EURASIP Journal on Wireless Communications and Networking

RESEARCH Open Access

Selfish node detection based on hierarchical game theory in IoT

Solmaz Nobahary[1], Hossein Gharaee Garakani[2*], Ahmad Khademzadeh[2] and Amir Masoud Rahmani[1]

Abstract

Cooperation between nodes is an effective technology for network throughput in the Internet of Things. The nodes that do not cooperate with other nodes in the network are called selfish and malicious nodes. Selfish nodes use the facilities of other nodes of the network for raising their interests. But malicious nodes tend to damage the facilities of the network and abuse it. According to reviews of the previous studies, in this paper, a mechanism is proposed for detecting the selfish and malicious nodes based on reputation and game theory. The proposed method includes three phases of setup and clustering, sending data and playing the multi-person game, and update and detecting the selfish and malicious nodes. The process of setup and clustering algorithm are run in the first phase. In the second phase, the nodes of each cluster cooperate with each other in order to execute an infinite repeated game while forwarding their own or neighbor nodes' data packets. In the third phase, each node monitors the operation of its neighbor nodes for sending the data packets, and the process of cooperation is analyzed for determining the selfish or malicious nodes which forwarded the data packets with delay or even not sent them. The other nodes reduce the reputation of the nodes which does not cooperate with them, and they do not cooperate with the selfish and malicious nodes, as punishment. So, selfish and malicious nodes are stimulated to cooperate. The results of simulation suggest that the detection accuracy of the selfish and malicious nodes has been increased by an average of 12% compared with the existing methods, and the false-positive rate has been decreased by 8%.

Keywords: Internet of Things (IoT), Selfish node, Malicious node, Game theory

1 Introduction

Nowadays, the Internet of Things (IoT) is introduced as a global infrastructure to establish communication between the physical and virtual worlds via the available technologies. The IoT is an intelligent network, and things attempt to transfer information through the network equipment. Its applications impress all human life aspects, including smart cities, smart environment, smart water control, security and emergency, smart transportation, smart agriculture, industrial control, and health. Its purposes are to facilitate the works and increase the quality of life. The IoT is popular for its capability to connect different kinds of things to the virtual world, and sensing data from different detectors are sent to the center [1–4]. Due to the improvement in wireless communications, it becomes a way to send and receive the data packets in IoT. However, the low range of wireless communications makes it possible for multi-hop communications, and the life of these communications depends on the cooperation of each node [5, 6].

One of the most important challenges is the lack of cooperation in some nodes due to the connection between the things and the Internet in the IoT network when sending data in multi-hop communications [7, 8]. Such nodes are called selfish nodes. The selfish nodes use the network facilities for personal purposes and only send their own data packets but do not help to forward the other neighbor nodes' data packets to save their energy power. The other group of nodes is called malicious nodes which tend to harm and exploit the network facilities. By increasing the number of such nodes, the network throughput and lifetime will reduce, and the energy consumption, average end-to-end delay, and network traffic will increase. As a result, it will disturb the network operation [7–9].

* Correspondence: gharaee@itrc.ac.ir
[2]Iran Telecom Research Center (ITRC), Tehran, Iran
Full list of author information is available at the end of the article

To overcome the side effects of the selfish and malicious nodes, it is necessary to detect and identify them. To this purpose, different strategies have been proposed, and the reputation-based method is one of the most popular methods in which each node receives a specific reputation according to the nodes' feedback. The nodes with higher reputations are recognized by the network as more reliable nodes, and the nodes with lower reputation are known as selfish nodes. The throughput of these methods is low, the energy consumption is high, the selfish and malicious nodes can collude, there are no punishments or incentives in the selfish nodes, and no second chances are given to the selfish or malicious nodes to cooperate with other nodes [10–15]. The other group of strategies discover the selfish and malicious nodes in the credit-based methods in which the nodes should pay the cost to send the data packets and/or the nodes trade the data packet and sell it at a higher price when they have purchased a packet. Collision attack, lack of punishment, and incentives are the disadvantages of these methods [16–22] (Table 1). The acknowledgment-based methods guarantee to send a packet of a node by using an acknowledgment message. In these methods, the throughput is low; it suffers from the collusion of the selfish and malicious nodes; it has high overhead (communicative, data packet, etc.) and end-to-end delay increases in the network due to the high traffic generated by the acknowledgment messages [23–26]. Another method to detect selfish nodes is game theory-based methods. The game theory is an applied mathematical theory that models and analyzes the systems where each individual attempts to find the best strategy selected by others to reach success [27]. Game theory-based approaches take advantage of the incentive mechanisms through payoffs and have lower false-positive rate and overhead (time, supervisory, hardware) compared to the other aforementioned methods in the network to detect the selfish and malicious nodes. However, the prior game theory-based methods have less throughput and more end-to-end delay, and the proposed methods attempt to increase these parameters [28–31].

The proposed mechanism is a multi-step method based on reputation and game theory for the stimulation of selfish and malicious nodes in Internet of Things, and the mechanism has been designed in three steps including setup and clustering, sending data and playing the multi-person game, and update and detecting selfish and malicious nodes. In the first step, a set of things are placed in a cluster due to being communicated with the destination base station, and they choose the base station as a cluster head. In the second step, the nodes cooperate with each other to forward the data packets to the cluster head, and for this purpose, during sending and forwarding their own or the neighbor nodes' data packets, they run a multi-person game and the results of this game is sent to the next step and determine the reputation of each of the nodes. In the third phase, the reputation of each node is assigned a value in the network and updated by its neighbor node. Each of the nodes whose reputation value is less than the predetermined threshold is detected as a selfish or malicious node. In the following, we highlight the novel contributions of our paper:

- The proposed method is a hybrid method which takes advantage of reputation-based method and game theory-based method to detect and stimulate the non-cooperation nodes. The strategy assigns a reputation value earned by playing the game to all nodes. The nodes with a low reputation cannot be active and send the data packet so the node tries to earn more reputation. The nodes that want to earn reputation should cooperate with other nodes, and it means stimulation of the nodes.
- If the reputation of a node is not less than the threshold, the mentioned node is provided the opportunity of cooperating with other nodes.

Table 1 Advantages and disadvantages of systems

Systems	Advantages	Disadvantages
Reputation-based	High throughput Less end-to-end delay High detection rate Less channel traffic	High energy consumption High overhead and complex systems No robustness against collusions The high false-positive rate
Credit-based	Less channel traffic High throughput Less end-to-end delay	No robustness against collusions No second chance Less detection rate Less energy consumption
Acknowledgment-based	The less false-positive rate High detection rate	High channel traffic Less throughput High end-to-end delay
Game theory-based	Less end-to-end delay High detection rate Less channel traffic	No second chance High energy consumption The high false-positive rate Less throughput

- We propose a multi-phase method to detect non-cooperation nodes based on the multi-person game in each round and prevent resending the data packets to such nodes. Then, it does not have high energy consumption. The network throughput is increased due to the source node that does not need to resend the data packets. Also, less traffic is caused by not resending the data packets, and it leads to decreased average of end-to-end delay of the data packets in the network. We have a lower false-positive rate and false-negative rate and a higher detection accuracy of selfish nodes. It causes the throughput of the networks to increase, and resending the data packets is prevented which might be needed due to failure of the data packets to reach the destination.
- We carry out several simulations to survey the different metrics of the theoretical results.

The remaining parts of the paper are as follows: Section 2 addresses the related works. In Section 3, the system model is described, and the multi-person repeated game is formalized. The strategy of detecting the cooperation stimulation node is presented in Section 4. The performance evaluation is addressed in Section 5. The conclusion is presented in Section 6.

2 Related works

Several approaches have been developed to detect non-cooperative nodes and stimulate them to cooperate with other nodes in the network. These approaches, based on their nature, are classified reputation-based approaches, credit-based approaches, acknowledgment-based approaches, and game theory-based approaches. In reputation-based methods, network nodes cooperate with each other to provide feedback for a set of particular nodes. Each node updates the assigned reputation value with respect to its feedback. The nodes that have more reputation value are recognized as cooperative nodes, and the nodes have a lower reputation are recognized as non-cooperative nodes. The most popular approach is the watchdog method based on the reputation mechanism [10]. The approach is proposed for fair cooperation nodes in MANET [11]. A node is used as a reputation manager called CONFIDANT, and it is responsible for maintaining the credibility of watchdog nodes and pathrater. The Observation-based Cooperation, Enforcement in Ad hoc Networks (OCEAN) approach is the DSR protocol developed based on the reputation-based approach and monitoring methods [12]. An intelligent central organization approach called the Separation of Detection Authority (SDA) is designed to consider the reliability of the network [13]. A Payment Punishment Scheme (PPS) is proposed to send messages, monitor, and report the neighbor nodes by using three watchdogs and stimulate them to cooperate with the network [14]. The method has clustered the nodes, and the cluster head applies the modified Extended Dempster-Shafer model by using watchdogs to detect the selfish node. A trust model approach is proposed based on the Dempster-Shafer evidence theory [15]. The proposed method is a clustered approach based on the direct trust of nodes to each other and the indirect trust of the neighboring nodes based on the Dempster-Shafer evidence theory. The cluster's nodes send their trust values about other neighboring nodes to their cluster head. The cluster head uses Dempster-Shafer theory to compute the indirect trust, and if a node trust is less than the threshold, it will be known as a non-cooperative node and isolated in the network.

In credit-based, if the nodes have a data packet to send, they pay for it, or the nodes trade their data packets between themselves and sell it at a higher price after buying a packet. The approach is proposed to detect the selfish nodes in the network using Nuglets [16]. It is the combination of a packet purse model (PPM) and a packet trade model (PTM) by the credit-based approaches [17, 18]. A credit-based approach called SPRIT (s simple, cheat-proof, credit-based system) is presented to stimulate the nodes to cooperate with other nodes [19]. I each node after receiving a message, the receipt of the message will store in the node's memory reporting to the credit clearance service (CCS) by transferring receipts for each sent/received message. The SPRITE approach has been improved, called MODSPRITE [20]. In this approach, if a node receives a message, it will store the receipt of the message and then communicates with the cluster head, which is responsible for providing credit and charging it for the other cluster member nodes. The new Nuglet approach is a combination of PPM and PTM approaches to identify selfish nodes in the network [21, 22]. In this approach, some virtual currency is generated by the source node using the PPM approach, and it is traded between nodes by the PTM approach until the content reaches zero.

In acknowledgment-based methods, it ensures sending a packet to a node using an acknowledgment message. Balakrishnan et al. have developed TWOACK to detect selfish nodes in the network [23]. Each intermediate node is sent a TWOACK message with a specific packet identifier to the previous node, and it continues until the packet is received by the destination node. The S-TWOACK scheme is actually an improved TWOACK method [24]. It sent an acknowledgment packet after receiving a certain number of the data packets. The EAACK method consists of three main partitions: ACK, S-ACK, and MRA malicious authentication [25]. It is basically an end-to-end acknowledgment model. For all three consecutive nodes on the path, the third node must send an acknowledgment S-ACK packet to the first node, but the source node does not immediately rely on a misbehaving report and needs to change its MRA state and approve misbehavior reporting. Each of the three EAACK sections uses a digitally signed digital signature and retrieves the message. Bounouni et al. proposed an approach consists of four models [26]. The monitoring model is responsible for controlling the sending of routing packets and data packets by using the acknowledgment packet. The reputation model evaluates each nodes' neighbors. Stimulator model manages and updates nodes' credit accounts, and malicious and selfish nodes are punished by an isolator model.

