Selfish Dynamic Punishment Scheme: Misbehavior Detection in MANETs Using Cooperative Repeated Game

Ashraf Al Sharah[1], Mohammad Alhaj[2], Mohammad Hassan[2],

[1]Electronic and Communication Engineering,
[2]Computer Engineering Faculty of Engineering, Al-Ahliyya Amman University,
Jordan, Amman, Jordan

Summary
A Mobile Ad Hoc Network is a self-organized and infrastructure-less and dynamic network. In this type of networks, nodes organize themselves in a non-centralized form where additional effort is applied on network members. Hence, nodes in the network attempt to maximize its own benefits and saves its own resources in a form called rational behavior. Thus, selfish behavior arises as a problem that may cause a severe fault for the network and highly affects the network functionalities and performance. There is a need to bring a suitable punishment method for this kind of behaviors. We propose a slave mode selfish dynamic punishment scheme, using cooperative repeated game to avoid selfish behavior in Mobile Ad Hoc Network, and motivate selfish nodes to cooperate. The scheme is used to pertain a cooperative punishment from all network nodes, to exhaust the punished node which can stimulate this node to cooperate with other members.

Key words:
MANETs, Selfishness behavior, Cooperative game, Misbehaving, Punishment.

1. Introduction

The Mobile Ad Hoc Networks (MANETs) are a group of autonomous mobile nodes that are connected through wireless links without any fixed infrastructure with a dynamic network topology. MANETs are self-organized networks that require distributed, reliable, and flexible networks to provide interdependency and rational decision making. Nodes in MANETs must carry out all network functions by themselves that would be reflected on the willingness of cooperation between them, without concentrating on network resources for the sake of maximum payoffs, and saving their own resources. This kind of selfish behavior puts more stress on all nodes to maintain the network performance and would cause a critical problem for the MANET networks.

The selfish behavior is considered as one of the challenges that directly affects the ad hoc network performance and its resources and leads to less cooperation among network members. It can be caused due to either the selfish behavior nature of the nodes regardless of their residual energy and resources; or due to resource constraints such as power concentration and data processing [1] [2]. However, there is a significant difference between selfish nodes and malicious nodes of an ad hoc network. The malicious node aims to collapse and crash the network functions by applying different attacks or schemes on a network; while the selfish nodes misbehave to save their own resource privileges. There is a need to adapt a punishment strategy for promoting the cooperation among network nodes that ensures the best network performance. The selfish nodes detection and punishment methods and schemes solve such kind of problems and correct their behavior that leads to lower the unnecessary loss of resources and reduce the network delays.

Due to the nature of interaction between nodes in an ad hoc network, using the game theory would be a suitable choice as a research topic to solve the selfish behavior facing MANETs. Game theory can provide an analytical framework by using helpful mathematical tools to study the complex interactions among rational players in the game. Games are categorized as cooperative and non-cooperative games; a cooperative game is played between nodes that have mutual relationship with each other, while the non-cooperative game is played between nodes that do not seem to coexist mutually [3]. Cooperative game theory provides useful method to analyze and study the rational behavior for the network members when they cooperate [4][5][6]. Cooperative game can be played in different ways, but the most intuitive way is when players form groups, which defined as a coalition game.

In this paper we propose a dynamic punishment scheme to curb rational behavior occurred by selfish nodes, motivating them through slave mode punishment, by imposition their own resources for a certain time which will lead to improve network performance. The rest of this paper is organized as follows; in Section 2 presents the background and related; in Section 3 we presents the network model; Section 4 describes the proposed punishment scheme; in Section 5 presents the simulation

Manuscript received March 5, 2020
Manuscript revised March 20, 2020

and result of the dynamic punishment scheme, and finally in Section 6 the conclusion and future work.

