Energy-Efficient Algorithm for Reliable Routing of Wireless Sensor Networks

Habib Mostafaei

***Abstract*—Quality of service (QoS) routing is one of the critical challenges in wireless sensor networks (WSNs), especially for surveillance systems. Multihop data transmission of WSNs, due to the high packet loss and energy-efficiency, requires reliable links for *end-to-end* data delivery. Current multipath routing works can provision QoS requirements like end-to-end reliability and delay, but suffer from a significant energy cost. To improve the efficiency of the network with multiconstraints QoS parameters, in this paper we model the problem as a multiconstrained optimal path problem and propose a distributed learning automaton (DLA) based algorithm to preserve it. The proposed approach leverages the advantage of DLA to find the smallest number of nodes to preserve the desired QoS requirements. It takes several QoS routing constraints like end-to-end reliability and delay into account in path selection. We simulate the proposed algorithm, and the obtained results verify the effectiveness of our solution. The results demonstrate that our algorithm has a better performance than current state-of-the-art competitive algorithms in terms of end-to-end delay and energy-efficiency.**

***Index Terms*—Distributed learning automaton (DLA), quality of service (QoS), routing, wireless sensor networks (WSNs).**

I. Introduction

Wireless sensor networks (WSNs) consist of small and tiny devices that have been scattered over a distant area to monitor many events such as border surveillance, temperature, and humidity [1]–[11]. These small nodes suffer from many resource constraints such as battery and computation. Therefore, they should be preserved as long as possible to reach the monitoring goals [12]–[14]. It is also necessary to design the algorithms that consume low resources from sensor devices.

Providing reliability for communication links in WSNs is a challenging problem due to leveraging shared error-prone communication medium, which causes packet loss. Multiple retransmissions may need to forward a packet at each hop resulting in resources consumption and long delays [15], [16].

Provisioning quality of service (QoS) for routing is a challenging issue in WSNs. QoS routing should take into account multiple constraints such as end-to-end reliability, delay, and energy-efficiency. Some of these constraints are contradictory and only soft QoS provisioning is achievable. Here, soft QoS means that the route meets the QoS requirements with a probability that is good enough for the specific purpose [15].

Several works have been done to utilize multiple paths to meet the QoS requirements [17]. Surely, sending the sensed packets via multiple paths can provision the QoS requirements, but it has some disadvantages such as exploiting more nodes results in consuming more energy from power constraints nodes and sending packets from multiple links may create interference on wireless channels and retransmissions are needed [15]. Therefore, selecting a single path with multiple constraints leads to multiconstrained optimal path (MCOP) problem, which is an NP-complete problem [18].

In this paper, we propose reliable routing with distributed learning automaton (RRDLA) algorithm, which considers dynamics of links in finding a path from a source to a destination by considering QoS constraints such as end-to-end reliability and delay. The rationale behind leveraging DLA is that it requires $O(N)$ to find a solution for NP-complete problems [19], [20].

RRDLA exploits a pruning rule to improve the convergency of the algorithm, and it saves more energy for resource constraints sensor nodes.

The main contributions of this work are as the follows.

1) We model the WSN with a network of learning automaton (LA) and the problem as multiconstrained QoS routing.
2) We propose a distributed algorithm to find a small number of nodes with highly reliable links between them to transfer the data.
3) Through several performance comparisons, we demonstrate the effectiveness of our approach for multiconstrained QoS routing in WSNs.

The rest of this paper is organized as follows. Section II reports state-of-the-art works in the reliable routing of WSNs. Section III shows our network model. The basic concept of LA is introduced in Section IV. The proposed routing algorithm is shown in Section V. Section VI demonstrates the results of our method. Section VII concludes our work.

II. Related Work

The authors of [21] provided a good survey on routing in WSNs by categorizing the recent research works into several directions as well as discussing the open research directions.

Manuscript received November 13, 2017; revised February 10, 2018, April 8, 2018, and July 1, 2018; accepted August 20, 2018. Date of publication September 13, 2018; date of current version February 28, 2019.

The author is with the Department of Engineering, Roma Tre University, 00154 Rome, Italy (e-mail: mostafae@dia.uniroma3.it).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIE.2018.2869345

Several routing protocols have been proposed to increase the efficiency of the network [22]–[28]. In this section, we give an overview of the works that consider the similar QoS requirements in designing the routing algorithms.

Opportunistic routing is widely used to improve the performance of reliable routing in WSNs [17]. It leverages the broadcast features provided by wireless medium to overhear data packets from several nodes and selects one of them as next-hop forwarder. This technique results in reducing the waiting time for the sender. Geographic random forwarding (GeRaF) is an opportunistic routing mechanism that exploits priority among the relay forwarding nodes to send the packets at every single hop [29]. Efficient QoS-aware geographic opportunistic routing (EQGOR) [15] leverages the geographical location information of the nodes to provide a set of prioritized nodes to steer the packets based on QoS requirements such as end-to-end delay. GeRaF and EQGOR exploit several control messages among the nodes to find a suitable sleep scheduling for neighbors, which results in adding an overhead to these protocols. Greedy perimeter stateless routing (GPSR) [30] leverages immediate neighbors in the network topology to send a packet to a destination. It also leverages global positioning system (GPS), which is not easy to install on board each node, and it increases the cost of sensor nodes.

Huang and Fang [31] leveraged multipath diversity to guarantee multiconstrained QoS in WSNs so that the delay and the reliability are main concerns. However, exploiting multiple paths results in more energy consumption. Charfi *et al.* [32] studied multipath data transmission by considering end-to-end reliability and energy consumption. This work finds a tradeoff between reliability and energy consumption, but the protocol causes a high end-to-end delay. DRINA [33] is a lightweight routing protocol that considers the reliability of links in selecting a path for data routing. It builds network's tree from source to sink node and tries to find the route from the tree. However, DRINA does not consider end-to-end delay for route selection.

Li *et al.* [34] proposed an energy-efficient and reliable routing algorithm, named REER, for WSNs by considering unreliable communication links. REER tries to minimize the energy consumption of the nodes to transfer the packet. GAMER [35] also considers the reliability of path in selecting a route to the destination in a network with unreliable communication links. However, REER and GAMER neglect to consider the end-to-end delay in path selection. DETR [36] considers several QoS constraints like end-to-end reliability and delay in path selection by obtaining the information from neighbor nodes. RRDLA algorithm is energy-efficient than DETR and REER in sending network traffic.

In this paper, we propose the RRDLA algorithm to transfer the information of events via provisioning the QoS requirements in the network. The distributed learning automaton (DLA) models the network as a directed graph and considers one QoS constraint to solve the problem, while the RRDLA approach leverages the same graph and takes several QoS parameters into account in finding the solution. In contrast with other works, it can guarantee both reliability and efficiency.

III. Network Model

In this section, we state the network model and problem formulation.