Game theory is an applied mathematical theory, it models and analyzes systems in which each person tries to find the best strategy that has been chosen by others to find success [27]. It is primarily used in economics to model competition between companies. The game consists of a principle and the finite set of players as $N = \{1, 2 \ldots n\}$. Each of them chooses a $s_i \in S_i$ strategy aimed at improving the utility function U_i (s): S → R denotes the sensitivity of each player to everyone's actions. The game theory has been classified into cooperation/non-cooperation games, dynamic/static games, repeated/one-interaction games, finite/infinite games, and n-person/two-person games. An approach based on a dynamic auction framework, non-cooperative, and finite-repetition game theory is presented based on the second-lowest price [28]. In the approach, the source node is trying to find a route with the lowest cost to send packets and at auction uses the second-lowest payment bid. A dynamic and self-learning repeated game was proposed to improve transmission efficiency by considering the non-cooperative network nodes [29]. In this approach, each node has two stages of decision-making, which is the first decision to send its packet, and the next decision is to forward the packets of other nodes. A game theory-based approach has been developed to associate users of wireless stations to prevent heterogeneous and poor performance based on joint resource allocation and association of wireless stations [30]. The payoff of wireless stations is based on the individual power of each station. Vijayakumaran et al. proposed a novel detection of the selfish node, which consists of two phases [31]. The generation phase includes routing task confirmation phase and the routing-report generation phase, and coordination-confirmation report generation phase. In the confirmation phase, the supervising agent will send the request for approval to the middle relay nodes.

The proposed scheme is a selfish node detection and prevention method called SENDER [32]. The scheme consists of two phases: the detection and prevention phases. In the detection phase, an adaptive threshold algorithm has been used to identify all nodes. In the prevention phase, selfish behavior is avoided based on the repeated game. The number of forwarded packets should be compared between current behavior and normal behavior to identify selfish behavior, which consists of three phases. Initially, the threshold value is set to the previous values. Next, the packet forward ratio (PFR) is calculated. Finally, the comparative threshold algorithm is used to compare with a threshold value to determine whether the current node shows selfish behavior or not. If the PFR is lower than the threshold value, the node is selfish, and the alarm is raised. Otherwise, the threshold value will be updated in accordance with the current PFR and the new threshold value for the next interval. In the prevention phase, the proposed method uses repeated games to prevent selfish behavior, and the game with payments are designed so that nodes gain lower payoffs if the nodes choose the selfish strategy; hence, they are unwilling to choose this strategy, and if some nodes sometimes choose the selfish strategy, they will tend to choose a normal strategy after a certain period of time due to reduced payoffs. Therefore, in the prevention phase, selfish behavior can be prevented by choosing a normal strategy.

3 System model

Each of the nodes is aware of the set of its neighbor nodes located at its domain. Each node is either normal or selfish. The nodes are not aware of the selfish or normal nature of their neighbor nodes. Each node uses all of its information and expectations from the behavior of other nodes to find the best strategy for itself. In order to be able to forward the data packets to the destination, the nodes do this by cooperating with each other due to the limitation of the sent frequency range. During the network operation, nodes gain some information about their neighbor nodes' cooperation. Each node tries to achieve the best result and the most payoff in the network. For this purpose, it tries to interact with normal nodes to forward its data packet to the destination. All notation and its description are used in the system model collected in Table 2.

The game is defined as $G = \{N, A^k, u^k\}$. In this definition, N is the number of nodes existing in the network, A^k is the set of the actions of the nodes, and u^k is the set of utility function which is the outcome of players in one round. The round is named as round k. For further explanation, $A^k = [A_1^k, A_2^k, \ldots, A_N^k]$ is the set of the node actions in round k. For example, A_i^k is the action of node i in round k. $A_i^k \in [0,\ \max(\text{pow}_i))$, pow$_i$ is the highest sending energy power required by node i for sending a packet.

A_i^k is the total actions of each node i in the repeated game in one round, and it can be forwarded or not be

Table 2 Notations and its description of systems

Notation	Description
N	The number of nodes
A^k	The set of the actions of the nodes in round k
$\boldsymbol{A_i^k}$	The action of node i in round k
u^k	Set of the utility function
$\boldsymbol{u_i^k}$	The utility function of node i in round k
pow$_i$	The highest sending power required by node i
$\boldsymbol{r_0}$	The constant value considered as a reward for node i
$\boldsymbol{n_{iCH_i}^k}$	The total number of the received packets from cluster head CH$_i$ by node i
$\boldsymbol{n_{Rij}^k}$	The total number of the forwarded packets of node j
$\boldsymbol{n_i^k}$	The number of node i packet in round k
$\boldsymbol{c_i}$	The total energy required for sending the data packets
$\boldsymbol{d(i,j)}$	The distance between node i and node j
p_i	The probability of node i to run one strategy
$\boldsymbol{P_i}$	The set of probabilities for node i to run all the strategies
$\boldsymbol{s_i}$	One strategy of node i
$\boldsymbol{S_i = \{F, NF\}}$	The set of all strategies (forwarding and not forwarding)
$p_i(s_i)$	The probability for every pure strategy of $\boldsymbol{s_i}$
$\boldsymbol{\Delta(S_i)}$	Strategies by node i in a mixed game
$\boldsymbol{p_i^*}$	New probability for node i
$\boldsymbol{\Delta p}$	The probability changes rate in each round
$\boldsymbol{u_i(p_1^*, \ldots, p_n^*)}$	Payoffs function of node i in new probability
$\boldsymbol{\pi_1(F, NF)}$	The probability distribution for node 1
$\boldsymbol{\pi_i(F, NF)}$	The probability distribution for node i
$\boldsymbol{\pi_i^k(F, NF)}$	The probability distribution for node i in round k
$\boldsymbol{E_{n_1}(F \mid p_2^*, p_3^*, \ldots, p_n^*)}$	The expectation value for node 1
$\boldsymbol{E_{n_i}(F \mid p_{-i})}$	The expectation value for node i for forwarding the packets
$\boldsymbol{E_{n_i}(NF \mid p_{-i})}$	The expectation value for node i for not forwarding the packets

forwarded to the data packet of node j (F, NF). Each player (nodes of each cluster) should choose one of these strategies in each round. After adopting the strategies in the games, the player gains a profit, the value of which is calculated by utility function and $u^k = \{u_1^k, u_2^k, \ldots, u_N^k\}$ is the efficiency value and utility function expected by the nodes in round k, and u_i^k is the utility function of node i in round k according to Eq. (1).

$$u_i^k = r_0\left(n_{i\mathrm{CH}_i}^k + n_{Rij}^k\right) - c_i n_i^k \quad (1)$$

In Eq. (1), r_0 is the constant value considered as a reward for node i. $n_{i\mathrm{CH}_i}^k$ is the total number of the received packets from cluster head chi (cluster head of cluster CH$_i$) by node i, n_{Rij}^k is the total number of the forwarded packets of node j (member of cluster CH$_i$) by node i in round k, and n_i^k is the number of node i's packet in round k that has been sent to the destination and c_i is according to Eq. (2).

$$c_i = \sum_{j \in \mathrm{CH}_i} \frac{\mathrm{pow}_i}{d(i,j)} \quad (2)$$

If a node forwards the data packet of the other node, the node consumes the energy power for it. The energy consumption of forwarding the packets should pay by the owner of the packets. In Eq. (2), c_i is the total energy power required for sending the data packets by node i and have been sent to other nodes that are members of cluster CH$_i$ in round k. The notation $d(i,j)$ is the distance between node i and node j. It is clear that c_i is the cost of the data packets which node i should pay it as the owner of the packet.

The proposed method models the system as a mixed or random strategy. The method allocates probabilities for different strategies of each player in Fig. 1.

When node N$_1$ wants to send the data packet to the destination N$_n$, it sends the packet to node N$_2$ but N$_2$ may forward it or not. According to our definition of mixed strategy, N$_2$ is independent to select the A_i^kas the strategy of s_i with the probability of p_i then it selects node N$_3$ to forward the data packet to the destination this procedure is continued up to the data packet forwarded to the destination or it is dropped by nodes. According to the mixed strategy definition, P_i that is $P_i : S_i \overset{s_i \in S_i}{\longrightarrow} [0,1]$ allocates a probability equal to $p_i(s_i)$ for every pure strategy of $s_i \in S_i$, and p_i is the probability of node i to run s_i as one strategy. The set of all strategies are in S_i and one strategy of node i defined s_i, so that $\sum_{s_i \in S_i} p_i(s_i) = 1$. In other words, according to the probability rules, the total probability of all strategies run by each player is equal to 1. With the increase of node number in each cluster, the game is expanded and a multiplayer game is done by the cluster member nodes similar to Fig. 1.

If the game assumes as the mixed strategy in Fig. 1, with n nodes, the game is represented $G_{ME} = \{N, (\Delta(S_i)), (u_i)\}$, in which $\Delta(S_i)$ is defined as:

$$\Delta(S_i) = \left\{ \begin{array}{c} (p_{i1}, \ldots, p_{im}) \in R^m : p_{ij} \geq 0 \\ , \forall j = 1, \ldots, m \\ , \sum_{j=1}^{m} p_{ij} = 1 \end{array} \right\} \quad (3)$$

At first, to define $u_i(p_1, \ldots, p_n)$, it is mentioned that the random variables p_1, …, p_n are pairwise independent;

therefore, the probability of a pure position of $(s_1, \ldots, s_n)$ is:

$$p(s_1, \ldots, s_n) = \prod_{i \in N} p_i(s_i) \tag{4}$$

Regarding the fact that after each round of the game, the probabilities change and depend on node *i* cooperation to forward the data packets; it has been changed to Eq. (5) and is defined as follows:

$$p_i^* = p_i \mp \Delta p, \forall i = 1, \ldots, n \tag{5}$$

Accordingly, we can define the pay-off functions of $u_i(p_1^*, \ldots, p_n^*)$ as follows:

$$u_i(p_1^*, \ldots, p_n^*) = \sum_{(s_1, \ldots, s_n) \in S} P^*(s_1, \ldots, s_n) u_i(s_1, \ldots, s_n) \tag{6}$$

For the game in Fig. 1, we have a probability distribution for nodes in Fig. 1 when $S_i = \{F, NF\}$ and $u_i(p_1^*, \ldots, p_n^*)$ defined in Eq. (7):

$$\pi_i^k(F, \mathrm{NF}) = \sum_{s_1, s_2, \ldots, s_n \in S} p^*(s_1, s_2, \ldots, s_n) * u_i^k(s_1, s_2, \ldots, s_n),$$

$$S = s_1 \times s_2 \times \ldots \times s_n = \{(F, F, \ldots, F), (F, F, \ldots, \mathrm{NF}), \ldots, (\mathrm{NF}, \mathrm{NF}, \ldots, \mathrm{NF})\} \tag{7}$$

The expectation values for all nodes are calculated for the choice of sending or not sending the data packets with Eqs. (8) and (9).

$$E_{n_i}(F|p_{-i}) = \sum_{s_1, s_2, \ldots, s_n \in S_{-i}} p^*(S_{-i}) \times u_i^k(S|s_i = F) \tag{8}$$

$$E_{n_i}(\mathrm{NF}|p_{-i}) = \sum_{s_1, s_2, \ldots, s_n \in S_{-i}} p^*(S_{-i}) \times u_i^k(S|s_i = \mathrm{NF}) \tag{9}$$

In other nodes, the expectation value is calculated by equivalent to each node in Eqs. 8 and 9; the Nash equilibrium is obtained. By calculating the Nash equilibrium, the probability values of each player are calculated. By getting these values at each round of the game, each player chooses the best action against the opponent. Given the fact that no nodes know when the game is over, a repeated game is infinite which has *N* players. In a repeated game at the beginning of round *k*, node *i* makes the decision based on the nodes' behavior in the past.

