CS 161 W22: Section 10 Problems

March 2022

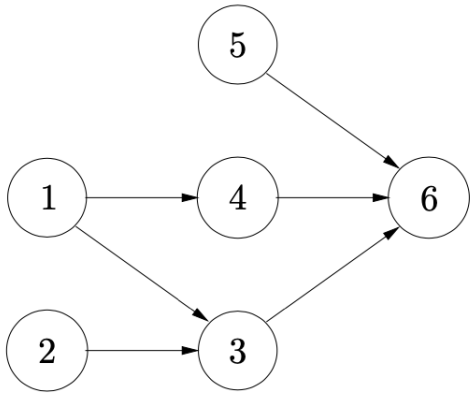
Exercise 0

Tasks before Tasks

Suppose we have n tasks, and we know how long each task will take. Additionally, tasks can be done in parallel with each other, but some tasks need to wait on other tasks to fully finish (e.g. perhaps task 5 needs the data from task 4). How long will it take to complete all our tasks?

Let's be more precise. We are given, for each task i, the time t_i it takes to complete. Additionally, we are given a set of *precedence constraints* $\mathcal{P}$, which are ordered pairs (i, j) which state that task j must wait for i to complete. We're given that $\mathcal{P}$ defines a directed acyclic graph (hint hint), where tasks are nodes and $(i, j) \in \mathcal{P}$ represents an edge from task i to task j.

For example, here is a representation of 6 tasks and 6 precedence constraints:



$$\mathcal{P} = \{(1,4), (1,3), (2,3), (3,6), (4,6), (5,6)\}$$

(a) What is a useful subproblem to consider here? How would you get the total runtime from the subproblems?

(b) Write out the recurrence relationship for this subproblem.

(c) Write an DP algorithm which finds the time it takes for all tasks to complete. To start, consider: is it more sensible to write something top-down or bottom-up?

(d) What is the runtime of this algorithm?

Fun fact: this problem is a simpler variant of a homework problem in Convex Optimization (EE364A). The original problem allowed you to adjust the speeds at which tasks completed, but limited the total "energy" used (generally higher speed meant more energy), and asked you to minimize the total time taken. Such a problem isn't meant to be solved analytically, for the record; the challenge is formulating the problem in a "convex" form, which you then feed into a numerical solver, and presto! Cool right? Hey, where are you going?

Exercise 1

Buy Low Sell High

Note that CS 161 does not endorse insider trading :')
Suppose you are investing. You want to buy low, then sell high. You have an array A of integers representing future prices, and can make one buy followed by one sell. Regardless of the price at which you buy, you will only purchase a single unit. What is the maximum profit you can make on this investment?

(a) Design an $O(n \log n)$ divide-and-conquer algorithm to return the maximum potential profit, and justify its runtime.

(b) (bonus) Design an $O(n)$-time algorithm to solve this problem, and justify its runtime. (Hint 1: what are the two numbers you need to know to determine max profit? Hint 2: consider the subproblem of max profit up for the first n days.)

Exercise 2

Ahh Quicksand!!

We are travelling through a marsh which can be mapped to an $M \times N$ grid. The marsh is mostly solid ground, but some parts are quicksand pits located throughout the marsh that are unsafe to travel through. In addition, the locations adjacent (up / down / left / right) to the quicksand pits are also unsafe. At each timestep, you can travel to any of the locations adjacent to your current location in the marsh (diagonal moves are *not* allowed). Design an algorithm that returns a shortest safe path from one side of the marsh to the other (starting at any location in the leftmost column of the grid and ending at any location in the rightmost column), and analyze its runtime.