Last update: June 8, 2022 Translated From: e-maxx.ru

Range Minimum Query

You are given an array $A[1..N]$. You have to answer incoming queries of the form (L, R), which ask to find the minimum element in array A between positions L and R inclusive.

RMQ can appear in problems directly or can be applied in some other tasks, e.g. the Lowest Common Ancestor problem.

Solution

There are lots of possible approaches and data structures that you can use to solve the RMQ task.

The ones that are explained on this site are listed below.

First the approaches that allow modifications to the array between answering queries.

- Sqrt-decomposition - answers each query in $O(\sqrt{N})$, preprocessing done in $O(N)$. Pros: a very simple data structure. Cons: worse complexity.
- Segment tree - answers each query in $O(\log N)$, preprocessing done in $O(N)$. Pros: good time complexity. Cons: larger amount of code compared to the other data structures.
- Fenwick tree - answers each query in $O(\log N)$, preprocessing done in $O(N \log N)$. Pros: the shortest code, good time complexity. Cons: Fenwick tree can only be used for queries with $L = 1$, so it is not applicable to many problems.

And here are the approaches that only work on static arrays, i.e. it is not possible to change a value in the array without recomputing the complete data structure.

- Sparse Table - answers each query in $O(1)$, preprocessing done in $O(N \log N)$. Pros: simple data structure, excellent time complexity.
- Sqrt Tree - answers queries in $O(1)$, preprocessing done in $O(N \log \log N)$. Pros: fast. Cons: Complicated to implement.
- Disjoint Set Union / Arpa's Trick - answers queries in $O(1)$, preprocessing in $O(n)$. Pros: short, fast. Cons: only works if all queries are known in advance, i.e. only supports off-line processing of the queries.
- Cartesian Tree and Farach-Colton and Bender algorithm - answers queries in $O(1)$, preprocessing in $O(n)$. Pros: optimal complexity. Cons: large amount of code.

Note: Preprocessing is the preliminary processing of the given array by building the corresponding data structure for it.

Practice Problems

- SPOJ: Range Minimum Query
- CODECHEF: Chef And Array
- Codeforces: Array Partition

Contributors:
jakobkogler (47.62%) anthony-huang (19.05%) adamant-pwn (14.29%) tcNickolas (9.52%)
kartikayx (7.14%) tanmay-sinha (2.38%)