2. Background and Related Work

MANET is a decentralized type of networks where each node contributes in forwarding data to the neighbor nodes. MANET does not operate on the common mobile communication infrastructure such as routers or access points, but rather the data is forwarded dynamically through the nodes based on the defined network connectivity and the routing algorithm. The behavior of MANETs can be simulated using a common simulator called Network Simulator NS-2 [7]. NS-2 provides a discrete event simulating for TCP, routing, and multicast protocols over wired and wireless networks. It is an open source simulation tool used for research and development.

Several approaches and mechanisms have been followed recently to protect MANETs against their selfish behavior. Jijeesh, and Prajapati in [8] present the watchdog mechanism that detects the misbehaved and selfish members in the networks. Watchdog works as each node monitors its own neighboring nodes to identify the misbehaving nodes by keeping a buffer that holds lately sent packets. When a sending node has forwarded a packet to the next hop node, it checks whether the next hop node in the path will forward this packet or not. If the node does not forward the packet, then that node will be considered as selfish. Then the watchdog will mitigate such selfish nodes from the network. Dais, and Haroon in [9] present the pathrater mechanism to avoid transmitting packets over selfish nodes. Pathrater is used for selecting the reliable path, which is running in and calculated for each network node. The path metric is calculated by averaging the node ratings in the path, and the path with the highest metric will be selected as the most reliable path to avoid paths that include selfish nodes. While Wankhade in [10] introduces the 2ACK-scheme used by routing protocols, for example OLSR. It uses two packets as an additional acknowledgement packets called 2ACK to ensure that the data have been successfully received by the destination. 2ACK maintains a predefined route of two nodes in the reverse direction to the data packet, If the sending node doesn't receive the two-ack, it will assume that the next-hop forwarding link is misbehaving. Senthilkumar and William in [11] used the CONFIDANT mechanism that is closely similar to watchdog. The mechanism contains four components which are monitoring system, reputation system, trust manager and path manager. The mechanism monitors the neighboring node behavior and record the misbehavior to the Reputation system to take an action. When a specific node reaches the threshold value (of what) the path manager will take an action(what actions), and the trust manager will warn other nodes about that misbehaving node. Giannis et.al. in [12] presents the CORE mechanism where each network member in CORE keeps track of other member's collaboration using reputation technique. The reputation metric is calculated based on data observed by the local node, and information collected by other nodes involved in network operations. Simply nodes with a good reputation can use the network resources while nodes with a bad reputation can't. The main key is how the nodes cooperate with others. Also, Rama, and Sumithra in [13] uses the Credit-based Mechanism which focuses in providing a credit to the selfish nodes, to encourage and motivate them to cooperate. Credit-based is an incentive mechanism that uses credits to charge the members that send packets, and to reward those members which are transmitting packets. This can motivate the selfish nodes to transmit packets to earn credits, applying fairness by recompensing credits to the members that transmit more packets. In summary, our proposed mechanism differs from the previous mechanisms in a way that uses the cooperative game theory to detect and monitor the selfish behavior node in MANETs. Also, it proposes a punishment scheme to the misbehaved nodes.

3. The Proposed System Model

In this section, we describe the arithmetic semantics of our system model. In order to model our proposed system, the following notations are used:

- S defines the number of nodes in the coalition.
- i defines any given node in the coalition.
- N defines the neighboring nodes.
- M defines the number of misbehaving actions.
- O defines the number of observations.
- N_i defines the neighboring nodes for any given node.
- M_i defines the misbehavior table for any given node.
- $O(N)$ defines the observations by neighboring nodes.
- $O_i(t)$ defines the observations for any give node at time t.
- T defines the total time for a node that spends in the coalition.
- $M_i(T)$ defines the total misbehaving actions for any given node over total time.
- $N_i(t)$ defines the number of neighboring nodes for any given node at time t.
- α and β defines the weight parameters for the punishment function.

