Tạp chí Tin học và Điều khiển học, T. 16, S. 3 (2000), 47–55

TỐI ƯU HÓA CÂY NHỊ PHÂN MỘT CHIỀU VỚI THÔNG TIN CHỨA Ở CÁC ĐỈNH TRONG TRÊN TẬP KHÓA HỮU HẠN

ĐỖ ĐỨC GIÁO, ĐẶNG THỊ NGA, VÀ VŨ LÊ TÚ

Abstract. The notion of search tree plays an impotant role in computer science, especially in the theory of data structures. The efforts to optimize one dimensional binary search tree as introduced in this paper quite useful for practical applications, especially for the representation of range queries, where the information about secondary keys defined on range are organized as a binary tree.

In this paper we intend to further develop the conception of the binary search tree in [3], [5] and prove a theorem which shows that each binary search tree can be uniquely transformed into optimal binary search tree in the finite set of keys.

1. ĐẶT VẤN ĐỀ

Trong [5] Thiele đã đưa ra định nghĩa cây nhị phân một chiều các thông itn chứa các đỉnh trong của cây trên tập khóa vô hạn N gồm các phần tử thỏa mãn quan hệ so sánh, đồng thời đưa ra một thuật toán để khẳng định hai cây bất kỳ có tương đương với nhau hay không trên tập khóa N.

Dựa vào thuật toán đó của Thiele trong [5], các tác giả trong [3] phát triển thêm thuật toán tìm cây tối ưu của một cây bất kỳ cho trước trên tập khóa vô hạn N.

Trong bài báo này, chúng tôi trình bày việc xây dựng thuật toán tương đương và thuật toán tối ưu trên lớp cây nhị phân một chiều với các thông tin chứa ở các đỉnh trong trên tập khóa hữu hạn bất kỳ $K \subset N$. Thực chất nội dung của bài này là cơ sở xây dựng thuật toán phân rã để tối ưu hóa cây nhị phân với các thông tin chứa ở các đỉnh trong. Chúng tôi sẽ đề cập đến thuật toán phân rã trong một bài báo khác.

2. SỰ TƯƠNG ĐƯƠNG CỦA CÂY NHỊ PHÂN TRÊN TẬP KHÓA HỮU HẠN *K*

2.1. Định nghĩa cây nhị phân

Giả sử I là tập hữu hạn không rỗng các phần tử nào đó. I được gọi là tập các thông tin .N là tập không rỗng các phần tử trên nó thỏa mãn quan hệ so sánh. Gọi N là tập khóa, mỗi phần tử $k \in N$ gọi là một khóa (key).

Đặt $I^+ = I \cup \{\tau\}$, ở đây τ là ký hiệu thông tin rỗng (không có thông tin).

Định nghĩa 1.

1. τ là một cây.
2. Nếu T_1, T_2 là hai cây thì dãy ký hiệu $[k, i]\langle T_1, T_2\rangle$, với $k \in N$, $i \in I$, là một cây.

Tập tất cả các cây định nghĩa như trên ta ký hiệu TREE và gọi là tập tất cả các cây nhị phân một chiều với thông tin chứa ở đỉnh trong. Để cho ta gọi TREE là tập các cây nhị phân.

Định nghĩa 2. Lấy $T \in$ TREE và $1 \in N$. Ta định nghĩa hàm làm việc (hay hàm kết quả) f : TREE $\times N \to I^+$ là một ánh xạ từ tập tích Đề các TREE $\times N$ vào tập I^+ như sau:

1. Nếu $T \in$ TREE mà T có dạng cây rỗng τ thì $f(\tau, l) = \tau$ với $\forall l \in N$.
2. Nếu $T \in$ TREE mà T có dạng $[k, i]\langle T_1, T_2\rangle$ thì

$$f([k,i]\langle T_1,T_2\rangle,l)=\begin{cases} f(T_1,l) & \text{nếu } l<k\\ i & \text{nếu } l=k\\ f(T_2,l) & \text{nếu } l>k\end{cases}$$

Ta đưa vào ký hiệu $T_1 \equiv T_2$ ($T_1 \not\equiv T_2$) có nghĩa là cây T_1 đồng nhất bằng cây T_2 (cây T_1 không đồng nhất bằng cây T_2).

