CS 3000: Algorithms & Data — Spring '20 — Jonathan Ullman

Homework 8
Due Friday April 10 at 11:59pm via Gradescope

Name:
Collaborators:

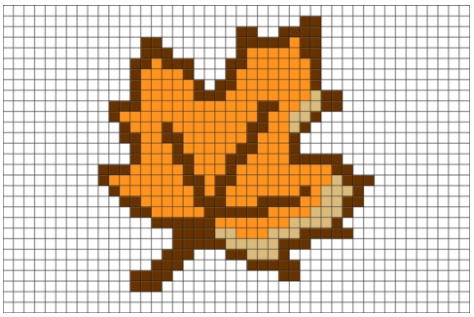
- Make sure to put your name on the first page. If you are using the LaTeX template we provided, then you can make sure it appears by filling in the `yourname` command.
- This assignment is due Friday April 10 at 11:59pm via Gradescope. No late assignments will be accepted. Make sure to submit something before the deadline.
- Solutions must be typeset in LaTeX. If you need to draw any diagrams, you may draw them by hand as long as they are embedded in the PDF. I recommend using the source file for this assignment to get started.
- I encourage you to work with your classmates on the homework problems. *If you do collaborate, you must write all solutions by yourself, in your own words.* Do not submit anything you cannot explain. Please list all your collaborators in your solution for each problem by filling in the `yourcollaborators` command.
- Finding solutions to homework problems on the web, or by asking students not enrolled in the class is strictly forbidden.

Problem 1. *Tiling*

In the ruins of Pompeii, I remember seeing the House of the Tragic Poet with a famous mosaic floor proclaiming visitors to "Beware of the Dog." In Boston, a less tragic and wealthier poet has commissioned a mosaic using 1kg bars of solid gold, specifically the type CreditSuisse mints in the dimension 80mmx40mm.

Design an algorithm that takes as input a grid of 40mmx40mm squares that are either colored or white. The algorithm determines if the colored squares in the design can be *entirely covered* with gold bullion bars. We call this the *tiling problem.* The algorithm should use a reduction to the bipartite matching problem.[1]

Note that gold bars can never be split in half! Each gold bar covers exactly two of the squares. As an example, consider the gold bars on the left, and the pixel art on the right. Can the leaf be covered in gold?



(a) Precisely specify the input and output for the bipartite matching problem that your reduction will use.

Solution:

(b) How do you map an input to the tiling problem into an input into the bipartite matching problem?

Solution:

(c) How do you map an output for the bipartite matching problem to an output for the tiling? Be sure to state what properties of the output you are relying on.

Solution:

(d) Give a rigorous argument that your algorithm correctly solves the tiling problem provided that you get a correct output to the bipartite matching you created in part (b).

Solution:

(e) What is the running time of the entire algorithm, including the time to compute the bipartite matching?

Solution:

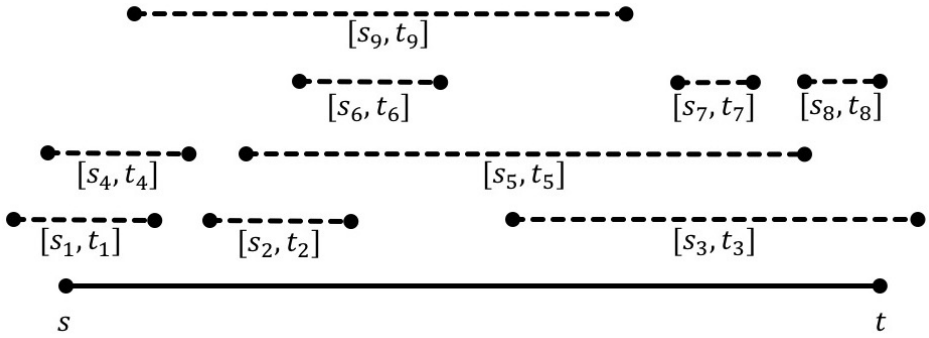
[1] **Hint:** Suppose we put a checkerboard pattern on the grid. Then each gold bullion bar covers exactly one red square and one black square.

