An Optimal Routing Policy in Networks using Q-Learning

Mohan Sai Ambati
Department of Computer Science, University of Nebraska at Omaha.

Abstract

The main feature of nodes in Networks is by cooperating with neighbors to propagate data. Misusing this feature, malicious nodes cooperate with normal nodes to disrupt network operation and reduce its efficiency. One of the effective ways of detecting malicious nodes and find the optimal routing policy is by learning the network. Focusing on this issue, in this paper, an approach is proposed to learn the network and predict the behavior of the neighbors using Q-Learning [9]. A learning module is embedded into each node of the network and each node learns the routing policy by minimizing the number of hops a packet will take in all possible routes available by predicting the behavior of the neighboring nodes and avoiding the malicious nodes. It does this by experimenting with different routing policies and gathering the statistics about which decisions minimizes the number of hops by avoiding the malicious nodes. The learning is continual and online, uses only local information, and is robust in the face of irregular and dynamically changing networks.

Keywords*:* Network security, malicious node, q-learning, game theory

1. INTRODUCTION

Network is the collection of nodes communicating with each other. Nodes in the network do everything such as routing. This means nodes are responsible for the routing, sending data, receiving data, network security controlling, recognizing malicious nodes, and other factors discussed in network security issues. This issue implies the important role of securing each node. Nodes of the networks are classified into two categories “malicious” and normal. The *normal nodes* perform their routing activities in the network without any destruction. The *malicious nodes* that have destructive goals in the network tries to maximize disturbing the network performances and efficiency, to deceive the normal nodes, to ignore, not sent, or eliminate the packets received from neighbors and to waste the energy of the other nodes. A malicious node adopts the following actions: (1) It cooperates with normal nodes to gain their trust; (2) It attacks to degrade the network performance or achieve other destructive goals; (3) It attempts to escape before being detected; (4) Enters the network as a new comer [7]. Identifying of malicious nodes plays an important role in finding the optimal routing policy. This paper proposes a Q-Learning based routing algorithm by learning the routing policy by detecting the malicious nodes and avoiding them and balances minimizing the number of hops a packet will take with the congestion along other possible routes. It does this by experimenting with different routing policies and gathering statistics about which decisions minimize total number of hops. The learning is continual and online, uses only local information, and is robust in the face of irregular and dynamic networks.

2. ROUTING AS A Q-LEARNING TASK

A Routing policy answers the question: to which adjacent node should the current node send its packet to get it quickly as possible to its eventual destination? Let $Q_x(d,y)$ be the confidence that node x estimates that it delivers packet P to node y by passing it to neighbor node d [1]. When node x pass the packet P to node d it will in response get a reward for its action as follows:

$$R_x(d,y) = 100/\delta$$

Where δ = Number of hops required for packet P to travel from node d to node y.

The information of number of hops is obtained by the Manhattan distance matrix given to algorithm. By getting the reward node x updates it confidence Q-value as follows:

$$Q_x(d,y) = Q_x(d,y) + n*(\max(Q_d(e,y) + R_x(d,y)))$$

Where, n = learning rate, which is set to 1.

3. MALICIOUS NODE DETECTION

By Q-Learning algorithm a node always selects a neighbor which has highest confidence to reach the destination. But there could be few nodes in the middle which falsify the confidence values or doesn't forward the packet in the network. In order to predict the behavior of the neighboring nodes normal node plays a game with the neighboring node that claims highest confidence, by the end of game normal node obtains optimal decision whether to forward or not forward packet to that particular neighbor. The approach proposed in this paper is based on Bayesian game theory that tries to conserve energy [7][11]. Bayesian game [8] refers to a game, with incomplete information and different types of players. In this game, each player's action depends on the opponents' previous action. The selected action is based on using a possibility value calculated at each step. In this paper, the interaction between two nodes, which one of them is normal and another one type is unknown, is modeled as a game. Due to certain conditions of Bayesian game, and it's compatibility with the presented Q-Learning algorithm we used it in this paper. The sending and receiving results and neighbor's responses to inquiries are saved in nodes' recording table and used in the next decision-making. These interactions are continued to be stored at appropriate times to be used to identify malicious nodes.

- **Number of Players:** The game is considered between two nodes of the network. One of these nodes could be a malicious node and the second node is assumed to be normal node.
- **Non-cooperative:** The goal is recognizing, punishing or avoiding malicious nodes. The nodes are also not aware of the existence and location of malicious nodes. Therefore, the game is non-cooperative.
- **Incomplete Information:** Obviously, the information is incomplete because nodes are not known about the destructive nature of a node [7].