Định nghĩa 3. Ta nói cây T_1 tương đương cây T_2 trên tập N (ký hiệu $T_1 \approx (N) T_2$) khi và chỉ khi $f(T_1,l) = f(T_2,f)$ với $\forall l \in N$.

2.2. Quy tắc dẫn xuất và hệ tiên đề của TREE

Giả sử $T_1, T_2 \in$ TREE và $=$ là một ký hiệu mới không nằm trong tập $I^+ \cup N \cup \{[,],\langle,\rangle\}$, khi đó dãy ký hiệu $T_1 = T_2$ được gọi là một phương trình cây. Tập tất cả các phương trình cây tương ứng với TREE ta kí hiệu là EQU và được định nghĩa như sau: EQU $= \{T_1 = T_2 | T_1, T_2 \in \text{TREE}\}$.

2.2.1. Quy tắc dẫn xuất của TREE

Giả sử $X \subseteq$ EQU và $T_1 = T_2 \in$ EQU. Phương trình cây $T_1 = T_2$ là dẫn được từ lập X (ký hiệu $X \vdash T_1 = T_2$) khi và chỉ khi $T_1 = T_2 \in X$ hoặc $T_1 = T_2$ nhận được từ các phần tử của X bằng cách áp dụng số hữu hạn lần các quy tắc dẫn xuất sau đây:

Quy tắc 1 (R_1): Nếu $T \in$ TREE thì $X \vdash T = T$ với mọi $T \in$ TREE.

Quy tắc 2 (R_2): Nếu $X \vdash T_1 = T_2$ thì $X \vdash T_2 = T_1$ với mọi $T_1, T_2 \in$ TREE.

Quy tắc 3 (R_3): Nếu $X \vdash T_1 = T_2$ và $X \vdash T_2 = T_3$ thì $X \vdash T_1 = T_3$ với mọi $T_1, T_2, T_3 \in$ TREE.

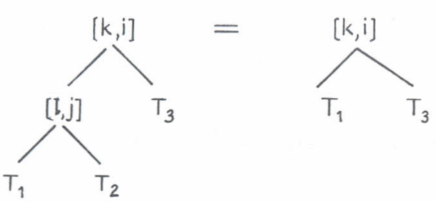
Quy tắc 4 (R_4): Nếu $X \vdash T_1 = T_1'$ thì $X \vdash [k,i]\langle T_1, T_2\rangle = [k,i]\langle T_1', T_2\rangle$ với mọi $T_1, T_2, T_1' \in$ TREE, $k \in K$ và $i \in I^+$.

Quy tắc 5 (R_5): Nếu $X \vdash T_2 = T_2'$ thì $X \vdash [k,i]\langle T_1, T_2\rangle = [k,i]\langle T_1, T_2'\rangle$ với mọi $T_1, T_2, T_2' \in$ TREE, $k \in K$ và $i \in I^+$.

2.2.2. Hệ các tiên đề của TREE trên tập khóa hữu hạn K

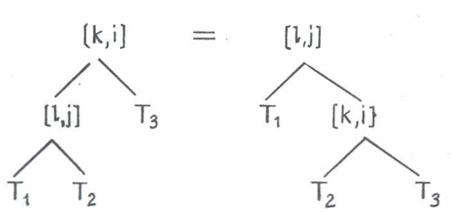
Giả sử $K \subset N$ là một tập hữu hạn các khóa. Do $K \subset K$ nên thỏa mãn quan hệ so sánh và trong K có tồn tại phần tử bé nhất và phần tử lớn nhất mà ta ký hiệu tương ứng là $k_{\min}$, $k_{\max}$. Trong các phương trình cây EQU ta chọn ra 5 phương trình cây sau đây làm hệ tiên đề của TREE trê tập hữu hạn K:

Tiên đề 1 (ax_1): Phương trình cây $[k,i]\langle [l,j]\langle T_1, T_2\rangle, T_3\rangle\rangle = [k,i]\langle T_1, T_3\rangle$ hay



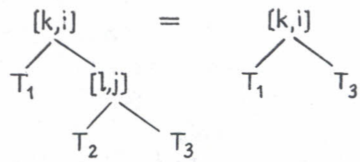
là một tiên đề nếu $k_{\min} \le k \le l \le k_{\max}$.

