

Weekplan: Lowest Common Ancestors and Range Minimum Queries

Inge Li Gørtz

References and Reading

- [1] The LCA problem revisited, M. A. Bender, M. Farach-Colton, Latin American Symposium 2000.
- [2] Scribe notes from MIT
- [3] Fast Algorithms for Finding Nearest Common Ancestors, D. Harel and R. E. Tarjan, SIAM J. Comput., 13(2), 338–355.

We recommend reading [1], [2] in detail before the lecture. [3] provides background on LCA.

Exercises

- 1 [w] **RMQ** Consider the array $A = [3, 5, 1, 3, 8, 6, 9, 2, 42, 4, 7, 12]$. Construct the sparse table for A .
- 2 **Sparse table** Show that we can do the preprocessing of the sparse table in $O(n \log n)$ time.
- 3 [w] **RMQ** Consider the array $A = [3, 4, 5, 4, 5, 4, 5, 4, 3, 2, 1, 0, 1, 0, 1, 2, 3, 4, 3, 4, 3, 2, 1, 2, 3, 2, 3, 4, 5, 6, 7, 6]$.
 - 3.1 Give the arrays A' and P used for the sparse table in the two level ± 1 RMQ data structure. Use block size 3.
 - 3.2 How many different tabulation tables do we need to store (how many different describing sequences/normalized blocks are there)?
- 4 **Size of blocks** In the ± 1 RMQ data structure we divided the array into blocks of length $\frac{1}{2} \log n$. What happens if we instead use a block size of
 - $\log n$
 - $\frac{3}{4} \log n$
- 5 **Two-Level Sparse Table** Consider the two-level sparse table solution, where we first divide the array A into $n/\log n$ blocks $A_1, \dots, A_{\lceil n/\log n \rceil}$. We construct a new table A' containing the minimum element from each block and construct a sparse table for this. For each block A_i we also construct a sparse table.
 - 5.1 Analyse the space usage of the data structure.
 - 5.2 Explain how to perform a query and analyse the running time.
 - 5.3 [*] What happens to space and query time if we recursively apply this idea, i.e., instead of storing sparse tables for the A_i s we store a two-level sparse table?
- 6 **Reduction between RMQ and LCA** In the lecture we saw how to reduce RMQ to LCA via a Cartesian tree and from LCA to RMQ.
 - 6.1 Build the Cartesian tree T for the array $A = [3, 5, 1, 3, 8, 6, 9, 2, 42, 4, 7, 12]$.
 - 6.2 Reduce LCA on T to ± 1 RMQ. That is, construct the array for the ± 1 RMQ instance.

7 Distance Queries in Trees Let T be a unrooted tree in which each edge has an integer weight. The distance between two nodes u and v is the sum of edge weights on the path between u and v . Give a linear-space data structure for T that can report the distance between any pair of nodes in constant time.

8 Range Smallest and Range Uniqueness Let A be an array of length n . Consider the following queries:

- $RS(i, j, t)$: return all integers $\leq t$ in $A[i, j]$.
- $RU(i, j)$: return the unique set of integers in $A[i, j]$. (i.e. only report every occurring integer in the interval once).

Solve the following exercises.

8.1 [w] Compute the result of $RS(4, 10, 3)$, and $RU(4, 10)$ on the array $A = [4, 1, 3, 2, 1, 4, 4, 3, 3, 1, 2, 5]$

8.2 Give a compact data structure that supports RS queries. Your query time should be output-sensitive.

8.3 Define the *predecessor array* P of A as the array P such that $P[i] = \max\{0 \leq j < i, A[j] = A[i]\} \cup \{-1\}$. Draw the predecessor array P of example array from exercise 7.1.

8.4 [$*$] Give a compact data structure that supports RU queries on A . Your query time should be output-sensitive. *Hint*: find a way to use the predecessor array.

9 Cartesian Trees Give an efficient algorithm for constructing the Cartesian tree of an array with n elements.