Wireless Sensor Networks: A WSN consists of N randomly deployed nodes $V = \{v_0, v_1, \ldots, v_n\}$ with v_0 as the sink node. Each node in a WSN has a sensing range (R_S) that enables the node to monitor the events or the objects and a communication range (R_C) that enables the node to communicate with other nodes. We use the network model in [37].

In our model, we assume that the MAC layer provides the facility to estimate links quality, i.e., the packet reception ratio (PRR), and each node knows the PRR of its one-hop neighbors. We use PRR to obtain the reliability of links for QoS provisioning.

We model QoS routing as MCOP problem [18], which is an *NP-complete* problem. Consider a network graph *G = (V, E)* where *V* illustrates the set of nodes and *E* indicates the set of edges. Each link in G, i.e, $(u, v) \in E$, is associated with a cost parameter $c(u, v)$ and K additive QoS parameter $w_k(u, v)$, for $k = 1, 2, \ldots, K$ [38]. With k constraints, the problem is to find a path like p from the source to the sink, such that

$$w_k(p) = \Sigma_{(u,v)\in p} \quad w_k(u, v) \leq c(u, v), \text{for } k = 1, 2, \ldots, K \tag{1}$$

and $c(p) = \Sigma_{(u,v)\in p} c(u, v)$ is minimized over all feasible paths satisfying (1). For $k = 1$, the problem is also known as the restricted shortest path problem, which is NP-complete [39].

We consider the end-to-end reliability of each link as the first constraint and the cost function is defined as the number of hops in path p. We also consider the end-to-end delay as the second constraint.

In our model, the probability of successful data delivery through n hops of a path like p if we consider $P(u, v)$ as the PRR between u and v can be computed as follows:

$$\text{PDR} = \prod_{\forall(u,v)\in p} P(u, v) \tag{2}$$

where PDR specifies the packet delivery ratio (PDR). Based on the number of sent packets (N_{sent}) and PDR, we can determine the number of successfully delivered packets (N_{success}) to the sink node as

$$N_{\text{success}} = N_{\text{sent}} \times \text{PDR}. \tag{3}$$

To compute single-hop media delay, we use the delay model in [16], where the signal propagation delay is ignored. The delay for path p, i.e., $D(p)$, can be computed as follows:

$$D(p) = \sum_{\forall(u,v)\in p} d(u, v) \tag{4}$$

where $d(u, v)$ is the media delay between u and v.

Formal Definition of the Problem: Considering a WSN with N deployed node including a sink schedule the nodes so that the PDR is maximized, the $D(p)$ minimized, and the total energy consumptions of nodes (E_{total}) is minimized. We can define the overall problem as two subproblems. We leverage links with high PRR to maximize PDR. Mathematically,

$$\text{Maximize} \quad \sum_{t=1}^{T} \text{PDR} \quad \text{subject to} \quad \text{PDR} \geq T_{\text{PDR}} \tag{5}$$

where T is the total time and T_{PDR} is the threshold value for PDR. The constraint PDR $\geq T_{\text{PDR}}$ confirms that the total number of transmitted packets is greater than zero and the threshold value.

We aim at minimizing the end-to-end delay of path p. Mathematically

$$\text{Minimize} \quad \sum_{t=1}^{T} D(p) \quad \text{subject to} \quad D(p) > 0 \tag{6}$$

where T is the total time and the constraint $D(p) > 0$ confirms that the end-to-end delay is greater than zero.

We also aim at minimizing the energy consumption of nodes in each path p from the source to the destination. Mathematically

$$\text{Minimize} \quad \sum_{t=1}^{T} E_{\text{total}}^{p} \quad \text{subject to} \quad E(v_i) > 0, \quad \forall v \in p. \tag{7}$$

where T is the total time. In (7), E_{total}^{p} is the total energy consumption of nodes in path p and for each node v_i in this path, $E(v_i)$ is always greater than zero.

IV. Basics on LA

LA is an abstract model to solve optimization problems. Each LA has a finite set of actions to choose at any time and each of them is associated with a probability. It randomly selects an action and the random environment (RE) generates a reinforcement signal for that action. The LA updates its probability of its action based on the received signal from RE. By iterating this procedure, an LA learns to pick an optimal action among its actions.

The environment can be described as a triple $E = \{\alpha, \beta, c\}$, where $\alpha = \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ indicates the finite input set (or actions), $\beta = \{\beta_1, \beta_2, \ldots, \beta_n\}$ indicates the output set (i.e., the reinforcement signals), and $c = \{c_1, c_2, \ldots, c_n\}$ indicates a set of penalty probabilities, where each element c_i corresponds to one input or action α_i. Environments can be categorized into three models based on reinforcement signals: P-model, Q-model, and S-model. In P-model, the environment β can assume only binary values, i.e., 0 or 1. This means that the output of environment can take one of these two values. Narendra and Thathachar [40] showed that it may need an arbitrary threshold to convert the actual output to a binary value in this model. In Q-model environment, β can take value in the interval (0, 1), which is a generalization of output values for P-Model. Finally, reinforcement signal varies in the generic interval $[a, b]$ in S-model environments. The environment responses in S-model can take a continuous value in this interval. More explanation for the models can be found in [40].

LA is classified into fixed-structure and variable-structure stochastics. For our solution, we consider only variable-structure automaton [40]. A learning algorithm T can be defined as

$$p(n+1) = T[p(n), \alpha(n), \beta(n)] \tag{8}$$

where $p(n)$ is the action probability vector. The automaton operates as follows. Based on the action probability set p, the automaton randomly selects an action α_i and performs it on the environment. After receiving the environment's reinforcement signal, automaton updates its action probability set based on

$$\begin{aligned} p_i(n+1) &= p_i(n) + a(1 - p_i(n)) \quad i = j \\ p_j(n+1) &= (1-a)p_j(n) \quad \forall j, j \neq i \end{aligned} \tag{9}$$

when the chosen action is rewarded by RE (i.e., $\beta(n) = 0$) and

$$\begin{aligned} p_i(n+1) &= (1-b)p_i(n) \quad i = j \\ p_j(n+1) &= \frac{b}{r-1} + (1-b)p_j(n) \quad \forall j, j \neq i \end{aligned} \tag{10}$$

when the chosen action is penalized by RE (i.e. $\beta(n) = 1$). In these two equations, a and b are reward and penalty parameters, respectively. For $a = b$, learning algorithm is called L_{R-P}, for $b \ll a$, it is called $L_{R\varepsilon P}$, and for $b = 0$, it is called L_{R-I}. Some applications of LA in WSNs can be found in [37] and [41].

