Greedy Algorithms

Reset Progress Reveal Solutions

1 Activity Selection

Suppose that we are given a list of intervals $[a_1, b_1], \ldots, [a_n, b_n]$. We want to check if a new interval $[c, d]$ intersects any of the $[a_i, b_i]$ intervals. What is the smallest runtime for an algorithm to do this check?

- $O(1)$
- $O(\log n)$
- [✓ $O(n)$]

Correct

Now suppose that all the $[a_i, b_i]$ intervals are *disjoint* and further sorted, i.e., $a_{i+1} > b_i$ for all i. What is the smallest runtime for an algorithm to check if a new interval $[c, d]$ intersects any of the $[a_i, b_i]$?

- $O(1)$
- [✓ $O(\log n)$]
- $O(n)$

Correct

What if we are further promised that $d \geq b_n$? What is the smallest runtime to find out if there is any intersection?

- [✓ $O(1)$]
- $O(\log n)$
- $O(n)$

Correct

What is the runtime of the greedy algorithm for activity selection? Assume the input is given in an arbitrary order.

- $O(n)$
- [✓ $O(n \log n)$]
- $O(n^2)$

Correct

In the activity selection problem, which of the following intervals are guaranteed to be part of some optimal solution?

- Shortest interval.
- [✓ Interval with largest start time.]
- Interval with smallest start time.
- All of the above.

Correct

2 Task Scheduling

Consider the task scheduling problem from lecture. Lucky and Plucky each have n tasks they need to do, and they each have already computed an optimal ordering for their tasks. Not being satisfied with how Lucky haphazardly performs tasks, Plucky insists on taking over and doing all the $2n$ tasks themself. What is the optimal runtime of finding the new optimal ordering of the $2n$ tasks?

- $O(\log n)$
- [✓ $O(n)$]
- $O(n \log n)$
- $O(n^2)$

Correct

Suppose that in an instance of the task scheduling problem, there is a unique optimal solution. If we pick two of the jobs A and B (from the n jobs in the input), then based on just the delay-costs and lengths of A and B:

- We can tell whether A and B appear next to each other in the optimal solution.
- [✓ We can tell which of A and B appears earlier in the optimal solution.]

Correct

Suppose that we have the following instance of the task scheduling problem:

$$\begin{array}{r c} & \begin{array}{ccccccc} A & B & C & D & E & F & G \end{array} \\ \begin{array}{r} \text{length} \\ \text{delay cost} \end{array} & \begin{bmatrix} 1 & 1 & 2 & 2 & 3 & 4 & 4 \\ 1 & 2 & 2 & 4 & 3 & 1 & 2 \end{bmatrix} \end{array}$$

How many different orderings of $A, B, \ldots, G$ result in an optimal solution to task scheduling?

12

Correct

3 Huffman Coding

Suppose that our alphabet consists of $n \geq 2$ letters, each appearing with nonzero frequency. In an optimal prefix-free coding, how large can the maximum length of a binary string assigned to the letters be?

- $O(\log n)$
- $n/2$
- [✓ $n - 1$]
- n

Correct

Suppose that we have two letters A and B, and we encode them as 10 and 1. Is this a prefix-free code?

- Yes
- [✓ No]

Correct

With this coding scheme, can there be an ambiguity in decoding some string of 0s and 1s?

- Yes
- [✓ No]

Correct