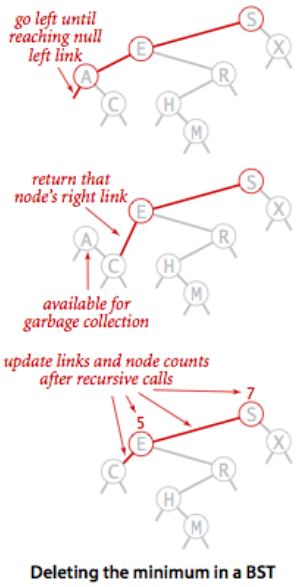
TH 02 - CÂY NHỊ PHÂN TÌM KIẾM (BINARY SEARCH TREE)

01/2018

1 Thực hành

1.1 Xoá node nhỏ nhất

Để xoá node chứa khoá nhỏ nhất trong cây, chúng ta sẽ liên tục duyệt nhánh bên trái cho đến khi tìm được node nhỏ nhất. Node nhỏ nhất này cũng chính là node không có cây con trái. Sau đó thay thế liên kết từ node cha đến cây con phải (nếu có).



Hình 1: Xoá node nhỏ nhất trong cây

```
private Node deleteMin(Node x) {
    if (x.left == null)
        return x.right;
    x.left = deleteMin(x.left);
    x.size = size(x.left) + size(x.right) + 1;
    x.height = 1 + Math.max(height(x.left), height(x.right));
    return x;
}
```

1.2 Xoá một node trong cây

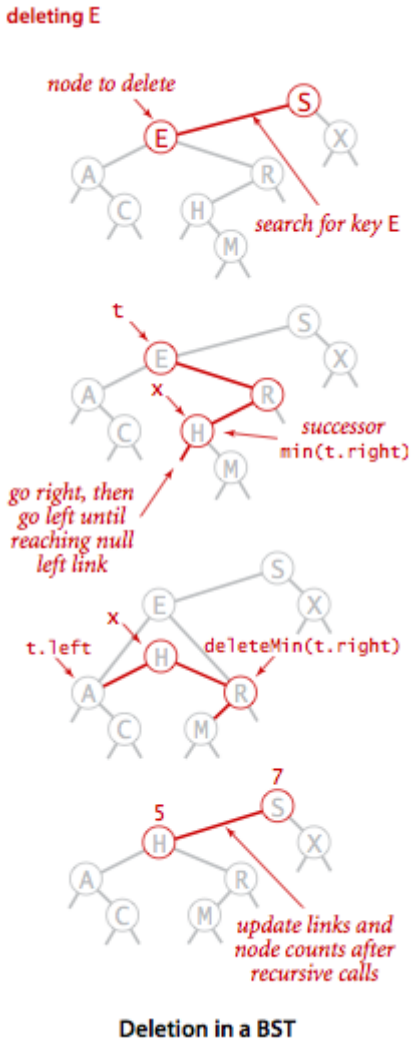
Khi node cần xoá chỉ có một cây con trái hoặc phải, sau khi xoá ta chỉ việc thay thế bằng node con trái hoặc phải.

Khi node cần xoá có đủ cả hai cây con trái và phải. Sau khi xoá ta sẽ thay node cần xoá bằng node trái nhất (node nhỏ nhất) của cây con phải hoặc node phải nhất (node lớn nhất) của cây con trái. Phương pháp này được đề xuất bởi T. Hibbard vào năm 1962 đó là xoá một node x và thay thế bởi node hậu duệ của nó. Bởi vì x có cây con phải, *hậu duệ* của nó chính là node có khoá nhỏ nhất trong cây con phải.

- Lưu node cần xoá vào biến t.
- Gán biến x trỏ đến hậu duệ cần thay thế *min(t.right)*.
- Gán liên kết phải của x đến *deleteMin(t.right)*, liên kết đến cây BST chứa tất cả các khoá lớn hơn *x.key* sau khi xoá.
- Gán liên kết trái của x (node không null) đến *t.left*.

```
private Node delete(Node x, Key key) {
    if (x == null) return null;

    int cmp = key.compareTo(x.key);
    if (cmp < 0)
        x.left = delete(x.left, key);
    else if (cmp > 0)
        x.right = delete(x.right, key);
    else {
        if (x.right == null)
            return x.left;
        if (x.left == null)
```



Hình 2: Xoá node trong cây

```
            return x.right;
        Node t = x;
        x = min(t.right);
        x.right = deleteMin(t.right);
        x.left = t.left;
    }
    x.size = size(x.left) + size(x.right) + 1;
    return x;
}
```

2 Bài tập

1. Viết hàm *deleteMax* xoá node lớn nhất trong cây.

```
public void deleteMax() {
    // your code
}
```

2. Viết hàm *delete1* xoá node trong cây và thay thế node vừa xoá bằng node con bên trái.

```
public void delete1(Key key) {
    // your code
}
```