The main idea is to predict the behavior of the neighboring node, initially normal node only the probability of a neighbor to be malicious. By that it models the game as follows:

<table>
<tr><td>NodeX/NodeY</td><td colspan="2">Node with highest confidence to reach destination (NodeY)</td></tr>
<tr><td>Normal Node which analysis to share or not
(NodeX)</td><td>NodeX thinks
P(NodeY == Normal) = P
<table><tr><td></td><td>Share</td><td>Not Share</td></tr><tr><td>Share</td><td>S,S</td><td>N,N</td></tr><tr><td>Not Share</td><td>0,0</td><td>0,0</td></tr></table></td><td>NodeX thinks
P(NodeY == Malicious) = 1-P
<table><tr><td></td><td>Share</td><td>Not Share</td></tr><tr><td>Share</td><td>S,N</td><td>N,S</td></tr><tr><td>Not Share</td><td>0,0</td><td>0,0</td></tr></table></td></tr>
</table>

Fig1. Bayesian Game played by NodeX in order to predict the behavior of NodeY. Where S >N

Probability of NodeY being Normal = P, which is obtained by history of the NodeY. If NodeY is normal then it shares the packet if NodeX shares the packet else it doesn't share the packet if NodeX doesn't share the packet. If NodeY is malicious it doesn't shares the packet if NodeX share. NodeX repeats the game n times in order get the optimal decision. So this behavior is modeled in Fig 1.
Considering S = 2 and N = -2 and calculating the payoffs for the Bayesian game as follows:

NodeX/NodeY	Share-Share	Share-Not Share	Not Share-Share	Not Share-Not Share
Share	2,2P-2(1-P)	2p-2(1-p),2	-2p+2(1-p),-2	-2,-2p+2(1-p)
Not Share	0,0	0,0	0,0	0,0

Fig 2. Final Payoff matrix computed by NodeX.

So always NodeX tries to compute the Nash Equilibrium, if P >= 0.5 then there will possibility that NodeX Share will be the equilibrium decision, but if P < 0.5 then NodeX Not Share will be the optimal decision. So NodeX follows the outcome of this game. If P < 0.3 then NodeY is marked as malicious and NodeX will try to avoid NodeY.

4. HIGH LEVEL OVERVIEW OF ALGORITHM

While learning does not converge **do**
Select normal neighbor node with max Q-value, otherwise select a random neighbor which is normal.
Find the p value from the past transactions, otherwise select p = ½.
Decide to share or notshare the packet depending upon Bayesian game.
If Node is Normal
Send packet to the selected neighbor.
Receive Reward R.
Update Q(present node) = Q(present node) + max(p*(Q(neighbor) + R))
Else
Skip the Node and mark it malicious
End

5. TIME COMPLEXITY

Suppose network with n nodes could have at least k neighbors and assume that each node plays g games with neighbor during the predicting the behavior of neighbor. Therefore, the time complexity of each node to select its neighbor is

(Number of iterations to learn the confidence)*(Number of Node's neighbors)*(Number of games played with neighbors)*(Time complexity of sending the packet) = O(n)*O(k)*O(g)*O(logn) = O(nkglog n)

6. SPACE COMPLEXITY

We assume that network contains n nodes, and each node has at most k neighbors. Each node needs to accumulate its neighbor's information. In this condition, a number or a string is presumed as storage unit. Each node should keep the few variable for each neighbor, which take a space in O(k), and each nodes maintains a routing table that contains the confidence values of n nodes which is of O(n).
Therefore, the total time complexity is = O(k+n)

7. RESULTS

We tested the algorithm on a 6X6 irregular grid shown in Fig 3. This algorithm successfully learns the network, adapts to the changes in network and learns the optimal routing policy by predicting the malicious behavior of the neighbors and avoiding malicious nodes while routing the packet.