4 Proposed method

A multi-step mechanism is designed based on the game theory to stimulate the cooperation between the selfish nodes in the Internet of Things in Fig. 2. A multi-step scenario is considered by setting up the nodes in the IoT networks, and the nodes try to distinguish their neighbor nodes by sending Hello messages. Some base stations (BS) are placed in a different location to collect the data packets. In the first step, nodes grouped in the cluster with the cluster head(s) and a BS to collect the data. The member nodes send the data packets in multi-hop to the cluster head, and the nodes cooperate in forwarding the data to the cluster heads or the destination of the base station. Then, they run a multi-person game in the second step (playing the multi-person game and sending data) while forwarding their own data packets or the neighboring nodes. By running the game, each node has

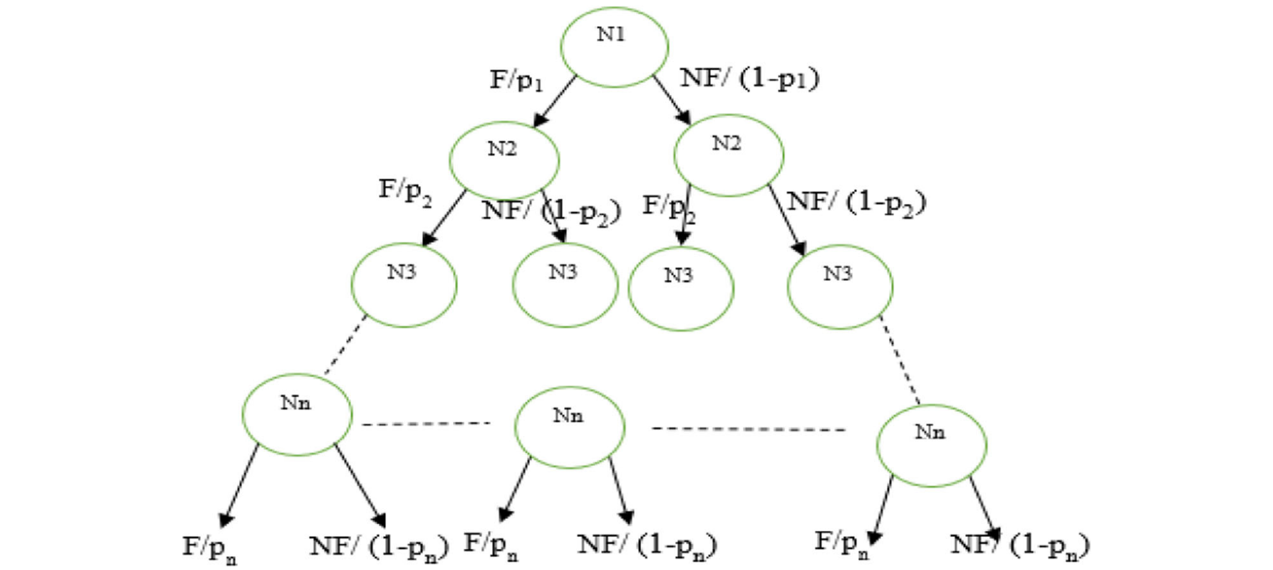


Fig. 1 Chart of forwarding or not forwarding the data packets

the data node to send the destination; it will select one neighbor node to play the game. The node will train about neighbor node status, and each node extracts features of the neighbor nodes. If the neighbor node forwards the data packet and cooperates with others, the node optimizes the probability of the neighbor node in the game (forward the data packets) for the next rounds. In the third step (detecting non-cooperation node and update), the nodes are assumed to be the selfish node and can select their strategy when forwarding the packets. The game theory and the cooperation process analysis are used to determine the selfish nodes and/or malicious nodes that forward the data packets by delay and/or do not send them at all so that the non-cooperative nodes are identified. The nodes classify the other nodes and update the reputation table. The playing multi-person game and sending data and detecting non-cooperation node and update phases are done as a repeated game.

The game theory-based mechanism takes advantage of the punishment-based and incentive mechanism. While the nodes are identified in misbehavior, the power of transferring their data packets (and/or even their cooperation with other nodes) is reduced so that the non-cooperative nodes are stimulated to cooperate via punishment, not cooperating with them reduced their reputation. By detecting non-cooperative nodes, the network throughput and detection accuracy are increased, and average end-to-end delay and energy consumption are decreased. Other metrics to compare the algorithm with other methods are false-positive rate and false-negative rate. If these metrics are low, the network will have a high performance, and the proposed method has a lower false-positive rate and false-negative rate in comparison with other detection methods.

The mechanism is designed based on the game theory to stimulate the cooperation between the selfish nodes in the Internet of Things which Fig. 3 shows the flow of data in the mechanism. Finally, the main focus of the present paper can be summarized as follows.

- The game theory analysis will be presented in the multi-person game. The game is modeled as an infinite repeated game, and the level of the cooperation power is found, which is achieved by the performance of the game by the reputation of each player proportionate with its cooperation with other nodes. It is also shown that the best mode for each thing is the cooperation option with other nodes in the network. The theoretical game approaches are used to stimulate the nodes to cooperate in the internet of things.
- A stimulating strategy based on the game theory and based on punishment is proposed to stimulate the nodes to cooperate in the Internet of Things. A motivational strategy is introduced and applied to overcome the challenge of non-cooperative behaviors in the network. The main idea is that each node monitors the behavior of other nodes and forwarding or not forwarding other nodes' data packets in a specific period of time. It is shown by the theoretical analysis the nodes are stimulated to cooperate by

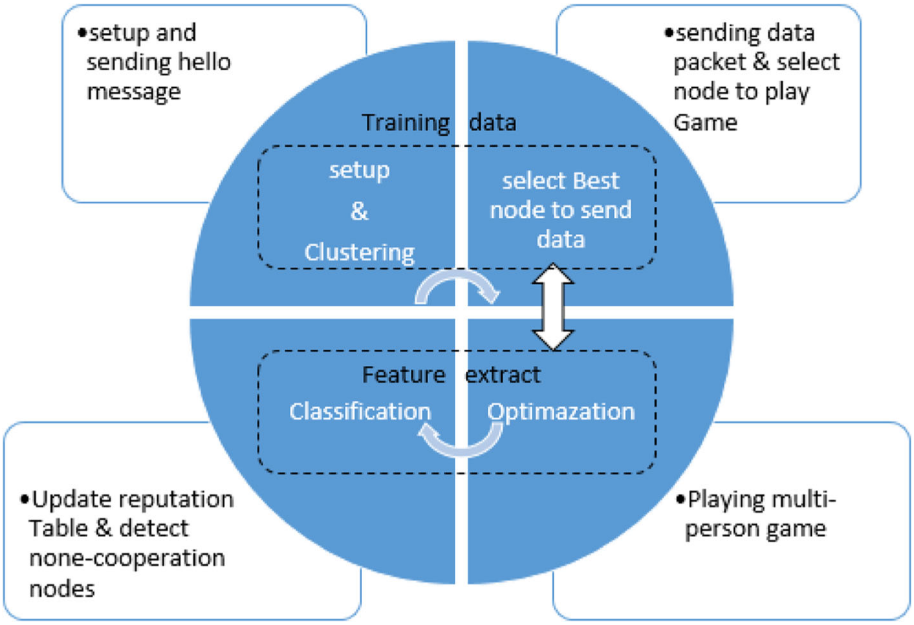

•setup and sending hello message
•sending data packet & select node to play Game
Training data
setup & Clustering
select Best node to send data
Feature extract
Classification
Optimazation
•Update reputation Table & detect none-cooperation nodes
•Playing multi-person game

Fig. 2 Multi-step detection mechanism in block diagram

this strategy because each deviation from forwarding the packets leads to less cooperation or even no cooperation, which reduces their reputation.

- The cluster head considers its cluster members for the status of each opponent nodes (neighboring nodes) saved in their reputation table. The nodes with reputations below the threshold are reported as selfish nodes to the cluster head. The cluster head will broadcast to all cluster nodes to know about their neighbors as a selfish node in the games so that they can stimulate the selfish nodes to cooperate with others. The results of the simulation show that the proposed strategy can approximately cooperate effectively to the desirable cooperation condition.

In the following, more details are discussed about each step of the proposed mechanism.

4.1 5-1 Setup and clustering

During this phase, all things are randomly distributed in the area. Then, each node broadcasts a “Hello” message, and the nodes replying to this message are known as a neighbor node. Each node will store information about the neighbor’s status in its database as a table consisting of four fields which are shown in Fig. 4

In the following, more details are discussed about each field, as shown in Fig. 4.

- Node’s ID: It has 16 bits to save the node’s identification.
- Number of hops up to the cluster head: It has 8 bits to save the number of hops between a node and the cluster head
- Node’s data: It is an array of n bits and saves the data of each node and its neighbors. The amount of n is per byte.
- Node’s status: It is an array of n bits and indicated the status of nodes. Because the nodes can have one of C, S, and LS statues, therefore, the length of each array is considered as 2 bits. The predefined status of this field is C, which is assumed as the cooperation node.

After the setup of the network, the neighbor nodes by using the clustering algorithm proposed by Kumar and Zaveri in 2016 [33], the performance of this method is such that all of the existing things with any features are assumed as a node, and they choose the nodes which are involved in the communications given the cluster heads. Overload decreases communications. Naturally, the nodes are heterogeneous in the Internet of Things, and they are connected to each other from different networks, and this approach has also considered the nodes heterogeneously. The clusters and their cluster head change in regular intervals, and they are dynamic due to the dynamic nature of the Internet of Things. This method promises saving energy by choosing different nodes as a cluster head.

