Course Name

Nume Prenume

October 25, 2024

Centrul Infogym Iasi

Computer Science - Algorithms

11-12 Classes

Advanced Course

Subtitle

Course description 1

Course description 2

Content

- Introdution
  - What is this about?
  - Problems solved using treaps
  - Basic operations
  - Advanced operations
- Extensions
- Case Study
- Practice Problems

Introdution

What is this about?

block title

Description

another description

$\forall\ key \in T_{left} \leqslant T_{key} < \forall\ T_{key} \in T_{right}$

$T_{pri} > \forall\ T_{pri} \in (T_{right} \cup T_{left})$

Problems solved using treaps

- arrays with indexes in range $[0, 10^{18}]$. $O(N)$ space
- keeps sorted array. $O(N)$ space, $O(logN)$ time for insert & delete
- Sum, Min, Max on interval. $O(logN)$ time
- Number of elements smaller than X. $O(logN)$ time
- Reverse elements on interval. $O(logN)$ time.
- Find Kth element. $O(logN)$ time
- Insert an element at Kth position in an array. $O(logN)$ time
- playing around with split & merge.

where N - numbber of elements.

Basic Operations

Split$(\mathbf{T}, \mathbf{X})$ $\mathcal{O}(\log N)$ Splits a treap **T** int two treaps, $\mathbf{T_1}$ and $\mathbf{T_2}$ by a key value **X**, such as $\forall\ \textit{element} \in T_1 \leqslant X < \forall\ \textit{element} \in T_2$.

Merge$(\mathbf{T_1}, \mathbf{T_2})$ $\mathcal{O}(\log N)$ Merges two treaps, $\mathbf{T_1}$ and $\mathbf{T_2}$ which respect the condition: $\forall\ \textit{element} \in T_1 \leqslant X < \forall\ \textit{element} \in T_2$, into a treap **T**

Search$(\mathbf{T}, \mathbf{X})$ $\mathcal{O}(\log N)$ Searches in **T** a node with the key value **X**. The implementation is the same as for an ordinary binary tree.

Insert$(\mathbf{T}, \mathbf{X})$ $\mathcal{O}(\log N)$ Inserts in **T** a new node with the key value **X**.

Erase$(\mathbf{T}, \mathbf{X})$ $\mathcal{O}(\log N)$ Searches in **T** a node with the key value **X** and removes it from the tree.

Advanced Operations

Build$(\mathbf{X_1}, ..., \mathbf{X_N})$ $\mathcal{O}(N)$ Builds a tree from a list of sorted values. This can be done in linear time (assuming that $X_1, ..., X_N$ are sorted).

Union$(\mathbf{T_1}, \mathbf{T_2})$ in $\mathcal{O}(M * log(N/M))$ Merges two trees, assuming that all the elements are different. An implementation with the same asymptotic behavior is possible if duplicate elements should be removed during merge.

Intersect$(\mathbf{T_1}, \mathbf{T_2})$ in $\mathcal{O}(M * log(N/M))$ Finds the intersection of two trees (i.e. their common elements). We will not consider the implementation of this operation here.

Extensions

Implicit Key Treap

A simple and very powerful modification. The idea is that the keys should be indices of the elements in the array. The keys are not kept explicit in the structure, they are calculated using the variable which keeps the size of the subtree.

In addition to the normal treap:

- Find Kth element. $O(logN)$ time
- Insert an element at Kth position in an array. $O(logN)$ time

Lazy Update Treap

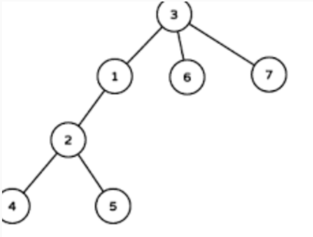
Exacty the same as lazy update on a Binary Indexed Tree.

Supports:

- Update on interval. $O(logN)$ time
- Reverse elements on interval. $O(logN)$ time.

Split - Normal Treap

```
void split(treap * root, int key,
           treap * &left, treap* &right)
{
  if(root == null)
    left = right = null;
  else if(key <= root->key) //split left child
  ␣split(root->left, key, left, root->left),
  ␣right = root;
  else                      //split right child
    split(root->right, key, root->right, right),
    left = root;
  update(root);
}
␣␣␣
```



Case Study

Codeforces 702F - T-shirts

- Problema cu siruri pare/impare
- https://acm.timus.ru/problem.aspx?space=1&num=1439
- Codeforces 702F
  http://codeforces.com/contest/702/problem/F

Practice Problems

- TREAP SPOJhttp://www.spoj.com/problems/TREAP/
- Zeap - infoarena, http://www.infoarena.ro/problema/zeap
- Codeforces 101174F, http://codeforces.com/problemset/gymProblem/101174/F
- Timus 1645, http://acm.timus.ru/problem.aspx?space=1&num=1645
- Timus 1439 http://acm.timus.ru/problem.aspx?space=1&num=1439
- Codeforces 702F http://codeforces.com/contest/702/problem/F
- Codeforces 455D http://codeforces.com/contest/455/problem/D
- Timus 1521 http://acm.timus.ru/problem.aspx?space=1&num=1521

Monster Mindcoding Final Round 2017
https://mindcoding.ro/pb/monster
https://acm.timus.ru/problem.aspx?space=1&num=1439

Bibliography

https://e-maxx-eng.appspot.com/data_structures/treap.html

http://codeforces.com/blog/entry/11148

http://codeforces.com/blog/entry/3767