CSCI 332, Fall 2024
Homework 9

Due before class on Tuesday, November 5, 2024–that is, due at 9:30am Mountain Time

---

Submission Requirements

- Type or clearly hand-write your solutions into a PDF format so that they are legible and professional. Submit your PDF on Gradescope.
- Do not submit your first draft. Type or clearly re-write your solutions for your final submission.
- Use Gradescope to assign problems to the correct page(s) in your solution. If you do not do this correctly, we will ask you to resubmit.
- You may work with a group of up to three students and submit ***one single document*** for the group. Just be sure to list all group members at the top of the document. When submitting a group assignment to Gradescope, only one student needs to upload the document; just be sure to select your groupmates when you do so.

Academic Integrity

Remember, you may access ***any*** resource in preparing your solution to the homework. However, you ***must***

- write your solutions in your own words, and
- credit every resource you use (for example: "Bob Smith helped me on this problem. He took this course at UM in Fall 2020"; "I found a solution to a problem similar to this one in the lecture notes for a different course, found at this link: www.profzeno.com/agreatclass/lecture10"; "I asked ChatGPT how to solve part (c)"; "I put my solution for part (c) into ChatGPT to check that it was correct and it caught a missing case.") If you use the provided LaTeX template, you can use the `sources` environment for this. Ask if you need help!

Grading

Remember, submitted homeworks are graded for completeness, not correctness. Correctness is evaluated using homework quizzes. Each submitted problem will be graded out of six points according to the following rubric:

- Does the solution address the correct problem?
- Does the solution make a reasonable attempt at solving the problem, even if not fully correct?
- Is the presentation neat?
- Is the explanation clear?

- Does the solution list collaborators or sources, or state that the student did not use any collaborators or outside resources?
- Is the solution written in the student's own voice (not copied directly from an outside resource)?

1. You plan to go trick-or-treating in a neighborhood that consists of one long road with n houses on it. Starting from one end of the road, the houses are located at distances $x_1, x_2, \ldots, x_n$. From previous Halloweens, you know exactly how many pieces of candy each house gives out: c_1 at the first house located at x_1, c_2 at the second house located at x_2, and so on.

   Your goal is to choose which houses to visit in order maximize the number of pieces of candy that you get as you walk from the start of the road to the end. However, the owners of these houses do not like to give candy to trick-or-treaters who are not children, and after they have handed you your candy, they can see that you are not a child. They will then call ahead to all of their neighbors within 300 yards and tell them not to give candy to anyone wearing your costume. However, they do not call any houses farther than 300 yards away, so you are free to get candy from a house that is farther along.

   (a) Suppose that the houses are located at (in yards) $x_1 = 100$, $x_2 = 350$, $x_3 = 1000$, $x_4 = 1100$, $x_5 = 1200$, $x_6 = 1300$, and c_1 through c_6 are $5, 8, 2, 12, 3, 7$. What is the maximum amount of candy that you can get? Which houses should you visit?

   (b) For house located at x_j, let $p(j)$ be the previous house that is closest to house j but more than 300 yards away. What are $p(1), p(2), p(3), p(4), p(5)$, and $p(6)$ for the example from (a)?

   (c) Let OPT(j) denote the number of pieces of candy from an optimal subset of houses among $x_1, x_2, \ldots, x_j$. Write a recurrence relation expressing OPT(j) as a function of c_j, OPT($p(j)$), and OPT($j-1$).

   (d) Write an algorithm that uses dynamic programming and the recurrence from (c) to compute the optimal amount of candy that you can collect.

   (e) Analyze the runtime of your algorithm.

2. Let $G = (V, E)$ be an undirected graph with n nodes. We call a subset of the nodes an *independent set* if no two of them are joined by an edge. Finding large independent sets is difficult in general, but here we'll see that it can be done efficiently if the graph is simple enough.

   Call a graph $G = (V, E)$ a *path* if its nodes can be written as $v_1, v_2, \ldots, v_n$, with an edge between v_i and v_j if and only if the the numbers i and j differ by exactly 1. With each node v_i, we associate a positive integer weight w_i.

   Consider, for example, the five-node path below. The weights are the numbers drawn in the nodes.

   

   The goal in this question is to solve the following problem: *Find an independent set in path G whose total weight is as large as possible.* For this homework, you will not design an algorithm for this problem yet; we will do that in class.

   (a) Give an example to show that the following algorithm *does not* always find an independent set of maximum total weight.

Start with S equal to the empty set
While some node remains in G:
  Pick a node v_i of maximum weight and add it to S
  Delete v_i and its neighbors from G
Return S

(b) Give an example to show that the following algorithm also *does not* always find an independent set of maximum total weight.

Let S_1 be the set of all v_i where i is odd
Let S_2 be the set of all v_i where i is even
(Note that S_1 and S_2 are both independent sets)
Determine which of S_1 and S_2 has greater total weight, and return this one