TOÁN TỬ ĐÁNH NHÃN VÀ CÁC PHÉP TOÁN QUAN HỆ

NGUYỄN XUÂN HUY

I – ĐẶT VẤN ĐỀ

Cho n miền $D_1, D_2, \ldots, D_n$ $(n \geqslant 1)$ chứa các dữ liệu nguyên tố, tức là các đơn vị dữ liệu không thể chia nhỏ được nữa trong một hệ cơ sở dữ liệu (HCSDL) cho trước. Quan hệ R với các thuộc tính $A_1, A_2, \ldots A_n$ là một tập các hàm $r: \{A_1, A_2, .., A_n\} \rightarrow \bigcup_{i=1}^{n} D_i$ sao cho với $\forall A_i \in \{A_1, A_2, ..., A_n\}$ thì $r(A_i) \in D_i$. Mỗi hàm như trên được gọi là một bộ của quan hệ R. Tổng số bộ trong một quan hệ được gọi là lực lượng của quan hệ đó, số lượng các thuộc tính được gọi là bậc của một quan hệ. Để chỉ rõ danh sách thuộc tính trong một quan hệ R ta viết $R(A_1, A_2, ..., A_n)$. Với mỗi quan hệ R ta ký hiệu $\mathcal{A}(R)$ là tập các thuộc tính của R. Nếu $B \subseteq \mathcal{A}(R)$ ta viết R.B để chỉ rõ xuất xứ của nhóm thuộc tính B; trong trường hợp không thể xảy ra nhầm lẫn ta chỉ viết B. Nếu B là một thuộc tính (hoặc nhóm thuộc tính) và r là một bộ của quan hệ R ta sẽ viết r.B để biểu diễn một (hoặc một nhóm) các giá trị thành phần trong r ứng với thuộc tính (hoặc nhóm thuộc tính) B. Chi tiết hơn, thay cho r ta thường viết $< r_1, r_2, .., r_n >$ hoặc $< r.A_1, r.A_2, ..., r.A_n >$ và như vậy ta đã ngầm qui định $r_i = r.A_i$ $(i = 1, 2, \ldots, n)$.

Hai thuộc tính A và B được gọi là khả hợp nếu chúng chứa những giá trị thuộc cùng một miền. Hai nhóm thuộc tính M và N được gọi là khả hợp nếu tồn tại một song ánh $\varphi: M \rightarrow N$ sao cho với $\forall A \in M$ thì A và $\varphi(A)$ là khả hợp. Mở rộng ra, hai quan hệ R và S được gọi là khả hợp nếu $\mathcal{A}(R)$ và $\mathcal{A}(S)$ là hai tập thuộc tính khả hợp. Trong thực tiễn cài đặt các HCSDL người ta thường chấp nhận định nghĩa chặt hơn như sau: Hai nhóm thuộc tính $M = \{A_1, A_2, \ldots, A_m\}$ và $N = \{B_1, B_2, ..., B_n\}$ được gọi là khả hợp nếu:

(i) $m = n$, và

(ii) $\forall i = 1, 2, ..., n$: A_i và B_i khả hợp.

Trong [2], [3] Codd – tác giả của mô hình quan hệ đã định nghĩa một đại số quan hệ với một số phép toán trên các quan hệ dùng để diễn đạt các yêu cầu tìm kiếm dữ liệu trong CSDL.

Trong [1] Beck đã đề xuất một số toán tử thao tác trên các bộ của quan hệ nhằm thể hiện các phép toán của đại số quan hệ nói riêng và các câu lệnh của các ngôn ngữ quan hệ (như SEQUEL, QBE, PDL) nói chung để tiến tới thuật toán tối ưu hóa các câu hỏi trong các ngôn ngữ quan hệ.

Mục II của bài này sẽ đề xuất thêm một vài toán tử thao tác bộ. Mục III trình bày các định nghĩa mở rộng và các thuật toán cải tiến cho các phép toán quan hệ. Việc mở rộng các phép toán quan hệ là một yêu cầu tất yếu để giữ cho việc diễn đạt các câu hỏi được gọn hơn và quá trình thực hiện các câu hỏi đó đỡ vụn hơn. Các thuật toán cải tiến về cơ bản dựa trên toán tử đánh nhãn bộ nhằm mục đích giảm số lần so sánh bộ. Điều đó tương đương với việc giảm số lần trao đổi giữa bộ nhớ trong và các thiết bị lưu trữ. Hiệu quả của các thuật toán được nêu trong mục IV.

