Final Project: Segtree-Range Minimum Query

Prachi Prashant Phatale(199005660)
Abiola Anderson(128001260)

There are various techniques for finding sum of ranges or minimum of the array or segment, to find range minimum query, we can very well just traverse the array and say minimum from a given range and return it and that would work fine for small array but when large array comes into picture, it is worse. Total time for this would be O(mn) where m is the number of queries and n is the size of the array. It might not scale very well if m and n are large numbers. Another option is it can make the range minimum matrix. It will answer the query in O(1) time, but it takes O(n^2) time to build this matrix and it takes O(n^2) space to maintain this matrix. And again, if n is a really big number, it does not scale very well as well. So, this is where segtree (segment tree) comes into picture. It takes O(n) time to build the segtree and O(n) space to maintain a segtree and answers query in O(logn) time.

A Segment Tree is a data structure that allows answering range queries over an array effectively, while still being flexible enough to allow modifying the array. It is a binary tree, and the elements of this array are the leaf of the binary tree.
Representation of Segment trees
1. Leaf Nodes are the elements of the input array.
2. Each internal node represents the minimum of all leaves under it.
If the segment arr is of size [0 . . . n-1]. and every time it divides the current segment into two halves (if it has not yet become a segment of length 1), and then it calls the same procedure on both halves, and for each such segment, it stores the minimum value in a segment tree node. All levels of the constructed segment tree will be completely filled except the last level. Also, the tree will be a Full Binary Tree because the segments are always divided in two halves at every level. Since the constructed tree is always a full binary tree with n leaves, there will be n-1 internal nodes. So total number of nodes will be 2*n – 1.

If an array has length of power of 2, then the size of segtree will be 2n-1 where n is the size of an array. If the length of the array is not power of 2 then subtract 1 from it to find the size of segtree. After creating an array of segtree, the left child can be found by 2i+1, right child can be found by 2i+2 and its parent can be found by (i-1)/2 where i is the position of an element of a segtree array. Worst case of the total size of an array would be 4n. Its space complexity will be O(n). time complexity will be O(n) and searching into a segtree would take O(logn) time.
To find the element which has the smallest value in the given range is Range Minimum Query.
Range query on the tree has three rules:

- Partial overlap of intervals
- Total Overlap of intervals
- No overlap of intervals

Suppose we have a segtree. In partial overlap of intervals, we need to look in both directions, go deeper accordingly and find the minimum of the range. In total overlap, we can straightway find

Prachi Prashant Phatale(199005660)
Abiola Anderson(128001260)

the minimum of the given range. And in no overlap, take maximum of it. Worst case will be O(4logn) which is almost equal to O(logn). After creating an array of segtree.

The Pseudo Code of the given algorithm is meant to compile the segment Tree and streamline the process of using the Minimum Range Query. Step by Step each function was built in a progressive nature as needed. We first converted the file from txt to int so that it will not input data as strings. Then after application of utility function to get the minimum of two numbers, we used a function to determine the position of the middle index from corner indices.Then we used the range minimum query function to apply its three rules which we mentioned above(Partial overlap of intervals, total overlap of intervals and no overlap of intervals). After using a query function to find the sum in the given range, we used an array to build the segtree and then we used it as our input. The datasets which we used are of 8 integer size to 8388608 integer data-size.

```
# Here we implement Libraries from the proper repository
def readInfoText():
    ## Import text file treat as number list

def minVal()
    # A utility function to get minimum of two numbers

def midPt()
    # A utility function meant to determine the position of
    # the middle index from corner indexes.


def RMQU():  # this where we can sort each array value through a given node
    # recursively
    ## Use an if Statement to look at whether or not the segment is inside the range
    ## Use an if Statement to see if it is outside of a given range
        ## implement an else  `
    ## there will also be an overlap

    # If segment of this node is within the range limit, the min of the segment
    # will be returned
    # If segment of this node is outside the given range
    # If segment component overlaps with the range return the minimum value

def RMQ():
    # Return minimum of elements in range from index qst (query start) to qbr (query
    # end). It mainly uses RMQUtil()
    # Make sure input values can be defined

    ### here we will open different indices to start and end the segment
    ## Use an "if statement"
    ## then we return the continued results of the tree


def Query(): # find the sum concerning each unique range
    # we will need to initialize the range value
    # there needs to be at least two variables
    # this allows us to find the sum in a given range which can be expanded
    ## We can use a 'while loop' to adjust with at least 2 'if' iterations

# A recursive function that constructs Segment Tree for array[start..se]. si is
# index of current node in segment tree seg
def buildUtil(): `# this is where the array is built to the segment tree
    ## here is the where we insert the leaf node into the tree
    ## there needs to be a parent node and at least 2 children nodes
    ## in this way we can reference data directly from an actual dataset

    # If there is one element in array, store it in current node of segment tree and return
    # If there are more than one elements, then recur for left and right subtrees and
    # store the minimum of two values in this node

def buildST(): here is where the tree is actually built
    ### this is done through the memory
    ## make sure that the memory is properly structured
    ## this is where the value of each array placement is used
    ## return the tree as built

    # Allocate memory for segment tree by;
    # finding the Height and maximum of the segment tree fill the memory then
    # return the constructed tree.


# Initial set Code
if __name__ == "__main__" : # here is where we set up the original setting
    ## reference the above array values to complete the SegTree construction
```

