Range Minimum Query

Dragan Marković

Range Minimum Query(RMQ) comprises all variations of the problem of finding the smallest element (or the position of the smallest element) in a contiguous subsequence of a list of items, usually numbers. This tutorial will deal with non-dynamic RMQ, or the one without any updates.

So our problem statement is as follows : Given an array of integers of length n and q queries of the type $[l, r]$ find smallest number within $a_l, a_{l+1}, ..., a_r$.

This is a well researched subject, and there are many different data structures that solve this problem efficiently including SQRT-Decomposition($\mathcal{O}(\sqrt{n})$ per query), Segment Tree($\mathcal{O}(\log n)$ per query) and similar. However the "classical" structure, the one most people think of when it comes to RMQ, and the fastest one is a sparse lookup table. It has a time complexity of $\mathcal{O}(1)$ per single query, and $\mathcal{O}(n \log n)$ memory complexity. Through use of Cartesian trees it is possible to achieve a memory complexity of $\mathcal{O}(n)$, however that approach is way too complicated, and most of the times not worth the effort.

Building the sparse table is a relatively simple process, as it turns out. Let's denote as $table_{i,j}$ minimum of the elements $a_i, a_{i+1}, ..., a_{i+2^j-1}$, or as many of these elements that exist(end of the array might be exceeded). Now, through a simple dynamic programming like approach we can build our table:

- First off set $table_{i,0} = a_i$ for all i's
- Now, we have that $table_{i,j} = min(table_{i,j-1}, table_{i+2^{j-1},j-1})$

Minimum of the block of size 2^j is simply the smaller of two minimums of blocks of the size 2^{j-1}.

To answer a query of $[l, r]$ we have to find such k that:

1. $[l, l + 2^k]$ and $[r - 2^k + 1, r]$ cover the entire interval $[l, r]$
2. These two intervals can intersect or overlap, it doesn't matter
3. These two intervals don't go out of bounds of $[l, r]$

It is easy to see that such k is exactly $floor(\log(r - l))$. So the answer is simply $min(table[l][k], table[r - 2^k + 1][k])$, which can be retrieved in $\mathcal{O}(1)$.