Tạp chí Tin học và Điều khiển học, T. 15, S. 4 (1999), 43–49

TỐI ƯU HÓA CÂY NHỊ PHÂN MỘT CHIỀU VỚI THÔNG TIN CHỨA Ở CÁC ĐỈNH TRONG

ĐỖ ĐỨC GIÁO, LÊ ANH CƯỜNG

Abstract. The notion of a search tree plays an important role in computer science, especially in the theory of data structures. For that reason we can find many papers concerned with the theory of search trees in literature. After having read these papers we noticed that, above all, questions of the optimal construction and inductive generation of search trees are studied, where equivalent transformation of search trees are often used.

In this paper we intend to further develop the conception of search trees in [4]. A prove of a theorem which shows that each binary search tree can be uniquely transformed into optimal binary search tree by using axiom schemes and the rules are introduced in this paper.

1. MỞ ĐẦU

Trong [4] Thiele đã đưa ra khái niệm về cây nhị phân với thông tin chứa ở các đỉnh trong, đồng thời đưa ra hệ tiên đề gồm ba phương trình cây và chứng minh tính phi mâu thuẫn, tính đầy đủ của hệ tiên đề đó.

Trong bài này, chúng tôi sử dụng tính phi mâu thuẫn và tính đầy đủ của hệ tiên đề do Thiele đưa ra để xây dựng thuật toán tìm cây tối ưu theo cấu trúc và theo thời gian trên lớp cây nhị phân được đưa ra trong [4] mà Thiele chưa đề cập đến. Thuật toán mà chúng tôi xây dựng giúp cho việc phân loại, tìm kiếm thông tin dưới dạng cây nhị phân một cách tối ưu nhất trên cơ sở tập các khóa (keys) là tập bất kỳ $\neq \emptyset$ các phần tử mà trên đó thỏa mãn quan hệ so sánh.

2. SỰ TƯƠNG ĐƯƠNG CỦA CÂY NHỊ PHÂN

2.1. Định nghĩa cây nhị phân

Giả sử I là tập không rỗng các phần tử, gọi là tập thông tin và K là tập khóa các phần tử mà trên nó thỏa mãn quan hệ so sánh, tức là $x, y \in K$ thì hoặc $x < y$, hoặc $x = y$ hoặc $x > y$.

Kí hiệu τ là cây rỗng. Đặt $I^+ = I \cup \{\tau\}$.

Định nghĩa 1.

a. τ là một cây.

b. Nếu T_1, T_2 là hai cây thì dãy kí hiệu $[k, i](T_1, T_2)$ với $k \in K$ và $i \in I$ cũng là một cây.

Tập tất cả các cây được định nghĩa như trên kí hiệu là TREE và gọi là tập tất cả các cây nhị phân một chiều với thông tin chứa ở các đỉnh trong, gọi tắt là tập tất cả các cây nhị phân.

Định nghĩa 2. Lấy $T \in$ TREE và $l \in K$ với $T \equiv [k, i](T_1, T_2)$. Định nghĩa hàm RESULT: TREE $\times K \rightarrow I^+$ như sau:

a. $\mathrm{RESULT}(\tau, l) = \tau$.

b. $$\mathrm{RESULT}([k, i](T_1, T_2), l) = \begin{cases} \mathrm{RESULT}(T_1, l) \text{ nếu } l < k \\ i \text{ nếu } l = k \\ \mathrm{RESULT}(T_2, l) \text{ nếu } l > k \end{cases}$$

Định nghĩa 3. Ta nói cây T_1 tương đương cây T_2 trên tập khóa K, và kí hiệu $T_1 \approx(K)\, T_2$ khi và chỉ khi:

$$\mathrm{RESULT}(T_1, l) = \mathrm{RESULT}(T_2, l) \text{ với } \forall l \in K.$$

Coi như lúc nào ta cũng xét trên tập K nên để đơn giản ta thay $T_1 \approx (K)\, T_2$ bởi $T_1 \approx T_2$.