A. Distributed LA

A distributed LA [42] is a network of LA, which cooperates to solve a problem. Formally, a DLA with an LA is defined by a graph (A, E), where A = {A$_1$, A$_2$, . . ., A$_z$} is the set of LA and $E \subset A \times A$ is the set of edges. Each edge $e(i, j)$ corresponds to action a$_j$ of automaton A_i. DLA acts as follows. At first, the LA of root randomly chooses one of its outgoing edges, which is equal to selecting one node in the network of LA. Then, the corresponding LA to the chosen node will be activated, and this process continues until the leaf automaton (i.e., it can be determined by the RE) is reached. The chosen actions among the traversed path from the root to the leaf node are applied to the RE. The RE gives a reinforcement signal to the DLA. The LA of selected nodes along the path updates their action probability vector based on reinforcement signal of RE. Selecting nodes from root to leaf restarts the same procedure until the action probability of a path reaches a close value to unity. Finally, each DLA has exactly one activated automaton in the root and one in the leaf automaton. DLA has been successfully used in many WSN applications such as target coverage [43], clustering [37], congestion control [41], and backbone formation [44] to state a few.

V. RRDLA Algorithm

In this section, we introduce the RRDLA approach. The RRDLA consists of four phases: *Initial*, *Learning*, *Transmitting*, and *Retransmitting*. In the first phase, the network of LA is formed in DLA manner (i.e. each node leverages an LA) and also the action-set of each LA is formed. Selecting a small number of nodes with high PDR is performed during *Learning* phase. Sending packets via selected nodes are performed in *Transmitting* phase. Undelivered packets will be sent in *Retransmitting* phase toward the sink node. We describe each phase in more detail in the subsequent sections.

Each node requires the following data structure to store to participate in finding a path to deliver the data.

1) T_k, the maximum number of iterations as a stopping condition for the algorithm.
2) $RELI\_SET$, the set of chosen nodes by our solution.
3) $P_{\text{threshold}}$, a threshold that is required for the termination of the learning process of our proposed approach that can be computed as the product of the probability of chosen nodes in $RELI\_SET$.
4) MIN_SET_SIZE, a dynamic threshold that keeps the cardinality of the lowest $RELI\_SET$ size. We initialize this value to network size.
5) k, a counter, which keeps the number of constructed $RELI\_SET$.

At the beginning of the algorithm, a `hello` message with T_k and $P_{\text{threshold}}$ values are flooded within the network to notify the nodes that the process is started. Algorithm 1 shows the pseudocode of RRDLA approach.

A. Initial

In this phase, the source node, i.e., the node that senses the information from an event or an object, plays the role of root node in DLA. The sink node will be the leaf node. To form the action-set of each sensor (say S_i), S_i propagates an `action` message to its one-hop neighbors within its transmission range. Upon receiving `action` message, each node replies to the sender and it forms its action-set. Therefore, each replied message by the neighbors is associated with an action in the action-set of S_i. Thus, the neighbors of each node specify the action-set size.

The above-mentioned process to form the action-set suffers from the fixed size action-set problem [37]. When the algorithm proceeds, this problem may result in selecting a node several times causes a loop. Thus, the situation decreases the convergency of RRDLA. To overcome this problem, we exploit variable action-set LA and introduce the following pruning rule to tackle it.

Convergency Rule: To prevent the nodes to be chosen by different nodes in the network, the LA of each node can prune its actions. To do so, the LA disables the actions that corresponds the earlier selected nodes. This rule improves the convergence speed of RRDLA and also decreases the running time.

Fig. 1(a) shows a sample DLA network of deployed nodes to detect an event. In Fig. 1(a), nodes A and J are the root and the leaf nodes, respectively. As described in Section IV-A, DLA tries to find a path from the root to the leaf. Each outgoing edge in this figure depicts an action for the LA of each node. Note that the reliability of each link does not appear in this figure for the simplicity. Table I shows the details of corresponding neighbors, action-set, and the initial value of each action for the DLA graph of the network in Fig. 1. As it was shown in Table I, each outgoing edge corresponds to an action in action vector of each node and their initial values are equal to one over the number of neighbors. The last column shows the probability of the actions after applying the learning process of the RRDLA algorithm. Note that unselected nodes were shown with the disabled label in this table.

Algorithm 1: RRDLA Algorithm 1.

```
1:  **Input** A network with $N$ node, $CR$, Source Node, Sink Node
2:  **Output** A set of reliable links $RELI\_SET$
3:  **Parameters:**
4:  $a$                ▷the reward parameter, where $0 < a < 1$
5:  $T_k$: The threshold value for iteration
6:  $T_D$: The threshold value for delay
7:  $T_{PDR}$: The threshold value for PDR
8:  **Begin**
9:  Construct a DLA from the WSN
10: Initialize $RELI\_SET$ with the nodes in the network
11: Initialize $T_k$ and $P_{\text{threshold}}$
12: $T_D \leftarrow \infty$
13: A hello message flooded in the network
14: Form action-set upon receiving a hello message
15: **repeat**
16: let $k \leftarrow 0$
17: **While** (sink node not visited)
18: The first LA is randomly chosen, denoted as $S_j$ and activated
19: **if** $S_j$ has no possible actions **then**
20:     Activated automata is traced back to find automaton with available actions
21:     The found LA is indicated as $S_i$
22: **end if**
23: $RELI\_SET \leftarrow RELI\_SET \bigcup S_i$
24: Automaton $S_i$ chooses one of its actions
25: Each automaton prunes its action-set to avoid the loop
26: Automaton $S_j$ is activated
27: Set $S_i$ to $S_j$
28: **EndWhile**
29: Compute $PDR$ and $D$ for the traversed path p.
30: //check the QoS constraints
31: **if** the modularity of $RELI\_SET <$ $MIN\_SET\_SIZE$ and $PDR > T_{PDR}$ and $D < T_D$ **then**
32:     The response from the environments $\beta_i(t) = 0$
33:     Reward the chosen actions of activated LAs using (9)
34:     $RELI\_SET \leftarrow MIN\_SET\_SIZE$
35:     $T_D \leftarrow D$
36:     Send the reward message to the selected nodes
37: **else**
38:     The response from the environments $\beta_i(t) = 1$
39:     Penalize the chosen actions of activated LAs using (10)
40:     Send the penalty message to the selected nodes
41: **end if**
42: $k \leftarrow k + 1$
43: Enable all disabled actions
44: **Until the stopping condition meet**
45: Send packets via nodes in $RELI\_SET$
46: **if** $PDR \leq 100$ **then**
47:     Resend undelivered packets
48: **end if**
49: **End Algorithm**
```

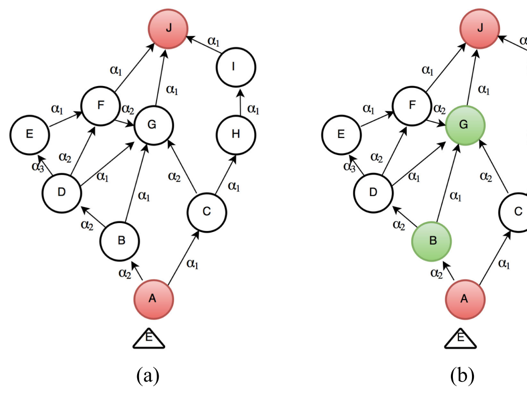

Fig. 1. An example of WSN. a) A network of DLA with node A as the root and node J as the leaf. b) Selected nodes by RRDLA in green color to transfer data.

