Infection Propagation Problem

Akshit Arora
akshit.arora@colorado.edu

Prashil Bhimani
prashil.bhimani@colorado.edu

1 Abstract

Cyber-attacks are increasing in the real-world and cause widespread damage to cyber-infrastructure and loss of information. It is not always economically possible to secure every single computer in the network. Hence, developing strategies in the event of an attack has always been traditionally one of the common interests of researchers in this area. We implemented a variant of minimax algorithm with pruning to produce strategies that minimize the number of nodes infected given one of the nodes in the network is compromised. We propose a new kind of pruning that can be applied with alpha-beta pruning and it performs significantly better than vanilla minimax algorithm.

2 Introduction

The increasing reliance of our information age industries, governments, and economies on cyber infrastructure makes them more and more vulnerable. In their most disruptive form, cyber-attacks target the enterprise, military and government, or other infrastructural resources of a country and its citizens [1]. According to Trustwaves 2015 Global Security Report, 98% of tested web applications were found vulnerable to cyber-attack. Based on the Department of Business, Innovation & Skills 2015 security survey 90% of the big organization and 74% of the small organization suffered from security breaches [2].

It is not always possible to secure all the computers in a network 100%. Therefore, it is of general interest to be able to devise strategies to selectively secure computers depending on the infection spread in the network. There are several possibilities in such a scenario, but for the scope of this paper, we restrict ourselves to securing one node at a time and we also assume that infection propagates one node at a time.

3 Approach

3.1 Problem Definition

Consider a network of connected computers. Infection starts with 1 computer and can propagate only if there is a connection between the 2 computers. Infection cannot propagate via a safe computer. Defender can mark any one computer as a safe computer by shutting it down. The infection ends when no more nodes can be infected or the protector cannot mark any more computers as safe. Goal is to minimize the the number of infected nodes. Defender can defend any 1 node in the network that, unlike attacker that can only propagate to only one of the nodes adjacent to already infected nodes.

3.2 Problem Formulation

The problem is formulated as a 2 player graph game. Each computer is a node and each connection is an edge. The infection can choose any one node to infect from the neighbors of the already infected nodes which are not safe. The protector can choose any one node which is neither safe or infected to save. Each player will select only one node at a time

3.3 Data

We generated random directed acyclic graphs using NetworkX library for python with edge/node ratios 1 through 5 and number of nodes varying from 10 to 50. Every node in the graph has an attribute 'infected' whose value is 1 if it is safe, -1 if it is infected and 0 (by default) indicating untouched. We assume that infection can only propagate to one untouched node at a time that is adjacent to an already infected node and cannot propagate through a safe node. Before starting the minimax algorithm, one of the nodes in the network is marked infected (i.e. 'infected' = -1) and the remaining nodes have 'infected' attribute set to 0.

We used this instance of network from NetworkX library, make a game tree out of it and then feed it to minimax algorithm. The initial values of alpha used is: -1000000000; and beta used is: 1000000.

3.4 Algorithm

Greedy Approach

This approach is quick but not optimal. In the worst case it will end up with N/2 infected nodes, where N is the total number of nodes in the graph.

Node Ranking

This approach involves assigning a heuristic rank to all the nodes in the graph and save the node with highest rank first, then the second highest, and so on. This is not an optimal approach since it does not take into consideration the start point and the current status of infection. It will always produce the same strategy.

Minimax Algorithm

Minimax is a recursive algorithm which is used to choose an optimal move for a player assuming that the other player is also playing optimally. It is used in games such as tic-tac-toe, go, chess, checkers, and many other two-player games. Such games are called games of perfect information because it is possible to see all the possible moves of a particular game. There can be two-player games which are not of perfect information such as Scrabble because the opponent's move cannot be predicted.

We formulate this problem as a two-player game and we implemented a vanilla minimax algorithm in which attacker (or the infection) is trying to maximize and defender (or network administrator) is trying to minimize the total number of nodes infected in the network. The game ends when there no more nodes available to be infected or when all the infected nodes are surrounded by only safe nodes. The heuristic value of leaf nodes is equal to the depth of leaf nodes since the depth is an indicator of how many nodes have been marked safe/infected (or how many turns have taken place in the game so far). It is reasonable to assume that the attacker is trying to maximize (in order to infect more and more nodes) this value while the defender is trying to minimize (in order to stop the infection from propagating further) this value.