We consider a model for the system as a coalition game with imperfect information. This game will be repeated at each iteration. The model will consist of (1, 2, . . . , N) numbers of normal nodes and S (0, 1, . . . , (N/2) − 1) numbers of selfish nodes, where the number of selfish nodes would not exceed the number of legitimate nodes. Any node would be able to join the coalition because it acts like a regular node at the beginning, which permits it to become a member of the coalition. On joining the coalition, a new node has a number of neighboring nodes which would start watching and monitoring the other nodes behavior to find out their willingness to share their own resources with their neighbors. When nodes refuse to share their resources, the action will be recorded from its own neighbors. A node who refuse to cooperate for a set number of iterations will be tagged as a selfish node and the punishment will be assign for this specific node from the coalition.

Our system model consists of the legitimate grand coalition which will include selfish nodes. The legitimate grand coalition is designed to detect insider selfish nodes by counting the number of misbehaviors (Mi). However, the legitimate coalition can also apply the punishment model for any given number of nodes. Nodes in the coalition rely heavily on a stored misbehaving actions for all neighboring nodes. This mechanism will depend on storing observation actions in to misbehaving table (Mi) define misbehaving nodes, which was a former legitimate node. The nodes will form a table in order to make a strategic security defense decision, to maintain the coalition by building and updating misbehaving table according to the misbehaving actions for all nodes and then punish any misbehaving node that has a misbehaving value below the threshold value. Each node will have a misbehaving table for all neighboring nodes.

$$M_i=\sum O_i(t) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(1)$$

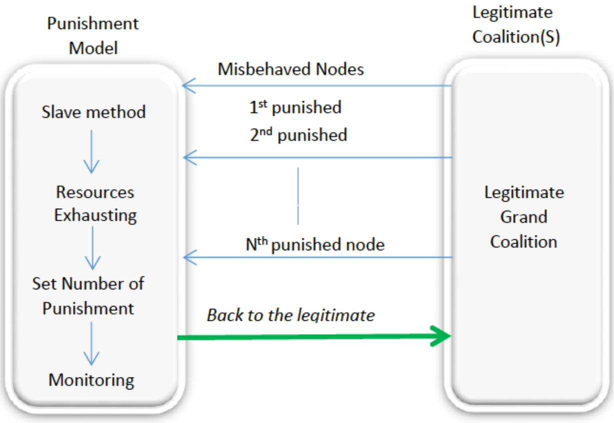


Fig. 1 System Model.

4. The Proposed System Model

Punishment model will depend on the first-hand information, which means that only neighbors who have a direct connection with a specific node can testify the degree of cooperation for that node and only these nodes can update them misbehaving table. The punishment depends on two main parameters: number neighboring nodes (N) and number of observations (O), and each parameter is assigned by a given variable factor where the sum of these factors is less than or equal one.

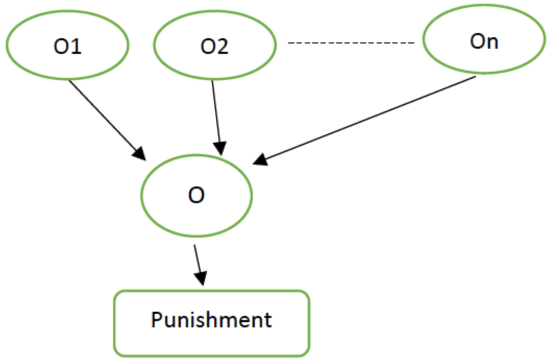


Fig. 2 Observation graph.

Neighboring nodes are node who can testify this node actions Ni, and the observations are the misbehaviors the observed by neighboring nodes (Oi). Where Neighboring nodes are assigned by any given time slot t is:

$$N_i(t)= O_i - 1 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2)$$

And Where observations made by nodes are assigned by any given time slot t is:

$$O_i(t)= max\ M_i(T) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3)$$

Incorporating these two parameters, we can write the punishment function by weighing each parameter. The punishment function proposed is then:

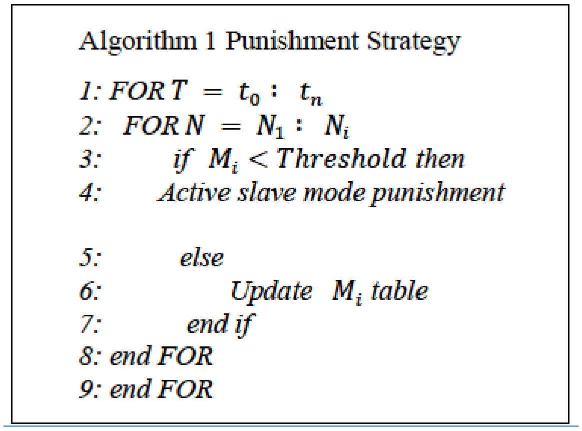
$$C(t)= \alpha(N_i(t)) + \beta(O_i(t)) \quad \ldots\ldots\ldots\ldots\ldots\ldots(4)$$

where α and β are weight parameters and $\alpha + \beta = 1$
These weight parameters can be used to help provide variability for the punishment function of the nodes, where α helps to weigh the support parameter that is responsible for the number of neighbors of a node, and β provides a weight value for the number of observations made by neighbors. For example, if we need to depend more on testified node we increase the value of α factor, and if we need to depend at the number of observations we increase the value of β factor, in this way we can give the system more flexibility for each individual case.

4.1 The Punishment Strategy

The main goal of our punishment strategy is to ensure that the nodes will start cooperating again and not causing any further problem for the coalition, also focus on promoting and motivating nodes rather than just excluding them. When the node stops cooperating according to the neighbor's nodes, then an alarm will be triggered all over the coalition and enter that node into punishment state. The punishment strategy depends on four main step and it is described in algorithm1 below:

- Slaved mode, where all neighbors will use this node to forward or exchange data with other nodes.
- Resource exhausting, where the punished node resources is used in a greedy way to rehabilitate its own behavior.
- Set number of punishment iteration, where the number of iterations stay at the slave mode to ensure the rehabilitation of the selfish node.
- Monitoring, where the coalition keeps watching the selfish node, to monitor its act before it is considered a legitimate node.



Algorithm 1 Punishment Strategy

```
FOR T $= t_0 : t_n$
  FOR N $= N_1 : N_i$
      if $M_i <$ Threshold then
     Active slave mode punishment

      else
          Update $M_i$ table
      end if
end FOR
end FOR
```

5. Simulation and Evaluation of the Results

In this section we present the results of our simulations. We implemented our approach using the network simulator NS-2. We will focus on how our approach can make a different in solving selfish issues in MANETs and shows the benefits of coalition game in detecting and promote selfish nodes to cooperate again. Furthermore, we show the number of selfish node before and after applying our scheme. Selfish impact on network performance will be shown by applying a various numbers of selfish nodes. There would also be a comparison between the Number of detecting selfish nodes with different coalition size. The system delay will be presented before and after applying our rehabilitation or punishment scheme. Table 1 describes the parameters of our simulated case study.

Table 1: Parameters for Simulation

Parameter	Level
Area	2300 × 1300
Speed	15 m/s
Radio range	250m
MAC	802.11
Simulation time	110 s
Number of mobile nodes	80
Network interface type	Wireless
Channel type	Wireless channel
Transmission rate	1–11Mbps

Figures 3, 4, 5, and 6 illustrate the results of our simulated case study. Figure 3 shows the different numbers of selfish nodes before and after applying the punishment scheme over the time. It is obvious that the numbers of selfish nodes decrease after applying our scheme because the impact of selfishness for a given node to its own resources is less than our punishment scheme.