Dưới đây ta kí hiệu $T_1 \equiv T_2$ và $T_1 \not\equiv T_2$ chỉ cây T_1 đồng nhất bằng cây T_2 và không đồng nhất bằng cây T_2.

2.2. Hệ tiên đề và quy tắc dẫn xuất

Giả sử $T_1, T_2 \in$ TREE. Ta kí hiệu $T_1 = T_2$ là một phương trình cây. Tập tất cả các phương trình cây tương ứng với tập TREE ta kí hiệu qua EQU $= \{T_1 = T_2 \mid T_1, T_2 \in \text{TREE}\}$.

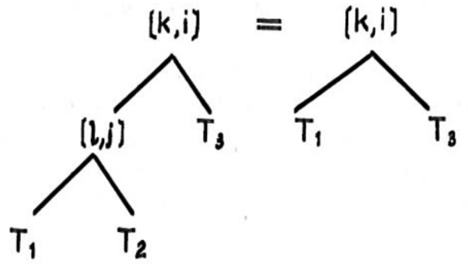
2.2.1. Quy tắc dẫn xuất

Phương trình cây $T_1 = T_2$ là dẫn xuất được từ tập X (kí hiệu $X \vdash T_1 = T_2$) khi và chỉ khi $T_1 = T_2$ là phần tử trong X hoặc $T_1 = T_2$ dẫn được từ các phần tử trong X qua việc áp dụng một số hữu hạn lần các quy tắc sau đây:

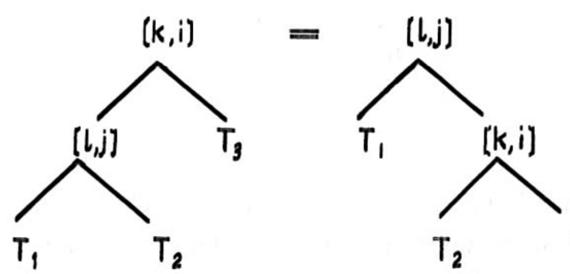
Quy tắc 1 (R_1): Nếu $T \in$ TREE thì $X \vdash T = T$.
Quy tắc 2 (R_2): Nếu $X \vdash T_1 = T_2$ thì $X \vdash T_2 = T_1$.
Quy tắc 3 (R_3): Nếu $X \vdash T_1 = T_2$ và $X \vdash T_2 = T_3$ thì $X \vdash T_1 = T_3$.
Quy tắc 4 (R_4): Nếu $X \vdash T_1 = T_2$ thì $X \vdash [k,i](T_1, T) = [k,i](T_2, T)$.
Quy tắc 5 (R_5): Nếu $X \vdash T_1 = T_2$ thì $X \vdash [k,i](T, T_1) = [k,i](T, T_2)$.

2.2.2. Hệ tiên đề

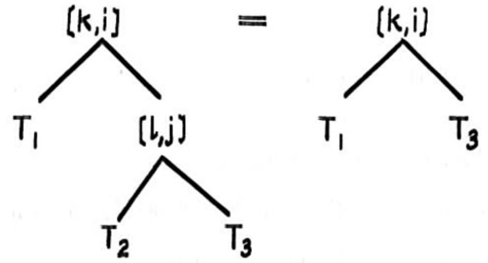
Tiên đề 1 (ax_1):
$[k,i]([l,j](T_1, T_2), T_3) = [k,i](T_1, T_3)$ với $k \le l$ là một tiên đề.



Tiên đề 2 (ax_2):
$[k,i]([l,j](T_1, T_2), T_3) = [l,j](T_1, [k,i](T_2, T_3))$ với $k > l$ là một tiên đề.



Tiên đề 3 (ax_3):
$[k,i](T_1, [l,j](T_2, T_3)) = [k,i](T_1, T_3)$ với $k \ge l$ là một tiên đề.



Đặt $AX = ax_1 \cup ax_2 \cup ax_3$ và gọi là hệ tiên đề TREE hay hệ tiên đề AX.

Định lý 1. *Giả sử $T_1, T_2 \in$ TREE.*

a. Nếu $AX \vdash T_1 = T_2$ thì $T_1 \approx T_2$.

b. Nếu $T_1 \approx T_2$ thì $AX \vdash T_1 = T_2$.

Chứng minh.

a. Chứng minh bằng quy nạp theo chiều dài dẫn xuất từ AX, áp dụng các quy tắc dẫn xuất và hệ tiên đề.

b. Sẽ chứng minh sau Định lý 3.

2.3. Cây chuẩn tắc

Định nghĩa 4. $N \in$ TREE, N được gọi là cây chuẩn tắc nếu N là một trong hai dạng sau:

1. $N \equiv \tau$.

2. $N \not\equiv \tau$ thì $N \equiv [k_1, i_1](\tau, [k_2, i_2](\ldots, [k_n, i_n](\tau, \tau) \ldots))$ với $k_i \in K$, $i_i \in I$ $(i = 1, 2, \ldots, n)$ và $k_1 < k_2 < \ldots < k_n$.

Định lý 2. *Giả sử N_1 và N_2 là 2 cây chuẩn tắc, nếu $N_1 \approx N_2$ thì $N_1 \equiv N_2$.*

Chứng minh. Ta phân ra các trường hợp sau:

a. $N_1 \equiv \tau$ và $N_2 \equiv \tau$. Định lý hiển nhiên đúng.

b. $N_1 \equiv \tau$ và $N_2 \equiv [k_1, i_1](\tau, [k_2, i_2](\ldots, [k_n, i_n](\tau, \tau) \ldots))$ với $n > 0$, $k_1 < k_2 < \ldots < k_n$. Do $N_1 \not\equiv N_2$ nên $N_1 \not\approx N_2$ (dùng phương pháp phản chứng).

c. $N_1 \equiv [k_1, i_1](\tau, [k_2, i_2](\ldots, [k_n, i_n](\tau, \tau) \ldots))$ với $n > 0$, $k_1 < k_2 < \ldots < k_n$.
$N_2 \equiv [k'_1, i'_1](\tau, [k'_2, i'_2](\ldots, [k'_m, i'_m](\tau, \tau) \ldots))$ với $m > 0$, $k'_1 < k'_2 < \ldots < k'_m$.
Không mất tổng quát ta giả sử $n \le m$. Chứng minh theo quy nạp ta có $k'_i = k_i$, với $i = 1, \ldots, n$. Từ đó chứng minh $n = m$ và suy ra N_1 trùng với N_2.

Định lý 3. *Với mọi $T \in$ TREE, tồn tại duy nhất cây chuẩn tắc N sao cho:*

a. $T \approx N$.

b. $AX \vdash T = N$.

Chứng minh.

a. Câu *a* được suy ra từ câu *b* nhờ áp dụng Định lý 1.

b. Ta chứng minh câu *b* bằng quy nạp theo định nghĩa của cây T. Ta phân ra các trường hợp sau:

1. $T \equiv \tau$. Chọn $N \equiv \tau$.

2. $T \equiv [k, i](T_1, T_2)$. Theo giả thiết quy nạp thì tồn tại N_i là dạng chuẩn của T_i $(i = 1, 2)$. Xét 4 khả năng xảy ra:

2.1. $N_1 \equiv \tau$, $N_2 \equiv \tau$. Chọn $N \equiv [k, i](\tau, \tau)$.
2.2. $N_1 \equiv \tau$, $N_2 \equiv [k_1, i_1](\tau, [k_2, i_2](\ldots, [k_n, i_n](\tau, \tau) \ldots))$ với $n > 0, k_1 < k_2 < \ldots < k_n$.
2.3. $N_2 \equiv \tau$, $N_1 \equiv [k_1, i_1](\tau, [k_2, i_2](\ldots, [k_n, i_n](\tau, \tau) \ldots))$ với $n > 0, k_1 < k_2 < \ldots < k_n$.
2.4. $N_1 \equiv [k_1, i_1](\tau, [k_2, i_2](\ldots, [k_n, i_n](\tau, \tau) \ldots))$ với $n > 0, k_1 < k_2 < \ldots < k_n$,
$N_2 \equiv [f_1, e_1](\tau, [f_2, e_2](\ldots, [f_m, e_m](\tau, \tau) \ldots))$ với $m > 0$, $f_1 < f_2 < \ldots < f_n$.

Các trường hợp trên áp dụng hệ tiên đề AX và quy tắc dẫn xuất R_3 sẽ xây dựng được cây chuẩn N cho từng trường hợp.

Chứng minh tính duy nhất của N: Giả sử tồn tại hai cây chuẩn tắc khác nhau N_1 và N_2 sao cho $AX \vdash T = N_1$, $AX \vdash T = N_2$, áp dụng R_2, R_3 suy ra $N_1 \approx N_2$. Theo Định lý 2 suy ra N_1 trùng N_2, mâu thuẫn. Vậy N là duy nhất.

Chứng minh Định lý 1.b. Áp dụng Định lý 2 và Định lý 3 ta suy ra Định lý 1.b. Thật vậy: Theo Định lý 3 tồn tại duy nhất cây chuẩn tắc $N_1 \approx T_1$, $AX \vdash T_1 = N_1$ và cây chuẩn tắc $N_2 \approx T_2$, $AX \vdash T_2 = N_2$. Vì $T_1 \approx T_2$, áp dụng Định lý 2 suy ra N_1 trùng N_2 gọi chung là N.
Vậy $AX \vdash T_1 = T_2$. Định lý 1.b được chứng minh.

2.4. Bảng mã của cây

Định nghĩa 5. $T \in$ TREE, bảng mã của T là tập hữu hạn tất cả các khóa khác nhau $k \in K$ sao cho RESULT$(T, k) \neq r$. Ta kí hiệu bảng mã của T là $m(T)$. $m(T)$ còn được gọi là tập bảng mã cốt yếu của cây T. Nếu $T \equiv r$ thì $m(T) \equiv \emptyset$.

Sau đây là một số bổ đề đơn giản mà ta sẽ không chứng minh.

Bổ đề 1. *$T \in$ TREE với N là cây chuẩn tắc và $N \approx T$.
Nếu $N \equiv [k_1, i_1](r, [k_2, i_2](\ldots, [k_n, i_n](r, r) \ldots))$ với $n > 0$, $k_1 < k_2 < \ldots < k_n$ thì: $m(T) = \{k_1, k_2, \ldots, k_n\}$ (tập tất cả các khóa ở các đỉnh trong của N).*

Bổ đề 2. *Với $T_1, T_2 \in$ TREE và $T_1 \approx T_2$ thì $m(T_1) = m(T_2)$.*

3. CÂY NHỊ PHÂN TỐI ƯU

3.1. Cây hoàn chỉnh

$T \in$ TREE, ta có các khái niệm sau:

$\gamma(T)$ là số tất cả các đỉnh và lá của cây T. Mức của đỉnh là 0. Nếu đỉnh cha có mức là n thì đỉnh con kế tiếp có mức là $n+1$.

Với r là một lá (cây rỗng) nào đó của T. Kí hiệu Deep (r) là số các cung của đường đi từ gốc của T tới r. Kí hiệu $h(T)$ là chiều cao của cây T và ta định nghĩa $h(T)$ = mức lớn nhất của cây T.

