## University of Waterloo CS 466 — Advanced Algorithm Spring 2013 Problem Set 2 Siwei Yang - 20258568

- 1. [8 marks: The Need for Paid Exchanges] In discussing Move to Front and other self-organizing heuristics the model was very important. In the model in which swapping the requested element with one immediately in front of it (and perhaps repeating this many times) as a free exchange, bring the question as to whether paid exchanges are necessary to achieve the offline optimal. Clearly they are not necessary to come within a factor of two of optimal under this model.
  - a. Show that the offline optimal to service the request  $a_3$ ,  $a_2$ ,  $a_3$ ,  $a_2$  is 8. The list is initially in the order  $a_1$ ,  $a_2$ ,  $a_3$ .

Assume we know the procedures to achieve offline optimal. Then, let  $\delta_0, \delta_1, \delta_2, \delta_3$  represent the ordering before the before first, second, third and fourth accesses; let  $c_0, c_1, c_2, c_3$  represent the cost for first, second, third and fourth accesses.

We know that  $c_0, c_1, c_2, c_3 \ge 1$ . Assuming  $c_0 = 1$ , then the access cost for  $a_2$  in  $\delta_0$  is greater than 1. Since  $c_1$  is at least access cost for  $a_2$  in  $\delta_0$ , we have  $c_1 \ge 2$ . Thus,

$$c_0 + c_1 \ge 3 \tag{1}$$

Otherwise  $c_0 \geq 2$ , and  $c_1 \geq 1$ , we still have equation 1.

And for the same reason, we have

$$c_2 + c_3 \ge 3 \tag{2}$$

Now, looking at the initial ordering  $\delta_0 = a_1, a_2, a_3$ , we have  $c_0 = 3$ . if no paid exchange is made during first access,  $c_1 \ge 2$ . Thus, we have

$$c_0 + c_1 \ge 5 \tag{3}$$

which leads to

$$c_0 + c_1 + c_2 + c_3 > 8 \tag{4}$$

Thus, the total cost is at least 8.

If paid exchange costing k is made during first access, Then we have  $c_1 \geq 1$ . Therefore, we easily have

$$k + c_0 + c_1 > 5 \tag{5}$$

which leads to

$$k + c_0 + c_1 + c_2 + c_3 \ge 8 \tag{6}$$

Thus, the total cost is at least 8.

From above, we conclude the optimal **total cost is at least 8**. And we actually achieve the total cost of 8 if we have  $\delta_1 = a_2, a_3, a_1$  is derivied from  $a_1, a_2, a_3$  after one paid and one free exchanges as well as  $\delta_1 = \delta_2 = \delta_3$ . Therefore, the optimal **total cost is 8**.

b. Also show that the optimal offline algorithm without using these paid exchanges is 9.

Consider  $\delta_1$ , if  $a_3$  move forward with free exchange, then  $\delta_1$  has  $a_2$  at it's end which means  $c_1 = 3$ . Thus

$$c_0 + c_1 = 6 (7)$$

Combining with equation 2, we have

$$c_0 + c_1 + c_2 + c_3 \ge 9 \tag{8}$$

Otherwise,  $\delta_1$  has  $a_3$  at it's end which means  $c_3 = 3$ . Thus

$$c_2 + c_3 > 4$$
 (9)

Combining with equation 3, we have

$$c_0 + c_1 + c_2 + c_3 \ge 9 \tag{10}$$

From above, we conclude the optimal **total cost is at least 9**. And we actually achieve the total cost of 9 if we have  $\delta_1 = a_1, a_2, a_3$  is derivied from  $\delta_0$  with no exchanges;  $\delta_2 = a_2, a_1, a_3$  is derivied from  $\delta_1$  with one free exchanges as well as  $\delta_2 = \delta_3$ . Therefore, the optimal **total cost is 9**.

2. [6 marks] In a splay tree we use double rotations to move an element to the root. It was mentioned that moving the requested element to the root by a sequence can give very bad amortized behavior. Prove that this amortized cost can be  $\Theta(n)$  (actually you can get it to n)for an arbitrarily long sequence of requests on a tree with n nodes, with a starting configuration of your choice. (You need not deal with either insertions or deletions.)

The adversary sequence is a repeatition of the list of sorted elements. The initial configuration doesn't matter as the first batch of accesses to all nodes in the tree will reshape the tree into a linear tree(each node has only left or right child except the leaf). Counting access cost after that, each batch of accesses to all nodes will have an aggregated cost of(considering each comparison and link travesal, or rotation cost 1)

$$(n+n+n-1+...+2)*2$$
 (11)

and the tree end up retuning to a linear tree. That sums up to  $n^2 + 3n - 2$ . Amortized cost for each access is  $\frac{n^2 + 3n - 2}{n}$  within this batch. As the adversary sequence goes arbitrarily long, the amortized cost of the whole sequence converge to  $\frac{n^2 + 3n - 2}{n}$  which is in  $\Theta(n)$ .