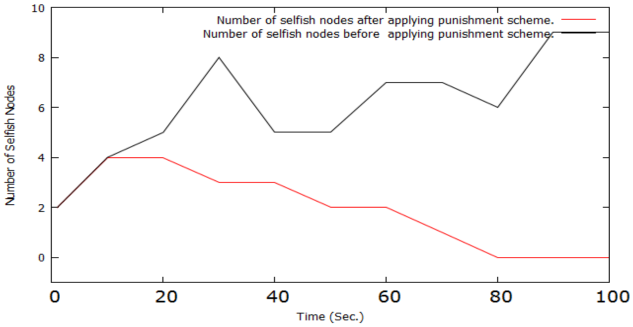


Fig. 3 Number of selfish nodes before and after applying punishment scheme.

Figure 4.a and 4.b illustrates the selfishness impact at the network performance before and after applying our scheme. The network performance is better after applying our scheme because the nodes can see what happed to other nodes that did behave in a selfish way so they know the impact of the punishment scheme, also the maximum number of nodes decrees after applying the punishment scheme.

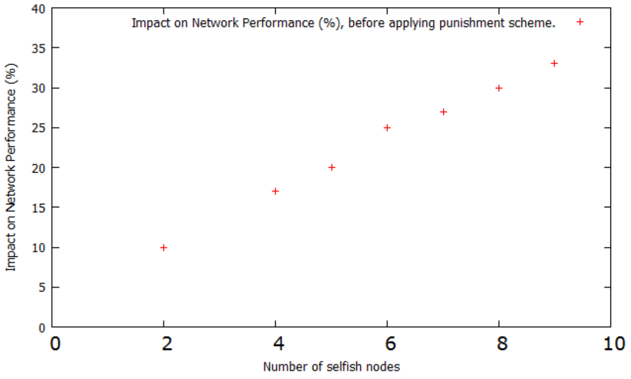


Fig. 4.a Network performance before applying our scheme.

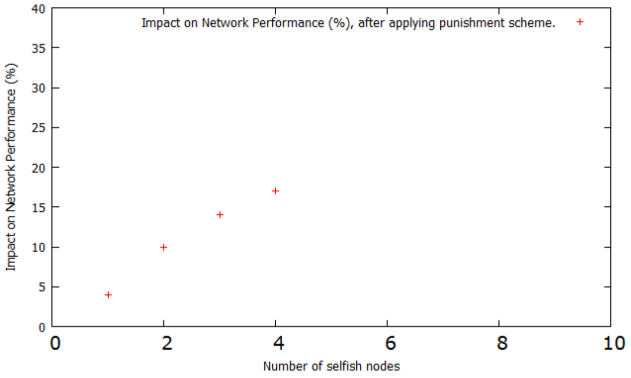


Fig. 4.b Network performance after applying our scheme.

Figure 5 shows the number of detecting selfish node using the coalition game with different coalition sizes. As we see as more nodes we have at the coalition, selfish nodes detection will be faster because more observations can be recorded, this leads to affect the network performance in a positive way.

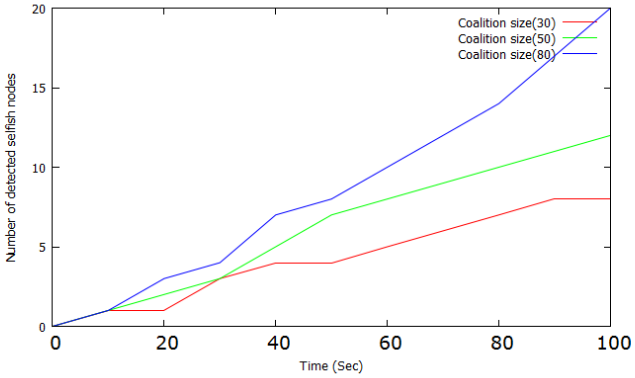


Fig. 5 Detecting selfish nodes with different coalition sizes.

Figure 6 shows the system delay over the time. It is obvious that delay is reduced significantly over the time. The delay is seen to be improved as the coalition cooperate in detecting selfish nodes.