Bổ đề 3. *$T, N \in$ TREE với N là cây chuẩn tắc và $T \approx N$ thì $\gamma(N) \leq \gamma(T)$.*

Định nghĩa 5. Giả sử $B \in$ TREE. Ta nói B là cây hoàn chỉnh nếu B là một trong hai dạng sau:

a. $B \equiv r$.

b. Nếu $B \not\equiv r$ thì B thỏa mãn hai điều kiện:

b1. Khóa của một đỉnh bất kì trong B nhỏ hơn tất cả các khóa ở cây con bên phải nó và lớn hơn tất cả các khóa ở cây con bên trái nó.

b2. Với r_1, r_2 là hai lá bất kì của B thì $|\text{Deep}(r_1) - \text{Deep}(r_2)| \leq 1$.

Định lý 4. *Với mỗi cây $T \in$ TREE tồn tại cây hoàn chỉnh B sao cho:*

a. $T \approx B$.

b. $AX \vdash T = B$.

Chứng minh.

a. $T \approx B$ được suy ra từ b nhờ Định lý 1.

b. Theo Định lý 3 tồn tại duy nhất cây chuẩn tắc N sao cho $AX \vdash T = N$. Từ cây chuẩn N ta nhận được cây hoàn chỉnh B bằng cách áp dụng một số hữu hạn lần ax_2, tức là thực hiện phép bẻ đôi N và các cây con của nó sau mỗi lần áp dụng tiên đề ax_2. Dựa vào tính chất cây hoàn chỉnh ta có: với B là cây hoàn chỉnh thì $m(B)$ chính là tập tất cả các đỉnh trong của B.

Bổ đề 4. *$T, B \in$ TREE với B là cây hoàn chỉnh và $T \approx B$ thì $\gamma(B) \leq \gamma(T)$.*

Chứng minh. Từ cách chứng minh của Định lý 4 thì B nhận được từ cây chuẩn $N \approx T$ bằng cách áp dụng tiên đề ax_2, mà tiên đề ax_2 không làm thay đổi số các đỉnh của cây, hay $\gamma(B) = \gamma(N)$. Do Bổ đề 3 thì $\gamma(B) \leq \gamma(T)$.

3.2. Cây nhị phân tối ưu theo cấu trúc

Định nghĩa 6. Giả sử $T_0 \in$TREE. T_0 gọi là tối ưu theo cấu trúc nếu T_0 thỏa mãn các điều kiện sau:

1. $\gamma(T_0) = \min\{\gamma(T) \mid T \in \text{TREE và } T \approx T_0\}$.
2. $h(T_0) = \min\{h(T) \mid T \in \text{TREE và } T \approx T_0\}$.
3. Khóa của đỉnh cha bất kì trong T_0 nhỏ hơn khóa của tất cả các đỉnh ở cây con bên phải và lớn hơn khóa của tất cả các đỉnh ở cây con bên trái.

Như vậy cây tối ưu theo cấu trúc sẽ có số đỉnh và độ cao bé nhất so với tất cả các cây tương đương với nó, đồng thời cây tối ưu theo cấu trúc chỉ chứa thông tin cốt yếu.

Định lý 5. *Nếu $B \in TREE$ là cây hoàn chỉnh thì B là cây tối ưu theo cấu trúc.*

Chứng minh. Từ Bổ đề 4 ta suy ra $\gamma(B) = \min\{\gamma(T) \mid T \in \text{TREE và } T \approx B\}$. (1)
Mặt khác vì số đỉnh của $B \leq$ số đỉnh của T (theo định nghĩa) và do tính chất của cây hoàn chỉnh thì tại mỗi mức của B số đỉnh sẽ $\geq$ số đỉnh tại mức tương ứng của T, chiều cao của B sẽ $\leq$ chiều cao của T. (2)
$h(B) = \min\{h(T) \mid T \in \text{TREE và } T \approx B\}$. Vì B là cây hoàn chỉnh nên B có tính chất b.1 của Định nghĩa 5. (3).