TABLE I
THE ACTION-SET AND THE INITIAL VALUES FOR EACH ACTION OF THE NODES OF GRAPH IN FIG. 1

Node	Neighbors	Action-set	Initial Probability	Final Probability
A	{B,C}	$\{\alpha_1, \alpha_2\}$	{0.5,0.5}	{0.05,0.95}
B	{D,G}	$\{\alpha_1, \alpha_2\}$	{0.5,0.5}	{disabled}
C	{G,H}	$\{\alpha_1, \alpha_2\}$	{0.5,0.5}	{0.95,0.05}
D	{E,F,G}	$\{\alpha_1, \alpha_2, \alpha_3\}$	{0.33,0.33,0.33}	{disabled}
E	{F}	$\{\alpha_1\}$	{1.0}	{disabled}
F	{G,J}	$\{\alpha_1, \alpha_2\}$	{0.5,0.5}	{disabled}
G	{J}	$\{\alpha_1\}$	{1.0}	{1.0}
H	{I}	$\{\alpha_1\}$	{1.0}	{disabled}
I	{J}	$\{\alpha_1\}$	{1.0}	{disabled}

B. Learning

The learning phase starts by selecting a random action by an LA of source node. This is equal to choosing a node and activating one of the out edges automaton. The node sends a proper packet to the selected node to activate it. This process continues until the selected action by one of the nodes visits the sink node. When the sink node reaches, RRDLA puts all activated nodes in $RELI\_SET$ (*lines 15–28*).

Now, the RE should provide a reinforcement signal to the taken actions by the LA of a path from the root to the sink node. The RE checks the following conditions to reward/penalize the chosen action. If the number of activated nodes (or the number of hops from source to sink node) and the end-to-end delay are less from the values of previous iteration and the PDR is greater than PDR of previous round, it rewards the chosen actions in the path. Otherwise, it penalizes the selected actions. The chosen actions get a reward from the RE when $\beta(n) = 0$. Then, the sink node sends a `reward` message to the chosen actions by the LA of each node in $RELI\_SET$. In this case, the probability of selected actions updates according to (9). Otherwise, the RE penalizes the selected actions (i.e. $\beta(n) = 1$) according to (10) and the sink node sends a `penalty` message to the nodes. In RRDLA algorithm, we use L_{R-I} reinforcement schema for each LA to update the probability of actions. Therefore, the penalty message does not change the probability of actions in $RELI\_SET$. After that all actions will be activated for next round (*lines 29–44*). A new iteration will start if the stopping condition is not met.

The *learning* phase will finish when one of the conditions satisfies. First, the probability of the chosen nodes during the last process is computed. Then, the product of the probability of selected actions during the last iteration is measured. If the value is greater than $P_{\text{threshold}}$, the stop condition meets. Second, the number of iterations reaches the maximum iteration value (T_k).

Let us provide an example of this process in more detail. The LA of the first node (i.e., A) randomly selects an action and suppose that the chosen action is α_1. Node A sends a packet to C and LA of C is activated. Node C randomly selects one of its actions (e.g., α_2) and sends a packet to activate it. After selecting node C by LA of node A, LA of node A can prune its action-set by disabling node B. Now, LA of node G is activated and it has one possible action and selects it. Finally, the algorithm reaches the leaf node. Now, the nodes in $RELI\_SET$ are A, C, G, and J.

iter1: $A \longrightarrow C \longrightarrow G \longrightarrow J.$

The RE evaluates this path and gives a reinforcement signal to it. In the first step, it compares the path with the threshold value (MIN_SET_SIZE) and it rewards the path because the size of $RELI\_SET$ is less than the network size. Therefore, the probability of the chosen action updates accordingly by sending a `reward` packet. Then, it checks for the stopping condition of the algorithm by computing the product of the probability of the nodes with $P_{\text{threshold}}$ and replacing the value of MIN_SET_SIZE with the cardinality of the selected nodes. Afterward, a new message that includes the selected nodes during the last iteration will be sent to the nodes in the network. Upon receiving this message, each node finds that a new iteration has been completed. For instance, we provide the next two iterations of RRDLA algorithms as follows:

iter2: $A \longrightarrow C \longrightarrow H \longrightarrow I \longrightarrow J$
iter3: $A \longrightarrow B \longrightarrow D \longrightarrow F \longrightarrow J.$

After each iteration, the RE evaluates the chosen action and gives a reward/penalty for them. Here, in the second and third iterations, the LA of selected nodes gets a penalty from the RE because the number of selected nodes is higher than previous iterations. In this case, the probability of actions remains fixed.

C. Transmission

At the end of $Learning$ phase, the nodes in $RELI\_SET$ will be selected as active nodes for data forwarding. For example, as it was shown in Fig. 1(b), the following links between active nodes are selected to transmit the data packets:

$$A \longrightarrow C \longrightarrow G \longrightarrow J$$

The reason to do this is due to the fact that these actions have a higher value than others in the action probability vector of these nodes. The final probability column of Table I verifies the probability of selected actions. After transferring the packets by active nodes, RRDLA measures the value of PDR according to 2. As it uses the minimum possible number of nodes to deliver the data, therefore, the idle nodes can save their residual energy. It is worth noting that in the simulation results, we study the dynamics of links quality to measure the PDR by our algorithm

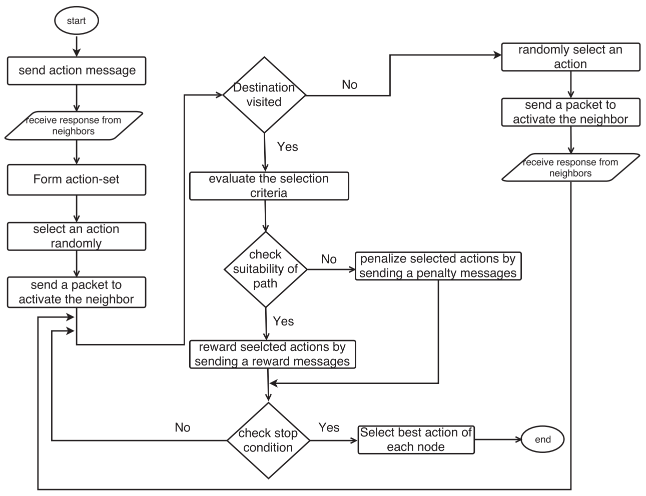
start
send action message
receive response from neighbors
Form action-set
select an action randomly
send a packet to activate the neighbor
Destination visited
No
Yes
randomly select an action
send a packet to activate the neighbor
receive response from neighbors
evaluate the selection criteria
check suitability of path
No
penalize selected actions by sending a penalty messages
Yes
reward seelcted actions by sending a reward messages
No
check stop condition
Yes
Select best action of each node
end

Fig. 2. Flowchart of RRDLA algorithm.

for each chosen path from a source to the destination. Fig. 2 shows the flowchart of RRDLA algorithm in selecting the nodes.

