İTÜ

BLG 335E – Analysis of Algorithms I
Fall 2013, Recitation 6
25.12.2013

R.A. Atakan Aral
aralat@itu.edu.tr – Research Lab 1
R.A. Doğan Altan
daltan@itu.edu.tr – Research Lab 3

Outline

- Augmenting Data Structures
- Amortised Analysis

Question 1

- [Course Book Exercises **14.3-6**]

Show how to maintain a dynamic set Q of numbers that supports the operation MIN-GAP, which gives the magnitude of the difference of the two closest numbers in Q. For example, if Q = { 1, 5, 9, 15, 18, 22}, then MIN-GAP(Q) returns 18 - 15 = 3, since 15 and 18 are the two closest numbers in Q. Make the operations INSERT, DELETE, SEARCH, and MIN-GAP as efficient as possible, and analyze their running times.

Solution 1

- Underlying data structure?
- A red-black tree in which the numbers in the set are stored simply as the keys of the nodes.
- SEARCH is then just the ordinary TREE-SEARCH for binary search trees, which runs in $O(\lg n)$ time on red-black trees.

Solution 1

İTÜ

- The red-black tree is augmented by the following fields in each node x:

1. *min-gap*[x] contains the minimum gap in the subtree rooted at x. It has the magnitude of the difference of the two closest numbers in the subtree rooted at x. If x is a leaf (its children are all *nil*[T]), let *min-gap*[x] $=\infty$.
2. *min-val*[x] contains the minimum value (key) in the subtree rooted at x.
3. *max-val*[x] contains the maximum value (key) in the subtree rooted at x.

Solution 1

- Maintaining the information:
- The three fields added to the tree can each be computed from information in the node and its children.
- Therefore, they can be maintained during insertion and deletion without affecting the $O(\lg n)$ running time:

Solution 1

$$
\textit{min-val}[x] = \begin{cases} \textit{min-val}[\textit{left}[x]] & \text{if there's a left subtree ,} \\ \textit{key}[x] & \text{otherwise ,} \end{cases}
$$

$$
\textit{max-val}[x] = \begin{cases} \textit{max-val}[\textit{right}[x]] & \text{if there's a right subtree ,} \\ \textit{key}[x] & \text{otherwise ,} \end{cases}
$$

$$
\textit{min-gap}[x] = \min \begin{cases} \textit{min-gap}[\textit{left}[x]] & (\infty \text{ if no left subtree) ,} \\ \textit{min-gap}[\textit{right}[x]] & (\infty \text{ if no right subtree) ,} \\ \textit{key}[x] - \textit{max-val}[\textit{left}[x]] & (\infty \text{ if no left subtree) ,} \\ \textit{min-val}[\textit{right}[x]] - \textit{key}[x] & (\infty \text{ if no right subtree) .} \end{cases}
$$

Solution 1

- Why to define min-val and max-val?
- In order to make it possible to compute *min-gap* from information at the node and its children.

Solution 1

- What about MIN_GAP?
- MIN-GAP simply returns the *min-gap* stored at the tree root.
- Thus, its running time is $O(1)$.

Solution 1

- How to get the closest numbers? (This is not asked in the question)
- Starting from the root, look for where the minimum gap (the one stored at the root) came from.
- At each node *x*, simulate the computation of *min-gap*[*x*] to figure out where *min-gap*[*x*] came from.
- If it came from a subtree's *min-gap* field, continue the search in that subtree.
- If it came from a computation with *x*'s key, then *x* and that other number are the closest numbers.
- Complexity? O(lg n)

Question 2

- [Course Book Exercises **17.1-3** & **17.2-2**] A sequence of n operations is performed on a data structure. The *ith* operation costs i if i is an exact power of 2, and 1 otherwise.
- Determine the amortized cost per operation:
  a. Using the aggregate analysis.
  b. Using the accounting method.

Solution 2

a. Total cost:

$$\sum_{i=1}^{n} c_i \le n + \sum_{j=0}^{\lg n} 2^j = n + (2n - 1)$$
$$< 3n$$

Operation	Cost
1	1
2	2
3	1
4	4
5	1
6	1
7	1
8	8
9	1
10	1
⋮	⋮

- Average cost of an operation < 3
- Amortized cost per operation is *O(1)*.

Solution 2

b. Charge each operation 3, then:
   - If i is not an exact power of 2, pay 1, and store 2 as credit.
   - If i is an exact power of 2, pay i, using stored credit.
   - Amortized cost is $3n$
   - Actual cost is less than $3n$ (from a.)
   - Credit never runs out (is never negative)
   - Amortized cost per operation is $O(1)$

Operation	Cost	Actual cost	Credit remaining
1	3	1	2
2	3	2	3
3	3	1	5
4	3	4	4
5	3	1	6
6	3	1	8
7	3	1	10
8	3	8	5
9	3	1	7
10	3	1	9
⋮	⋮	⋮	⋮