II – CÁC TOÁN TỬ THAO TÁC BỘ

Xét tập các toán tử sau [1]:

1. for each $r \in R$ do M;

Ngữ nghĩa: với mỗi bộ r thuộc quan hệ R thực hiện toán tử M.

2. fos each $r \in R$ with b do M;

Ngữ nghĩa: với mỗi bộ r thuộc quan hệ R, nếu điều kiện logic b được thỏa mãn thì thực hiện toán tử M.

3. if $\exists s \in S$ with b then M else N;

Ngữ nghĩa: Nếu tồn tại một bộ s trong quan hệ S thỏa mãn điều kiện (logic) b thì thực hiện toán tử M, ngược lại thực hiện toán tử N. Toán tử 3, còn có dạng ngắn là:

if $\exists s \in S$ with b then M;

M và N trong các dạng trên có thể là những toán tử trống.

4. construct $t = \langle t_1, t_2, \ldots, t_n \rangle$;

Ngữ nghĩa: Tạo một bộ t (trên miền nhớ trung tâm) từ các giá trị $t_1, t_2, \ldots, t_n$ cho trước.

5. add t to T;

Ngữ nghĩa: Đưa thêm bộ t vào quan hệ T.

Ngoài ra ta xét thêm một toán tử và hai thủ tục sau đây:

6. label r in R;

Ngữ nghĩa: đánh nhãn cho bộ r trong quan hệ R. Bộ được đánh nhãn sẽ không tham gia vào quá trình xét duyệt quan hệ R; ta nói rằng bộ đó được loại về mặt logic khỏi quan hệ R. Việc này được thực hiện thông qua việc lập chỉ dẫn trên các bộ của quan hệ R chẳng hạn. Mỗi phần tử của chỉ dẫn bao gồm hai thành phần: khóa tuyển chọn và địa chỉ (con trỏ) tới bộ. Việc đánh nhãn cho một bộ tương đương với việc loại bỏ tạm thời một phần tử của chỉ dẫn trỏ tới bộ đó.

7. t in R.

Đây là một hàm logic nhận giá trị đúng (true) nếu bộ t có mặt trong quan hệ R, ngược lại hàm nhận giá trị sai (false).

8. test (t, b).

Đây cũng là một hàm logic nhận giá trị đúng nếu bộ t thỏa mãn biểu thức logic b, ngược lại hàm nhận giá trị sai.

Ký hiệu A, B là các thuộc tính, C là một giá trị hằng, θ là một phép so sánh:

$$\theta \in \{ =, \neq, \leqslant, <, >, \geqslant \}.$$

Các dãy ký hiệu dạng sau đây được gọi là các biểu thức đơn:

(i) A θ B,

(ii) A θ C.

Biểu thức được xây dựng từ các biểu thức đơn và các phép toán $\vee$ (OR), $\wedge$ (AND), $\neg$ (NOT) theo kiến trúc đệ quy tương tự như việc kiến thiết các biểu thức trong các ngôn ngữ lập trình bậc cao.

9. *Định nghĩa.* Cho b là một biểu thức, t là một bộ chứa các giá trị ứng với các thuộc tính A_i ($i = 1, 2, \ldots, n$); cụ thể là $t = \langle t.A_1, t.A_2, \ldots, t.A_n \rangle$. Ta nói rằng bộ t thỏa mãn biểu thức b nếu sau khi thay mọi tên thuộc tính A_i trong b bằng giá trị $t.A_i$ của bộ t thì ta được một giá trị hằng đúng. Nếu giá trị nhận được là sai, ta nói rằng bộ t không thỏa mãn biểu thức b. Hàm test hoạt động theo nguyên tắc nói trên.

10. Sau cùng ta xét hàm card (R) xác định lực lượng của quan hệ R.

Các toán tử 1., 3. và 2., 7. đòi hỏi trao đổi với bộ nhớ ngoài. Các toán tử 4. và hàm 8. được thực hiện hoàn toàn trên bộ nhớ trung tâm. Toán tử 5. và hàm 10. đòi hỏi cùng lắm là một lần trao đổi với bộ nhớ ngoài. Thật vậy, khi cần đưa thêm một bộ vào quan hệ cho trước người ta thường đưa bộ đó vào vùng đệm trên bộ nhớ trung tâm. Chỉ khi nào vùng đệm đầy người ta mới đổ nó vào bộ nhớ ngoài. Với một cài đặt tốt (như dùng chỉ dẫn chẳng hạn), ta có thể tăng sức chứa của vùng đệm, thay vì dùng nó để lưu trữ nội dung của một bộ ta chỉ cần lưu trữ một phần tử của chỉ dẫn tới bộ đó.

