Weekplan: Nearest 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] and [2] in detail before the lecture. [3] provides background on NCA.

Exercises

1 Reduction from RMQ to RMQ In the lecture we saw how to reduce RMQ to LCA via a Cartesian tree and from LCA to RMQ.

1.1 Build the Cartesian tree T for the array $A = [3, 5, 1, 3, 8, 6, 9, 2, 42, 4, 7, 12]$.

1.2 Reduce LCA on T to RMQ. That is, construct the array for the RMQ instance.

1.3 Prove that the reduction from LCA to RMQ is correct (in general—not just on the instance from the previous exercise).

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

3 Range X Queries We saw how to support range minimum queries on an array A of n elements in linear space and constant time. Try to support the following similar queries on A:

- Range Maximum Queries
- Range Sum Queries
- Range Median Queries

Let S be a set and c be a constant, and consider a function $f : S \to [n^c]$. Formulate a general and sufficient condition for supporting *range f queries* in linear space and constant time. Such a query takes indicies $1 \leq i \leq j \leq n$ and returns $f(\{A[i], A[i+1], \ldots, A[j]\})$.

4 Longest Common Prefixes Let S be a set of strings and $n = \sum_{x \in S} |x|$ be their total length. Give an $O(n)$-space data structure that supports the following query in constant time:

- LCP(i, j): Return the length of the longest common prefix of the two strings $x_i, x_j \in S$.

E.g., if $x_i = \texttt{algorithms}$ and $x_j = \texttt{alcohol}$ then LCP(i, j) $= |\texttt{al}| = 2$.

5 Size of blocks In the 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

- $\log n$
- $\frac{3}{4}\log n$

6 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.

7 Level ancestor In the level ancestor problem we want to support the following query in a tree T:

- $\text{LA}(x, k)$: Return the kth ancestor of x in T.

7.1 Give an $O(n^2)$ space and $O(1)$ time solution to the level ancestor problem.

7.2 Use jump pointers to give a $O(n \log n)$ space and $O(\log n)$ time solution to the level ancestor problem.

7.3 Use a ladder decomposition to give a $O(n)$ space and $O(\log n)$ time solution to level ancestor.

7.4 Combine jump pointers and the ladder decomposition to give a $O(n \log n)$ space and $O(1)$ time solution to the level ancestor problem.

8 Minimum Path Queries in Trees Let T be a unrooted tree in which each edge has an integer weight. Give a time and space efficient data structure for T that can report the minimum weight edge between any pair of nodes.