This approach presents a hierarchical model in which the lower layer is known as layer number 2, and it has tools and equipment which have an ID code. The things include sensors, RFID tools, and people. The important assumption is that these things do not have internet protocol due to high energy consumption, and they cannot be directly or indirectly connected to a cloud system. But they are critical points of the networks which are needed by application programs. In this layer, the creation of cluster and choosing the cluster heads are done in a dynamic way. Cluster heads send the accumulated information to the upper layer of the base station.

In the upper layer which is known as layer number 1, the tools and things have an internet protocol, and they are usually less likely to face the problem of an energy shortage, and they have immediate communications and

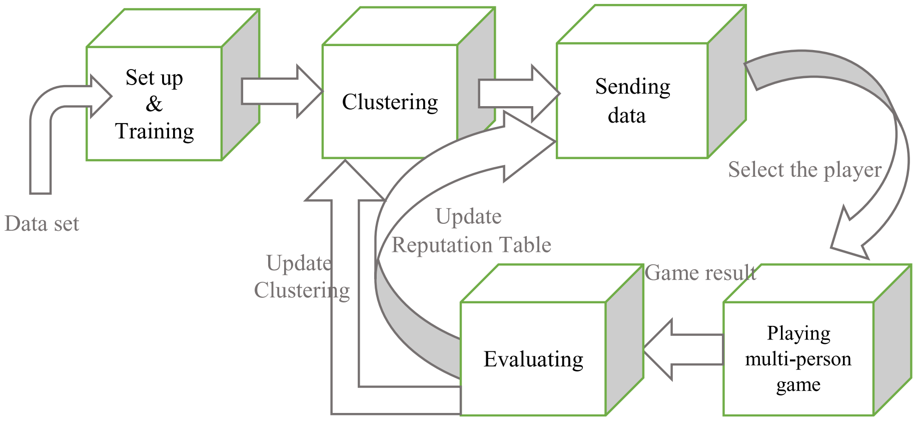
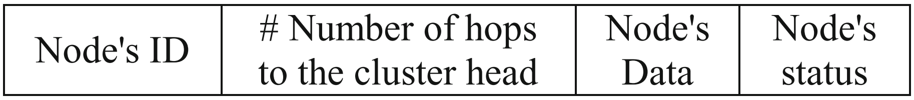


Fig 3 General diagram of the proposed mechanism



Node's ID	# Number of hops to the cluster head	Node's Data	Node's status

Fig. 4 Format of data in cluster heads

processes. These things provide the possibility of communication between all the things existing in the network. The nodes are communicated with their cluster head, and the cluster head is also communicated with the base station. In this way, all the things can communicate with each other, and in the case of necessity, they can exchange information. The proposed algorithm works on the origin of counting the number of neighbors, and the remaining energy of the node and the clusters are formed in the radio range of the node. The proposed algorithm runs three rounds to determine the cluster and the cluster head in layer number 2.

In the first round, a broadcast message is sent, and each of the nodes that receive this message sends a confirmation message in response to the sender node, and finally, all the nodes recognize their neighbors. The ID code of the node or the internet protocol is entered into the neighbor's list. In the second round, a multicast message is sent, and the node declares the number of its neighbors and its remaining energy to all of its neighbors which are located at its own radio range. In the third round, the cluster head selection process is done based on the information of the neighbor nodes, and the node which has the highest amount of remaining energy and the highest number of neighbors is selected as the cluster head.

4.2 5-2 Sending data and playing multi-person game

In the proposed method, sending a packet is considered between two nodes in a cluster from the source node to the destination node as a hierarchical multiplayer game. The players make their decisions independently, and each of them is faced with a minimum probability of losing profit. This game is repeatedly done between all the network nodes, and the number of the repeated game is very large. So, the game is an infinite repeated game. Also, the game is dynamic, because if one of the players cooperate at first, other players choose their actions being aware of the first player's action. Since the nodes do not have any information about the selfish nodes, the game will be done with incomplete information. With the start of network operation and after the setup and clustering phase, all the neighboring nodes in a cluster play with each other many times during the life of the network and exchange their data packets to reach the destination node.

At the beginning of the game, none of the nodes has any information about its neighboring nodes, and they choose one of their neighboring nodes every time of sending data. During sending data packets and receiving an acknowledgment message from the destination, the nodes will be informed of the success or failure of sending the data packet and the status of the neighboring node. However, in order to prevent the data packet from extra forwarding and increased energy consumption in the network, it will even be stuck in the infinite loop; the data packet can pass through all the nodes in the network; the limited lifetime in the data packets are used to solve this problem. In each round and each game, the nodes send the game results to the third phase, and the entry of the related node in the table is updated to take advantage of this information for successfully sending data packets in the next rounds. In other words, as a result of successful or unsuccessful sending of the data packets in each game and per round, each node sends the game results to the third phase, and it is stored in the reputation table by the neighboring nodes.

According to the distance between the nodes calculated using Eq. (1) and the information of the previous games such as the number of unsuccessful and successful sent to the third step then it is sending by the neighboring nodes and the reputation values of each neighboring node, payoff values change in reputation table. In this way, the players play more carefully, and so the selfish nodes cannot reduce the network efficiency by sending unsuccessful packets and unnecessary use of bandwidth.

At each step, node i intends to send the packet to the next step, if it knows the status of all its neighboring nodes, that is, if at least once played with them. At this moment, the mentioned node reviews its reputation table that contains the reputation record of the neighboring node. According to the number of successful and unsuccessful sending in the opponent node, it updates its payoff values and obtains the probability values of choosing each action and their reputation values, which has received the third phase. The node will be able to choose the best strategy and optimize its guess about the nature of the neighbors. But if node i does not know the status of all neighboring nodes, it will continue to play with a neighboring node that does not have any information about it, and this procedure is done for intermediate nodes so that all nodes have the opportunity to play and attend the operation to send the data packet. Further nodes also have the ability to cooperate in network operations. Figure 5 shows the performed operation of sending the data packet by the source node to the destination node.

4.3 5-3 Update and detecting selfish and malicious nodes

The reputation of the network nodes are updated by using the information received from the network nodes from the neighboring nodes for sending data packets, and it becomes possible to identify the selfish nodes in each cluster and consequently in the whole network. The way of detecting the selfish nodes in the network is based on the results of games that directly affect the detecting and updating of the reputation of each node. The game results are analyzed in the third phase, after the nodes play the game in each round. Getting the acknowledgment packet from the node which the game is played, the reputation of the node is increased by a predetermined constant shown by pd_Rep in Fig. 6; in the next round of playing, if the data packet sent to a node to forward them, it will be played according to the previous experience with that node. But if the acknowledgment packet is not received from the node played with it in a game, the reputation of that node will be reduced by a predetermined constant. If the reputation of each node is not lower than the predefined threshold value, this node will have the second opportunity to cooperate with other nodes and participate in the games. So, the selfish and malicious node will increase their reputation with other nodes. And with that single failure, that node is not considered as a selfish or malicious node. Reducing network performance can be prevented by excluding the selfish or malicious nodes. But if with the second opportunity of the nodes the reputation is less than the predefined threshold, the node will be considered as a selfish and malicious node and will be avoided to cooperate in the network.

At certain times, the cluster head controls the cluster members about the situation of each of their opponent nodes (neighboring nodes) that have stored in their reputation table. The nodes with a reputation value of lower than the predefined threshold are reported to the cluster head as selfish nodes. The node that is known as a selfish node based on the most reports of its neighbors in the games will be dismissed from the cluster, and all the cluster nodes will be informed by a message from the cluster head. Detecting the selfish node in the network will increase the network throughput and reduce energy consumption in nodes.

5 Simulation and evaluation

This study is faced with the problem of the selfish and malicious nodes in the Internet of Things. The selfish nodes are the nodes that use the facilities of the network for their own interests. These nodes do not participate in processes of sending and forwarding the packets, so they do not help to save energy and communicating with other nodes. The malicious nodes are considered as those nodes which dropped the data packets or sent them by the delay that the packets are discarded due to the expiration of the packets lifetime [34]. As a result, the throughput of the network is significantly decreased in the presence of such nodes. In order to stimulate the mentioned problem, simulation of such behaviors is done in the network for indicating its effect on the delivery percentage of the data packets, throughput percentage, and end-to-end delay. In this section, the proposed method is evaluated and compared with other similar methods in different metrics. Firstly, the evaluation criteria are introduced. Secondly, the simulation and results are presented.

5.1 Evaluation criteria

The proposed mechanism is evaluated using different criteria. The definition of the evaluation metrics are presented in the following.

5.1.1 Detection accuracy

The detection rate of the selfish node indicates the ratio number of selfish nodes detected to all the selfish nodes in the network as DA, where TP denotes the number of selfish nodes detected, and FN indicates the number of

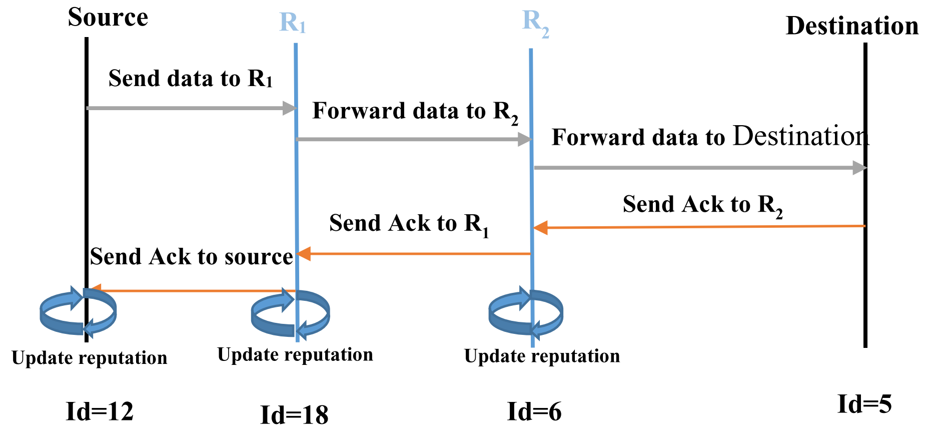
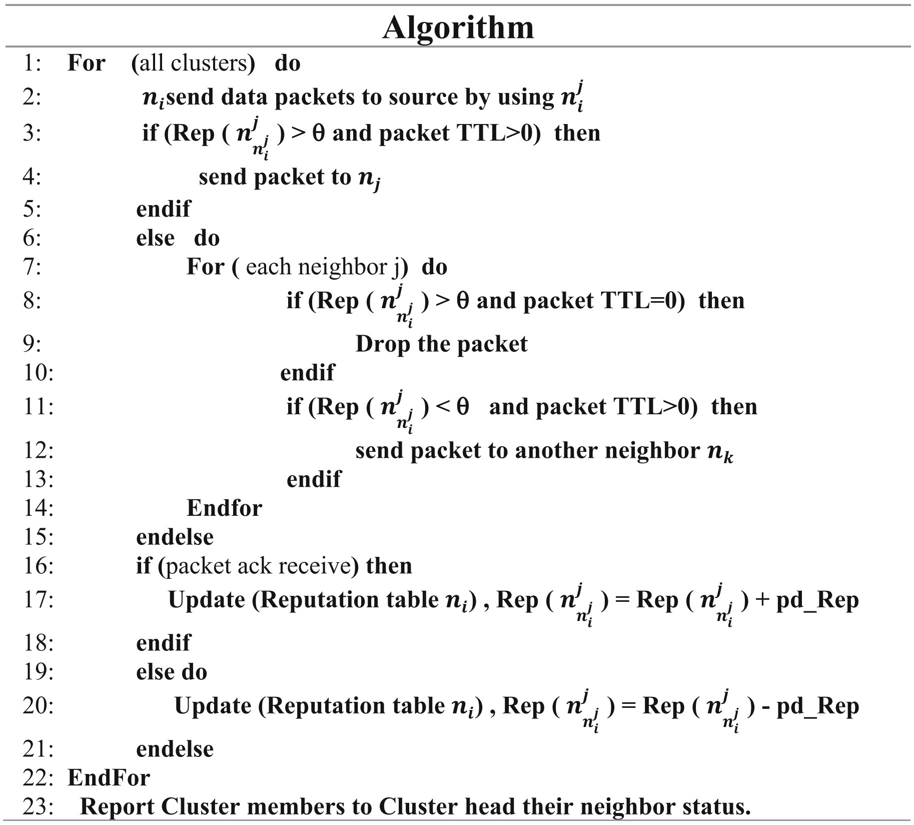


Fig. 5 Example of the packet forwarding with two relay nodes



```
Algorithm
For (all clusters) do
    $n_i$send data packets to source by using $n_i^j$
    if (Rep ( $n^{j}_{n_i^{j}}$ ) > θ and packet TTL>0) then
        send packet to $n_j$
    endif
    else do
        For ( each neighbor j) do
            if (Rep ( $n^{j}_{n_i^{j}}$ ) > θ and packet TTL=0) then
                Drop the packet
            endif
            if (Rep ( $n^{j}_{n_i^{j}}$ ) < θ and packet TTL>0) then
                send packet to another neighbor $n_k$
            endif
        Endfor
    endelse
    if (packet ack receive) then
        Update (Reputation table $n_i$) , Rep ( $n^{j}_{n_i^{j}}$ ) = Rep ( $n^{j}_{n_i^{j}}$ ) + pd_Rep
    endif
    else do
        Update (Reputation table $n_i$) , Rep ( $n^{j}_{n_i^{j}}$ ) = Rep ( $n^{j}_{n_i^{j}}$ ) - pd_Rep
    endelse
EndFor
Report Cluster members to Cluster head their neighbor status.
```

Fig. 6 Simi-code of algorithm

nodes which are selfish nodes but detected as normal nodes in the network. The detection rate of the selfish node is according to Eq. (10).

$$\mathrm{DA} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \quad (10)$$

5.1.2 False-positive rate

Another parameter for evaluating selfish node detection is the false-positive rate in the network. The false-positive rate indicates the ratio of the normal node number detected as a selfish node by error to the total number of normal nodes detected by mistake (FP) and the number of normally detected nodes (TN) in the network. Therefore, the false-positive rate (FPR) is calculated using Eq. (11) and is defined in the following:

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}} \quad (11)$$

5.1.3 False-negative rate

The parameter is used to evaluate the efficiency of selfish node detection methods; the false-negative rate is defined in Eq. (12), which is the ratio of the number of the selfish nodes detected in the normal node by error to the total number of selfish nodes detected by normal node (FN) and the number of detected normally nodes (TP) in the network.

$$\mathrm{FNR} = \frac{\mathrm{FN}}{\mathrm{FN} + \mathrm{TP}} \quad (12)$$

5.1.4 Throughput

Throughput is one of the evaluation parameters of the network in most of the fields of IoT. Throughput is actually the number of data packets that are successfully delivered to the destination. Therefore, the average throughput is the ratio of the average number of the data packets delivered to the destination by all nodes to the total number of the packets produced in the network.

5.1.5 End-to-end delay

The average end-to-end delay is the arrival time of a packet from the source node to the destination.

5.1.6 Energy consumption

IoT system nodes consist of sensor nodes, and the nodes have mobility similar MANET nodes. So, each node uses the energy model.

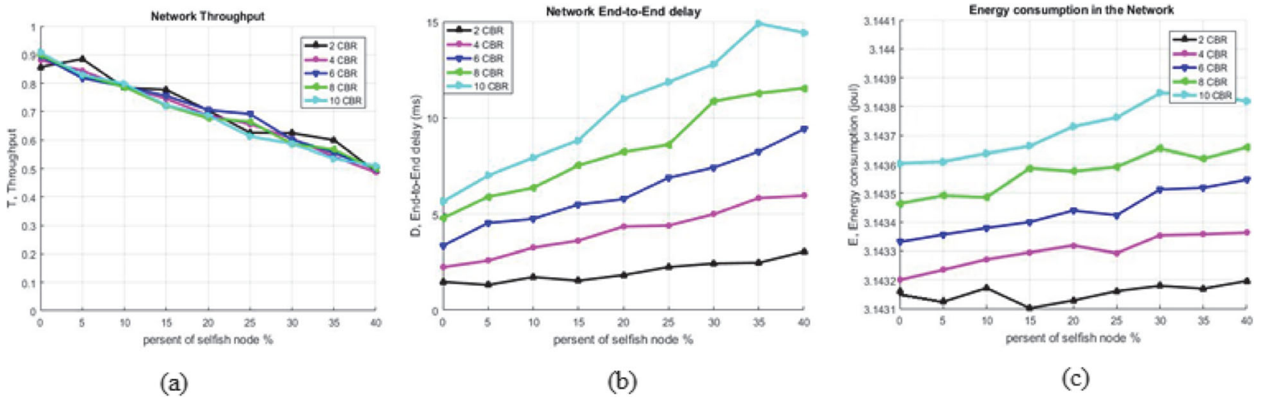


Fig. 7 Impact of the selfish node on the **a** throughput of the network, **b** end-to-end delay, and **c** energy consumption

5.2 Simulation result

A network is consistently distributed in an environment with an area of 1000 × 1000 m^2, and the nodes are randomly dispersed in IoT for 4 different types of IoT nodes with different numbers and parameters. The considered internet network includes immovable things or limited energy source similar to wireless sensor networks with 4 different types of nodes which can be used in agricultural fields as sensor type 1 (controlling water), sensor type 2 (controlling soil), sensor type 3 (controlling weather), and sensor type 4 (controlling temperature). All of these networks have wireless communications. The simulation environment is MATLAB, and in the performed simulation, at the center of all of

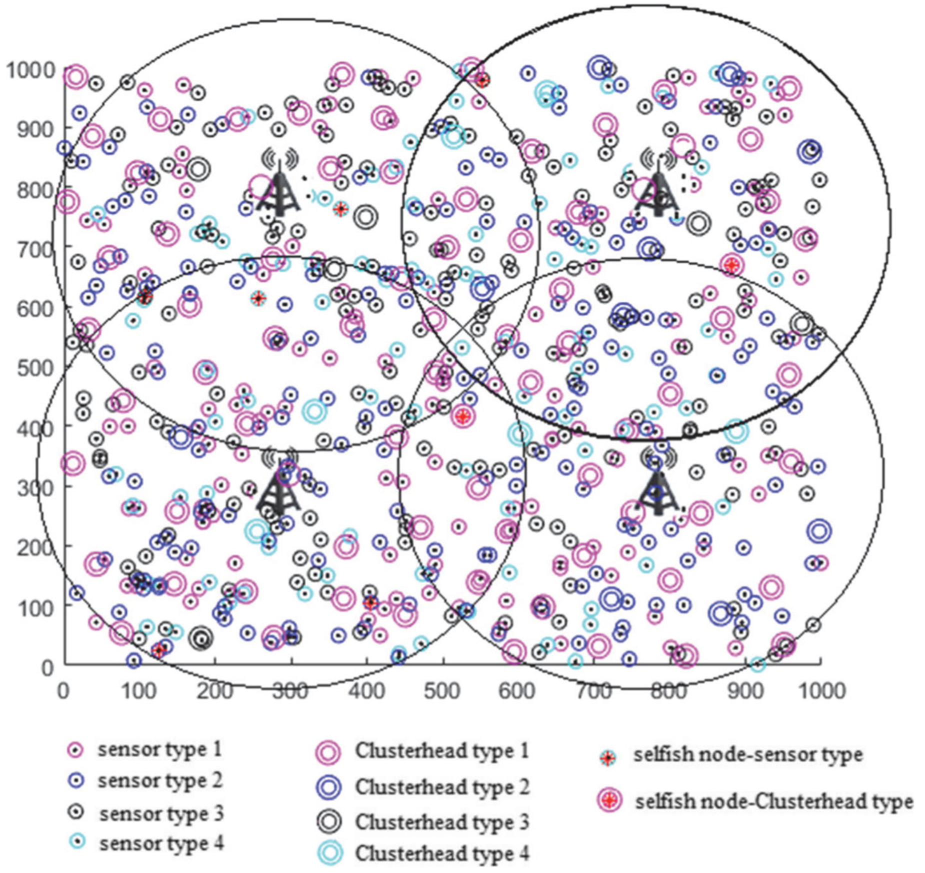


Fig. 8 System models in clustered IoT

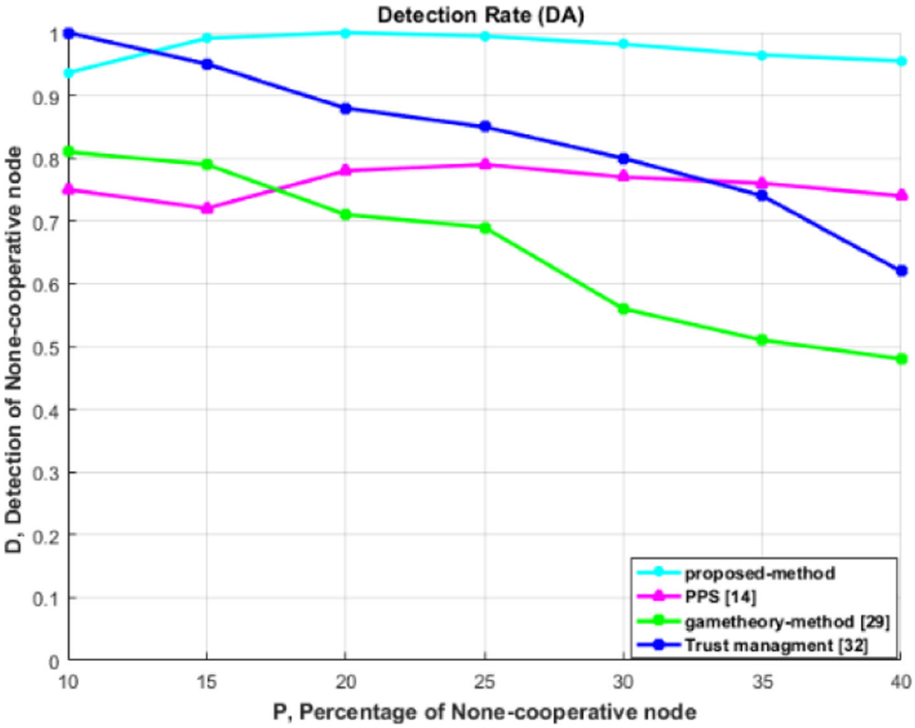


Fig. 9 Comparison of detection accuracy (DA) in IoT

the 4 types of networks; a base station is placed to collect data. Firstly, the network is simulated without any detection mechanism. In Fig. 7, the network service as usual and nodes collect data and send them to the destination, and metrics of throughput, end-to-end delay, and energy consumption are measured in different traffic on the network.

The percentage of the selfish nodes used in the simulation gradually reaches from 0% (without selfish node) to 40% of the total nodes. It means that when some nodes' energy level decreases, they will change to a selfish node. When the nodes forward the data packets, they lose energy and the number of such nodes increases in the network. At first, there is no selfish node, then the percentage of the selfish nodes is 0% of all nodes. The nodes start to collect the data and forward them to the destination, they use energy power, and some of them change their status to the selfish nodes. For example, when we have 100 nodes in the network and 5 nodes change their status to the selfish nodes, we have 5% of total nodes as selfish. In the simulation environment, we assume that the nodes' energy level is lower than the predefined energy level which became a selfish node and numbered them to calculate the percentage of the selfish nodes. In Fig. 7a, network throughput reduces by increasing the selfish and malicious node number. Imposing traffic on networks 2, 4, 6, 8, and 10 (CBR) leads to the increase in the sending of the packets. It is expected that by increasing the transmission of the data packets and the delivery rate of the data packets in the network, the existence of selfish nodes prevents it. In Fig. 7b, the average end-to-end delay for the data packets in the network is represented by the presence of the selfish nodes. Increasing network traffic leads to increase the end-to-end delay in the delivery of the data packets; when the selfish nodes are present in the network, both dropping the data packets and resending them also increases network traffic. Not only have the selfish nodes sent the data packets by delay, but also increasing the network traffic leads to high average end-to-end delay in the data packets. In Fig. 7c, the energy consumption to send the data packets increases by increasing the transmission of the data packets from the source node. When the data packets are dropped by the selfish nodes, they are sent in such a delay that they are not actually destined to be redundant energy consumption, which wastes the energy resources of the nodes. Hence, the detection of selfish nodes in the network and stimulate them to cooperate on the purposes of this article. If we can provide a model to detect the selfish nodes in the IoT, we have provided conditions for increasing throughput, reducing the average end-to-end delay in the delivery rate of the data packets and energy consumption in the network.

Secondly, the proposed mechanism is simulated and evaluated in different metrics. The mechanism clustered the nodes, and the cluster heads have communication

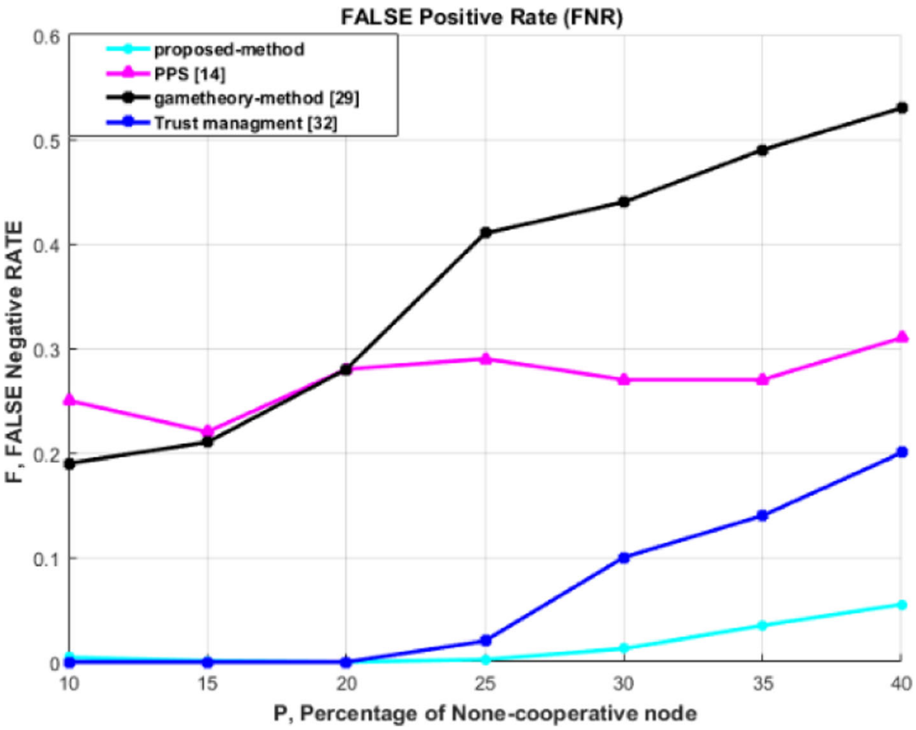


Fig. 10 Comparison of the different algorithm in false-positive rate (FPR) metrics

with cluster members in clusters 1 to 4, as mentioned in Section 4.1. According to the fact that each of these different networks has different simulation models, we will first have a clustering based on the different types of nodes according to Fig. 8. The cluster heads have 4 types: cluster head type 1 for sensor nodes type 1, cluster head type 2 for sensor nodes type 2, cluster head type 3 for sensor nodes type 3, and cluster head type 4 for sensor nodes type 4. In the simulated model, the base stations (cluster heads) are located in the position (250, 250), (750, 750), (250,750), and (750,250), respectively. However, the initial energy of the nodes in clusters 1 to 4 is 0.5, 1.5, 1, and 1.1 J and 200, 100, 200, and 200 number of nodes in clusters with a radio range of 80, 70, 75, and 70 m, respectively. But the energy model and the type of nodes are the same.

The proposed approach has made decisions about both of the cooperation and selfish nodes by cluster heads. To evaluate the proposed scheme, we were simulated in Window 8.1 basic (64-bit), core i7 processors, 370 M processors, 2.40 GHz of speed with a memory of 8 GB, and MATLAB 2015 software. The performance of the proposed method is compared with the PPS [14], game theory-based [29], and trust management [32] protocols for evaluation metrics such as detection accuracy, the percentage of false-positive rate, false-negative rate, throughput, average end-to-end delay, and energy consumption. The simulation result is performed 100 runs, and the average result has shown in different metrics.

In order to examine the important parameter of the detection accuracy (DA) of the selfish node in IoT, at the beginning of the simulation, 10% of the total nodes are selfish nodes in the network; further, the rate of selfish nodes gradually increased by 15%, 20%, and 40%. It is worth mentioning that in the real world, as time goes on, due to lower initial energy levels, nodes tend to maintain their energy resources and they will refuse to send the other packets, and they will be as selfish nodes. According to the diagram in Fig. 9, increased detection accuracy of the selfish node is clearly observed. When the percentage of selfish nodes increases in the network, it leads to a larger number of games running in the proposed method. The reputation of each node is updated during these games and while forwarding the data packets through different nodes. Therefore, when only 10% of the nodes of the network are selfish nodes, lower games have been run for preventing more energy waste, and the reputation of the network nodes is not well known, and only 92% of the selfish nodes have been detected. However, with the increased percentage of the selfish nodes in the network and increased number of games, the reputation of the games is updated, and detection is done accurately, and up to 100% of the selfish nodes will be detected. However, an increased number of selfish nodes in the network needs to run a large number of games, and due to the resulted energy consumption, it is not a rational way. Also, with this

Table 3 Different metrics of the proposed methods in comparison with other methods

Presence of selfish node, algorithms	Metrics	10	15	20	25	30	35	40
Proposed algorithm	Detection accuracy (DA)	0.93	0.99	1	0.99	0.98	0.96	0.95
PPS [14]		0.75	0.72	0.78	0.79	0.77	0.76	0.74
Game theory-based [29]		0.81	0.79	0.71	0.69	0.56	0.51	0.48
Trust management [32]		1	0.95	0.88	0.85	0.8	0.74	0.62
Proposed algorithm	False-positive rate (FPR)	0	0	0	0.002	0.0127	0.0347	0.0549
PPS [14]		0.25	0.22	0.28	0.29	0.27	0.27	0.31
Game theory-based [29]		0.19	0.21	0.28	0.41	0.44	0.49	0.53
Trust management [32]		0	0	0	0.02	0.1	0.14	0.2
Proposed algorithm	Throughput	75.85	73.05	74.14	78.92	86.71	91.07	95
PPS [14]		48	41	43	38	35	32	19
Game theory-based [29]		50	42	41	36	33	30	15
Trust management [32]		85	78	81	84	82	71	73
Proposed algorithm	End-to-end delay (ms)	16.35	17.01	17	12	7.93	4.1	1.47
PPS [14]		48	41	43	38	35	32	19
Game theory-based [29]		50	42	41	36	33	30	15
Trust management [32]		85	78	81	84	82	71	73
Proposed Algorithm	Energy consumption (μJ), 2 CBR	3.14300	3.14302	3.14303	3.14299	3.14301	3.14301	3.14302
PPS [14]		3.1	3.25	4.2	4.8	4.9	5.01	5.2
Game theory-based [29]		2.8	2.9	3.5	3.6	3.66	3.78	3.9
Trust management [32]		2.9	3.1	3.4	3.9	4.12	4.52	4.6

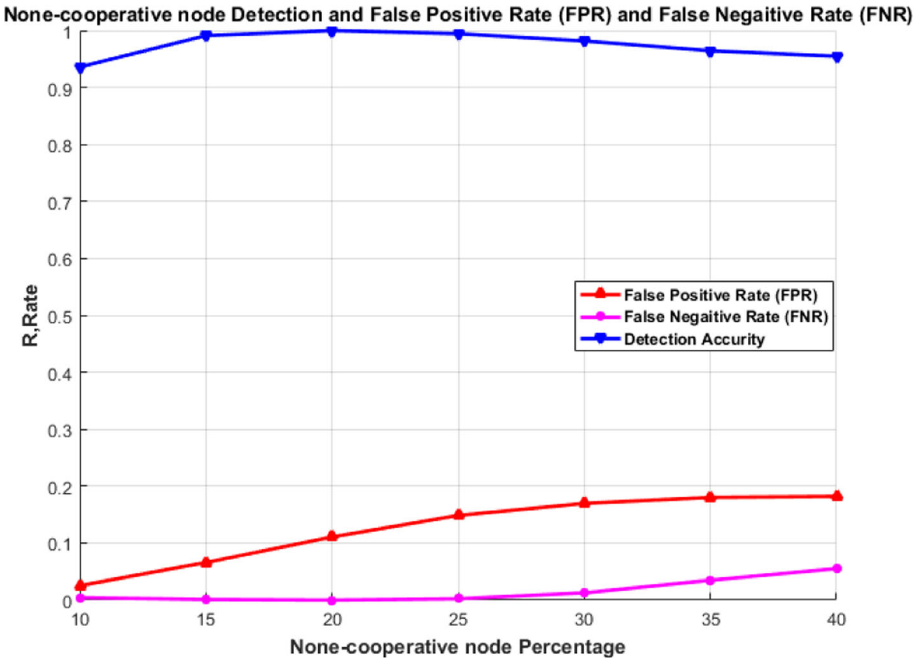


Fig. 11 DA, FPR, and FNR metrics in the proposed method

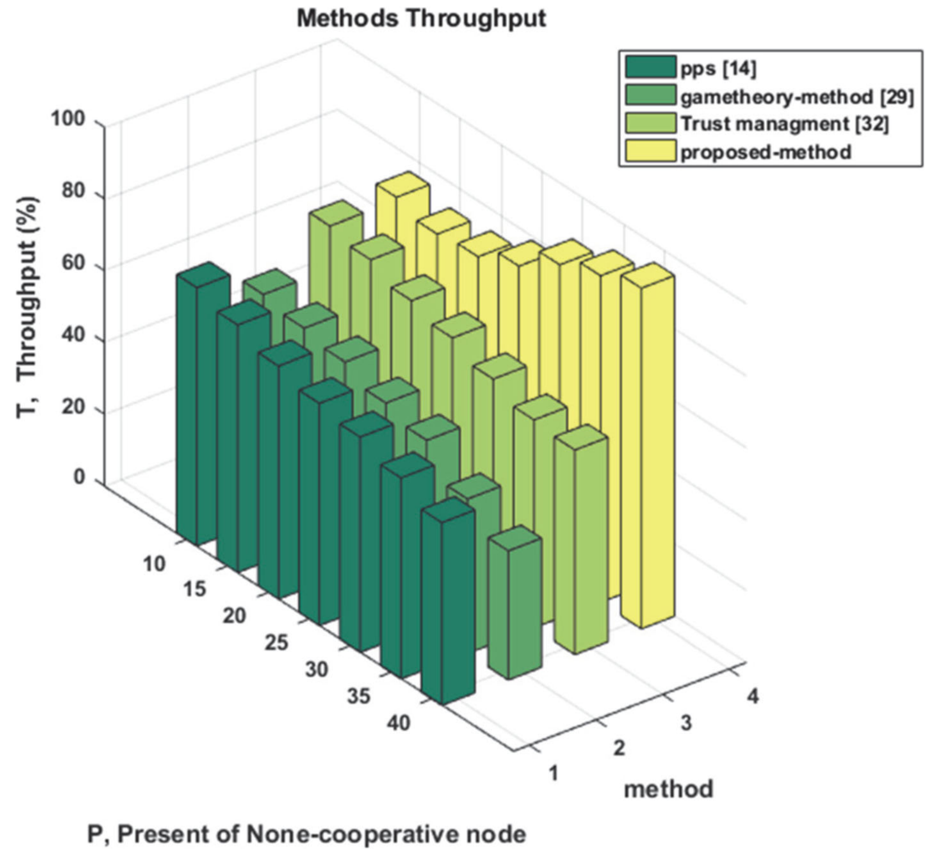


Fig. 12 Comparison of the throughput metrics in a different algorithm

number of games, an acceptable percentage of selfish node detection will be achieved even by a high percentage of the existence of the selfish nodes. The numerical values resulted in comparing the proposed approach showing that with the increased percentage of selfish nodes, the algorithm detection accuracy will be higher than other algorithms and has a slighter slope compared with other similar methods.

The fact that the proposed method has a slighter slope compared to other methods is shown in Table 2. This is obvious that the proposed method uses acknowledgment messages and the reputation of the nodes in each cluster to detect the selfish nodes. While other methods in higher percentages of the selfish nodes are usually unable to detect them in high detection accuracy.

In fact, FPR has an inverse relationship in the network, so its low level shows the accuracy of the proposed approach. That is, the higher the number of cooperation nodes correctly detected, the network performance will also be higher, because the network refuses to cooperate with the node when it detects as a selfish and malicious node. If the detection was mistaken, it would reduce the network efficiency and forward lower the data packets to the destination. Twenty games have been repeated to evaluate the proposed algorithm, and the simulation result is performed 100 runs, then the average result has shown that the higher number of games is the higher selfish node detection accuracy. The algorithm is also implemented in rounds to evaluate the performance of this algorithm in different situations. The numerical comparison has shown that the increase of the selfish nodes in the network, the false-positive rate of the proposed algorithm is lower than the other algorithms. As shown in Fig. 10, the FPR is better than other algorithms when more than 20% of the network nodes are a selfish node and less error than others, but up to 20% is roughly the same as the methods PPS [14], game theory-based [29], and trust management [32] protocols. The fact that the proposed method has a lower false-positive rate compared to other methods is shown in Table 3.

Figure 11 illustrates the proposed algorithm for the detection accuracy (DA), the false-positive rate (FPR), and the false-negative rate (FNR) metrics in the number of selfish nodes from 10 to 40%. The only FNR parameter is the negative point in the proposed method, which increases with the increase in the number of selfish nodes in the network. But it has a low percentage, then it has a disproportionate effect on network performance;

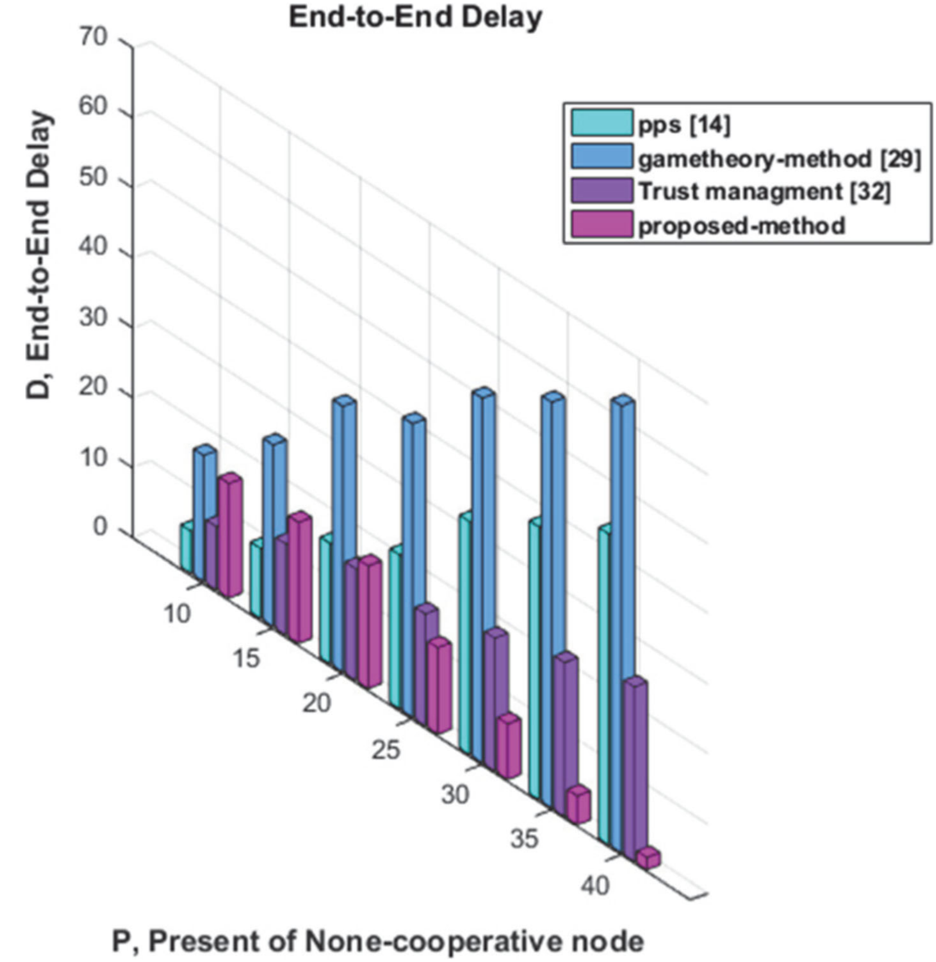


Fig. 13 Comparison of the average end-to-end delay by a different algorithm

considering the diagrams in Fig. 11, this weak point of the proposed approach was negligible, and a further work on this issue will be examined further.

The maximum throughput shows the high efficiency of the proposed method, because when the selfish nodes do not send data packets in the network and the acknowledgment message is not received in the source node, this packet is sent in the network again and it leads to increased traffic and the total number of the produced and sent the packets in the network, which is practically disadvantageous. So, the more throughput in the network led to proper use of the network resources, including bandwidth or limited energy resources in the nodes. According to Fig. 12, high throughput of the proposed approach is observed due to early detection of the selfish nodes in the network which prevents the production of repeated the data packets, increased network traffic, and average delay of the packets in the network. Table 2 shows the network throughput in the proposed method and other methods PPS [14], game theory-based [29], and trust management [32] protocols. Which has a direct relation to the selfish node detection accuracy in which the detection accuracy of the selfish and malicious nodes is higher than other approaches which cannot handle unsuccessful data packets from the nodes, and the data packets are sent from trusted paths to be forwarded to the destination.

Figure 13 depicts the average end-to-end delay in the proposed method compared with different methods. The average end-to-end delay of the proposed method is lower than other algorithms. The increase in the number of selfish nodes increases the average end-to-end delay. As the number of selfish nodes increases, it takes a lot of time to get a packet to the destination. Because of dropping the packets by selfish nodes or delayed, the network has to resend the data packets. Resending the data packets in the network will cause network power loss and decreases network lifetime and increase the average end-to-end delay. So, the proposed method can detect the selfish nodes soon, and it will reduce the end-to-end delay of the data packets. Some selfish nodes on the malicious nature also exist in the network, which increases

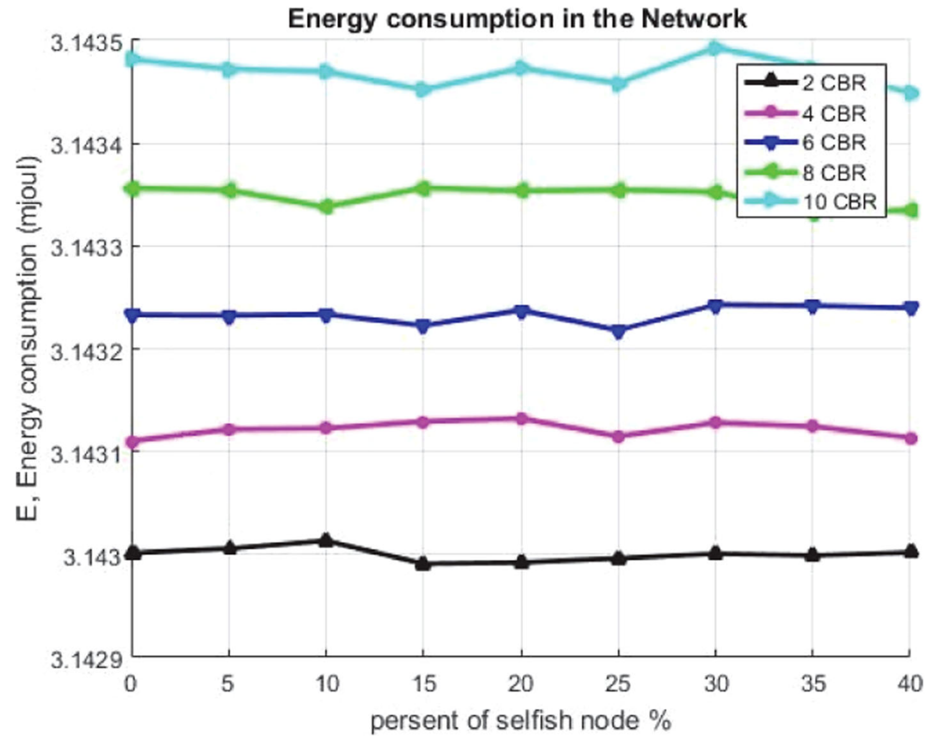


Fig. 14 Comparison of energy consumption in different traffic (2, 4, 6, 8, 10 CBR)

the average end-to-end delay in the network, so that the data packets remain in the buffer of the intermediate node until its survival, then it is sent and the packet to be discarded, because the survival time of the packet is expired, and this increases the average end-to-end delay in the network. Table 2 shows the numerical comparison of this delivery time in milliseconds. The advantages of the proposed method are detecting selfish nodes at high speeds, so the effects on the network are less. The node is detected, and the packets are lost less than other methods at the very beginning of selfish behavior. The proposed method with low end-to-end delay is appropriate for emergency applications, and real-time applications play critical role times and can provide service in that particular application.

