Segment trees

JJV CP Algorithms

29 septembre 2023

Structure

For a given array a

- Compute segment sum $a[\ell \dots r]$
- Update element $a[i]$

Segment tree

Each node:

- handles segment $[\ell, r]$
- has one or several attributes/values (ex. min or sum of $a[\ell, r]$)
- has children $[\ell, m]$ and $[m+1, r]$ where $m = \lfloor (\ell + r)/2 \rfloor$

Building the tree has complexity $O(n)$

The height is $O(\log n)$, the complexity of queries is $4 \log n = O(\log n)$

Sum queries

Requested $[\ell, r]$

If current node is $[t\ell, tr]$, three cases:

- $[\ell, r] = [t\ell, tr]$: return current value
- $[\ell, r] \subset [tm, tr]$: 1 recursive call to the left
- Otherwise: 2 recursive calls on left and right

Update queries

Update element at position i

If current node is $[t\ell, tr]$:

- If $i \in [t\ell, tm]$: 1 recursive call to the left
- If $i \in [tm, tr]$: 1 recursive call to the right

Segment trees defined by arrays

Just like heaps

- Childrens of i are $2i$ and $2i + 1$
- Parent of i is $\lfloor i/2 \rfloor$

Variants

Should wonder: what attributes at each node, how to merge children info upwards

- Min / Max / GCD / LCM instead of Sum: easy
- Max and number of occurrences of the max
- Count number of zeroes / finding the k-th zero
- Given value x find smallest i such that $a[i] \geq x$
- Finding subsegments of maximal sum: slightly harder

Union of rectangles

- https://www.spoj.com/problems/NKMARS/
- https://lightoj.com/problem/rectangle-union

More variants: lazy

- Adding x to all cells in a range $[\ell, r]$
- Get $a[i]$

Attribute is "how much is added to this segment".
Then we will compute the actual value of a cell only if requested, in $O(\log n)$.

- Assign x to all cells in a range $[\ell, r]$
- Get $a[i]$

Or:

- Adding x to all cells in a range $[\ell, r]$
- Query for max in a range $[\ell, r]$

One can also lazy build the segment tree (grow node only if needed)

Variant: persistent

- What is the kth smallest element in range $a[\ell : r]$

What's in the notebook

- 4.5 Binary Tree is a sum segment tree
  - update cell
  - query sum
- 4.6 Binary Tree with Lazy Propagation
  - update cell
  - assign range
  - query sum
- 4.7 Persistent Binary Tree == 4.8 Persistent Segment Tree
- 4.10 Heavy-Light decomposition
  - decomposition of trees into heavy/light paths
  - can use segment trees for heavy paths
- 4.11 Range min query
  - uses sparse table: queries $O(1)$

https://cp-algorithms.com/data_structures/segment_tree.html