CSC 4120 Project

Pandemic Control

November 21, 2022

Important: This project is due on 23:59 pm, Dec 18, 2022.

Topics: computational theory, greedy method, approximation algorithm.

1 Background

Suppose the government has become aware of the presence of a new deadly variation of the COVID-19 virus. This new variation is highly contagious and all close contacts will be infected quickly. If one person gets infected, all of her/his neighbours will get infected by the next day. The vaccine for this virus is so hard to make that the government can only produce 1 vaccine per day[1].

For simplicity, the interactions between people is modeled as a **tree**, with two people sharing an edge if the virus can be transmitted from one to the other. Now consider a strategy to **save as many people as possible**. It can be modeled as a game as follows:

- The virus begins at the root r of the tree $T = (V, E)$.
- The government can vaccinate 1 person (vertex) per day, and then the virus spread from all currently infected people to infect all of their neighbours who have not been vaccinated. The infection and vaccination are permanent. Once a person has been infected, she cannot be saved. Once a person has been vaccinated, she cannot be infected.
- Specifically, let $F^{(i)} \subseteq V$ be the set of vertices that have been infected by round i, and

[1]In the general case of this problem (not your project) one can produce d vaccines per day.

let $P^{(i)} \subseteq V$ be the set of vertices that have been vaccinated by round i. At the start of the game, $F^{(0)} = \{r\}$ and $P^{(0)} = \emptyset$.

In each round i, starting from $i = 1$, the government is free to choose to vaccinate any individual that is not yet infected or vaccinated in the earlier rounds. More precisely, it chooses a vertex $p^{(i)} \in V \backslash (F^{(i-1)} \cup P^{(i-1)})$ to vaccinate, creating an updated set of vaccinated people $P^{(i)} := P^{(i-1)} \cup \{p^{(i)}\}$.
Following the choice of the government, the infection spreads to all individuals that can be possibly infected in one step given the spread $F^{(i-1)}$ of the virus so far and the updated set $P^{(i)}$ of vaccinated people. Hence, it produces the new infections $\bigcup_{u \in F^{(i-1)}} N(u) \backslash P^{(i)}$, where $N(u)$ is the set of neighbours of vertex u in the graph, and at the end of the round the total infections are $F^{(i)} := F^{(i-1)} \cup (\bigcup_{u \in F^{(i-1)}} N(u) \backslash P^{(i)})$.

- The game ends at the time t_{max} when the virus can no longer spread to any vertices, i.e., at the first t for which $F^{(t)} = F^{(t-1)}$.

- The goal of the game is for the government to save as many vertices as possible by the end of the game. Note that a vertex doesn't have to be included in $P^{(t)}$ in order to be saved; it simply must not get infected when the game ends, and this happens if it is in a subtree of a vaccinated node before the infections spreads to that node. Hence, the goal is to maximize $|V \backslash F^{(t_{max})}|$.

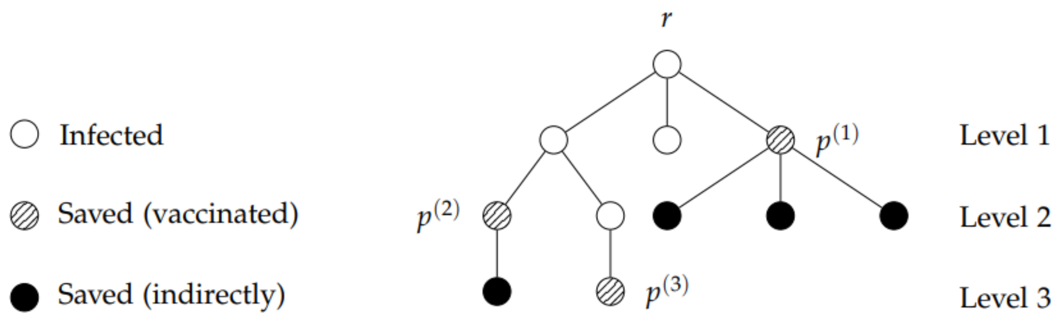

Figure 1: Example ($d = 1$). The saved vertices consist of those vaccinated (hashed), and of others saved because the virus cannot reach them (black).

2 The project deliverables

We divide the deliverables of the project into two sets depending on their interdependence. We next provide some definitions.

In any given instance of the problem, let S_{greedy} and $S_{optimal}$ denote the number of vertices saved by the greedy algorithm and the optimal solution respectively. Define as $c = \frac{S_{greedy}}{S_{optimal}}$ the performance of the greedy algorithm compared to the optimal[2]. We are interest to know how bad c can become if an adversary is free to choose the structure of the tree. In general, there is nothing to preclude that there are trees for which c can be arbitrarily small. The challenge is to come up with a greedy algorithm that has a bounded worse case, i.e., for any possible tree, $c \geq \hat{c}$ for some fixed constant $\hat{c} > 0$.

Let us give you some more information. There is no algorithm with running time polynomial in $|V|$ to compute the optimal strategy in our problem. However, a simple greedy algorithm exists that saves at least a fixed constant fraction $\hat{c}$ of the number of people saved by an optimal solution. This algorithm gives priority to vertices whose subtree has the largest number of nodes. One of your tasks is to analyze this algorithm.

2.1 Part 1

1. [**1 point**] Assume that the root r of the tree corresponds to level 0, and number the levels in increasing order going down the tree. Argue that move i (i.e., in round i) by the government should always be to vaccinate a vertex at level i in the tree (remember that the government is free to choose to vaccinate any unvaccinated individual).

2. [**1 point**] Consider the following 'degree-heavy' greedy algorithm for the government: in round i, choose $p^{(i)}$ to be the vertex at the i-th level of the tree that could be infected and has the largest degree (i.e., number of children). Show that there does not exist a constant $c \in (0, 1]$ such that this algorithm saves a guaranteed fraction c of the number of vertices saved by the optimal solution.

[2]In our lecture on approximation algorithms we will define as the 'approximation ratio' of the greedy algorithm the value of $\frac{S_{optimal}}{S_{greedy}}$. This counts how many times better is the optimal compared to the greedy.

Hint: Do a proof by counterexample. Construct a tree with a special structure for which the ratio $\frac{S_{greedy}}{S_{optimal}}$ goes to 0 as the tree gets large.

3. [**2 points**] Consider the following 'subtree-heavy' greedy algorithm: in round i, choose $p^{(i)}$ to be the vertex at the i-th level of the tree that can be infected and whose sub-tree has the largest number of nodes.

   Implement both degree-heavy and subtree-heavy algorithms and compare their performance on the given tree data sets.

4. [**1.5 points (OPTIONAL)**] Prove that the subtree-heavy heuristic actually saves at least a constant fraction $c > 0$ of the number of vertices saved by the optimal solution. Full credit will be received for $c \geq \frac{1}{2}$.

   Hint: Try to divide the optimal strategy into two parts and make $S_{optimal} = S^A_{optimal} + S^B_{optimal}$. In order to show $S_{greedy} \geq \frac{1}{2} S_{optimal}$, it suffices to show $S^A_{optimal} \leq S_{greedy}$ and $S^B_{optimal} \leq S_{greedy}$.

   Note: This question is hard so we make it optional. It is worth 1.5 points. But the maximum score one can get from this project is 10 points. If you do all questions correctly, including this optional question, you still get at most 10 points.

2.2 Part 2

[**4 points**] We define the following zero-sum game based on the previous model. The nodes of the tree are of 2 types: the blue nodes and the red nodes. Instead of having a single player (the government), now we have a blue player and a red player. Each player acts in alternating rounds, e.g., the blue player in rounds $1, 3, 5, ...$ and the red player in rounds $2, 4, ...$, where the player that plays first is chosen with prob. $1/2$.

Each player chooses to vaccinate a single not infected individual (not necessarily of the same color), and by doing so it protects all individuals (of any color) in the corresponding subtree. A player wins if by the end of the game there are more saved (i.e., non-infected) individuals of his own color compared to his opponent, and his payoff from the game is how many more individuals of the same color have been saved at the end.

More precisely, if by the end of the game there are x saved nodes of color X and y saved nodes of color Y, the payoff (profit) of player X is $x - y$ and the payoff of player y is $y - x$. Since the sum of payoffs is zero, this is a case of a zero-sum game.