D. Retransmission

In the case of undelivered packets, it can be asked to the sender of packets to resend the lost packets. To do this, the PDR is computed and if the value of this parameter is less than 100%, the algorithm starts to resend undelivered packets. Note that in this phase the currently selected nodes are used to resend undelivered packets.

E. Convergency Proof

In order to prove the convergency of proposed algorithm, we use the *inductive* approach. We consider the reward equation of the selected action i.

$$\begin{aligned} p_i(t+1) &= p_i(t) + a(1-p_i(t)) \quad j = i \\ p_j(t+1) &= (1-a)p_j(t) \quad \forall j, j \neq i \end{aligned} \tag{11}$$

Proof: To prove that $p_i(t+1)$ converges to unity, we use *inductive* approach. We rewrite the equation in (11) as follows

$$\begin{aligned} p_i(t+1) &= a + a(1-a) + a(1-a)^2 + \cdots + a(1-a)^t \\ &\quad + a(1-a)^{t+1}p_i(0) \\ p_j(t+1) &= (1-a)^{t+1}p_j(0). \end{aligned} \tag{12}$$

From (12), we can obtain

$$\begin{aligned} p_i(t+1) &= a + \left[1 + (1-a) + (1-a)^2 + \cdots + (1-a)^t\right] \\ &\quad + a(1-a)^{t+1}p_i(0) \\ p_j(t+1) &= (1-a)^{t+1}p_j(0). \end{aligned} \tag{13}$$

With geometric progression, we can get

$$\begin{aligned} p_i(t+1) &= a\left(\frac{1-(1-a)^{t+1}}{1-(1-a)}\right) + (1-a)^{t+1}p_i(0) \\ &= 1 - (1-a)^{t+1}[1-p_i(0)] \\ p_j(t+1) &= (1-a)^{t+1}p_j(0). \end{aligned} \tag{14}$$

Now, we should prove that by increasing the value of t, $p_i(t+1)$ converges to unity and $p_j(t+1)$ tends to zero, where $0 < a < 1$. Thus, $0 < 1-a < 1$.

For $p_i(t+1)$, we have

$$\lim_{t\to\infty} p_i(t+1) = \lim_{t\to\infty} 1 - (1-a)^{t+1}[1-p_i(0)] = 1 \tag{15}$$

and for $p_j(t+1)$, we have

$$\lim_{t\to\infty} p_j(t+1) = \lim_{t\to\infty} p_j(t+1) = (1-a)^{t+1}p_j(0) = 0. \tag{16}$$

Penalty equations: The penalty equation is defined as follows:

$$\begin{aligned} p_i(t+1) &= (1-b)p_i(t)) \quad j = i \\ p_j(t+1) &= \frac{b}{r-1} + (1-b)p_j(t) \quad \forall j, j \neq i. \end{aligned} \tag{17}$$

For the penalty equations, we have to prove that $p_i(t+1)$ converges to 0 and also the sum of the probabilities of other actions $(p_j(t+1))$ converges to 1. ■

Proof: To prove that $p_i(t+1)$ converges to unity, we use the *inductive* approach. We rewrite the equation in (17) as follows:

$$\begin{aligned} p(2) &= (1-b)p(1) \\ p(3) &= (1-b)^2 p(1) \\ p(t+1) &= (1-b)^t p(1) \end{aligned} \tag{18}$$

since $0 < b < 1$ then $0 < 1-b < 1$, so $(1-b)^t$ is very big and converges to 0.

To prove the sum of the probabilities of other actions $p_j(t+1)$ converges to unity, we act as follows:

$$p_j(t+1) = \frac{b}{r-1} + (1-b)p_j(t) \tag{19}$$

and therefore we can have

$$p_j(1) = \frac{b}{r-1} + (1-b)p_j(0). \tag{20}$$

The sum of $p_i(1)$ and $p_j(1)$ can be computed as follows:

$$p_i(1) + p_j(1) = \frac{b}{r-1} + (1-b)p_j(0) + (1-b)p_i(0) = 1. \tag{21}$$

By one iteration, we can compute $p_j(2)$ as follows:

$$\begin{aligned} p_j(2) &= \frac{b}{r-1} + (1-b)\left[\frac{b}{r-1} + (1-b)p_j(0)\right] \\ &= \frac{b}{r-1} + \frac{b(b-1)}{r-1} + (1-b)^2 P_j(0). \end{aligned} \tag{22}$$

If we iterate (22) to find $p_j(3)$, we have

$$\begin{aligned} p_j(3) &= \frac{b}{r-1} + (1-b)\left[\frac{b}{r-1} + \frac{b(b-1)}{r-1} + (1-b)^2 P_j(0)\right] \\ &= \frac{b}{r-1} + \frac{b(1-b)}{r-1} + \frac{b(1-b)^2}{r-1} + (1-b)^3 P_j(0). \end{aligned} \tag{23}$$

Subsequently, we can find $p_j(t+1)$ by computing the previous values for $p_j(t)$, and we have

$$\begin{aligned} p_j(t+1) &= \frac{b}{r-1} + (1-b) \\ &\quad \times \left[\frac{b}{r-1} + \frac{b(b-1)}{r-1} + (1-b)^2 P_j(0)\right] \\ &= \frac{b}{r-1} + \frac{b(1-b)}{r-1} + \frac{b(1-b)^2}{r-1} + \cdots + \frac{b(1-b)^t}{r-1} \\ &\quad + (1-b)^{t+1} P_j(0) \\ &= \frac{b(1-b)}{r-1}[1 + (1-b) + (1-b)^2 \\ &\quad + \cdots + (1-b)^t] + (1-b)^{t+1} P_j(0) \\ &= \frac{b}{r-1}\left[\frac{1-(1-b)^{t+1}}{\frac{1-(1-b)}{b}}\right] + (1-b)^{t+1} P_j(0) \\ &= \frac{1-(1-b)^{t+1}}{r-1} + (1-b)^{t+1} P_j(0) \end{aligned} \tag{24}$$

and when *t* is a great value, we can have

$$\begin{aligned} \lim_{t\to\infty} p_j(t+1) &= \frac{b}{r-1}\left[\frac{1-(1-b)^{t+1}}{\frac{1-(1-b)}{b}}\right] + (1-b)^{t+1} P_j(0) \\ &= \frac{1}{r-1} \\ \lim_{t\to\infty} p_j(t+1) &= 0. \end{aligned} \tag{25}$$

Therefore, from (25), we obtain

$$p_j(t+1) = \frac{1}{r-1} \tag{26}$$

if the probability of one action tends 0, then the sum of the probability of the other actions needs to approach unity. ■

VI. Performance Evaluation

In this section, we investigate the performance of the proposed approach through simulations and experiments. We first provide the evaluation setup of our algorithm in Section VI-A. Then, we report the simulation results.