Lực lượng của một quan hệ thường được lưu trữ cùng với quan hệ đó. Khi gọi tới một quan hệ thì giá trị của lực lượng được đọc lên miền nhớ trung tâm, do đó hàm card chỉ đòi hỏi một lần trao đổi ở giai đoạn đầu tiên khi gọi tới quan hệ.

Nhờ tổ chức chỉ dẫn, toán tử đánh nhãn 6. cũng không đòi hỏi thường xuyên phải trao đổi với bộ nhớ ngoài. Lưu ý rằng trong nhiều trường hợp chúng ta không thể nạp toàn thể các bộ của một quan hệ lên bộ nhớ trung tâm nếu như chiều dài của mỗi bộ quá lớn, nhưng chỉ dẫn cho quan hệ đó vẫn có thể đưa hết lên bộ nhớ trung tâm.

III – CÁC PHÉP TOÁN QUAN HỆ

Các thuật toán ở mục này được diễn đạt qua ngôn ngữ tựa ALGOL.

1. Phép chiếu.

1) Định nghĩa. Cho quan hệ R và Y là một nhóm thuộc tính của R. Phép chiếu quan hệ R trên nhóm thuộc tính Y sẽ cho quan hệ kết quả $T = R[Y]$ xác định như sau: trước hết từ R loại đi những thuộc tính không thuộc Y, sau đó trong số các bộ giống nhau chỉ giữ lại một bộ.

2) Thuật toán $T = R[Y]$; $Y \subseteq \mathcal{A}(R)$

```
for each r ∈ R with (r.Y not in T) do
    add r.Y to T.
```

3) Định nghĩa (phép chiếu trên tập bù). Với nhóm thuộc tính Y của quan hệ R ta xác định $\bar{Y} = \mathcal{A}(R) \setminus Y$ và gọi $\bar{Y}$ là phần bù của Y trong quan hệ R. Một dạng khác của phép chiếu gọi là phép chiếu trên tập bù được định nghĩa như sau:

$$R / Y = R[\bar{Y}].$$

4) Thuật toán: $T = R / Y$; $\bar{Y} \subseteq \mathcal{A}(R)$

```
for each r ∈ R with (r.Ȳ not in T) do
    add r.Ȳ to T.
```

Phép chiếu bù tiện lợi trong trường hợp chúng ta muốn loại bỏ đi một vài thuộc tính của quan hệ R và số thuộc tính được giữ lại nhiều hơn số thuộc tính loại bỏ.

2. Phép chọn (mở rộng).

1) Định nghĩa. Cho b là một biểu thức logic trên tập các thuộc tính của quan hệ R Khi đó phép chọn các bộ của R thỏa mãn điều kiện b cho ta một quan hệ $T = R(b)$ xác định như sau:

$$T = \{r \in R : r \text{ thỏa mãn } b\}$$

2) Thuật toán $T = R(b)$

```
for each r ∈ R with test (r, b) do
        add r to T.
```

3. Tích Descartes.

1) Nếu r là một bộ của quan hệ R và s là một bộ của quan hệ S ta kí hiệu $\langle r, s \rangle$ là một hàm thỏa mãn:

$$\langle r, s \rangle|_{\mathcal{A}(R)} = r \text{ và } \langle r, s \rangle|_{\mathcal{A}(S)} = S.$$

Trong thực tiễn cài đặt, $\langle r, s \rangle$ phụ thuộc vào trật tự cụ thể của các thuộc tính trong các quan hệ R và S.

Thí dụ, nếu $r = \langle r_1, r_2, r_2, \ldots, r_n \rangle \in R$ và

$$s = \langle s_1, s_2, \ldots, s_m \rangle \in S \text{ thì}$$

$$\langle r, s \rangle = \langle r_1, r_2, \ldots, r_n, s_1, s_2, \ldots, s_m \rangle$$

Toán tử construct (4.) sẽ đảm bảo trật tự của các thuộc tính trong từng trường hợp cụ thể.

Định nghĩa. Tích Descartes giữa hai quan hệ R và S là quan hệ T = R * S được xác định như sau:

$$T = \{ \langle r, s \rangle : r \in R \text{ và } s \in S \}.$$

2) Thuật toán T = R * S

```
for each r ∈ R do
    for each s ∈ S do
     begin construct t = ⟨ r, s ⟩:
      add  t  to  T
    end
```

4. Phép kết nối (mở rộng).

1) Định nghĩa. Cho hai quan hệ R và S và B là một nhóm thuộc tính của chúng: $B \subseteq \mathcal{A}(R) \cup \mathcal{A}(S)$, b là một biểu thức logic. Phép kết nối quan hệ R với quan hệ S theo điều kiện b, B cho ta quan hệ T = R.b, B.S được xác định như sau:

$$T = R * S(b)/B \qquad (1)$$

Theo trật tự từ trái sang phải, thoạt tiên lấy tích Descartes R * S, sau đó thực hiện phép chọn trên kết quả này và cuối cùng (nếu cần) loại bỏ đi nhóm thuộc tính B. Như vậy, phép kết nối còn có hai trường hợp riêng là T = R b.S = R * S(b) và T = R..S = R * S. Trường hợp riêng thứ hai (R..S) tương đương với trường hợp b là một biểu thức hằng đúng.

Lưu ý rằng biểu thức (1) chỉ có ý nghĩa mô tả các bộ của quan hệ kết quả T, việc cài đặt thuật toán cho T được thực hiện như sau:

2) Thuật toán T = R.b, B.S

```
for each r ∈ R do
    for each s ∈ S do
     begin construct t = ⟨r, s⟩;
                  if (test (t, b)) AND (t.B̄ not in T)
                  then add t.B̄ to T
     end
```

5. Phép hợp.

1) Định nghĩa. Cho hai quan hệ khả hợp R và S, phép hợp R và S sẽ cho quan hệ T = R ∪ S xác định như sau:

$$T = \{ t : t \in R \text{ hoặc } t \in S \}.$$

2) Thuật toán T = R ∪ S

```
for each r ∈ R do
     add r to T;
for each s ∈ S do
if s in R then label s in R
                  else add s to T
```

6. Phép giao.

1) Định nghĩa. Giao của hai quan hệ khả hợp R và S cho ta quan hệ T = R ∩ S được xác định như sau:

$$T = \{ t : t \in R \text{ và } t \in S \}.$$

2) Thuật toán T = R ∩ S

```
for each r ∈ R do
 if r in S then begin
 add r to T;
            label r in S
              end
```

Cho b_1 và b_2 là hai biểu thức logic trên R, dễ dàng kiểm tra các hằng đẳng thức sau:

$R(b_1)(b_2) = R(b_1 \bigwedge b_2) = R(b_1) \cap R(b_2)$

$R(b_1 \bigvee b_2) = R(b_1) \cup R(b_2)$.

Từ đó thấy rằng việc mở rộng phép chọn sẽ tiết kiệm được các phép giao và hợp trung gian.

7. Phép trừ.

1) Định nghĩa. Hiệu của hai quan hệ khả hợp R và S là quan hệ $T = R \setminus S$ được xác định như sau:

$$T = \{t : t \in R \text{ và } t \overline{\in} S\}.$$

2) Thuật toán $T = R \setminus S$.

for each $r \in R$ do
if r in S then label r in S
else add r to T

8. Phép chia. Cho quan hệ R và B là một nhóm thuộc tính của R. Với mỗi $y \in R/B$ ta định nghĩa tập

$$y_B = \{x : x \in R[B] \text{ và } \langle y, x \rangle \in R\}$$

và gọi y_B là tập ảnh của y trên B.

1) Định nghĩa. Cho quan hệ R và nhóm thuộc tính $B \subseteq \mathcal{A}(R)$ và quan hệ S sao cho R[B] và S là khả hợp. Phép chia quan hệ R cho quan hệ S theo nhóm B cho ta quan hệ $T = R:B:S$ được xác định như sau:

$$T = \{y : y \in R/B \text{ và } y_B \supseteq S\}.$$

Nói cách khác, T gồm những phần tử của R/B có tập ảnh trên B chứa S.

2) Thuật toán $T = R:B:S$

```
n = card (S);
for each r ∈ R do
    begin k = 0; construct t = ⟨ r.B̄ ⟩;
   for each q ∈ R with q.B̄ = t do
   begin
   if q.B in S then k = k + 1;
   label q in R
   end;
if k = n then add t to T
end
```

9. Tóm tắt.

Trong [1] không trình bày phép chia, các phép toán chọn và kết nối được trình bầy với các biểu thức đơn.

