạng $x.S$, $(x \in r)$. Như vậy đối với phần tử $x \in r$ thì x bị chia thành hai phần: phần $x.T$ sẽ được lưu trong mảng left, còn phần $x.S$ được lưu trong mảng right. Do đó, sau khi duyệt một lượt các phần tử của khối r thì mảng left sẽ chứa các phần tử dạng $x.T$ là đại diện cho các lớp tương đương, số phần tử của mảng left chính là số các lớp tương đương theo quan hệ α, còn đối với các phần tử tương ứng của mảng right thì mỗi phần tử của mảng chứa phần còn lại $x.S$ của các phần tử trong một lớp tương đương ứng với một phần tử của mảng left, (trong thực hành nếu không thể lưu trữ các mảng left, right trong bộ nhớ trong thì ta lưu các phần tử của chúng ở các thiết bị nhớ ngoài và các mảng này sẽ chỉ chứa các con trỏ trỏ tới các phần tử đó).

Trong thuật toán, count là một mảng và count(k) đếm số phần tử của lớp tương đương thứ k, hàm find($x1$) với $x1 = x.T$ cho chỉ số k của mảng left sao cho left(k) = $x1$ (nếu tìm được) và find($x1$) = 0 trong trường hợp ngược lại.

Mệnh đề 3.4. *Thuật toán chia có độ phức tạp thời gian là:*

$$\sum_{i=1}^{u} f(i) + (m-u)f(u) + \sum_{m_k \geq n} \theta_k f(n)$$

và khi $s = \Pi_s(r)$ *thì độ phức tạp thời gian là:*

$$\sum_{i=1}^{u} f(i) + (m-u)f(u),$$

ở đó $m = \mathrm{card}(r)$, $n = \mathrm{card}(s)$, $u = \mathrm{card}(r/\alpha)$,

m_k là số phần tử của lớp tương đương thứ k,
θ_k là số phần tử của lớp tương đương thứ k được xét đủ để kết luận rằng đếm $1 = n$.

Chứng minh. Khi xây dựng danh sách left ta phải dùng đến hàm find, nên phải chi phí thời gian là:

$$\sum_{i=1}^{u} f(i).$$

Sau khi đã có các phần tử đại diện left(k), $(k = 1, ..., u)$ của các lớp tương đương trong r/α thì các phần tử còn lại của r sẽ được tìm bởi hàm find với thời gian $(m - u)f(u)$.

Nhờ phép kiểm tra count(k) >= n nên ta chỉ hạn chế xem xét các lớp tương đương nào có số phần tử lớn hơn hoặc bằng n. Đối với mỗi lớp tương đương thứ k thỏa điều kiện này, để xác định đếm $1 = n$ ta phải duyệt θ_k phần tử của chúng. Như vậy, thời gian để thực hiện phép kiểm tra θ_k phần tử của lớp tương đương thứ k là $\theta_k f(n)$. Do vậy thời gian thực hiện của thuật toán này là:

$$\sum_{i=1}^{u} f(i) + (m - u)f(u) + \sum_{m_k \geq n} \theta_k f(n).$$

Khi $s = \Pi_s(r)$ thì do không phải mất thời gian để kiểm tra (y in s) trong các lớp tương đương đã chọn (ở đây chỉ cần xét xem có lớp tương đương nào có số lượng các phần tử bằng n hay không) nên độ phức tạp thời gian chỉ còn là:

$$\sum_{i=1}^{u} f(i) + (m - u)f(u).$$

TÀI LIỆU THAM KHẢO

[1] L. L. Beck, A Generalized Implementation Method for Relational Data Sublanguages, *IEEE Transactions on Software Engineering* **Se-6** (2) (1980) 152-162.

[2] J. Demetrovics, Nguyen Xuan Huy, Closed sets and translations of relation schemes, *Computers Math. Applic.* **21** (1) (1991) 13-23.

[3] D. Maier, *The Theory of Relational Databases*, Computer Science Press, Rockville, Md, 1983.

[4] Nguyễn Xuân Huy, Toán tử đánh nhãn và các phép toán quan hệ, *Tạp chí Khoa học tính toán và Điều khiển* **1** (3) (1985) 1-7.

[5] Nguyễn Thanh Thủy, Nguyễn Xuân Huy, Một vài thuật toán thực hiện phép chia trong mô hình quan hệ, *Tạp chí Khoa học tính toán và Điều khiển* **1** (4) (1985) 1-6.

[6] Nguyễn Xuân Huy, Trịnh Đình Thắng, Mô hình cơ sở dữ liệu dạng khối, *Tạp chí Tin học và Điều khiển học* **14** (3) (1998) 52-60.

Nhận bài ngày 28 - 4 - 1999
Nhận lại sau khi sửa ngày 1 - 6 - 1999

Nguyễn Xuân Huy, Viện Công nghệ thông tin.
Trịnh Đình Thắng, Trường Đại học Sư phạm Hà Nội 2.