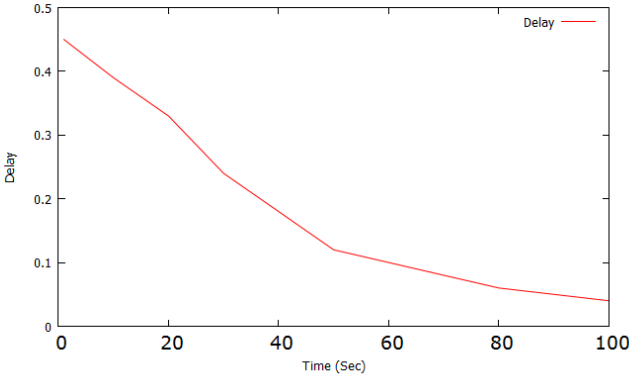


Fig. 6 System delay.

6. Conclusion

In the paper, we proposed a mechanism that concerns in rehabilitation rather than excluding the selfish node by tracking them in the network. The proposed mechanism is based on the game theory where node coalition are used to monitor the selfish behavior and uses a punishment strategy to reduce the effect of selfish nodes on the MANET resources. As a result of our proposed mechanism, we illustrates through simulation that the mechanism detects and decreases selfish nodes in a mobile ad hoc network. Also, the number of selfish nodes reduced and the performance of MANET has been improved. As a future work, we are planning to test our model in different game theory schemes and make a comparison between them.

Acknowledgments

The authors would like to thank the Faculty of Engineering at Al-Ahliyya Amman University for providing us the necessary tools to carry out this research successfully.

References

[1] Lei T., Wang S., Li J., You I., and Yang F. (2016) Detecting and preventing selfish behaviour in mobile ad hoc network. The Journal of Supercomputing 72, no. 8: 3156-3168.

[2] Tarag F., and Askwith R. (2006) A node misbehaviour detection mechanism for mobile ad-hoc networks. In proceedings of the 7th Annual Post Graduate Symposium

on The Convergence of Telecommunications, Networking and Broadcasting, pp. 1-6.

[3] Giacomo B., Lasaulce S., Saad W., and Sanguinetti L. (2016) Game theory for networks: A tutorial on game-theoretic tools for emerging signal processing applications. IEEE Signal Processing Magazine 33, no. 1: 94-119.

[4] Al-Sharah A., Oyedare T., and Shetty S. (2016) Detecting and mitigating smart insider jamming attacks in MANETs using reputation-based coalition game. Journal of Computer Networks and Communications.

[5] Stutzman W. L., and Thiele G. A. (2013) Antenna theory and design. John Wiley & Sons.

[6] Goudarzi P. (2013) A non-cooperative quality optimization game for scalable video delivery over MANETs. Wireless networks 19, no. 5: 755-770.

[7] The Network Simulator - ns-2, https://www.isi.edu/nsnam/ns/

[8] Jijeesh B., and Prajapati J. (2014) A review paper on watchdog mechanism in wireless sensor network to eliminate false malicious node detection. Int. J. Res. Eng. Technol 3, no. 1: 381-384.

[9] Dais J., and Haroon R. P. (2014) Selfish Node Isolation & Incentivation using Progressive Thresholds" International Journal on Network Security 5, no. 1: 68.

[10] Kafi M. A., Othman J. B., and Badache N. (2017) A survey on reliability protocols in wireless sensor networks. ACM Computing Surveys (CSUR) 50, no. 2: 31.

[11] Senthilkumar S. and William J. (2014) A survey on reputation based selfish node detection techniques in mobile ad hoc network" Journal of Theoretical & Applied Information Technology 60, no. 2.

[12] Marias G. F., Georgiadis P., Flitzanis D., and Mandalas K. (2006) Cooperation enforcement schemes for MANETs: A survey. Wireless Communications and Mobile Computing 6, no. 3: 319-332.

[13] Abirami K. R., and Sumithra M. G. (2018) Evaluation of neighbor credit value based AODV routing algorithms for selfish node behavior detection. Cluster Computing: 1-10.