Minimax Algorithm with Alpha-Beta (A-B) Pruning

When we try to scale up the minimax algorithm to apply it on larger graphs, it generates huge game trees and goes to every possible leaf node. Instead with A-B pruning, the performance of the nave minimax algorithm may be improved dramatically, without affecting the result. The algorithm works with two values, alpha and beta,

Edge/Node ratio	A-B and BFS pruning	BFS pruning	A-B Pruning
1	455.64	281.25	5.32
2	16.96	1	16.96

Table 1: Average reduction in number of nodes explored using different pruning techniques

which represent the minimum score that the maximizing player is assured of and the maximum score that the minimizing player is assured of respectively. Initially alpha is negative infinity (we used -1000000000) and beta is positive infinity (we used 1000000), i.e. both players start with their worst possible score. It can happen that when choosing a certain branch of a certain node the maximum score that the minimizing player is assured of becomes less than the minimum score that the maximizing player is assured of (beta ¡= alpha). If this is the case, the parent node should not choose this node, because it will make the score for the parent node worse. Therefore, the other branches of the node do not have to be explored.

BFS Pruning

From the start of the infection the infection will reach a particular node in at least n steps where n is its shortest distance from that node If there are K nodes at depth M and if K $<$ M we do not need to consider the nodes after depth M.

3.5 Complexity Analysis

This problem resembles with Generalized Geography problem which is a PSPACE-Complete problem. Only chooses one path Back edges can be represented as the leaf nodes of MiniMax Tree.

3.6 Results and Discussion

We used the average of the ratio of number of nodes explored by the algorithm and total number of nodes in the graph to evaluate the different pruning techniques discussed above. The BFS pruning and A-B pruning perform significantly better than other pruning techniques when Edge/Node ratio is 1, as shown in Table 1. However, when Edge-Node ratio is 2, it performs just as good as A-B pruning applied independently.

Note: The code related to this project can be found in this GitHub repository: https://github.com/aroraakshit/algorithms_project

The current problem definition is very limited in scope as to the speed with which the infection is spreading and that the infection can only add one node at a time. The proposed algorithms works well for sparse graph but the efficiency decreases as the graph becomes more dense (more edge/node ratio). Approximate approaches may not give the optimal solution but will definitely may be a lot quicker.

4 Future work

Here are some ideas that are yet to be explored in this area:

- As presented here, our algorithm used two different approaches to prune the game tree. However better pruning can be achieved by using other techniques e.g. iterative deepening.

- In addition, a follow up of this work would be to scale up the graphs and observe the behavior of our proposed BFS pruning technique and check its applicability on larger networks.

- Real world networks can be cyclic, therefore this algorithm needs modifications so it can perform better with cyclic graphs as well.

- We assume that there is only one infected node in the beginning, however there can be more than one nodes infected at different locations in the network.

- Right now, the infection can propagate to only one adjacent node at a time, however this can be changed to more than one adjacent nodes too.

- The concept of defending the network using honey-pot servers can also be used to trap the attacker by faking an infection. Strategies can be modified accordingly. [1]

- Besides an approach like Minimax algorithm, we can also try out Reinforcement learning algorithm on this kind of two-player game.

Ideally, we would like to identify loopholes in the BFS pruning technique and therefore try to combine it with other techniques that solve those loopholes.

Acknowledgments

The authors wish to thank Dr. Ashutosh Trivedi.

References

[1] P. Aggarwal, C. Gonzalez, and V. Dutt. Cyber-security: Role of deception in cyber-attack detection. In *Advances in Intelligent Systems and Computing*, vol. 501, pp. 85–96, 2016. doi: 10.1007/978-3-319-41932-9_8

[2] Trustwave. 2016 Trustwave Global Security Report. Technical report, 2016.