The lower the energy consumed in the networks, the higher the efficiency of the proposed method. Due to the random acceleration of mobile nodes in the IoT network, energy consumption varies in a certain range. The average energy consumption in all nodes in the IoT network varies 3.1429~3.1435 μJ. When traffic change in network and the total number of sent packets increase, the power consumption also increases. Figure 14 shows average energy consumption in the network during 20 rounds which are the network continues the normal work, and the proposed method detects the selfish nodes and the network applies different traffic 2 to 10 CBR. Figure 14 illustrates less energy consumption than Fig. 7c. Due to the proposed method, detecting the selfish and malicious nodes prevents further energy consumption and reduces energy consumption.

6 Conclusion

The paper presented a new multi-person game theory based, which is used to detect the selfish and malicious node in the IoT. The proposed method combines advantages from reputation and game theory-based methods. The proposed method is a multi-step method that is performed games between nodes in a clustered network when sending or forwarding the node's data packets. Each player independently chooses their own strategy for forwarding or not forwarding; during the game, each player tries to increase their payoffs in the game. The performance of the method has been tested on the network and compared with PPS [14], game theory-based [29], and trust management [32]. The results have shown that the proposed method can detect selfish and malicious nodes efficiently and prevent increasing of end-to-end delay of the data packets to reach the destination and consumption of node resources (energy, battery, memory, etc.). The average throughput which is the percentage of successful delivery of the data packets to the destination is up to 20%, and simultaneously the average end-to-end delay in the delivery of the data packets is reduced by 12%. Also, the percentage of selfish and malicious nodes increased by 10% compared to similar methods, and the false-positive rate and false-negative rate, indicating the accuracy of the selfish node detection, decreased by 8%. Ultimately, the proposed approach gives the malicious and selfish nodes the second opportunity to cooperate with other nodes, and it does not isolate the nodes immediately from the network, because by increasing the percentage of such nodes, the network actually loses its performance.

Abbreviations

DA: Detection accuracy; FN: Number of nodes which are selfish nodes but detected as normal nodes; FNR: False-negative rate; FP: Normal selfish node detected as normal node; FPR: False-positive rate; TN: Total number of normal nodes detected by mistake; TP: Number of selfish nodes detected

Acknowledgements

Not applicable

Authors' contributions

SN and HGG contributed to the main idea and drafted the manuscript, algorithm design, and performance analysis. AK performed the statistical analysis. AMR conceived the study, participated in its design and coordination, and helped to draft the manuscript. All authors read and approved the final manuscript.

Funding

The research was not funded.

Availability of data and materials

We declare that the MATLAB code used for the simulation will not be shared, and we assure that we will send it on demand.

Competing interests

The authors declare that they have no competing interests.

Author details

[1]Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran. [2]Iran Telecom Research Center (ITRC), Tehran, Iran.

Received: 14 May 2019 Accepted: 12 September 2019
Published online: 27 November 2019

References

1. J. Guth *et al.*, Institute of Architecture of Application Systems A detailed analysis of IoT platform architectures: concepts, similarities, and differences (2018).
2. G. Kotonya, IoT architectural framework: connection and integration framework for IoT systems, pp. 1–17, 2018.
3. Liu, Xin, et al. Multi-modal cooperative spectrum sensing based on Dempster-Shafer fusion in 5G-based cognitive radio. IEEE Access 6. 199-208, 2018.
4. Liu, Xin, et al. A novel multi-channel Internet of Things based on dynamic spectrum sharing in 5G communication. IEEE Internet of Things Journal, 2018.
5. F. Olivier, G. Carlos, and N. Florent, New security architecture for IoT network, *Procedia - Procedia Comput. Sci.*, vol. 52, no. BigD2M, pp. 1028–1033, 2015.
6. T. Revathi, Applied fuzzy heuristics for automation of hygienic drinking water supply system using wireless sensor networks, J. Supercomput., 2018.
7. B. Kim, K. Psannis, H. Bhaskar, Special section on emerging multimedia technology for smart surveillance system with IoT environment. *J. Supercomput.* **73**(3), 923–925 (2017)
8. X. Liu et al., 5G-based green broadband communication system design with simultaneous wireless information and power transfer. *Physical Communication* **28**, 130–137 (2018)
9. J. Choi, Y. In, C. Park, S. Seok, H. Seo, and H. Kim, Erratum. Secure IoT framework and 2D architecture for end-to-end security Erratum J Supercomput, *J. Supercomput.*., vol. 2, no. i, p. 112–127, 2016.
10. S. Marti, T. J. Giuli, K. Lai, and M. Baker, Mitigating routing misbehavior in mobile ad hoc networks, *Proc. 6th Annu. Int. Conf. Mob. Comput. Netw. MobiCom 00*, vol. 1, no. 18, pp. 255–265, (2000).
11. S. Buchegger, Performance analysis of the CONFIDANT protocol (Cooperation Of Nodes: Fairness In Dynamic Ad-hoc NeTworks) background: the DSR protocol, pp. 226–236.
12. S. Bansal and M. Baker, Observation-based Cooperation Enforcement in Ad hoc Networks. http://arxiv.rog/pdf/cs.NI/0307012. (2003)
13. G. B.-M., K.N. B.-C., Z.-K. Chong, S.-W. Tan, Outwitting smart selfish nodes in wireless mesh networks. Int. J. Commun. Syst. **23**(5), 633–652 (2010)
14. A. Jesudoss, S. V. Kasmir Raja, and A. Sulaiman. Stimulating truth-telling and cooperation among nodes in VANETs through payment and punishment scheme, *Ad Hoc Netw.*, vol. 24, no. PA, pp. 250–253, 2015.
15. W. Zhang, S. Zhu, J. Tang, and N. Xiong, A novel trust management scheme based on Dempster–Shafer evidence theory for malicious nodes detection in wireless sensor networks, J. Supercomput., 2017.
16. L. Buttyán and J. Hubaux, Stimulating Cooperation in Self-Organizing Mobile Ad Hoc, pp. 579–592, 2003.
17. Srikanth, B. Detecting selfish nodes in MANETs. Doctoral dissertation, 2014.
18. S. Nobahary, S. Babaie, A credit-based method to selfish node detection in mobile ad-hoc network. *Applied Computer Systems* **23**(2), 118–127 (2018)
19. J. Chen and Y. R. Yang, Sprite: A simple, cheat-proof, credit-based system for mobile ad-hoc networks, vol. 00, no. C, pp. 1987–1997, 2003.
20. R. Kaushik and J. Singhai, MODSPIRITE: a credit based solution to enforce node cooperation in an ad-hoc network, vol. 8, no. 3, pp. 295–302, 2011.
21. J.S.S. Uma, Enhanced intrusion detection & prevention mechanism for selfishness in MANET. *Int. J. Innov. Res. Comput. Commun. Eng.* **3**(10), 10131–10138 (2015)
22. Kumar, Sunil, Kamlesh Dutta, and Girisha Sharma. A detailed survey on selfish node detection techniques for mobile ad hoc networks. Parallel, Distributed and Grid Computing (PDGC), 2016 Fourth International Conference on. IEEE, 2016.
23. K. Balakrishnan, J. D. J. Deng, and V. K. Varshney, TWOACK: preventing selfishness in mobile ad hoc networks, IEEE Wirel. Commun. Netw. Conf. 2005, vol. 4, no. C, pp. 0–5, 2005.
24. Kejun Liu, Jing Deng, P. K. Varshney, and K. Balakrishnan, An acknowledgment-based approach for the detection of routing misbehavior in MANETs, *IEEE Trans. Mob. Comput.*., vol. 6, no. 5, pp. 536–550, 2007.
25. E. M. Shakshuki, S. Member, N. Kang, and T. R. Sheltami, EAACK—a secure intrusion-detection system for MANETs, vol. 60, no. 3, pp. 1089–1098, 2013.
26. M. Bounouni, Acknowledgment-based punishment and stimulation scheme for mobile ad hoc network, J. Supercomput., 2018.
27. Basar, Tamer, and Geert Jan Olsder. Dynamic noncooperative game theory. Vol. 23. Siam, 1999.
28. Z. Ji, W. Yu, K.J.R. Liu, A game theoretical framework for dynamic pricing-based routing in self-organized MANETs. *IEEE J. Sel. Areas Commun.* **26**(7), 1204–1217 (2008)
29. A. H. Networks, Y. Sun, Y. Guo, Y. Ge, S. Lu, and J. Zhou, Improving the transmission efficiency by considering non-cooperation in, vol. 56, no. 8, 2013.
30. M. Touati, R. El-Azouzi, M. Coupechoux, E. Altman, J.M. Kelif, A controlled matching game for WLANs. *IEEE J. Sel. Areas Commun.* **35**(3), 707–720 (2017)
31. C. Vijayakumaran, An integrated game theoretical approach to detect misbehaving nodes in MANETs, pp. 173–180, 2017.
32. W. Zhang et al., A novel trust management scheme based on Dempster–Shafer evidence theory for malicious nodes detection in wireless sensor networks. J. Supercomput. **74**(4), 1779–1801 (2018)
33. K. J. Sathish, M.A. Zaveri, Hierarchical clustering for dynamic and heterogeneous Internet of Things. Procedia Computer Science **93**, 276–282 (2016)
34. T. Sheltami, et al. Video transmission enhancement in presence of misbehaving nodes in MANETs. *Multimedia Systems* **15**(5), 273–282 (2009)