We study the performance of RRDLA algorithm in terms of the following:

1) end-to-end delay;
2) packet delivery ratio (PDR);
3) network lifetime;
4) the number of data transmissions.

We compare the performance of RRDLA with similar state-of-the-art QoS provisioning algorithms.

1) REER [34] leverages the reliability of links to find an energy efficient path to transfer the network data.
2) DETR [36] considers the reliability of links and energy efficiency for QoS routing. We use the same network parameters of [29] and [30] in our comparisons.

A. Evaluation Setup

We use ns-2 network simulator [45] for the simulations in which N nodes are randomly deployed in a two-dimension area with a size of 150×150 meter. All nodes have 40-m transmission ranges with 1 joule of initial energy.

The quality of each wireless link in our simulation is derived from the Nakagami fading model in which the PRR is related to the distance between two nodes [46].

We use the Nakagami distribution as

$$f(x, m, \Omega) = \frac{m^m x^{m-1}}{\Gamma(m)\Omega^m} \exp\left(-\frac{mx}{\Omega}\right) \tag{27}$$

to determine the power x of a received signal, where Γ is the Gamma function, *m* shows the Nakagami fading parameter, and Ω is the average received power. In our simulation, we set $m = 1$. By considering two-ray ground signal propagation, Ω can be stated as a function of d between a sender and a receiver as follows:

$$\Omega(d) = \frac{P_t G_t G_r h_t^2 h_r^2}{d^n L} \tag{28}$$

where P_t is the transmission power, G_t and G_r are the transmit and receive antenna gain, respectively, h_t and h_r are the antenna heights, and *n* is the path-loss exponent. We use $G_t = G_r = 1$, $h_t = h_r = 1.5$ m, and $n = 4$ in our simulation according to model in [16], [46].

We put the source and the sink node in the network diameter. Consequently, the coordinates of the source and the sink nodes are the network diagonal. We set the value of T_k in our algorithms to 100 iterations, which is obtained empirically and $P_{\text{threshold}}$ to 0.9. Note that choosing a small value less than 0.9 for the threshold may lead to a suboptimal solution because the stop condition for the algorithm meets after a few iterations. We also set the learning rate to 0.1.

B. Simulation Results

We report the obtained results from our simulations. We first study the effect of node density and then report effect of reliability requirements on the performance of all algorithms.

1) Node Density: Fig. 3(a) reports the end-to-end delay of all algorithms. From the figure, we notice that the RRDLA delivers the packets with a shorter end-to-end delay than those of REER and DETR. The reason behind this fact is the behavior of each LA in each running cycle because it tries to minimize the end-to-end delay to transfer the event information. It is worth noting that only successful end-to-end transmissions are considered in our results. We also notice that the end-to-end delay in RRDLA remains almost fixed by increasing the node density in the network.

Fig. 3(b) reports the PDR for all algorithms. We observe from the figure that increasing the number of nodes slightly improves PDR and they have a similar trend. The reason behind this fact can be summarized as follows. By increasing the node density of the network, all algorithms have better opportunities to investigate a highly reliable route to steer the network traffic toward the destination.

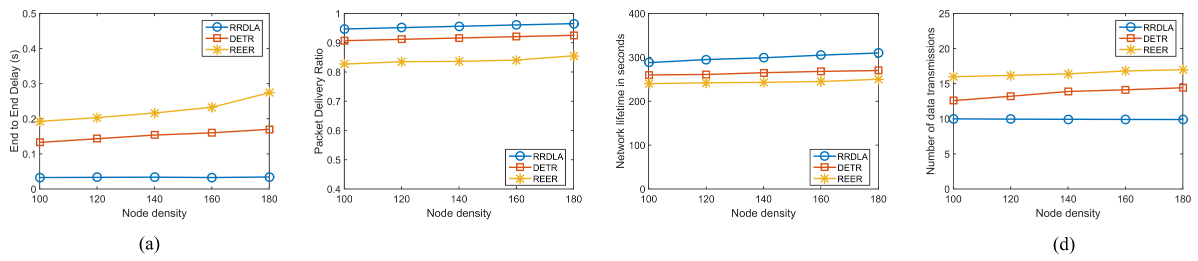

Fig. 3. Impact of the node density by keeping the end-to-end reliability fixed to 0.95. a) End-to-end delay. b) PDR. c) Network lifetime. d) Number of data transmissions.

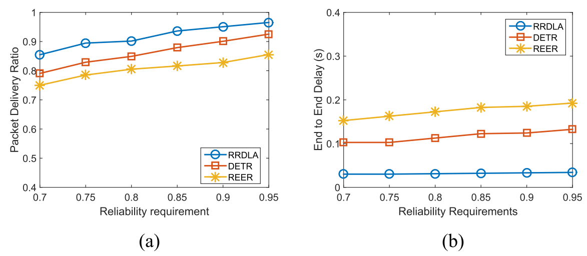

Fig. 4. Impact of QoS reliability with 100 nodes. a) PDR. b) End-to-end delay.

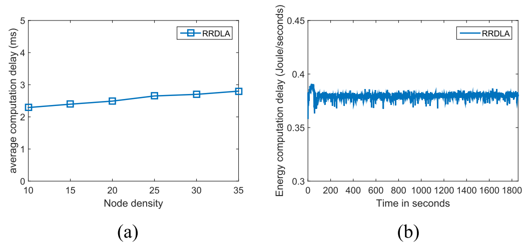

Fig. 5. Performance of RRDLA with A8 nodes of IoT-LAB [48]. a) Average computation delay. b) Energy consumption.

Fig. 3(c) depicts the obtained network lifetime of all algorithms. We use the energy model in [47] to compute the network lifetime. We also measure network lifetime until the first node drains its residual energy. Fig. 3(c) shows that RRDLA gains better lifetime than REE and DETR. The reason behind this fact is that it leverages a small number of nodes for packet delivery.

Fig. 3(d) depicts the total number of data transmissions under different node density for a successful end-to-end packet delivery. The results also reflect the number of hop counts from the source to the destination in the routing path. It is clear from the figure that RRDLA requires less number of data transmissions than REER and TEDR because it exploits links with a high PDR for packet delivery.

2) Reliability Requirements: We examine the performance of RRDLA, DETR, and REER by varying the reliability requirements. We set the node density to 100-node. Fig. 4(a) depicts that RRDLA gains better PDR than DETR and REER when the reliability requirement increases from 0.70 to 0.95. The results report that increasing the reliability requirements causes a better PDR for all algorithms.