Final Project: Segtree-Range Minimum Query

Prachi Prashant Phatale(199005660)
Abiola Anderson(128001260)

## the given array will allow the segment tree to compile in stages
## show the starting index
## show the ending index
## Print the tree results
## order the tree results within the range
## then print the proper array value

We run the code on different datasets starting its size from 8 int to 8388608 int. The results for it are as follows:

Size of database	Runtime(ms)
8 int	7.336230283
32int	8.070910644
128int	8.878664045
512int	9.754644355
1024int	9.889781885
4096int	10.25110004
4192int	10.71749693
8192int	10.71654866
16384int	11.4002723
32768int	11.52264783
65536int	11.83730287
131072int	12.4296171
262144int	13.67191538
524288int	13.8355573
1048576int	16.16887314
8388608int	16.65704837

It takes O(n) time to build the segtree and O(n) space to maintain a segtree and answers query in O(logn) time.

Final Project: Segtree-Range Minimum Query

Prachi Prashant Phatale(199005660)
Abiola Anderson(128001260)

There are multiple worldwide real life applications of range minimum query. It has several use cases in computer science, such as the lowest common ancestor problem and the longest common prefix problem (LCP).

Sports Rankings: In numerous sports such as; Basketball, Football and Soccer there are numerous ways to define the success of a team within a given league or tournament. In leagues such as the British Premier League, there is a complex system that determines the rankings and point systems regarding the placement of specific soccer clubs within the league.
There is a points system that measures the success of teams in regarding wins and losses.
If a team wins, then they receive a total of 3 points
If they lose they receive 0 points
If they tie a game then both teams receive a 1 point
As the given soccer season progresses, each team begins to fill a specific rank within the Premier League based on the number of games played, who beats who and who loses. At the end of the regular season, the lowest ranked teams within the league are relegated to a lower tier league. When this happens the winning teams of the lower tier league are then allowed to enter into the premier league in the hopes of gaining a seed among the top soccer clubs. What determines a teams ranking prior to relegation involves looking at their standing in the general season and sometimes their success in numerous different tournaments. There are a total of 20 clubs with the bottom 5 being relegated at the end of a season. These 5 lowest teams are eliminated through the scores and goal differentials in the League. The new teams are ordered and ranked based on the points earned throughout the previous season and then the new season begins.
During a given tournament that can take place during the regular season a competitive bracket in the form of two opposing trees are then formed. Although these trees make work in reverse the construction of a segmented tree as previously set the losses accumulated will result in the elimination of a team.

Gaming is another key area where a segment tree and range minimum query variant can make a rather unconventional showing. When certain tournaments are played with numerous players certain points can be accumulated by players in various first person shooter games during different types of gaming formats where the lowest ranked player can be eliminated. This is one instance where it can be readily seen from the front end. On the other hand certain aspects of programming for gaming can use range minimum query. When looking at online gaming, the processing and study of large amounts of data across different metrics can shed some light on the gaming habits of users. When there is heap-like data involved in the study of graphics knowing the lowest integer could yield some interesting results in terms of how to improve the users gaming experience. In summation, the use of segment trees and range minimum query will continue to be used concerning static data structures given its efficiency and its versatility.

Final Project: Segtree-Range Minimum Query

Prachi Prashant Phatale(199005660)
Abiola Anderson(128001260)

Bibliography:
1.DanielP. “Range Minimum Query and Lowest Common Ancestor.” Topcoder, © 2021 Topcoder, 31 Oct. 2018, www.topcoder.com/thrive/articles/Range%20Minimum%20Query%20and%20Lowest%20Common%20Ancestor.
2.Crick, Joseph. “Practical Data Structures for Frontend Applications: When to Use Segment Trees.” Hackernoon.com, 1 May 2018, hackernoon.com/practical-data-structures-for-frontend-applications-when-to-use-segment-trees-9c7cdb7c2819. Accessed 22 Apr. 2021.
3.Yaniv, Aviv. “8 Real-Life Applicable Trees Data Structures for Your Toolkit.” Medium, 19 Jan. 2020, levelup.gitconnected.com/8-real-life-applicable-trees-data-structures-for-your-toolkit-7c0931ebe0c1. Accessed 18 Apr. 2021.
4.“Segment Tree - Competitive Programming Algorithms.” Cp-Algorithms.com, 2014-2021 translation by http://github.com/e-maxx-eng, cp-algorithms.com/data_structures/segment_tree.html.
5. Jain, Rachit. “Segment Trees - the Best Introduction in 10 Mins.” *Www.youtube.com*, 9 Mar. 2019, www.youtube.com/watch?v=Ic7OO3Uw6J0. Accessed 24 Apr. 2021.
6. Roy, Tushar. “Segment Tree Range Minimum Query.” *Www.youtube.com*, Autumn 2015, www.youtube.com/watch?v=ZBHKZF5w4YU&t=1001s. Accessed 25 Apr. 2021.
7. “Range Minimum Query | 3 Methods | Segment Tree.” *Www.youtube.com*, Tech Dose, 10 Feb. 2020, www.youtube.com/watch?v=DpSYj7t1sbQ. Accessed 3 May 2021.