Problem 2. *Greedy Algorithms*

You are running a convention on a shoestring budget, and need to staff the registration desk. The convention runs for the interval $[s, t]$. There are n volunteers, each of which is able to cover the interval $[s_i, t_i]$. You need to select a set of volunteers $S \subseteq \{1, \ldots, n\}$ to *cover* the entire convention, meaning that $\bigcup_{i \in S} [s_i, t_i] \supseteq [s, t]$. Equivalently, for every time $z \in [s, t]$, there is some volunteer $i \in S$ such that $z \in [s_i, t_i]$. However, each staffer will be paid for their services with a coveted *Black Lotus* card, so you need to ensure $|S|$ is as small as possible.

In this problem you will design an efficient greedy algorithm that takes as input the numbers $s, t, s_1, t_1, \ldots, s_n, t_n$ and outputs a set S that covers the convention and uses the minimum number of staffers. The running time of your algorithm should be at most $O(n^2)$ and you will receive *5 bonus points* for a (fully correct) solution with an $O(n \log n)$ time algorithm. I strongly recommend making sure that you get at least a fully correct solution before you try for bonus points.

The following is an example input with 9 volunteers. One optimal solution is $S = \{1, 3, 9\}$.



(a) Describe your algorithm in pseudocode.

Solution:

(b) Analyze the running time of your algorithm.

Solution:

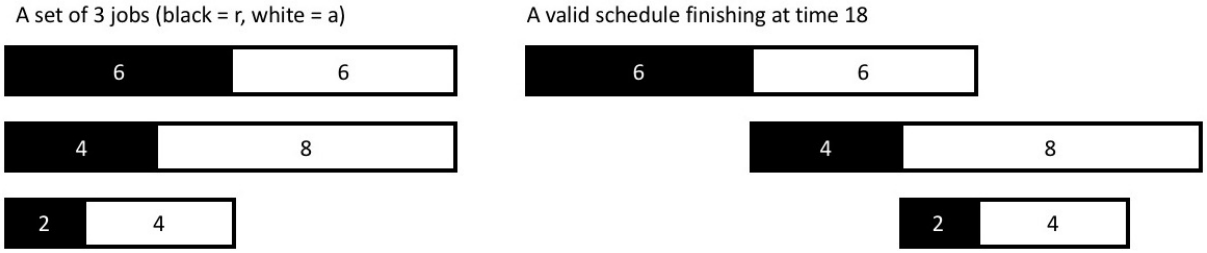
(c) Prove that your algorithm finds a set of staffers S of minimum size to cover the convention.

Solution:

Problem 3. *Seating Arrangements*

You are the great Dutch painter Rembrandt Harmenszoon van Rijn, renowned the world over for your prolific portrait work. However, you wouldn't be so prolific without your army of apprentices, and your skill with scheduling algorithms. You've been contracted by a wealthy aristocrat to produce portraits of each of his family members, and to preserve your reputation you must complete all the portraits as quickly as possible. Each portrait requires you to spend some of your time starting it, and then will require some amount of time for one of your interchangeable apprentices to finish. There is only one of you, but there is an infinite number of apprentices who can work in parallel. To satisfy your patron, you need to schedule the work so that the entire batch of portraits are finished as quickly as possible.

Design an algorithm that takes a set of n portraits, each with a pair $p_i = (r_i, a_i)$ where r_i is the amount of time you (Rembrandt) must spend on it and a_i is the amount of time it will take an apprentice to finish with it, and outputs a schedule for the work so that the last apprentice finishes the last portrait at the earliest possible time. An example of a valid input and valid (though not necessarily optimal) solution is given below.



(a) In what order does your greedy algorithm sort the inputs?

Solution:

(b) Give pseudocode for your greedy algorithm.

Solution:

(c) Prove that your algorithm finds an optimal schedule.

Solution:

(d) Analyze the running time of your algorithm?

Solution: