Assignment 11, CSC4120

Hanzhe Wu, 120090353; Shanglin Yang, 120090224; Zhexu Luo, 120090427

December 19, 2022

1 Problem 1

As the virus spread from all currently infected node to infect all of their neighbours who have not been vaccinated once in each day, we can conclude that a node with level smaller than i has been infected with the virus before the start of round i, if all of its ancestors (and itself) had not been vaccinated before or on the round of its level. Also, all the nodes with level greater than or equal to i has not been infected at the beginning of the i-th round, because there are only $(n-1)$ rounds of infect. Hence, in round i, there is no use to vaccinate any node with level smaller than i, because either it has been infected (so that it would not block the spread of the virus), or it has been vaccinated or protected by its ancestors. It is also not optimal to vaccinate a node with level greater than i, because we can vaccinate its ancestor with level i, which protects all the nodes that can be protected by the original node, and would protect strictly more nodes, since it has a larger subtree. Hence, the government should always be to vaccinate a vertex at level i in the tree.

2 Problem 2

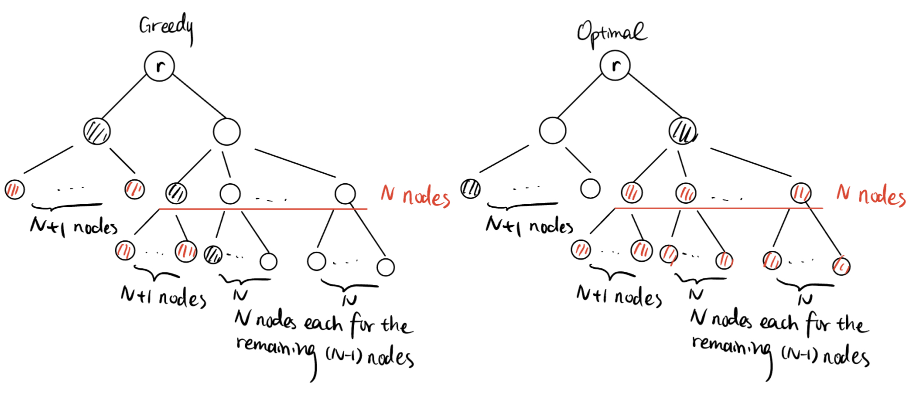
Consider the following node below. The number N can be adjusted with c to prove that there does not exist a constant $c \in (0, 1]$, such that this algorithm saves a guaranteed fraction c of the number of vertices saved by the optimal solution. In this case, the "degree-heavy" greedy algorithm would select the hashed vertices in the 3 levels. The red and black hashed nodes are those got protected, while others all get infected. In this case, there are only $1 + (N + 1) + 1 + (N + 1) + 1 = 2N + 3$ nodes that get protected. However, the optimal solution, which is on the right, protects $1 + 1 + N + (N + 1) + N(N - 1) = N^2 + N + 3$ nodes. Thus, as

$$\frac{S_{greedy}}{S_{optimal}} = \frac{2N + 3}{N^2 + N + 3} \to 0$$

as $N \to \infty$, for any $c \in (0, 1]$, there exists $N_0 \in \mathbb{N}$, such that for all $N \geq N_0$,

$$\frac{S_{greedy}}{S_{optimal}} = \frac{2N + 3}{N^2 + N + 3} < c$$

, which proves the claim (e.g., we can take $N = N_0$).


Greedy
r
N nodes
N+1 nodes
N+1 nodes
N
N
N nodes each for the remaining (N-1) nodes
Optimal
r
N nodes
N+1 nodes
N+1 nodes
N
N
N nodes each for the remaining (N-1) nodes

3 Problem 3 (optional)

Let $A \subset V$ be the nodes that get protected by the "subtree-heavy" greedy strategy, and $B = V \backslash A$ is the set of nodes infected. By Problem 1, and the definition of greedy strategy, both of the greedy and optimal strategy would contain a number of disjoint subtrees of T, whose roots are at level $1, 2, \cdots$. Consider the subtrees in the optimal strategy. Let $S^A_{optimal}$ and $S^B_{optimal}$ represents the number of protected nodes in the optimal strategy that lie in set A and B, respectively. Then, $S_{optimal} = S^A_{optimal} + S^B_{optimal}$. It is easy to see that $S^A_{optimal} \le S_{greedy}$, since $S_{greedy} = |A|$. To show that $S^B_{optimal} \le S_{greedy}$, we first have $S^B_{optimal}$ is no greater than the sum of the size of subtrees in the optimal strategy whose roots are in B, since some of those subtrees may have sub-subtree that is in A. $(*)$ Then, we claim that for those subtrees, if its root is in level l, then there is a subtree in the greedy strategy with level l, and the size of the subtree in the optimal strategy must be no greater than the size of the subtree chosen at the same level in the greedy strategy. For the first claim, as there is a node in B with root in level l, then the greedy strategy can at least choose this node at round l. For the second claim, if otherwise, as the root is in B, then the "subtree-heavy" greedy strategy would choose the node in the optimal strategy at round l, as it has a larger size of subtree. Hence, each of the subtree in the optimal solution with its root in B is no greater than its counterpart in the greedy strategy, and by $(*)$, we have $S^B_{optimal} \le S_{greedy}$. Thus, we have

$$\begin{aligned} S_{optimal} &= S^A_{optimal} + S^B_{optimal} \\ &\le S_{greedy} + S_{greedy} \\ &= 2S_{greedy} \\ c = \frac{S_{greedy}}{S_{optimal}} &\ge \frac{1}{2} \end{aligned}$$

4 Problem 4

4.1 1

Implementation of degree-heavy and subtree-heavy strategies

```
Algorithm 1: Degree-Heavy Algorithm
Input: Queue Q, Infected-set S
Output: Saved number m
Initialization: Q.put(node(1))

For i=1 to Max_level:
    While Q is not empty:
        u = Q.pop()
        For children in u:
            S.add(children)
    If S.length >= 1:
        target = argmax (n in S)[(n.degree)]
        If S.length > 1:
            S.pop(target)
            m += (target.size + 1)
            For node in S:
                Q.put(node)
        Else:
            m += (target.size + 1)
        break
    Else:
        Break

Algorithm 2: Subtree-Heavy Algorithm
Input: Queue Q, Infected-set S
Output: Saved number m
Initialization: Q.put(node(1))
For i=1 to Max_level:
```

```
While Q is not empty:
u = Q.pop()
For children in u:
    S.add(children)
If S.length >= 1:
    target = argmax (n in S)[(n.size)]
    If S.length > 1:
        S.pop(target)
        m += (target.size + 1)
        For node in S:
            Q.put(node)
    Else:
        m += (target.size + 1)
        break
Else:
    break
```

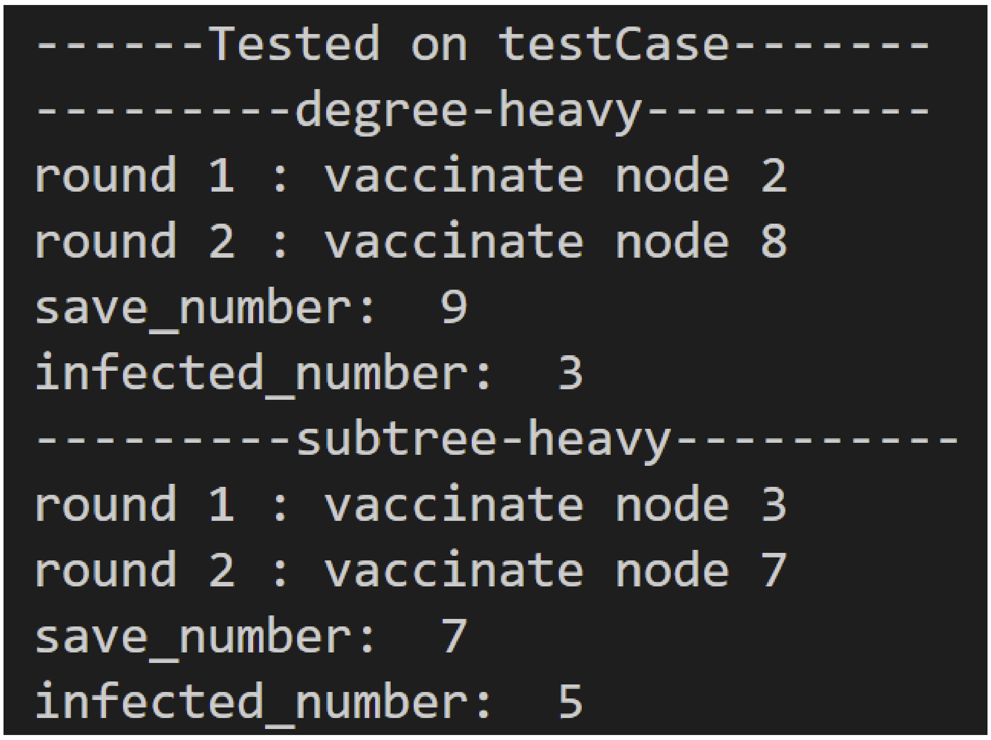
4.2 2

Pseudo code of SS strategy:

```
Q1 = a dictionary storing key = node index and value = the difference of blue
     descendents - red descendents at this node
Q2 = a dictionary storing key = node index and value = the difference of red
     descendents - blue descendents at this node
for i = 1 to max[level_list]:
    if all nodes whose level = i are safe:
        break
    if blue is playing:
        pop min element = x from Q1
        if x[value] < 0:
            blue takes no operation this round
        if x is not safe:
            vaccinate x and sign x as safe
        else:
            keep popping min element from Q1 until an unsafe one
        sign all descendents of x as safe
        iterate upwards through all ascestors of x, for each ancestor, substract its
            rsize and b size by the rsize and bsize of x separately
        update values in Q1 and Q2
    if red is playing:
        symmetric to blue
```

5 Problem 5

Demo output on testCase.txt:

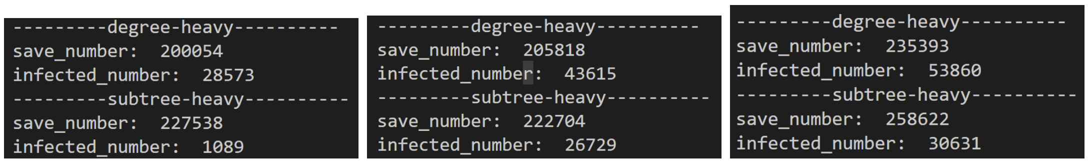


```
------Tested on testCase-------
---------degree-heavy----------
round 1 : vaccinate node 2
round 2 : vaccinate node 8
save_number:  9
infected_number:  3
---------subtree-heavy----------
round 1 : vaccinate node 3
round 2 : vaccinate node 7
save_number:  7
infected_number:  5
```

Figure 1: Demo output on testCase.txt

6 Problem 6

Performance comparison:



```
---------degree-heavy----------
save_number:  200054
infected_number:  28573
---------subtree-heavy----------
save_number:  227538
infected_number:  1089
```

(a) testCase1.txt

```
---------degree-heavy----------
save_number:  205818
infected_number:  43615
---------subtree-heavy----------
save_number:  222704
infected_number:  26729
```

(b) testCase2.txt

```
---------degree-heavy----------
save_number:  235393
infected_number:  53860
---------subtree-heavy----------
save_number:  258622
infected_number:  30631
```

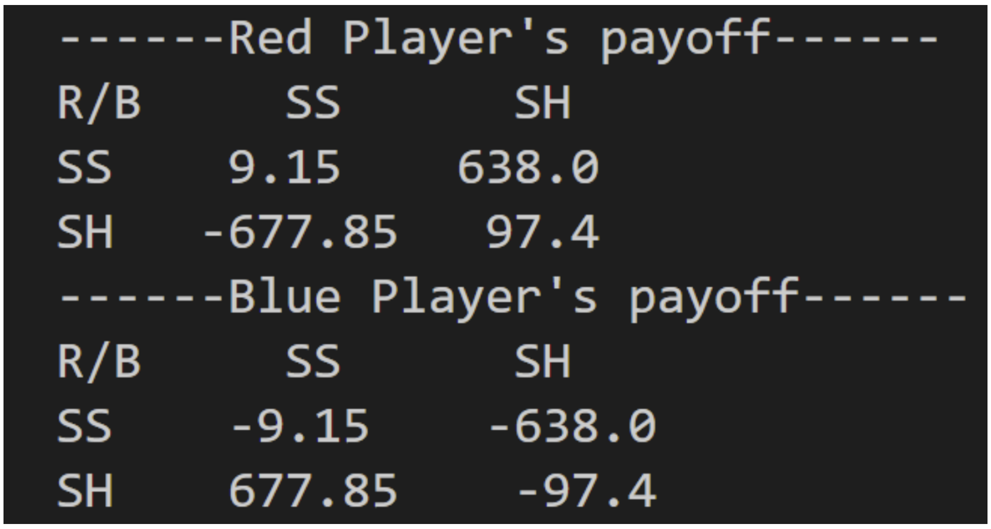
(c) testCase3.txt

Figure 2: Comparison between degree-heavy and subtree-heavy strategies

7 Problem 7

7.1

The average payoff matrix is shown as below. Each element in the matrix is the average result of 20 experiments (red plays first and blue play first for all 10 testCase.txt).



```
------Red Player's payoff------
R/B      SS       SH
SS    9.15    638.0
SH   -677.85   97.4
------Blue Player's payoff------
R/B      SS       SH
SS    -9.15    -638.0
SH    677.85    -97.4
```

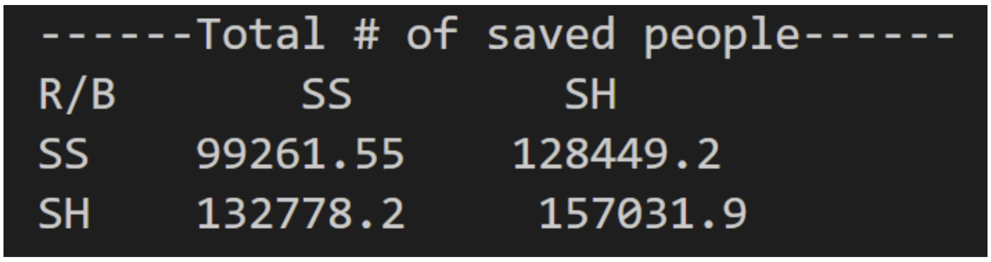
Figure 3: Average payoff matrices

7.2

From the average payoff matrix, we can find that only when $i = j =$ SS, there is a Nash equilibrium.

7.3

The average number of saved individuals are shown below. We can find that more people are saved if both i and j using SH strategy than using SS strategy.



```
------Total # of saved people------
R/B        SS         SH
SS    99261.55    128449.2
SH    132778.2     157031.9
```

Figure 4: Average matrix of total number of saved people

8 Problem 8

80 = 4 * 2 * 10 experiments are conducted. (For each of the four strategy-pair, red plays first and blue play first for all 10 testCasei.txt)