Kết hợp (1), (2), (3) ta kết luận B là cây tối ưu theo cấu trúc.

3.3. Cây nhị phân tối ưu theo thời gian

Nếu cho khóa $k \in K$ vào cây $T \in$TREE thì hàm RESULT(T, k) xác định duy nhất một đường đi từ gốc đến kết quả tương ứng (nút chứa k hoặc cây rỗng). Số các cung của đường đi đó được gọi là thời gian tìm kiếm ứng với khóa k, kí hiệu là $t(T, k)$.

Tuy nhiên do chỉ có các khóa thuộc $m(T)$ mới có đường đi đến nút chứa nó, nên ta chỉ xét khái niệm thời gian tìm kiếm trên các khóa thuộc $m(T)$. Thời gian tìm kiếm trung bình của một cây $T \in$ TREE, kí hiệu $W(T)$ được định nghĩa là:

Định nghĩa 7.

1. Nếu $T \equiv r$ thì $W(T) = 0$.
2. Nếu $T \not\equiv r$ thì $W(T) = \sum\limits_{k \in m(T)} t(T, k)/|m(T)|$.

Định nghĩa 8. Giả sử $T_0 \in$TREE. T_0 được gọi là cây tối ưu theo thời gian nếu nó thỏa mãn: $W(T_0) = \min\{W(T) \mid T \in \text{TREE và } T \approx T_0\}$.

Bổ đề 5. *Giả sử B_1, B_2 là hai cây hoàn chỉnh. Nếu $B_1 \approx B_2$ thì $W(B_1) = W(B_2)$.*

Chứng minh. Theo Định lý 5, ta thấy $h(B_1) = h(B_2)$ và $\gamma(B_1) = \gamma(B_2)$. Từ đấy kết hợp với tính chất cây hoàn chỉnh suy ra mỗi mức tương ứng của B_1 và B_2 có số đỉnh là như nhau. Bổ đề được chứng minh.

Định lý 6. *Cây hoàn chỉnh B là cây tối ưu theo thời gian.*

Áp dụng các định lý 3, 4, 5 và các bổ đề để chứng minh Định lý 6.

Cây hoàn chỉnh là cây tối ưu theo cấu trúc và tối ưu theo thời gian, gọi chung là cây tối ưu. Theo Định lý 4, với cây $T \in$ TREE bất kì bao giờ cũng tìm được cây tối ưu tương đương với nó.

4. MỘT SỐ THỦ TỤC THỰC HIỆN TÌM CÂY NHỊ PHÂN TỐI ƯU

Mục đích của chúng ta là từ cây nhị phân bất kì ban đầu ta đưa về cây nhị phân tối ưu tương đương với nó. Chúng ta sẽ thực hiện hai bước sau đây:

- Đưa cây bất kì về cây chuẩn tắc tương đương.
- Từ cây chuẩn tắc chuyển về cây hoàn chỉnh.

1.1. Cấu trúc cây

```
TreeNode=^Tree
Tree=record
        key: integer;
        info: string; {có thể một kiểu dữ kiện khác }
        left: TreeNode;
        right: TreeNode;
    end;
```

1.2. Các hàm xử lý hệ tiên đề

```
Procedure ax1(T: TREE)
begin
    if (tồn tại n^.key ≤ (n^.left)^.key)and(n ∈ T) then
        n^.left = (n^.left)^.left;
end;
Procedure ax2(T: TREE)
begin
    if (tồn tại n^.key > (n^.left)^.key)and(n ∈ T) then
begin
        n1 = n^.left;
        n^.left = (n^.left)^.right;
        n1^.right = n;
    end;
end;
Procedure ax3(T: TREE)
begin
    if (tồn tại n^.key ≥ (n^.right)^.key)and(n ∈ T) then
        n^.right = (n^.right)^.left;
end;
```

1.3. Chuyển về cây chuẩn tắc

input: $T \in$ TREE.

output: $N \in$ TREE là cây chuẩn tắc, $N \approx T$.

