TQOR: Trust-based QoS-Oriented Routing in Cognitive MANETs

Shima Asaadi*, Kiana Alikhademi†, Farshad Eshghi‡ and Juan E. Gilbert†
*Faculty of Computer Science
Technische Universität Dresden, Dresden, Germany 01062
Email: shima.asaadi@tu-dresden.de
†Computer and Information Science and Engineering
University of Florida, Gainesville, Florida 32611-612
Email: {kalikhademi, juan}@ufl.edu
‡Electrical and Computer Engineering Dept., Faculty of Eng.
Kharazmi University, Tehran, Iran 15719-14911
Email: farshade@khu.ac.ir

***Abstract*—Dynamicity and infrastructure-less nature of MANETs expose the routing in such networks to a variety of attacks, and moreover, make the conventional fixed policy routing algorithms inefficient. To deal with the routing challenges and varying behavior of malicious nodes in such networks, employing reinforcement learning algorithms and proper trust models seem promising. In this paper, we introduce a cognition layer in parallel and interacting with the network layer which comprises two cognitive processes: path learning (routing) and trust learning. The first process is based on machine learning algorithms and the latter is based on trust management. We compare our algorithm, TQOR, with a well known trust-based routing protocol, TQR, in terms of three measures of performance. The simulation results show better end-to-end delay and communication overhead which further improve as time progresses, without sacrificing the data packet delivery ratio.**

I. Introduction

Mobile Adhoc Networks (MANETs) comprise mobile devices connected by wireless links in a self-configuring and infrastructure-less fashion. Each device in a MANET can move independently in any direction, therefore, the topology of the network changes frequently. Each node in the network must forward transit traffic as well as its self-originating packets [1].

The main advantage of a MANET is its flexibility since there are no constraints on node movement and network topology [2]. However, the dynamicity of MANETs caused by node mobility has been a major challenge regarding routing in such networks [3]. Also, due to the mobility of nodes in an autonomous manner and lack of centralized control, MANETs are subjected to a variety of interior and exterior attacks by malicious nodes. Hence, security would be another important challenging issue in MANETs where the source node relies on intermediate relay nodes to deliver its information [4].

Many recent research efforts on secure routing in MANETs have focused on various trust computing techniques. Blaze et al. [5] introduced the term "Trust Management" in security services in networks. Trust management is to assign a trust value to each entity in the network which enables the trustor (e.g. the source) to detect misbehaving nodes. In MANETs, trust management is necessary when participating nodes, without any previous knowledge about each other, must establish a reliable route between a source and a destination [6].

Recently, Machine Learning (ML) techniques have received a considerable research attention in solving routing problems in MANETs [7]. ML algorithms have addressed the routing challenge by enabling wireless nodes to observe, gather information, and learn from their dynamic local operating environment. Thereafter, wireless nodes make efficient routing decisions while satisfying specific application requirements called Quality of Service (QoS) [8], on the fly.

Among different class of ML techniques, Reinforcement Learning (RL) methods are widely used in MANETs. RL is a biological-based ML approach in which the learning agents acquire knowledge by exploring their local operating environment without the need of external supervision [7]. Q-learning and SARSA (State-Action-Reward-State-Action) are among the most popular iterative methods in RL. In these algorithms, agents learn the optimal action selection policy given the state of the environment, by exploring the environment iteratively.

Thomas et al. [9] named such an adaptive data network, which optimizes its performance using machine learning algorithms, a Cognitive Network (CN). In cognitive networks, a cognition layer is added within which a cognitive process can perceive the current conditions of the network, and then, decide and act on those conditions. The network learns to make better future decisions from this process, while considering end-to-end goals [10]. So cognition can be used to improve QoS, security and many other network-wide goals.

978-1-5386-1465-5/17/$31.00

A. Contribution

To the best of our knowledge, so far, the trust management aspect of security has not been considered as a cognitive process in cognitive wireless networks. We propose a new trust-based routing protocol for cognitive networks in which the network can benefit from the cognitive properties in the cognition layer in order to construct a trusted sub-network for secure routing. In particular:

1) We propose two cognitive processes, path learning (routing) and trust learning, as learning phases in the cognition layer of a cognitive network. Expected-SARSA [11], [12], a class of RL methods has been used as the path learning process in our routing procedure.
2) We introduce a slight enhancement to an already existing trust model in [13] in the trust learning process to detect black-hole and gray-hole attackers.

Considering that throughput and end-to-end delay are among the most important measures of performance in MANETs, we consider the routing problem under three optimization objectives: delay, delivery ratio and communication overhead.

The rest of the paper is organized as follows. Related works are discussed in Section II. A detailed description of our proposed method is discussed in section III followed by simulation results and discussions in section IV. Finally, the conclusion remarks are drawn in section V.

II. Related Work

In this section, we study the recent works that focus on trust management methods for improving security, and machine learning schemes for intelligent routing.

A. Trust management in MANETs:

In MANETs, trust can be considered as the reliance on a node to forward the packets or offer services timely, integrally, and reliably. It is computed based on the evidence generated by the previous and current interactions among nodes within a network [1]. The current value of trust can be built both directly and indirectly. In direct trust computation, a node evaluates the target node based on direct observations of its behavior and the interactions it has with the target node. In indirect trust computation, also called recommendation, the node gathers second hand information from other available nodes, which have information or opinions about a target node [6], [14].