Tiên đề 2 (ax_2): Phương trình cây $[k,i]\langle [l,j]\langle T_1, T_2\rangle, T_3\rangle\rangle = [l,j]\langle T_1, [k,i]\langle T_2, T_3\rangle\rangle$ hay



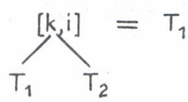
là một tiên đề nếu $k_{\max} \ge k > l \ge k_{\min}$.

Tiên đề 3 (ax_3): Phương trình cây $[k,i]\langle T_1,[l,j]\langle T_2,T_3\rangle\rangle = [k,i]\langle T_1,T_3\rangle$ hay



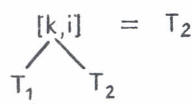
là một tiên đề nếu $k_{\max} \geq k \geq l \geq k_{\min}$.

Tiên đề 4 (ax_4): Phương trình cây $[k,i]\langle T_1,T_2\rangle = T_1$ hay



là một tiên đề nếu $k > k_{\max}$.

Tiên đề 5 (ax_5): Phương trình cây $[k,i]\langle T_1,T_2\rangle = T_2$ hay



là tiên đề nếu $k < k_{\min}$.

Đặt $AX = \{ax_1,\ ax_2,\ ax_3,\ ax_4,\ ax_5\}$ và gọi là hệ tiên đề của TREE trên tập hữu hạn K.

Chú ý: Các tiên đề ax_1, ax_2 và ax_3 là hệ tiên đề của TREE trên tập khóa vô hạn N trong [3] và ta dễ dàng kiểm tra mỗi tiên đề trong AX là một phương trình cây gồm 2 cây tương đương với nhau trên tập khóa K.

2.3. Cây chuẩn tắc trên tập khóa hữu hạn K

Định nghĩa 4. Giả sử $M \in$ TREE, M được gọi là cây chuẩn tắc trên K nếu M là một trong hai dạng sau đây:

1. $M \equiv \tau$.
2. Nếu $M \not\equiv \tau$ thì M có dạng $M \equiv [k_1,i_1]\langle \tau,[k_2,i_2]\langle \dots,[k_n,i_n]\langle \tau,\tau\rangle,\dots\rangle$ với $k_i \in K$ và $k_i < k_{i+1}$ $(i = 1,2,\dots,n-1)$.

Định lý 1. (Tính duy nhất của dạng chuẩn tắc) *Giả sử M_1, M_2 là hai cây chuẩn tắc. Nếu $M_1 \approx (K)M_2$ thì $M_1 \equiv M_2$.*

Chứng minh. Dùng phương pháp phản chứng dựa trên định nghĩa của cây chuẩn tắc M trên K.

Ta đưa vào ký hiệu $\delta(T) :=$ số các đỉnh của cây T.

Định lý 2. (về sự tồn tại dạng chuẩn) *Với mỗi $T \in$ TREE tồn tại duy nhất cây chuẩn tắc $M \in$ TREE sao cho:*

(a) $AX \vdash T = M$,

(b) $T \approx (K)M$.

Chứng minh. Tính duy nhất của M được suy từ Định lý 1.

(b) suy từ (a), vì mỗi tiên đề trong AX là một phương trình cây được lập từ 2 cây tương đương nhau trên K, đồng thời các quy tắc dẫn xuất bảo tồn tính tương đương.

(a) được chứng minh dựa trên định nghĩa đệ quy của T với sự áp dụng các tiên đề trong AX và các quy tắc dẫn xuất.

Định lý 3. *Nếu M là cây chuẩn tắc thì* $\delta(M) = \min\{\delta(T) \mid T \in \text{TREE}$ *và* $T \approx (K)M\}$.

Chứng minh. Theo phần (b) của Định lý 2 thì từ cây T đưa về cây M phải xuất hiện các cây trung gian T_i thỏa mãn các điều kiện: $T \equiv T_1 \approx (K)T_2 \approx T_3 \ldots \approx (K)T_{n-1} \approx (K)T_n \equiv M$, ở đây T_{i+1} nhận được từ T_i bằng cách áp dụng một tiên đề nào đó trong AX.