The following are the two heuristic strategies we consider in the game when vertices have multiple colors.

- The 'selfish' strategy (SS): choose the node that has the largest difference of (# same color descendants - #different color descendants).
- The 'subtree-heavy' strategy (SH) with no color priority: this is same as discussed earlier. A player acts altruistically (same as cooperatively) and is not concerned to save more individuals of the same color.

Notice that in the selfish strategy a player is concerned to save more same color individuals than his opponent. He might prefer saving fewer people of his own color if this implies hurting more his opponent.

We like to analyze this game assuming that players are free to choose any strategy in {SS, SH}. We like to answer the following issues: a) If players are rational (i.e., try to maximize their average payoffs as defined earlier), which pair of strategy choices by the two players is a Nash equilibrium? b) Will society be better-off in terms of total number of saved individuals if both players are altruistic and play the SH strategy compared to both playing the Nash equilibrium strategy?

The first question is related to finding the Nash equilibrium in a two player game with strategies SH, SS and payoffs the average payoffs of the players, averaged over many game instances. If there is a unique Nash equilibrium, then we expect in real life that the players will choose to use the strategies that correspond to the Nash equilibrium.

For the second question, you are asked to check the following 'sad' fact: although cooperation (i.e., altruism) by both players is beneficial to society and each saves a larger number of same color individuals, the fact that they are selfish leads to an outcome where society suffers the most.

To summarize the previous tasks, you are required to do the following:

1. Implement the SH and SS strategy to work on trees with nodes of colors red and blue.

2. Define the game G with players $= \{Red, Blue\}$, strategies $= \{SH, SS\}$, and the 2×2 matrix of the average payoffs of strategies SH, SS.

   More precisely, for each pair of strategies $i, j \in \{SH, SS\}$, you need to compute the pair $(\pi^R_{(i,j)}, \pi^B_{(i,j)})$ of average payoffs for players Red and Blue when Red plays i and Blue plays j. These average payoffs are computed by averaging the payoffs of the corresponding strategies over many tree instances. The elements of the 2×2 (average) payoff matrix are the pairs $(\pi^R_{(i,j)}, \pi^B_{(i,j)})$, $i, j \in \{SH, SS\}$. The rows (columns) of the matrix correspond to player Red (Blue) choosing strategies SH, SS.

3. Check which strategy pair $i, j \in \{SH, SS\}$ is a Nash equilibrium. In a Nash equilibrium no player is strictly better-off by deviating.

4. Calculate the total # of saved individuals when i) players choose the Nash equilibrium strategies and when ii) both players cooperate and play the SH strategy. What you observe?

One would expect that in the case of players using identical strategies SH,SH or SS,SS, the average payoffs should be zero since each player should save the same number of individuals as his opponent because both are using the same strategy. This is not the case in the simulations, where we observe one of the players is gaining and the other loosing. This is because the test tree sample is relatively small.

To improve the results and avoid artificial break of symmetry, you could do two simulations for the same tree with a different player starting first. In any case, the resulting payoff values of symmetric strategies can be thought as the effect of 'noise'. They are expected to be an order of magnitude smaller than the payoffs in the case of non-symmetric strategies. This large difference of values should be enough for you to determine the true Nash equilibrium and to convince yourself that it will not change if you do more experiments.

3 Submission

The deadline is 23:59 pm, Dec 18, 2022.

Each group should submit the source codes for all algorithms and simulation of the games. You should submit a report [**2 points**] including

- solution to theory questions
- implementation of all algorithms
- demo output on *testCase.txt*: output the node you vaccinate at each round and the number of nodes you save for 'degree-heavy' and 'subtree-heavy' algorithms
- performance comparison: compare the performance (in terms of the number of nodes saved at the end) of 'degree-heavy' and 'subtree-heavy' algorithms. You can run multiple tests on different random trees and compare the numbers.
- game analysis: compute the (average) payoff matrix, find the NE, and check that 'sad' fact.
- you should conduct at least 20 experiments (on 10 test cases)

Submit all your files through Bb, do not zip them. In your report, state clearly the student IDs of your group members.