In [15], a reputation and trust scheme is introduced in order to identify and avoid malicious nodes. Reputation is the perception that peers form about a node. Reputation is passive while trust is active. The mechanism does not consider the recommendation trust and historical information, therefore, the trust evaluation process may not be accurate enough.

Hui et al. [16] present a trust prediction model to evaluate the trustworthiness of the nodes in a MANET, which is based on the nodes' historical behaviors and fuzzy logic prediction method rules. A recommendation level is assigned to every node on a specific path and if this level is less than a pre-specified threshold, its recommendation will not be considered. Recommenders are not necessarily one-hop neighbors and generated recommendations travel through a path which might consist of misbehaving intermediate nodes resulting in misinformation.

In [1] and [13], a light-weight trust-based QoS routing algorithm for MANETs is designed. The proposed algorithms ensure the forwarding of packets through the trusted routes with least link delays. This is achieved by monitoring the forwarding behavior of other nodes and considering the target QoS constraints, using direct and indirect information. The drawback of the proposed trust models is that nodes trust all recommenders equally, and the trust may be compromised by some attackers posing as recommenders.

Authors in [17] propose a modified version of Adhoc On-Demand Distance Vector (AODV) protocol for cooperative wireless networks. In every step, the Final Trust Value (FTV) for a node is computed based on direct and indirect trust. After every packet transmitted, the trustor node switches to promiscuous mode seeking acknowledgment from neighbors. The number of acknowledged packets will be used in computation of the direct trust. Indirect trust is calculated in cases where direct trust is less than 0.5 (which is the initial trust for all nodes). In this model, every node will participate in the routing based on its FTV and energy level. For the best route selection, each node launches a down-counting timer and upon its expiry chooses the best route between all the routes for which it has received the related Route Request (RREQ) packet.

The trust management scheme proposed in [18] also comprises two components: direct trust and indirect trust. The direct trust value is obtained by Bayesian model and the indirect one is computed based on belief functions in Dempster-Shafer theory. In Wei's proposed routing protocol, the packet delivery ratio and throughput are improved while the average end-to-end delay and communication overhead are increased.

Trust management is also applied to other type of networks such as Delay Tolerant Networks (DTN). A dynamic trust management model for routing in DTNs is proposed by Malathi and Jayashri [19]. DTN recognizes selfish and malicious nodes based on their histories and send the information into Information Centric Networkings (ICN). ICNs change the path based on the information received from the DTNs. The trust concept in this work is based on whether the nodes have used their available capacity for participation in the data transmission or not. Comparing to other works, packet delivery ratio is improved, however, the communication overhead increases. Moreover, the criteria for recognizing the malicious nodes by the DTN is not clear enough.

Authors in [20] use a metric consisting of energy consumption and data forwarding ratio of sensor nodes to satisfy the data security in Wireless Sensor Networks(WSN). In their proposed algorithm, they use the broadcasting properties of WSNs and find the best wireless link by

applying the proposed metric. This work does not consider QoS such as end-to-end delay which can play a major role in opportunistic routing in WSNs.

B. Intelligent routing in wireless networks:

A cognitive routing algorithm based on SARSA is proposed in [21]. Rewards are computed using energy consumption rate of nodes, but each node receives the same reward for all the flows in the route discovery procedure, regardless of each flow's condition. Results show balanced energy depletion among all nodes.

A routing protocol based on Q-learning algorithm is proposed in [15]. Rewards in the learning phase are computed based on the distance of the next hop to the destination, so the nodes need to know the position of their neighbors which results in high communication overhead. Also, QoS requirements are not considered during the routing.

A MANET routing protocol which uses network status information such as link stability and bandwidth efficiency in selecting a route is introduced in [22]. The protocol uses distributed Q-learning to infer the above-mentioned information. It can efficiently handle network mobility by preemptively switching to a better route before current route fails.

A real-time Q-learning-based routing algorithm in MANETs is proposed in [23] which considers link stability and route shortness as the optimization objectives. Here, the action is to select a group of nodes in the first step, followed by the selection of a node within the group which offers lowest number of hops to the destination. The results show better performance than some previous approaches such as ant-colony based routing algorithm.

A multi agent RL-based routing protocol in MANETs is proposed by [24]. They formulate the routing problem as a Markov Decision Process (MDP) which captures both link delay and energy consumption as the MDP state dynamics, and define a cost function for a single-hop model based on these metrics.

III. Proposed Method:TQOR

In this work, we propose a Trust-based QoS-Oriented Routing in cognitive MANETs (TQOR) algorithm based on AODV protocol [25] at network layer. The cognition layer of the CN consists of two cognitive processes as learning phases, path learning (routing) and trust learning. We use expected-SARSA [11], [12], a class of RL methods, as the path learning process. The expected-SARSA is adopted due to its lower updates variance and faster convergence [11]. Trust learning is done according to a model which is the result of an slight enhancement we made to an already existing trust model in [13]. Based on these phases, each node interacts with its environment, using an RL method, to learn the best path to deliver its packets. Moreover, and at the same time, each node interacts with other nodes in order to learn their trustworthiness (trust learning). In what follows, first we introduce path learning and trust learning phases as has been illustrated in Figure 1 which are then used in the explanation of TQOR protocol.