Do cách chọn các tiên đề ax_i mà ta luôn có $\delta(T_{i+1}) \leq \delta(T_i)$ $(i = 0, 1, \ldots, n-1)$. Từ đó $\delta(T) = \delta(T_1) \geq \delta(T_2) \geq \cdots \geq \delta(T_{n-1}) \geq \delta(T_n) = \delta(M)$. Giả sử có tồn tại cây $T' \in$ TREE mà $T' \approx (K)M$ nhưng $T' \notin \{T_1, T_2, \ldots, T_n\}$. Dễ dàng suy ra $\delta(T') \geq \delta(M)$ bằng phương pháp phản ứng. Từ đó $\delta(M) = \min\{\delta(T) \mid T \in \text{TREE}$ và $T \approx (K)M\}$.

Định lý 4. *Giả sử* $T_1, T_2 \in$ TREE. $T_1 \approx (K)T_2$ *khi và chỉ khi* $AX \vdash T_1 = T_2$.

Chứng minh. Từ $AX \vdash T_1 = T_2$ suy ra $T_1 \approx (K)T_2$ do Định lý 2.

Chiều ngược lại chứng minh như sau: Với T_i có tồn tại duy nhất cây chuẩn tắc M_i sao cho $AX \vdash T_i = M_i$ và $T_i \approx (K)M_i$ $(i = 1, 2)$. Vì $T_1 \approx (K)T_2$ nên $M_1 \approx (K)M_2$. Theo Định lý 1 thì $M_1 \equiv M_2$, áp dụng quy tắc R_1 ta có $AX \vdash M_1 = M_2$. Từ các kết quả trên ta suy ra $AX \vdash T_1 = T_2$. Đó là điều cần chứng minh.

2.4. Thuật toán về sự tương đương của các cây nhị phân trên tập khóa K

Input: $T_1, T_2 \in$ TREE và tập khóa K.

Output: $T_1 \approx (K)T_2$ hay $T_1 \not\approx (K)T_2$.

Bước 1: Xây dựng cây chuẩn tắc M_1 của T_1 và M_2 của T_2 trên K.

Bước 2: So sánh M_1 với M_2.

a) Nếu $M_1 \equiv M_2$ thì $T_1 \approx (K)T_2$.

b) Nếu $M_1 \not\equiv M_2$ thì $T_1 \not\approx (K)T_2$.

3. CÂY NHỊ PHÂN TỐI ƯU TRÊN TẬP KHÓA HỮU HẠN K

3.1. Cây hoàn chỉnh trên tập khóa hữu hạn K

Giả sử cây $T \in$ TREE là một cây bất kỳ. Để phân biệt vị trí các lá trong T ta ký hiệu $\tau_1, \tau_2, \ldots, \tau_n$ là các thông tin rỗng ở các lá của T (với giả thiết T có n lá). Ngoài ra ta đặt Deep (τ_i) là số các cung của đường đi từ gốc đến lá chứa τ_i trong cây T và $h(T)$ là ký hiệu chiều cao của T được định nghĩa như sau: $h(T) = \max\{\text{Deep}\,(\tau_i)/\tau_i$ là lá trong cây$\}$.

Định nghĩa 6. Cây $B \in$TREE được gọi là cây hoàn chỉnh trên tập khóa hữu hạn K nếu B là một trong hai dạng sau:

1. $B \equiv \tau$.

2. Nếu $B \not\equiv \tau$ thì B phải thỏa mãn các điều kiện sau:

a) Khóa của đỉnh cha bất kỳ của cây cong trong B nhỏ hơn khóa của đỉnh con bên phải và lớn hơn khóa của đỉnh con bên trái.

b) Nếu τ_i, τ_j là 2 lá bất kỳ trong B thì: $|\text{Deep}\,(\tau_i) - \text{Deep}\,(\tau_j)| \leq 1$ với mọi $i \neq j$.