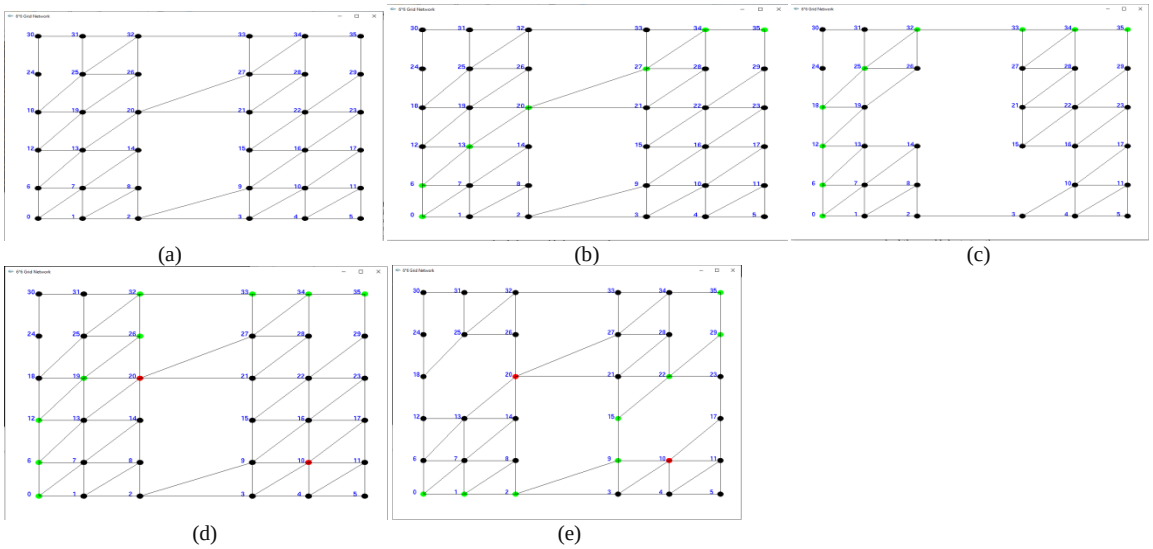

Fig 3. (a) 6X6 irregular network. (b) Nodes highlighted in green shows the optimal routing policy when there are no malicious nodes. (c) Adapting the network when there are no malicious nodes and finding the optimal routing policy. (d) Nodes highlighted in red are the malicious nodes detected by the algorithm and finds the optimal routing policy by avoiding them. (e) Adapting the network and avoiding the malicious nodes in the routing path.

8. REFERENCES

[1] Packet Routing in Dynamically Changing Networks: A Reinforcement Learning Approach
Justin A. Boyan, school of computer science, Carnegine Mellon University, Pittsburgh, PA 15213, Michael L. Littman, Cognitive Science Research Group, Bellcore, Morristown, NJ, 07962.

[3] Artificial Intelligence: A Modern Approach Textbook by Peter Norvig and Stuart J. Russell

[4] Computer Graphics using OpenGL (3rd Edition) by Francis S Hill Jr, Stephen M Kelley

[5] https://www.opengl.org/resources/

[6] A Game Theory Based Approach to the Generation of Optimal DDoS Defending Strategy
Liang Huang, Dengguo Feng, Yifeng Lian, Yingjun Zhang, and Yuling Liu,Trusted Computing and Information Assurance Laboratory, Institute of Software, The Chinese Academy of Sciences, Beijing, China{lancerhuang, feng, lianyf, yjzhang, ylliu}@tca.iscas.ac.cn

[7] A Game Theory Approach for Malicious Node Detection in MANETs YASER TAHERI1, HOSSEIN GHARAEE GARAKANI2 AND NASER MOHAMMADZADEH1 1Department of Computer Engineering Shahed University Tehran, 3319118651 Iran 2Department of Network Security and Information Technology Research Center Tehran, 1439955471 Iran E-mail: {y.taheri; mohammadzadeh}@shahed.ac.ir; gharaee@itrc.ac.ir

[8] Games of Incomplete Information Jonathan Levin February 2002

[9] Technical Note Q,-Learning CHRISTOPHER J.C.H. WATKINS 25b Framfield Road, Highbury, London N5 IUU, England PETER DAYAN Centre for Cognitive Science, University of Edinburgh, 2 Buccleuch Place, Edinburgh EH8 9EH, Scotland

[10] Playing Atari with Deep Reinforcement Learning Volodymyr Mnih Koray Kavukcuoglu David Silver Alex Graves Ioannis Antonoglou Daan Wierstra Martin Riedmiller DeepMind Technologies {vlad,koray,david,alex.graves,ioannis,daan,martin.riedmiller} @ deepmind.com

[11] Game Strategies in Network Security Kong-wei Lye 1 Jeannette Wing2 May 2002 CMU-CS-02-136

[12] Applying Game Theory to Analyze Attacks and Defenses in Virtual Coordinate Systems
Sheila Becker, Jeff Seibert, David Zage, Cristina Nita-Rotaru and Radu State Purdue University West Lafayette, IN 47906, USA Email: {jcseiber, zagedj, crisn}@cs.purdue.edu

[13] A Game Theoretical Approach to Communication Security
Assane Gueye EECS Department University of California, Berkeley Technical Report No. UCB/EECS-2011-19 March 14, 2011