A. Path learning phase

According to general definition, RL models an agent as a three-tuple $\{\mathcal{S}, \mathcal{A}, \mathcal{R}\}$. $\mathcal{S}$ denotes the set of states of environment which the agent observes. An agent has a set of available actions $\mathcal{A}$. At time step n, agent i observes state $s_i^n \in \mathcal{S}$, and based on its knowledge about the environment, it takes an action $a_i^n \in \mathcal{A}$ and receives a reward $r_i^{n+1} \in \mathcal{R}$. $\mathcal{R}$ can be considered as a subset of real numbers specific to the problem in hand. In RL, the long-term rewards that an agent can expect to receive for each possible action $a_i^n \in \mathcal{A}$, taken in state $s_i^n \in S$, is called the Q-value of that state-action pair. Each agent keeps track of the Q-values of possible state-action pairs $(Q(s_i^n, a_i^n))$ in a Q-table.

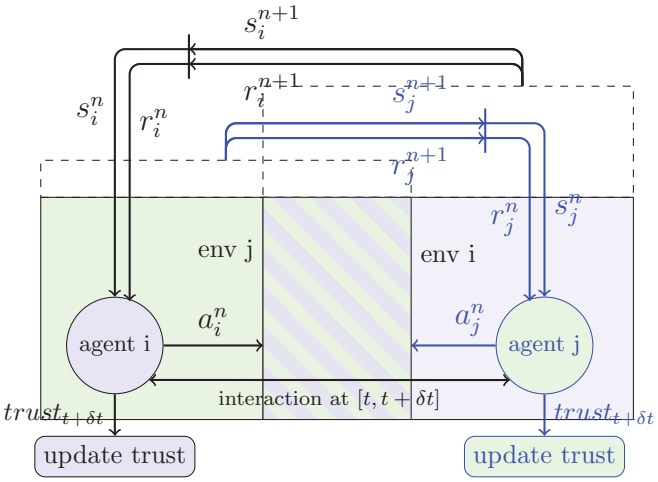


Fig. 1: Learning phases

Now, considering the network nodes as the agents of the RL method, we define the RL components of our routing protocol as follows:

- The whole network excluding a particular node is considered as its environment.
- Each state $s_i^n \in \mathcal{S}$ $(i \in \{1, \ldots, k\}$ in which k is the number of nodes in the network) is defined as the node which currently holds the RREQ packet at time index n.
- Each action $a_i^n \in \mathcal{A}$ represents the selection of a node i from the set of nodes directly connected to s_i, as the next RREQ forwarding node, at time index n.
- $Q(s_i^n, a_i^n)$ denotes the Q-value of selecting an action a_i at the state s_i at time index n, and ending up at state s_i^{n+1}. When an agent selects its next hop to forward the RREQ to, this value is updated.
- The reward $r_i^n \in \mathcal{R}$ is dependent on the quality of the path taken to the destination. Whenever a node selects an action a_i^n(selects the next forwarding node), the state of the environment is changed to a new state s_i^{n+1} and the node receives a new reward $r_i^{n+1} \in \mathcal{R}$ accordingly.

According to Figure 1, each node in the network (e.g. $agent\ j$) individually performs the learning task by consid-

ering other nodes (e.g $agent\ i$) as part of its environment, to achieve its end-to-end goals.

We assume that, at the beginning of the communication process, nodes know nothing about the whole network and all Q-values are initialized to 0. In general, the corresponding Q-value update rule according to [11] is the weighted sum of the old Q-value and the learned Q-value which the latter consists of the immediate reward and the future Q-value as follows:

$$Q(s_i^n, a_i^n) \leftarrow (1-\alpha)\cdot Q(s_i^n, a_i^n) + \alpha\Big(r_i^{n+1} + \gamma \sum_{a_i^{n+1}} \pi(s_i^{n+1}, a_i^{n+1})\cdot Q(s_i^{n+1}, a_i^{n+1})\Big) \quad (1)$$

In (1), α and γ are fixed learning rate and discount factor respectively, $\pi(s_i^{n+1}, a_i^{n+1})$ is the next intermediate forwarding node ε-greedy based selection policy, and r_i^{n+1} is the immediate reward. As mentioned earlier, Expected-SARSA algorithm is adopted as our RL method. In expected-SARSA the future Q-value is the weighted sum of all possible actions as follows:

$$\sum_{a_i^{n+1}} \pi(s_i^{n+1}, a_i^{n+1})\cdot Q(s_i^{n+1}, a_i^{n+1}) = (1-\varepsilon)\cdot \max_{a_i'^{n+1}} \Big(Q(s_i^{n+1}, a_i'^{n+1})\Big) + \varepsilon \cdot mean_{a_i^{n+1}} Q(s_i^{n+1}, a_i^{n+1}) \quad (2)$$

in which the best action with maximum Q-value, $a_i'^{n+1}$, is selected with probability of $1-\varepsilon$ and the mean of the Q-value of all other actions, a_i^{n+1}, are considered with the weight of ε .

The immediate reward r_i^{n+1} in (1) is considered to be equal to the inverse of the total delay taken to deliver the packet to the final destination through the next intermediate forwarding node. Total link delay is dependent on the path quality of that forwarding node to the destination. Path quality is represented through the Expected Transmission count (ETX) [26]. It estimates the number of transmissions required to deliver a packet to its final destination, by measuring the packet loss ratios between pairs of neighboring nodes towards the destination.

In our protocol, we use hello packets to obtain the ETX value. In a time interval $[t-\Delta t, t]$, a node i maintains the number of hello messages it broadcasts (e.g. $h_i[t-\Delta t, t]$) and records the number of hello messages it hears back from a neighbor node j's broadcast(e.g. $h_j[t-\Delta t, t]$). These values are carried with hello messages to the neighbors. ETX_{ij}^t corresponding to the ETX between node i and j at time t is computed by the following formula:

$$ETX_{ij}^t = \frac{1}{\frac{h_j}{h_i}} \quad (3)$$

The delay of a link consists of the transmission delay, the propagation delay, and also the queuing delay which is ignored in our computations.

So, the total delay at time t, d_{ij}^t, is computed based on ETX and the packet transmission time, t_pkt, as in the following formula (ignoring propagation and queuing delays):

$$d_{ij}^t = ETX_{ij}^t \times t\_pkt. \quad (4)$$

This parameter estimates the average total delay needed to deliver a packet to a neighboring node.

One common challenge in RL algorithms is to find an adequate trade off between exploration and exploitation. In this paper we apply ε-greedy strategy to select an action among all possible actions. In ε-greedy, the agent selects the best action (one that optimizes the Q-value function), with probability of $1-\varepsilon$, and selects uniformly an action at random with probability of ε, where $0<\varepsilon<1$. The adequate value of ε is obtained experimentally.

B. Trust learning phase

1) Trust model: In the trust learning phase of the algorithm each node interacts with its neighbors to learn their trustworthiness. As shown in Figure 1, $agents\ i$ and j interact with each other in a predetermined time interval $[t, t+\delta t]$ and, as a result, update the trust value of its own neighbors in time intervals of δt seconds. In order to decide about the trustworthiness of a node, a trust threshold λ is specified. If the trust value is equal or more than the threshold, the node is trusted. Otherwise, the node is untrusted and since there is no reconsideration, the node will be kept isolated until the end of the network's lifetime. Node isolation might be caused by false decisions and leads to a negative impact on the final results. This will be addressed as our future works.

Each node uses direct and indirect trust to detect and isolate black-hole and gray-hole attacks. In black hole attack, a malicious node drops all data packets which it is supposed to forward. However, it participates in the routing process in order to remain as a trusted node. In gray hole attack, a malicious node selectively drops data packets with a random dropping probability of 0.5, however, similar to black hole attack, it participates in the routing process. Trust between two neighbor nodes is assumed to be asymmetric. Also, we use previous (representing historical trust) and current trust evaluations for calculating the total/up-to-date trust. Inclusion of the previous trust evaluation in the calculation of the up-to-date trust is to not let abrupt trust level changes appear due to random occurrence of gray-hole attacks.

2) Trust computation: Direct trust is computed based on the interaction of two neighbor nodes at time t, using the following formula:

$$DT_{ij}^t = \frac{f_j}{f_{i,j}} \quad (5)$$

wherein node i is the trustor, node j is the trustee, and $f_{i,j}$ and f_j are respectively the number of packets forwarded from node i towards node j and the number of packets forwarded by node j during the time interval $[t-\delta t, t]$.

Each node uses recommendations from its neighbor nodes,

which have an evaluation about the trustee/target node. Since recommenders may be malicious nodes, node i assigns a weight to each of its recommenders based on their (total) trust value. So, the indirect trust level of the trustee node is computed based on the average weighted sum of the recommenders' corresponding trust values, as follows:

$$IT_{i,j}^{t} = \frac{\sum_{k \in N_i} T_{i,k}^{t\text{-}1} \cdot T_{k,j}^{t\text{-}1}}{N_i} \quad (6)$$

where N_i is the number of node i's neighbors excluding the target node, and $T_{i,k}$ and $T_{k,j}$ represent the total trust values of i towards k and k towards j, respectively. On the other hand, current trust of a particular node is computed based on the weighted sum of direct and indirect trusts, as follows:

$$CT_{i,j}^{t} = \omega_1 DT_{i,j}^{t} + \omega_2 IT_{i,j}^{t} \; ; \qquad \omega_1 + \omega_2 = 1 \; , \; \omega_1 > \omega_2 \quad (7)$$

where more weight has been put on the direct information resulted from a node's own observation and considered to be more reliable. Then, we use the weighted sum of the current trust and the historical trust for updating the total/up-to-date trust at time t, using the following formula:

$$T_{i,j}^{t} = \omega_1' CT_{i,j}^{t} + \omega_2' T_{i,j}^{t\text{-}1} \; ; \qquad \omega_1' + \omega_2' = 1 \; , \; \omega_1' > \omega_2' \quad (8)$$

As recent information is more important, we assign more weight to the current trust rather than the historical one.

C. TQOR Protocol

In TQOR, the cognition layer is in connection with the network layer to achieve end-to-end goals. The neighbor table and learning processes are implemented in this layer.

1) Protocol entities: The core entities of the protocol are the hello messages, the route request (RREQ), the route reply (RREP), the neighbor table, and finally the routing table. Since the RREQ and RREP are the same as in AODV protocol [25], just the remaining entities are introduced in the following.

Hello Messages: Compared to the hello message format in AODV, additional fields exist which hold the number of hello messages received from each current neighbor in the most recent time interval. As mentioned earlier, the information in these fields allow a node to determine the ETX of all its current neighbors.

Neighbor Table: Each node in the network maintains a separate neighbor table n_table_d (containing the information of the neighbors) for each destination d, during the route discovery phase. Note that in our protocol, to avoid extra storage consumption (resulting in efficient space complexity), the Q-values of the neighbors are also stored in the neighbor table. The configuration of the neighbor table of each node is illustrated in Figure 2. As shown in the figure, the information associated with each neighbor appears in four fields. The first field contains the ID of the neighbor node. The next field represents the Q-value for selecting this node as the next hop in the routing towards the intended destination. The T-value field maintains the trust value of the node about this neighbor ID. The Q-value and T-value fields are filled during the path learning and trust learning phases. Finally, the calculated ETX to this neighbor is recorded in the neighbor table. Since each node stores the information of its neighbors in a neighbor table, it anyways incurs extra storage cost compared to AODV protocol. However, this extra storage cost has a polynomial relation of $O(n^2)$ with the network size.

Routing table: In each node, the routing table conventionally maintains the next hop neighbors towards all final destinations. Since we are dealing with MANETs, each entry in the routing table has a lifetime. In our protocol, the routing table is constructed from the information of neighbor tables as follows. For each destination, the corresponding neighbor table is checked. From the neighbors with higher-than-threshold T-values (trusted neighbors), the next hop neighbor is selected to forward the RREQ by applying the routing policy to the Q-values of the trusted neighbors. This selection is reflected in Figure 2, where node ID 4 is selected by the current intermediary node as the forwarding node towards the destination node ID 1.

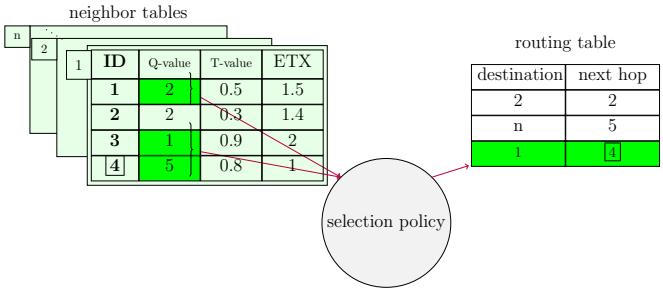


Fig. 2: Neighbor tables inside a node

Now we can proceed with explaining our routing protocol in a step by step manner.

2) Protocol description: When the upper layer at the source requires sending a data packet, network layer protocol should look up route in its routing table. If no route exists or it has expired, our protocol launches the route discovery phase which consists of path learning and trust learning. The detailed process is given as follows:

- **Step 1:** The source node s looks for a route entry to destination node d in its local routing table. If such a route is found, it proceeds with sending the data packets accordingly. Otherwise, the source node s initiates a route discovery procedure by sending out a new RREQ to a next hop node selected as below.
- **Step 2:** Upon initiation of a new RREQ (as is the case with the source node) or receiving an RREQ (as is the case with the intermediate nodes), the subject node checks for the existence and time validity of a next hop forwarder in its routing table for the destination of that RREQ. If a next hop forwarder does not exist or its validity has expired, the node checks the availability of the neighbor table n_table_d associated with the destination d for that RREQ. If there is no table, it constructs a new one; and otherwise, it updates the existing table. Using

the information in the n_table_d, the source node selects a set of neighbors with trust degree equal or greater than a trust threshold λ. The next hop is chosen from this candidate set, based on their Q-values, using ε-greedy selection rule. The ID of the selected next hop node is added to the routing table with a new expiry time, and the RREQ is forwarded to the next hop after adding the ID of the forwarder node to it (the list of forwarders is used by Route Reply (RREP)).

- **Step 3:** The receiving intermediate node checks if it itself is the final destination node. If so, it generates an RREP which is sent back to the source node along the reverse direction of the discovered route and the protocol continues to step 5. Otherwise, it proceeds to step 4.
- **Step 4:** Upon receiving an RREQ, the intermediate node first checks whether it has already received the same RREQ. If so, it drops the RREQ and stops the procedure. Otherwise, it processes the RREQ and proceeds as in step 2.
- **Step 5:** When the source node receives the RREP, it proceeds with transmitting the data packets.

For clarification purposes, it is worth emphasizing that there are five different procedures in our proposed protocol, namely: node movement, ETX calculation, trust calculation, Q-value calculation and routing table update in our proposed protocol. The *node movement* procedure is activated at each time step with predetermined length (τ_m). Although the movement affects the ETX of the nodes, it does not directly initiate any other procedures. ETX calculation procedure is based on observation of how hello packets are received in time intervals with predetermined length (τ_{ETX}). *Trust calculation* procedure is activated in predetermined periods (τ_T) and is based on monitoring past RREQs. *Q-value calculation* and *routing table* update procedures are RREQ-activated.

Our routing protocol discovers the path with higher-than-threshold trust and low link delay values instead of minimum hop count as is in AODV protocol [25]. As we will see shortly, the simulation results show how employing path and trust learning procedures in a cognition layer improves the performance of the routing.

IV. Simulation Results and Discussion

We simulate and compare the performance of TQOR with TQR [13] which is a recently proposed trust-based QoS routing protocol. TQR ensures packet forwarding through trusted and least link delay routes, as its QoS constraint, by computing direct and indirect trusts. TQR shows outperformance against AODV protocol as a baseline method, and other previous approaches as well. Moreover, being a trust/AODV-based and QoS oriented routing protocol and considering the same attack models, make TQR a good candidate to compare our proposed method with. As opposed to our work, TQR does not consider cognition which, as will be shown shortly, influence the measures of performance of interest.

Simulation Setup: The simulation has been implemented in the discrete event simulator OMNeT++ (version 4.4.1). We simulate our work in INET framework (version 2.3) of OMNeT++, which is an open-source communication networks simulation package for wireless and mobile networks. All implementations can be downloaded from https://github.com/sasaadi/TQOR-MANET.

TABLE I: Simulation Setup

Parameter	Value
Simulation area	$1000\ m \times 1000\ m$
Number of nodes	50
Concurrent source-destination flows	15
Simulation time	200 s
Number of malicious nodes	6, 10, 14
Data packet size	512 B
Maximum speed of nodes	5 m/s
Hello packet generation interval	1 s
RREQ generation interval	1 s
Data packet generation interval	1 s
Trust threshold	0.5
Trust evaluation time interval	5 s
ETX evaluation time interval	0.5 s
Mobility update interval	100 ms
Learning rate	0.9
Mobility model	Random WayPoint (RWP)
Number of runs	10/scenario
Confident interval level	95 %

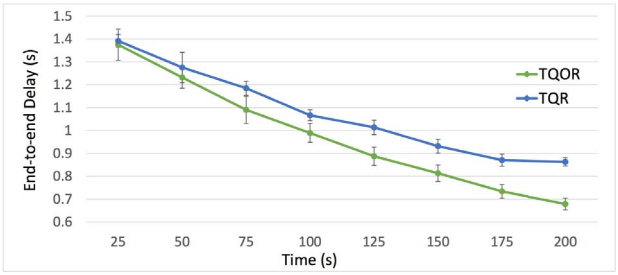
Our simulation setup is as per Table I. Source-destination pairs are randomly selected to generate traffic concurrently. Malicious nodes are chosen randomly out of 50 nodes. To reach the target confidence interval, each individual result is the average of 10 runs. Trust level of each node is quantified by a number in continuous interval [0, 1] with 0 and 1 indicating complete distrust and complete trust respectively. The initial trust level value of all nodes is set to 0.5 which means trust neutrality.

Herein, we are not explicitly concerned with MAC layer issues of wireless networks which are taken care of by the adopted simulation package. Not targeting WSNs, we are not concerned with the energy consumption of mobile nodes herein.

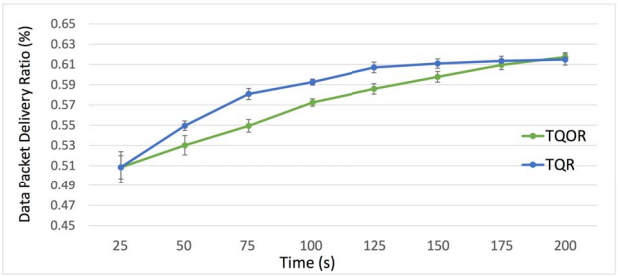
We simulate black-hole and gray-hole attackers as defined in [13]. Slander attackers which might send false recommendations are not considered in this work.

We consider average end-to-end delay, packet delivery ratio and communication overhead as our measures of performance. It should be noted that hello packets are not considered as communication overhead since they are not intended just to serve routing purposes.

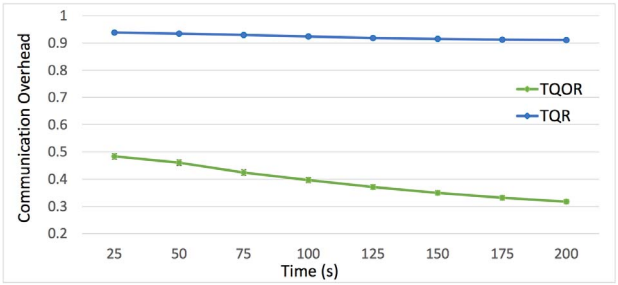
Results and Discussion: Figure 3 compares the performance of two protocols against the simulation time. These results are obtained by considering 6 malicious nodes in the network. Figure 3a shows the average end-to-end delay versus simulation time. As per the very nature of learning-based routing algorithms, by collecting more information about the environment over time, better routes are discovered. The results show that TQOR outperforms TQR and this outperformance grows significantly over time. This outperformance can be attributed to the delay being considered as the main parameter used for choosing the next action in the learning phase in TQOR, as opposed to TQR which considers the link



(a)



(b)



(c)

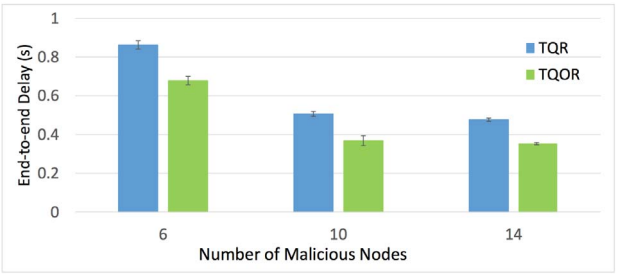
Fig. 3: Comparing TQOR and TQR in terms of performance measures of interest against simulation time. a) average end-to-end delay. b) data packet delivery ratio. c) communication overhead.

delay as the QoS constraint in routing. The results suggest that by adding the path learning phase to the protocol, more complete information about the discovered routes is acquired.

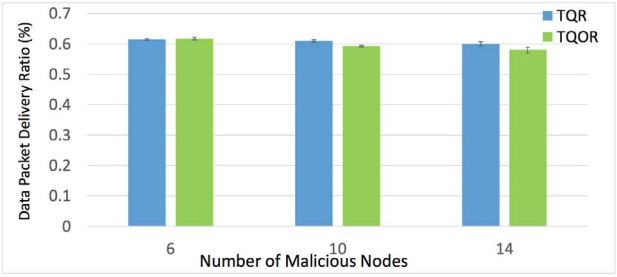
Regarding data packet delivery ratio, as Figure 3b shows, delivery ratio in both protocols improves as time passes. The ascendancy of the curves emphasizes that more accurate trust values become available as time goes on, and the routes with no malicious nodes are used for packet delivery. Regarding packet delivery ratio, TQOR performs almost the same as TQR, though at times slightly underperforms it. This is because in TQOR finding the most trusted node may take more time. While both protocols perform the same initially, TQR enjoys a faster delivery ratio improvement for some time. In the long term, not only TQOR catches up, but also seems to demonstrate better asymptotic performance.

Another effect of employing intelligent routing is also obvious from Figure 3c which shows a significant difference in the communication overhead between TQOR and TQR. This is mostly because TQOR unicasts RREQs during the route discovery using the RL method, as opposed to RREQ broadcast in TQR. Also, in TQR, to keep the most recent valid routes more control packets are needed continuously to check for the new connections. However, in TQOR the path learning phase finds the most stable routes which reduces the number of control packets significantly during the simulation time. Moreover, as time passes less control packets are produced in TQOR compared to TQR.

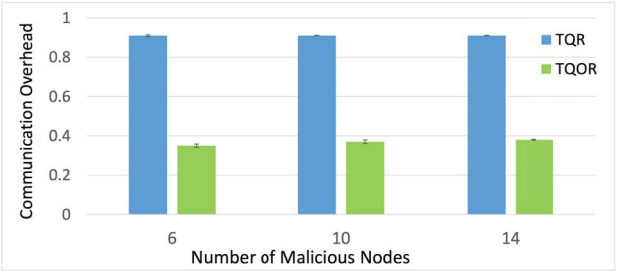
In Figure 4 we compare the performance of two protocols by varying the number of malicious nodes at a time instant of 200s. The results show the stability (resistance) of both protocols against increasing the number of malicious nodes.



(a)



(b)



(c)

Fig. 4: Comparing TQOR and TQR in terms of performance measures of interest against the number of malicious nodes. a) average end-to-end delay. b) data packet delivery ratio. c) communication overhead.

As shown in Figure 4a, the end-to-end delay in TQOR outperforms TQR for all number of malicious nodes which points to the efficiency of incorporating the path learning phase, and also partly as a result of lower communicational overhead of TQOR. The decline of the delay with the increase in the number of malicious nodes confirms the results obtained

in TQR, which shows that the discovered paths contain lower delay overhead through the routing process.

The data packet delivery ratio versus different number of malicious nodes is shown in Figure 4b. As is observed, TQOR performs almost the same as TQR, and the slight difference is because of the extra time required for finding the most trusted route in TQOR. This slight difference will die down in longer runs. The interesting point is the insensitivity of the data packet delivery ratio to the number of malicious nodes in TQOR, which confirms the robustness of the protocol.

According to Figure 4c, for all scenarios, TQOR outperforms TQR significantly, due to TQOR's unicast RREQs as opposed to TQR's broadcast RREQs in route discovery.

V. Conclusion and Future Works

In this paper, we propose a new trust-based QoS oriented routing in cognitive MANETs. In particular, we propose a cognition layer comprising two interrelated cognitive processes, path learning and trust learning. We use expected-SARSA as the path learning process in our routing protocol and the trust learning is done according to our enhanced version of the model in TQR. Owing mainly to the employment of reinforcement learning and unicast RREQs, simulation results show that the TQOR outperforms TQR in terms of end-to-end delay and communication overhead, while achieving nearly the same delivery ratio in the long run. Moreover, the stability of different measures of performance with varying number of malicious nodes show the robustness of TQOR against the dynamic topology of wireless networks. As the future work, we plan to consider a more flexible trust model in unsecured networks as follows. To prevent ordinary nodes from making unreliable routing decisions, reconsideration of isolated nodes will be added to the model, which might improve data packet delivery ratio. Moreover, minimizing energy consumption of wireless nodes will be considered as one of the optimization parameters.

VI. Acknowledgement

Authors would like to thank Babak Alipour for his constructive comments and feedback.

References

[1] C. Qu, L. Ju, Z. Jia, H. Xu, and L. Zheng, "Light-weight trust-based on-demand multipath routing protocol for mobile ad hoc networks," in *12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*, 2013, pp. 42–49.

[2] K. Wang, W.-C. Wong, and T. Y. Chai, "A manet routing protocol using Q-learning method integrated with bayesian network," in *IEEE International Conference on Communication Systems (ICCS)*, 2012, pp. 270–274.

[3] G. Weiß, *The Biology and technology of intelligent autonomous agents*. Springer Berlin Heidelberg, 1995, ch. Distributed Reinforcement Learning, pp. 415–428.

[4] R. Changiz, H. Halabian, F. R. Yu, I. Lambadaris, and H. Tang, "Trust management in wireless mobile networks with cooperative communications," in *8th IEEE/IFIP 8th International Conference on Embedded and Ubiquitous Computing (EUC)*, 2010, pp. 498–503.

[5] M. Blaze, J. Feigenbaum, and J. Lacy, "Decentralized trust management," in *Preceedings of 1996 IEEE Symposium on Security and Privacy*, 1996, pp. 164–173.

[6] J.-H. Cho, A. Swami, and R. Chen, "A survey on trust management for mobile ad hoc networks," *IEEE Communications Surveys and Tutorials*, vol. 13, no. 4, pp. 562–583, 2011.

[7] H. A. A. Al-Rawi, M. A. Ng, and K.-L. A. Yau, "Application of reinforcement learning to routing in distributed wireless networks: a review," *Artificial Intelligence Review*, vol. 43, no. 3, pp. 381–416, 2015.

[8] G. Santhi, A. Nachiappan, M. Z. Ibrahime, R. Raghunadhane, and M. Favas, "Q-learning based adaptive QoS routing protocol for MANETs," in *International Conference on Recent Trends in Information Technology (ICRTIT)*, 2011, pp. 1233–1238.

[9] R. W. Thomas, L. A. DaSilva, and A. B. MacKenzie, "Cognitive networks," in *First IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, Nov 2005, pp. 352–360.

[10] R. W. Thomas, D. H. Friend, L. A. Dasilva, and A. B. Mackenzie, "Cognitive networks: adaptation and learning to achieve end-to-end performance objectives," *IEEE Communications Magazine*, vol. 44, no. 12, pp. 51–57, 2006.

[11] v. S. H., v. H. H., W. S., and W. M., "A theoretical and empirical analysis of Expected Sarsa," in *IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning*, March 2009, pp. 177–184.

[12] R. S. Sutton and A. G. Barto, "Reinforcement learning: An introduction," 1998.

[13] B. Wang, X. Chen, and W. Chang, "A light-weight trust-based QoS routing algorithm for ad hoc networks," *Pervasive and Mobile Computing*, vol. 13, pp. 164–180, 2014.

[14] H. Yu, Z. Shen, C. Miao, C. Leung, and D. Niyato, "A survey of trust and reputation management systems in wireless communications," *Proceedings of the IEEE*, vol. 98, no. 10, pp. 1755–1772, 2010.

[15] Y. Naputta and W. Usaha, "RL-based routing in biomedical mobile wireless sensor networks using trust and reputation," in *International Symposium on Wireless Communication Systems (ISWCS)*. IEEE, 2012, pp. 521–525.

[16] X. Hui, J. Zhiping, J. Lei, L. Xin, and H.-M. S. Edwin, "Impact of trust model on on-demand multi-path routing in mobile ad hoc networks," *Computer Communications*, vol. 36, no. 9, pp. 1078 –1093, 2013.

[17] U. Venkanna, J. K. Agarwal, and R. L. Velusamy, "A cooperative routing for MANET based on distributed trust and energy management," *Wireless Personal Communications*, vol. 81, no. 3, pp. 961–979, 2015.

[18] Z. Wei, H. Tang, F. R. Yu, M. Wang, and P. Mason, "Security enhancements for mobile ad hoc networks with trust management using uncertain reasoning," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 9, pp. 4647–4658, 2014.

[19] M. Malathi and S. Jayashri, "Design and performance of dynamic trust management for secure routing protocol," in *IEEE International Conference on Advances in Computer Applications (ICACA)*, 2016, pp. 121–124.

[20] N. Kumar and Y. Singh, "An energy efficient and trust management based opportunistic routing metric for wireless sensor networks," in *Fourth International Conference on Parallel, Distributed and Grid Computing (PDGC)*, 2016, pp. 611–616.

[21] S. Chettibi and S. Chikhi, "An adaptive energy-aware routing protocol for MANETs using the SARSA reinforcement learning algorithm," in *IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS)*, 2012, pp. 84–89.

[22] C. Wu, K. Kumekawa, and T. Kato, "A MANET protocol considering link stability and bandwidth efficiency," in *International Conference on Ultra Modern Telecommunications and Workshops*. IEEE, 2009, pp. 1–8.

[23] A. Ghaffari, "Real-time routing algorithm for mobile ad hoc networks using reinforcement learning and heuristic algorithms," *Wireless Networks*, vol. 23, no. 3, pp. 703–714, 2017.

[24] M. Maleki, V. Hakami, and M. Dehghan, "A model-based reinforcement learning algorithm for routing in energy harvesting mobile ad-hoc networks," *Wireless Personal Communications*, pp. 1–21, 2017.

[25] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc on-demand distance vector AODV routing," Tech. Rep., 2003.

[26] D. S. De Couto, D. Aguayo, J. Bicket, and R. Morris, "A high-throughput path metric for multi-hop wireless routing," *Wireless Networks*, vol. 11, no. 4, pp. 419–434, 2005.