Định lý 5. *Với mọi $T \in$ TREE bao giờ cũng tồn tại một cây hoàn chỉnh $B \in$ TREE trên tập khóa hữu hạn K sao cho:*

(a) $AX \vdash T = B$.

(b) $T \approx (K)B$.

(c) $\delta(B) = \min\{\delta(T) \mid T \in \text{TREE}$ *và* $T \approx (K)B\}$.

(d) $h(B) = \min\{h(T) \mid T \in \text{TREE}$ *và* $T \approx (K)B\}$.

Chứng minh. (a) Theo Định lý 1 từ cây T bao giờ cũng tồn tại duy nhất một cây chuẩn tắc M của T trên tập hữu hạn K sao cho $AX \vdash T = M$. Từ cây M này áp dụng một số hữu hạn lần tiên đề ax_2 ta sẽ được cây hoàn chỉnh B thỏa mãn $AX \vdash M = B$. Theo quy tắc R_3 ta có $AX \vdash T = B$.

(b) Suy từ phần (a) như Định lý 2.

(c) Từ cây chuẩn tắc M áp dụng một số hữu hạn lần tiên đề ax_2 ta được cây hoàn chỉnh B trên tập khóa hữu hạn K. Vì mỗi lần áp dụng ax_2 số đỉnh trong cây không thay đổi nên $\delta(B) = \delta(M)$. Hay $\delta(B) = \min\{\delta(T) \mid T \in \text{TREE} \text{ và } T \approx (K)B\}$.

(d) Chứng minh dựa vào tính chất của cây hoàn chỉnh (xem [3]).

3.2. Cây nhị phân tối ưu trên tập khóa hữu hạn K

Định nghĩa 7. Cây $T_0 \in$ TREE được coi là cây tối ưu trên tập khóa hữu hạn K nếu T_0 thỏa mãn đồng thời các điều kiện sau:

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

Như vậy cây tối ưu có số đỉnh bé nhất, độ cao thấp nhất so với các cây tương đương với nó. Từ định nghĩa của cây hoàn chỉnh trên tập khóa hữu hạn K ta có các Định lý 6 và 7 sau đây:

Định lý 6. *Cây hoàn chỉnh chính là cây tối ưu trên tập khóa hữu hạn K.*

Định lý 7. *Mỗi cây nhị phân $T \in$ TREE tồn tại cây tối ưu T_0 sao cho:*

(a) $AXT = T_0$,

(b) $T \approx (K)T_0$.

Chứng minh. Suy ra từ các Định lý 2, 3, 4, 5 và 6.

4. MỘT SỐ THỦ TỤC TÌM CÂY NHỊ PHÂN TỐI ƯU TRÊN TẬP KHÓA HỮU HẠN K

Theo Định lý 7 thì cây nhị phân bất kỳ $T \in$ TREE, tìm được cây tối ưu $T_0 \in$ TREE sao cho $T \approx (K)T_0$. Việc xây dựng T_0 được thực hiện theo các bước sau:

+ Đưa cây nhị phân bất kỳ T về cây chuẩn tắc M trên tập khóa hữu hạn K.

+ Từ cây chuẩn tắc M chuyển về cây hoàn chỉnh B trên tập khóa hữu hạn K ($T_0 \equiv B$).

4.1. Cấu trúc cây

```
typedef struct tree
    {     int key;
          char info [n];
          struct tree *left;
          struct tree *right;
    } TREE;
```

4.2. Các hàm xử lý hệ tiên đề

Hàm xử lý tiên đề 1

```
int ax1 (TREE *t)
  {  if ((*p).key ≤ (*(*P).left)). key && P là đỉnh trong cây gốc t)
         {  (*P).left = (*((P).left)).left;
            return 1; }
     return 0;
  }
```

Hàm xử lý tiên đề 2

```
int ax2 (TREE *t)
  {  if((*p).key < (*(*P).left)). key && P là đỉnh trong cây gốc t)
          {  P1 = (*P).left;
             (*P).left = (*P1).right;
             (*P1).right = P;
             return 1;  }
     return 0;
  }
```

Hàm xử lý tiên đề 3.

```
int ax3 (TREE *t)
  {  if ((*p).key ≥ (*((*P).right)). key && P là đỉnh trong cây gốc t)
          {    (*P).right = (*((*P).right)).left;
               return 1; }
     return 0;
  }
```

