Longest Increasing Subsequence

Dragan Marković

The Problem

Given a sequence of integers $a_1, a_2, ..., a_n$. Find the length of the longest increasing subsequence(abbreviated as LIS). A sequence is defined as k-tuple $(a_{i_1}, a_{i_2}, ...a_{i_k})$ such that $1 \leq i_1 < i_2 < ... < i_k \leq n$. A sequence is increasing if $a_{i_1} < a_{i_2} < ... < a_{i_k}$.

Dynamic Programming Approach

This approach is rather straightforward and solves the problem in $\mathcal{O}(n^2)$ time and in $\mathcal{O}(n)$ extra space. The concept of the solution is quite simple:

- Let dp_i be the length of the longest increasing subsequence ending with a_i.
- $dp_{i+1} = max(dp_j + 1)$ for every $1 \leq j \leq i$ such that $a_{i+1} > a_j$.
- If such j doesn't exist then $dp_{i+1} = 1$ (every element is a LIS of length 1).
- Our answer is, of course, the largest element of dp array.

Example :
$a = \{1, 2, 1, 5, 1, 6, 2\}$
$dp = \{1, 2, 1, 3, 1, 4, 2\}$
$Answer = 4\ \{1, 2, 5, 6\}$

Greedy Approach

This approach uses $\mathcal{O}(n \log k)$ time, where k is size of LIS. We'll keep set s, initially empty, the following way:

1. Iterate through all the elements of our initial array.
2. After each iteration we insert a new element into the set, if not present.
3. After each insertion we delete the first larger element from the set(if it exists).
4. Size of s is actually the size of longest increasing subsequence.

Here is s after each iteration for array $a = \{1, 8, 9, 2, 4, 6, 3\}$

i	s
1	{1}
2	{1, 8}
3	{1, 8, 9}
4	{1, 2, 9}
5	{1, 2, 4}
6	{1, 2, 4, 6}
7	{1, 2, 3, 6}

Note that s doesn't have to necessarily contain elements of longest increasing at the end of the algorithm. However after each index i length of s is the same as length of LIS from 1 to i.

Greedy Approach Proof

Claim: After every index i size of s is the same as the size of longest increasing subsequence from a_1 to a_i, furthermore largest element of s is smallest element of all largest elements of all longest increasing subsequences until from a_1 to a_i. Corollary of this is that size of LIS of entire array is size of s after n iterations.

Proof: The proof is easily constructed using mathematical induction:

- For $i = 1$ size of s is 1, no matter what, and that is trivially size of longest increasing subsequence.
- Let's now assume that our claim holds for some $i > 1$.
- Let's denote largest element of s as m and size of s as l. For $i + 1$ we have two cases:
  1. $a_{i+1} > m$. l increases by 1, and this is indeed the size of longest increasing subsequence thus far. If it was $l + 2$, then simply remove the largest element and you have a LIS of size $l + 1$ from first i elements, which is a contradiction.
  2. $a_{i+1} < m$. At this step we add one element to the set but we also remove one element from the set, so l doesn't change. Claim here is that, thus far, the length of the longest increasing subsequence is l. Suppose it was $l + 1$ then a_{i+1} must be the largest element in that sequence, otherwise there would be subsequence of length $l + 1$ from the first i elements which is a contradiction. Now we know that for every increasing subsequence of length l, m is the smallest element out of largest elements of those increasing subsequences, so in order for a_{i+1} to be the largest element of increasing subsequence of length $l + 1$, a_{i+1} must be larger than some largest element of subsequence of length l, since m is smallest such element and $a_{i+1} < m$ this is impossible.