Fig. 4(b) shows end-to-end delay of all algorithms against reliability requirements. With increasing the reliability requirements, the end-to-end delay of all algorithms increases and RRDLA has a lower end-to-end delay than DETR and REER.

C. Experimental Results

We measure the computation delay of node selection at each hop of RRDLA algorithm on A8 nodes, which are provided by IoT-LAB [48]. This kind of computation cannot be measured in simulation. We vary the number of nodes from 5 to 25. As it was shown in Fig. 5(a), the average delay in node selection process of RRDLA algorithm is slightly increased by increasing the number of nodes in the network. These results show that RRDLA algorithm requires a computation delay between 2 and 3 ms. We can conclude that the overhead, i.e., the node selection delay, of the algorithm is negligible for a long-time network operation.

Fig. 5(a) depicts the average energy consumption of A8 nodes by varying the time. To perform this experiment, we set the number of nodes to 10 nodes. As it was shown in Fig. 5(a), the average energy consumption is 0.37 joule/second. It is worth stating that the A8 node is the most powerful node in IoT-LAB, which has the ability to run high-level operating systems like Linux.

VII. Conclusion

In this paper, we studied the QoS routing of WSNss. We modeled the reliable routing problem as an MCOP problem. We proposed a distributed LA based algorithm to provision the QoS requirements for packet routing in WSNs. In terms of energy efficiency, our method tried to select best possible nodes to save other nodes' residual energies. It used a small number of sensors with high-reliable links to transfer information of a particular event in a network. RRDLA algorithm achieved a good balance among multiple QoS constraints such as end-to-end delay and energy consumption. Simulation results confirmed that RRDLA is a good solution for QoS provisioning in a WSN.

References

[1] M. Obaidat and S. Misra, *Principles of Wireless Sensor Networks*. Cambridge, U.K.: Cambridge Univ. Press, 2014.

[2] D.-G. Zhang, W.-B. Li, S. Liu, and X.-D. Zhang, "Novel fusion computing method for bio-medical image of WSN based on spherical coordinate," *J. Vibroengineering*, vol. 18, no. 1, pp. 522–538, 2016.

[3] D. Zhang, X. Wang, and X.-D. Song, "New medical image fusion approach with coding based on SCD in wireless sensor network," *J. Elect. Eng. Technol.*, vol. 10, no. 6, pp. 2384–2392, 2015.

[4] D. Zhang, X. Wang, X. Song, and D. Zhao, "A novel approach to mapped correlation of ID for RFID anti-collision," *IEEE Trans. Services Comput.*, vol. 7, no. 4, pp. 741–748, Oct.– Dec. 2014.

[5] D.-G. Zhang and X.-D. Zhang, "Design and implementation of embedded un-interruptible power supply system (EUPSS) for web-based mobile application," *Enterprise Inf. Syst.*, vol. 6, no. 4, pp. 473–489, 2012.

[6] D. Zhang, X. Kang, and J. Wang, "A novel image de-noising method based on spherical coordinates system," *EURASIP J. Advances Signal Process.*, vol. 2012, no. 1, p. 110, 2012.

[7] D.-G. Zhang, X.-D. Song, X. Wang, K. Li, W.-B. Li, and Z. Ma, "New agent-based proactive migration method and system for big data environment (BDE)," *Eng. Comput.*, vol. 32, no. 8, pp. 2443–2466, 2015.

[8] D.-G. Zhang and Y.-P. Liang, "A kind of novel method of service-aware computing for uncertain mobile applications," *Math. Comput. Model.*, vol. 57, no. 3, pp. 344–356, 2013.

[9] Z. Ma, D.-G. Zhang, J. Chen, and Y.-X. Hou, "Shadow detection of moving objects based on multisource information in internet of things," *J. Exp. Theor. Artif. Intel.*, vol. 29, no. 3, pp. 649–661, 2017.

[10] D.-G. Zhang, "A new approach and system for attentive mobile learning based on seamless migration," *Appl. Intel.*, vol. 36, no. 1, pp. 75–89, 2012.

[11] D.-G. Zhang, K. Zheng, D.-X. Zhao, X.-D. Song, and X. Wang, "Novel quick start (QS) method for optimization of TCP," *Wireless Netw.*, vol. 22, no. 1, pp. 211–222, 2016.

[12] D.-G. Zhang, S. Zhou, and Y.-M. Tang, "A low duty cycle efficient MAC protocol based on self-adaption and predictive strategy," *Mobile Netw. Appl.*, pp. 1–12, 2017.

[13] Z. Ma *et al.*, "A novel compressive sensing method based on SVD sparse random measurement matrix in wireless sensor network," *Eng. Comput.*, vol. 33, no. 8, pp. 2448–2462, 2016.

[14] D.-G. Zhang, Y.-N. Zhu, C.-P. Zhao, and W.-B. Dai, "A new constructing approach for a weighted topology of wireless sensor networks based on local-world theory for the Internet of Things (IOT)," *Comput. Math. Appl.*, vol. 64, no. 5, pp. 1044–1055, 2012.

[15] L. Cheng, J. Niu, J. Cao, S. K. Das, and Y. Gu, "QoS aware geographic opportunistic routing in wireless sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 7, pp. 1864–1875, Jul. 2014.

[16] J. Niu, L. Cheng, Y. Gu, L. Shu, and S. K. Das, "R3e: Reliable reactive routing enhancement for wireless sensor networks," *IEEE Trans. Ind. Inform.*, vol. 10, no. 1, pp. 784–794, Feb. 2014.

[17] A. Boukerche and A. Darehshoorzadeh, "Opportunistic routing in wireless networks: Models, algorithms, and classifications," *ACM Comput. Surv.*, vol. 47, no. 2, pp. 22:1–22:36, Nov. 2014. [Online]. Available: http://doi.acm.org/10.1145/2635675

[18] J. M. Jaffe, "Algorithms for finding paths with multiple constraints," *Networks*, vol. 14, no. 1, pp. 95–116, 1984.

[19] H. Mostafaei, A. Montieri, V. Persico, and A. Pescapé, "A sleep scheduling approach based on learning automata for WSN partial coverage," *J. Netw. Comput. Appl.*, vol. 80, pp. 67–78, 2017.

[20] J. A. Torkestani and M. R. Meybodi, "An intelligent backbone formation algorithm for wireless ad hoc networks based on distributed learning automata," *Comput. Netw.*, vol. 54, no. 5, pp. 826–843, 2010.

[21] M. A. Kafi, J. B. Othman, and N. Badache, "A survey on reliability protocols in wireless sensor networks," *ACM Comput. Surv.*, vol. 50, no. 2, pp. 31:1–31:47, May 2017.

[22] D.-G. Zhang, X.-D. Song, X. Wang, and Y.-Y. Ma, "Extended AODV routing method based on distributed minimum transmission (DMT) for WSN," *AEU-Int. J. Electron. Commun.*, vol. 69, no. 1, pp. 371–381, 2015.

