CSCI 3104 Spring 2023
Instructors: Ryan Layer and Chandra Kanth Nagesh

Homework 21

Due Date ..... Apring 6, 2023
Name ..... **Blake Raphael**
Student ID **109752312**
Collaborators ..... **Brody Cyphers, Alex Barry, and Ben Kohav**

Contents

1 Instructions

- The solutions **should be typed**, using proper mathematical notation. We cannot accept hand-written solutions. Here's a short intro to LaTeX.
- You should submit your work through the **class Gradescope page** only (linked from Canvas). Please submit one PDF file, compiled using this LaTeX template.
- You may not need a full page for your solutions; pagebreaks are there to help Gradescope automatically find where each problem is. Even if you do not attempt every problem, please submit this document with no fewer pages than the blank template (or Gradescope has issues with it).
- You are welcome and encouraged to collaborate with your classmates, as well as consult outside resources. You must **cite your sources in this document. Copying from any source is an Honor Code violation. Furthermore, all submissions must be in your own words and reflect your understanding of the material.** If there is any confusion about this policy, it is your responsibility to clarify before the due date.
- Posting to **any** service including, but not limited to Chegg, Reddit, StackExchange, etc., for help on an assignment is a violation of the Honor Code.
- You **must** virtually sign the Honor Code (see Section 2). Failure to do so will result in your assignment not being graded.

2 Honor Code (Make Sure to Virtually Sign)

- My submission is in my own words and reflects my understanding of the material.
- Any collaborations and external sources have been clearly cited in this document.
- I have not posted to external services including, but not limited to Chegg, Reddit, StackExchange, etc.
- I have neither copied nor provided others solutions they can copy.

Agreed (I agree to the above, Blake Raphael). □

3 Standard 21 - Dynamic Programming: Identify the Precise Subproblems

The goal of this standard is to practice identifying the recursive structure. To be clear, you are **not** being asked for a precise mathematical recurrence. Rather, you are being asked to clearly and precisely identify the cases to consider. Identifying the cases can sometimes provide enough information to design a dynamic programming solution.

3.1 Problem 1 (2 Points)

Problem 1. Consider the Stair Climbing problem, defined as follows.

- Instance: Suppose we have n stairs, labeled $s_1, \ldots, s_n$. Associated with each stair s_k is a number $a_k \geq 1$. At stair s_k, we may jump forward i stairs, where $i \in \{1, 2, \ldots, a_k\}$. You start on s_1.
- Solution: The number of ways to to reach s_n from s_1.

Your job is to clearly identify the recursive structure. That is, suppose we are solving the subproblem at stair s_k. What precise sub-problems do we need to consider?

Answer. The subproblems we need to consider are all problems where we jump forward 1 stair, to 2 stairs, all the way to a_k stairs from stair s_k. That is we need to consider $s_1 \to s_2$, $s_1 \to s_3$, ..., $s_1 \to s_k$ for all of our potential subproblems.

We can put this in a function that represents all the possible ways to reach s_n from s_k.

$F(s_k) = F(s_k + 1) + F(s_k + 2) + ... + F(s_k + a_k)$

This is the precise subproblem we need to address in our stair climbing problem. □

3.2 Problem 2 (2 Points)

Problem 2. Fix $n \in \mathbb{N}$. The *Trust Game* on n rounds is a two-player dynamic game. Here, Player I starts with \$100. The game proceeds as follows.

- **Round 1:** Player I takes a fraction of the \$100 (which could be nothing) to give to Player II. The money Player I gives to Player II is multiplied by 1.5 before Player II receives it. Player I keeps the remainder. (So for example, if Player I gives \$20 to Player II, then Player II receives \$30 and Player I is left with \$80).
- **Round 2:** Player II can choose a fraction of the money they received to offer to Player I. The money offered to Player I increases by a multiple of 1.5 before Player I receives it. Player II keeps the remainder.

More generally, at round i, the Player at the current round (Player I if i is odd, and Player II if i is even) takes a fraction of the money in the current pile to send to the other Player and keeps the rest. That money increases by a factor of 1.5 before the other player receives it. The game terminates if the current player does not send any money to the other player, or if round n is reached. At round n, the money in the pile is split evenly between the two players.

Each individual player wishes to maximize the total amount of money they receive.

Your job is to clearly identify the recursive structure. That is, at round i, what precise sub-problems does the current player need to consider? [**Hint:** Do we have a smaller instance of the Trust Game after each round?]

Answer. The subproblem we have to consider with the Trust Game is that we have n number of rounds to our game. We start with 100 dollars which Player 1 chooses an amount to give to Player 2 that multiplies by a factor of 1.5x. The next round, Player 2 then chooses an amount from their pool to return back to Player 1 and multiply that amount by 1.5x as well. The only thing that changes is we now have $n - k$ number of rounds left where k represents the amount of rounds played thus far.

The key differences round to round are as follows: we have less rounds to play each turn that is taken, and the dollar amount in the pool of money changes based on the decision made by the player in control in the previous round. The finally in round n, the pool is split evenly between players. This is the only round where the even split of the pool is a guaranteed outcome.

□