Những tác giả không cài đặt phép chia thường giả thiết rằng phép toán này có thể được biểu diễn qua các phép chiếu, phép trừ và tích Descartes như sau:

$$R:B:S = R[\overline{B}] \setminus ((R[\overline{B}] * S) \setminus R)[\overline{B}]$$

hoặc dùng phép chiếu trên tập bù thay cho $[\overline{B}]$ ta có:

$$R:B:S = R/B \setminus ((R/B * S) \setminus R)/B.$$

IV – ĐÁNH GIÁ MỘT SỐ THUẬT TOÁN

1. Với các qnan hệ R và S cho trước ta đặt:

$m = \text{card}(R)$,
$n = \text{card}(S)$,
$p = \text{card}(R \cap S)$.

Giả thiết rằng nếu quan hệ R có m bộ thì việc xác định xem một bộ t cho trước có mặt hay không trong R đòi hỏi f(m) phép so sánh bộ t với các bộ của R. Ở đây ta cũng chấp nhận thêm rằng việc tìm kiếm cả bộ t hay một phần t' của nó trong R đều đòi hỏi f(m) phép so sánh, có nghĩa là hàm f(m) không phụ thuộc vào chiều dài của bộ trong R.

Nói chung f(m) là một hàm phụ thuộc vào m và tổ chức sắp xếp các bộ trong R. Đương nhiên, với mỗi tổ chức sắp xếp cho trước f(m) là một hàm đồng biến [4].

Trong các thuật toán 5, 6 và 7 ở mục III ta thấy nếu $r \in R$ thì việc kiểm tra (r in S) có thể cần f(q) phép so sánh với $q \leqslant n$. Nếu trước đó trong S đã có một số bộ được đánh nhãn thì $q < n$. Ta xét trường hợp xấu nhất là trường hợp p bộ của $R \cap S$ sắp xếp ở cuối của quan hệ R. Theo giả thiết này việc kiểm tra điều (r in S) cho m-p bộ đầu tiên của quan hệ R đòi hỏi f(n) phép so sánh cho mỗi bộ, tức là đòi hỏi cả thảy (m-p).f(n) phép so sánh. Với p bộ cuối của R, vì chúng cũng có mặt trong quan hệ S nên sau mỗi bước chúng ta đánh nhãn được một bộ trong S và do đó số các phép so sánh cần thiết cho p bộ này là $\sum_{i=1}^{p} f(n-i+1)$.

Tổng cộng lại, trong trường hợp xấu nhất, các thuật toán 5, 6 và 7. đòi hỏi

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

phép so sánh.

Nếu ta bỏ các toán tử đánh nhãn trong các thuật toán 5, 6 và 7 thì sẽ nhận được các thuật toán tương ứng cho các phép hợp, giao và hiệu hai quan hệ nêu trong [1]. Vậy các thuật toán tương ứng trong [1] đòi hỏi m.f(n) phép so sánh,

Tổng kết lại ta có:

Đánh giá 1.

a) Trong trường hợp xấu nhất, các thuật toán 3.5, 3.6 và 3.7 đòi hỏi $(m-p).f(n) + \sum_{i=1}^{p}(n-i+1)$ phép so sánh $(p \geqslant 1)$.

b) Các thuật toán tương ứng trong [1] đòi hỏi m.f(n) phép so sánh.

c) So với các thuật toán tương ứng trong [1], việc dùng toán tử đánh nhãn trong các thuật toán 3.5, 3.6. và 3. 7 tiết kiệm được

$$mf(n) - [(m-p)f(n) + \sum_{i=1}^{p} f(n-i+1)] = pf(n) - \sum_{i=1}^{p} f(n-i+1) = \sum_{i=1}^{p} [f(n) - f(n-i+1)]$$

phép so sánh $(p \geqslant 1)$.

Trong trường hợp $p = 0$, tức là các quan hệ R và S không có phần tử chung thì các biểu thức của a và b là như nhau, và do đó biểu thức của c bằng 0.

2. Xét thuật toán chia 8. 2. Với nhóm thuộc tính B cho trước ta thực hiện một phân hoạch R theo quan hệ nhị nguyên ρ sau đây:

$$\forall r, r' \in R : r \rho r' \overset{def}{\Leftrightarrow} r.\overline{B} = r'.\overline{B}.$$

Dễ thấy ρ là quan hệ tương đương. Gọi v là số lớp tương đương trong phân hoạch R/ρ. Số bộ trong lớp k (k = 1, 2,..., v) được ký hiệu là m_k. Ta có :

$$m = card(R) = \sum_{k=1}^{v} m_k.$$

Trong thuật toán 3. 8. 2 ta thấy toán tử đánh nhãn được đặt cho mỗi bộ của quan hệ R. Do đó sau mỗi bước ở vòng for trong, số bộ của R được giảm đi 1. Mặt khác, mỗi vòng for ngoài quét hết một lớp tương đương. Không làm giảm tổng quát, ta coi vòng for ngoài thứ k sẽ quét hết lớp tương đương thứ k và do đó số bộ của R được giảm bớt m_k phần tử so với vòng for ngoài trước đó.