Thuật toán:

```
k = 0;
While k <> 0 do
begin
        áp dụng ax1 đối với T, nếu có thay đổi k = k + 1;
        áp dụng ax2 đối với T, nếu có thay đổi k = k + 1;
        áp dụng ax3 đối với T, nếu có thay đổi k = k + 1;
end;
N = T;
```

1.4. Chuyển về cây hoàn chỉnh

input: $n \in$ TREE là cây chuẩn tắc.

output: $B \in$ TREE là cây hoàn chỉnh $B \approx N$.

Thuật toán:

```
Proceduce Bedoi(T: TREE)
begin
    if h(T) > 2 then
        begin
                bedoi T;
            1. bedoi(T.right);
            2. bedoi(T.left);
        end;
end;
```

5. KẾT LUẬN

Cây nhị phân mà ta đang xét trong bài toán này là cây nhị phân một chiều. Ta có thể mở rộng thành cây nhị phân n chiều theo định nghĩa đệ quy sau đây:

a. Kí hiệu r là một cây (cây n chiều rỗng).

b. Nếu T_1, T_2 là hai cây n chiều khi đó ký hiệu $[p,k,i]$ (T_1, T_2) hay $[p,k,i]$ cũng là cây n chiều

$T_1 \quad T_2$

ở đây p $(1 \le p \le n)$ là chỉ số của khóa cần lựa chọn, $k \in K$ và $i \in I$. Tập tất cả các cây định nghĩa như trên ta kí hiệu qua TREE(n).

Đặt $K(n) = K \times K \times \ldots \times K$ (tích Đề Các n lần của tập K) $= \{(k_1, k_2, \ldots, k_p, \ldots, k_n) \mid k_i \in K,\ i = \overline{1,n}\}$.

Khi đó hàm kết quả RESULT: $\mathrm{TREE}(n) \times K(n) \to I^+$ được xác định như sau:

a. $\mathrm{RESULT}(r, l) = r\ \forall l = (k_1, k_2, \ldots, k_p, \ldots, k_n) \in K(n)$.

b. $$\mathrm{RESULT}([p,k,i](T_1,T_2), l) = \begin{cases} \mathrm{RESULT}(T_1, l) & \text{nếu } k < k_p \\ i & \text{nếu } k_p = k \\ \mathrm{RESULT}(T_2, l) & \text{nếu } k_p > k \end{cases}$$

Với các khái niệm mở rộng như trên thì khi cho $n = l$, ta thu lại toàn bộ kết quả trong [4] cũng như kết quả của bài bào này.

TÀI LIỆU THAM KHẢO

[1] Do Duc Giao and Tjoa A. M., Optimization for one dimension binary search trees with information in leafs, *HNU Journal of Science, Nat. Sci.* **2** (1995) 49-61.

[2] Đỗ đức Giáo và Phạm Trung Kiên, Thuật toán về sự tương đương của cây tam nguyên một chiều với các thông tin chứa ở lá, *VNU. Journal of Science, Nat. Sci, Proc. of Conference on Information Technology* (1998) 39-47.

[3] Đỗ Đức Giáo và Phạm Trung Kiên, Thuật toán tìm cây tam nguyên tối ưu một chiều với các thông tin chứa ở lá, *VNU. Journal of Science, Nat. Sci. Proc. of Conference on Information Technology* (1998) 62-72.

[4] Helmut Thiele, On equivalent transformation of one dimensional binary search trees with information in nodes, *Pr. IPI. Pan.* **411** (1980) 87-88.

[5] Knuth D. E., Optimal binary search tree, *Acta Informatica* **1** (1971).

Nhận bài ngày 12 - 8 - 1998
Nhận lại sau khi sửa ngày 15 - 9 - 1999

Khoa Công nghệ thông tin,
Đại học Quốc gia Hà Nội.