Hàm xử lý tiên đề 4

```
int ax4 (TREE *t)
  {  if ((*P).key > kmax && P là đỉnh trong cây gốc t)
          {  P = (*P).left;
             return 1; }
     return 0;
  }
```

Hàm xử lý tiên đề 5.

```
int ax5 (TREE *t)
  {  if ((*P).key < kmin && P là đỉnh trong cây gốc t)
          {  P = (*P).right;
             return 1;  }
     return 0;
  }
```

4.3. Thuật toán chuyển về cây chuẩn tắc

Input: $T \in$ TREE, tập khóa hữu hạn K.

Output: $M \in$ TREE là cây chuẩn tắc trên tập khóa hữu hạn K, $M \approx (K)T$.

Thuật toán

```
{ k = 1
  while (k := 0)
    { 1. Áp dụng ax4, nếu có thay đổi thì k++;
      2. Áp dụng ax5, nếu có thay đổi thì k++;
    }
  k = 0;
  do { 3. Áp dụng ax1 nếu có thay đổi thì k++;
       4. Áp dụng ax2 nếu có thay đổi thì k++;
       5. Áp dụng ax3 nếu có thay đổi thì k++;
    } while (k = 0)
  M ≈ (K)T;
}
```

Với thuật toán này thì các bước "duyệt ax_1, ax_2, ax_3" không cần kiểm tra các khóa có thuộc tập K hay không vì các bước "duyệt ax_4", "duyệt ax_5" đã loại bỏ toàn bộ những khóa không thuộc K.

4.4. Thuật toán chuyển từ cây chuẩn tắc về cây hoàn chỉnh

Khi đưa cây $T \in$ TREE về cây chuẩn tắc thì toàn bộ các đỉnh trong của cây chuẩn tắc có các khóa đều thuộc tập hữu hạn K. Ta chỉ việc bẻ đôi cây chuẩn tắc M một số hữu hạn lần bằng việc áp dụng tiên đề ax_2. Thuật toán trên có thể mô tả như sau:

Input: Cây chuẩn tắc M trên tập khóa hữu hạn K của cây $T \in$ TREE.

Output: Cây cân bằng $B \approx (K)M$.

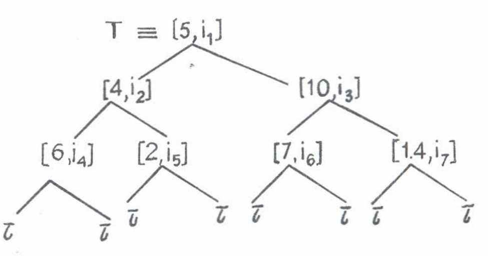
Thuật toán

```
void Bedoi (TREE *M)
  { if (h(M) > 2)
    {  Bedoi (M);
       1. Bedoi ((*M).righ);
       2. Bedoi ((*M).left);
    }
  }
```

5. VÍ DỤ MINH HỌA

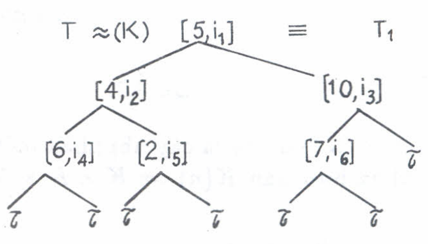
Để đơn giản ta xét tập khóa N là tập các số tự nhiên và lấy $K = \{5, 6, 7, 8, 9, 10, 11, 12\} \subset N$.

Cho cây $T \in$ TREE có dạng sau:

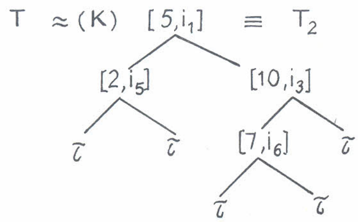


Tìm cây tối ưu của T trên tập khóa hữu hạn $K = \{5, 6, 7, 8, 9, 10, 11, 12\}$, trên tập khóa K có $k_{\min} = 5$, $k_{\max} = 12$.