[23] J. So and H. Byun, "Load-balanced opportunistic routing for duty-cycled wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 16, no. 7, pp. 1940–1955, Jul. 2017.

[24] D. Zhang, G. Li, K. Zheng, X. Ming, and Z.-H. Pan, "An energy-balanced routing method based on forward-aware factor for wireless sensor networks," *IEEE Trans. Ind. Inform.*, vol. 10, no. 1, pp. 766–773, Feb. 2014.

[25] D.-g. Zhang, K. Zheng, T. Zhang, and X. Wang, "A novel multicast routing method with minimum transmission for WSN of cloud computing service," *Soft Comput.*, vol. 19, no. 7, pp. 1817–1827, 2015.

[26] D.-G. Zhang, X. Wang, X.-D. Song, T. Zhang, and Y.-N. Zhu, "A new clustering routing method based on pece for WSN," *EURASIP J. Wireless Commun. Netw.*, vol. 2015, no. 1, pp. 1–13, 2015.

[27] D.-G. Zhang, S. Liu, T. Zhang, and Z. Liang, "Novel unequal clustering routing protocol considering energy balancing based on network partition & distance for mobile education," *J. Netw. Comput. Appl.*, vol. 88, pp. 1–9, 2017.

[28] D. Zhang, C.-P. Zhao, Y.-P. Liang, and Z.-J. Liu, "A new medium access control protocol based on perceived data reliability and spatial correlation in wireless sensor network," *Comput. Elect. Eng.*, vol. 38, no. 3, pp. 694–702, 2012.

[29] M. Zorzi and R. R. Rao, "Geographic random forwarding (GeRaF) for ad hoc and sensor networks: energy and latency performance," *IEEE Trans. Mobile Comput.*, vol. 2, no. 4, pp. 349–365, Oct.–Dec. 2003.

[30] B. Karp and H.-T. Kung, "GPSR: Greedy perimeter stateless routing for wireless networks," in *Proc. 6th Annu. Int. Conf. Mobile Comput. Netw.*, 2000, pp. 243–254.

[31] X. Huang and Y. Fang, "Multiconstrained QoS multipath routing in wireless sensor networks," *Wireless Netw.*, vol. 14, no. 4, pp. 465–478, 2008.

[32] Y. Charfi, N. Wakamiya, and M. Murata, "Trade-off between reliability and energy cost for content-rich data transmission in wireless sensor networks," in *Proc. 2006 3rd Int. Conf. Broadband Commun., Netw. Syst.*, Oct. 2006, pp. 1–8.

[33] L. A. Villas *et al.*, "Drina: A lightweight and reliable routing approach for in-network aggregation in wireless sensor networks," *IEEE Trans. Comput.*, vol. 62, no. 4, pp. 676–689, Apr. 2013.

[34] X. Y. Li, Y. Wang, H. Chen, X. Chu, Y. Wu, and Y. Qi, "Reliable and energy-efficient routing for static wireless ad hoc networks with unreliable links," *IEEE Trans. Parallel Distrib. Syst.*, vol. 20, no. 10, pp. 1408–1421, Oct. 2009.

[35] Q. Dong, S. Banerjee, M. Adler, and A. Misra, "Minimum energy reliable paths using unreliable wireless links," in *Proc. 6th ACM Int. Symp. Mobile Ad Hoc Netw.Comput.*, ser. MobiHoc '05., 2005, pp. 449–459.

[36] Z. Liu, L. Dai, L. Xue, X. Guan, and C. Hua, "Reliability considered routing protocol in wireless sensor networks," in *Proc. 30th Chinese Control Conf.*, Jul. 2011, pp. 5011–5016.

[37] J. A. Torkestani and M. R. Meybodi, "Clustering the wireless ad hoc networks: A distributed learning automata approach," *J. Parallel Distrib. Comput.*, vol. 70, no. 4, pp. 394–405, 2010.

[38] T. Korkmaz and M. Krunz, "Multi-constrained optimal path selection," in *Proc. IEEE INFOCOM 2001. 20th Annu. Joint Conf. IEEE Comput. Commun. Societies.*, 2001, vol. 2, pp. 834–843.

[39] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, "Network Flows: Theory, Algorithms, and Applications," Englewood Cliffs, NJ, USA: Prentice-Hall, 1993.

[40] K. S. Narendra and M. A. L. Thathachar, *Learning Automata: An Introduction*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1989.

[41] S. Misra, V. Tiwari, and M. S. Obaidat, "Lacas: learning automata-based congestion avoidance scheme for healthcare wireless sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 27, no. 4, pp. 466–479, May 2009.

[42] H. Beigy and M. R. Meybodi, "Utilizing distributed learning automata to solve stochastic shortest path problems," *Int. J. Uncertainty, Fuzziness Knowledge-Based Syst.*, vol. 14, no. 05, pp. 591–615, 2006.

[43] H. Mohamadi, A. S. Ismail, and S. Salleh, "Utilizing distributed learning automata to solve the connected target coverage problem in directional sensor networks," *Sensors Actuators A: Physical*, vol. 198, pp. 21–30, 2013.

[44] J. A. Torkestani and M. R. Meybodi, "An intelligent backbone formation algorithm for wireless ad hoc networks based on distributed learning automata," *Comput. Netw.*, vol. 54, no. 5, pp. 826–843, 2010.

[45] "The network simulator - nNS-2," Aug. 2017. [Online]. Available: https://www.isi.edu/nsnam/ns/

[46] K. Zeng, W. Lou, J. Yang, and D. R. Brown, "On throughput efficiency of geographic opportunistic routing in multihop wireless networks," *Mobile Netw. Appl.*, vol. 12, no. 5, pp. 347–357, Dec. 2007.

[47] M. Calle and J. Kabara, "Measuring energy consumption in wireless sensor networks using gsp," in *Proc. IEEE 17th Int. Symp. Personal, Indoor Mobile Radio Commun.*, 2006, pp. 1–5.

[48] "Iot-lab: a very large scale open testbed." [Online]. Available: https://www.iot-lab.info/, Jan. 2018.

Habib Mostafaei received the M.S. degree in software engineering from the Islamic Azad University, Arak branch, Arak, Iran, in 2009. He is currently working toward the Ph.D. degree with the Department of Engineering, Roma Tre University, Rome, Italy.

From February to August 2018, he was a Visiting Researcher with the University of Tuebingen/Germany. Prior to the Ph.D. training at Roma Tre University, from 2009 to 2015, he was a Lecturer with the Computer Engineering Department, Islamic Azad University. His research interests include software defined networking (SDN), interdomain routing, and wireless networks.

He has served as a Reviewer for a number of journal and conference papers, and he is the recipient of an outstanding reviewer award for Elsevier Journal of Networks and Computer Applications in October 2016 and for Journal of Computational Science in April 2017.