Với n = card (S) ta thấy toán tử if q.B in S đòi hỏi trung bình f(n) lần so sánh cho mỗi $q \in R$, do đó m bộ của R sẽ đòi hỏi m.f(n) phép so sánh trong S cho toán tử if q.B in S và $\sum_{i=1}^{m} f(m-i+1)$ phép so sánh trong bản thân R cho toán tử for each $q \in R$ with $q.\overline{B} = t$.

Ngoài ra, ta để ý rằng sau mỗi vòng for ngoài, với phần tử r ở đầu lớp tiếp theo thì việc xác định $q \in R$ with $q.\overline{B} = t$ thực chất là việc chọn ngay q = r, do đó các phần tử ở đầu mỗi lớp tương đương sẽ không đòi hỏi phép so sánh nào cho vòng for trong đầu tiên của lớp đó. Đặt $m_0 = 0$, ta thấy sau lớp thứ k − 1 thì số bộ của R được giảm đi $\sum_{j=0}^{k-1} m_j$ phần tử (k = 1, 2,..., v), và do đó R còn $m - \sum_{j=0}^{k-1} m_j$ bộ ở lúc bắt đầu xét lớp thứ k, do đó nếu không chọn q = r thì sẽ cần $f(m - \sum_{j=0}^{k-1} m_j)$ phép so sánh cho phần tử đầu tiên của lớp thứ k. Tổng cộng cho các phần tử đầu tiên của v lớp ta có: $\sum_{k=1}^{v} f(m - \sum_{j=0}^{k-1} m_j)$. Trừ giá trị này vào $\sum_{i=1}^{m} f(m-i+1)$ ta được tổng số các phép so sánh trong bản thân R cho toán tử for each $q \in R$ with $q.\overline{B} = t$. Tổng kết lại ta có:

Đánh giá 2. Thuật toán chia 3. 8. 2 đòi hỏi trung bình

$$\sum_{i=1}^{m} f(m-i+1) - \sum_{k=1}^{v} f(m - \sum_{j=0}^{k-1} m_j) + m.f(n) \qquad (2)$$

phép so sánh, trong đó m = card (R), n = card (S), v là số lớp tương đương, m_j là số bộ trong lớp j (j = 0,1,.., v) với $m_0 = 0$.

Để ý rằng $\sum_{i=1}^{m} f\, m-i+1) = \sum_{i=1}^{m} f(i)$ ta viết lại (2) dưới dạng sau:.

$$\sum_{i=1}^{m} f(i) + m.f(n) - \sum_{k=1}^{v} f(m - \sum_{j=0}^{k-1} m_j).$$

Cuối cùng người viết xin chân thành cám ơn phó giáo sư Hồ Thuần, Viện Khoa học tính toán và điều khiển, đã đọc và góp ý kiến tỉ mỉ cho bài này, đặc biệt là cách diễn đạt phần chứng minh cho đánh giá 2.

Nhận ngày 25-9-1984

TÀI LIỆU THAM KHẢO

1. L.L. **Beck**, A Generalized Implementation Method for Relational Data Sublanguages. IEEE Transactions on software engineering, vol. Se − 6, N° 2, March 1980, pp. 152 − 162.

2. **E.F. Codd**, A Relationnal Model of Data for Large Shared Data Banks. Commun. Assoc. Comput. Mach., vol. 13, June 1970, pp. 377 − 397.

3. **E.F. Codd**, Relational Completeness of Data Base Sublanguages. In Courant Computer Science Symposia, vol. 6. Data Bata Systems. Englewood Cliffs, NJ: Prentice − Hall, 1971, pp. 65 − 99.

4. **D.E. Knuth**, The Art of computer Programming, vol. 3. Reading, Mass: Addison − Wesley, 1973.

ABSTRACT

A labelling operator and relational operattons

A set of algorithms of relational operations in the realization of DBMS is described. A quantitative efficiency of these algorithms with using of labelling operator is derived.