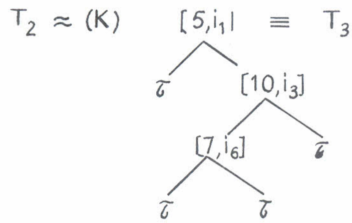
Theo ax_4 (tiên đề 4) thì tại đỉnh $[14, i_7]$ có $k = 14 > k_{\max}$ nên



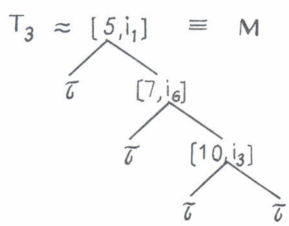
Theo ax_5 tại đỉnh $[4, i_2]$ có khóa $k = 4 < k_{\min} = 5$ nên



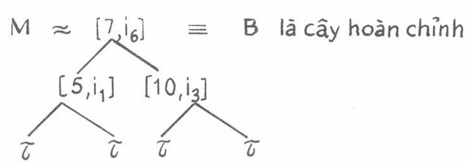
Theo ax_5 tại đỉnh $[2, i_5]$ có khóa $k = 2 < k_{\min} = 5$ nên



Sau khi duyệt ax_4 và ax_5 một số lần thì ta được cây T_3 gồm các đỉnh có khóa thuộc tập K. Tiếp tục duyệt ax_2 với $7 < 10$ ta có



Từ cây chuẩn M duyệt ax_2 (tức là bẻ đôi xây M) ta được



Rõ ràng $B \approx (K)T$ và B là cây tối ưu của T trên tập khóa hữu hạn K.

6. KẾT LUẬN

Cây nhị phân mà ta đang xét trong bài này là cây nhị phân một chiều. Ta có thể mở rộng thành cây nhị phân n chiều trên tập khóa hữu hạn $K(n) := K \times K \times K \times ... \times K$ (n lần) với K là tập $K \subset N$ được định nghĩa như sau:

1. Ký hiệu τ là một cây (cây n chiều rỗng).

2. Nếu T_1, T_2 là hai cây n chiều thì dãy ký hiệu $[p, k, i]\ \langle T_1, T_2\rangle$, với $k \in K$, $i \in I$, p là chỉ số của khóa ($I \leq p \leq n$), là một cây n chiều. Tập tất cả các cây định nghĩa như trên được ký hiệu là TREE(n) và gọi là tập các cây nhị phân n chiều với các thông tin chứa ở đỉnh trong trên tập khóa hữu hạn $K(n)$.

Khi đó hàm làm việc (hay hàm kết quả) f : TREE$(n) \times K(n) \rightarrow I^+$ được xác định như sau:

1. Nếu $T \in \mathrm{TREE}(n)$ mà T có dạng cây rỗng τ thì $f(\tau, l) = \tau$ với $\forall l = \{k_1, k_2, \dots, k_p, \dots, k_n\} \in K(n)$.

2. Nếu $T \in \mathrm{TREE}$ mà T có dạng $[p, k, i]\langle T_1, T_2\rangle$ thì

$$f([p,k,i]\langle T_1, T_2\rangle, l) = \begin{cases} f(T_1, l) & \text{nếu } k < k_p \\ i & \text{nếu } k = k_p \\ f(T_2, l) & \text{nếu } k > k_p \end{cases}$$

Ta có thể mở rộng thuật toán tương đương và thuật toán tối ưu của TREE (n) trên tập hữu hạn $K(n)$ mà với $n = 1$ thì ta thu lại toàn bộ kết quả trong bài báo này.

TÀI LIỆU THAM KHẢO

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

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

[3] Đỗ Đức Giáo, Lê Anh Cường, Tối ưu hóa cây nhị phân một chiều với thông tin chứa ở các đỉnh trong trên tập khóa vô hạn, *Tạp chí Tin học và Điều khiển học* **15** (4) (1999) 43-49.

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

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

Nhận bài ngày 17 - 9 - 1999
Nhận lại sau khi sửa ngày 7 - 7 - 2000

Khoa Công nghệ thông tin,
Trường Đại học Khoa học tự nhiên - ĐHQG Hà Nội.