Wireless Pers Commun (2017) 95:3119–3139
DOI 10.1007/s11277-017-3987-8

A Model-Based Reinforcement Learning Algorithm for Routing in Energy Harvesting Mobile Ad-Hoc Networks

Meisam Maleki[1] · **Vesal Hakami**[2] · **Mehdi Dehghan**[1]

Published online: 3 February 2017

Abstract Dynamic topology, lack of a fixed infrastructure and limited energy in mobile ad-hoc networks (MANETs) give rise to a challenging operational environment. MANET routing protocols should consider dynamic network changes (e.g., link qualities and nodes residual energy) in such circumstances and be able to adapt to these changes to efficiently handle the traffic flows. In this paper, we assume an energy harvesting MANET in which the nodes have recharging capability and thus their residual energy level is randomly changing with time. We present a bi-objective intelligent routing protocol that aims at reducing an expected long-run cost function composed of end-to-end delay and the path energy cost. We formulate the routing problem as a Markov decision process which captures both the link state dynamics due to node mobility and energy state dynamics due to nodes rechargeable energy sources. We propose a multi-agent reinforcement learning-based algorithm to approximate the optimal routing policy in the absence of a priori knowledge of the system statistics. The proposed algorithm is built using the principles of model-based RL. More specifically, we model each node's cost function by deriving an expression for the expected value of end-to-end costs. Also the transition probabilities are estimated online using a tabular maximum likelihood method. Simulation results show that our model-based scheme outperforms its model-free counterpart and operates closely to standard value-iteration which assumes perfect statistics.

✉ Mehdi Dehghan
dehghan@aut.ac.ir

Meisam Maleki
meisam.maleki@aut.ac.ir

Vesal Hakami
vhakami@iust.ac.ir

[1] Mobile Ad Hoc and Wireless Sensor Networks Lab, Department of Computer Engineering and Information Technology, Amirkabir University of Technology, 424 Hafez Avenue, PO Box 15875-4413, Tehran, Iran

[2] Department of Computer Engineering, Iran University of Science and Technology, Hengam St., Resalat Sq., 16846-13114, Tehran, Iran

Keywords Mobile ad-hoc networks · Routing · Model-based reinforcement learning · MDP · Energy harvesting

1 Introduction

Mobile-ad-hoc-networks (MANETs) are self-configuring networks of mobile nodes which communicate by wireless links. Since the network topology continuously varies due to node mobility, the main challenge in MANET management is to allow each node to correctly route the packets to the other nodes. Beside node mobility, the routing algorithms must face other challenges, such as energy limitations of the nodes. Recent technological advancements have enabled energy harvesting capabilities for wireless nodes as a means to mitigate energy scarcity through recharging of a renewable energy source [1, 2]. However, to fully exploit the benefits of this technology, one needs to make special design considerations. In fact, in an energy harvesting MANET, the amount of energy consumed and/or harvested by a node is randomly changing with time. Also, by nature, node mobility renders the link conditions stochastic and time-varying as well. Hence, a principled way to optimize MANET routing is to model routing as a stochastic optimization problem with long-run system-wide objectives (e.g., expected end-to-end delay, energy consumption, etc.).

Within this perspective, many works have modeled the MANET routing problem as a Markov decision problem (MDP) [3–9]. An MDP-based model directly reflects the stochastic evolution of the system state and also makes its possible to come up with analytical guarantees on different long-run objectives. However, in general, the formulated MDP cannot be solved in exact offline fashion given that a prior knowledge of the link and energy level dynamics is typically not available at design time. The alternative is to resort to reinforcement learning schemes to approximate the optimal routing policy online, i.e., during system run time. Empowering the MANET with reinforcement learning (RL) allows the nodes to autonomously perceive the network condition, learn its dynamics from local observations, and to adapt their routing decisions accordingly [10].

In what follows, we review the related work in MANET routing which draws on reinforcement learning paradigms for system optimization. In one taxonomy, we can distinguish between two major trends in RL-based MANET routing: the algorithms built on the popular Q-routing scheme [11] and those based on multi-agent reinforcement learning (MARL).

Many studies in RL-based MANET routing have built and improved upon the pioneer Q-routing algorithm [3, 4, 7, 8], which is among the very first RL-based routing algorithms for computer networks. Q-routing draws on the well-known Q-learning scheme of the RL literature. In Q-routing, each node makes its routing decisions based on its local information. Each destination constitutes a system state and the actions are a nodes choice among its neighbors to be the next forwarder of the message. The reward (or cost) is defined to reflect different routing objectives such as delay, throughput, channel utilization, etc. For example, in [8], relays are chosen in such a way that average end-to-end delay is minimized in the long-run. The authors in [3] have extended the basic Q-routing scheme to come up with energy-efficient routing. Routing decisions in [3] are shaped in a way that paths with the least energy consumption and the most remaining energy will prevail in the long run. The system state is composed of two components: one indicating the total energy consumption along the path, and the other reflecting the least energy level among the nodes on the path. A node's action is its choice among the available paths and its cost is determined by the energy consumed on the chosen path. In [7] and [9], a Q-routing-based

algorithm has been proposed which features a dynamic discount factor parameter. The main objective is to reduce the number of route discovery attempts due to link failures. To this end, each node selects the next forwarder based on a combination of metrics concerning link stability, bandwidth utilization, and residual energy. These metrics are all incorporated into the discount factor parameter. Much in the same way as basic Q-routing, in [11], each destination represents a system state and actions are the selections of next-hop relays. A node is rewarded only if its packet reaches successfully to the destination. A common drawback with algorithms based on Q-routing is their greedy basis for local optimization of routing objectives, which does not necessarily lead to globally optimal network performance. In particular, the nodes in a wireless network communicate through a shared medium and thus mutually affect each others' performance. Another shortcoming is that the existing Q-routing-based algorithms learn in a purely model-free fashion. In model-free RL, the nodes rely on repeated interactions with the system to gradually build their experience from the acquired sample costs and observed states. Hence, it may take a huge number of iterations for a model-free algorithm to converge to an optimal policy. In contrast, a model-based scheme relies on a model of expected costs along with online estimation of transition probabilities to decrease the number of samples required for algorithm convergence. As a result, a model-based algorithm can achieve convergence much faster.

In MARL-based routing algorithms, on the other hand, each node cooperatively exchanges its individual observations with its neighbors to achieve global optimization besides local learning [12]. A MARL-based MANET routing algorithm, namely SAMPLE, is proposed in [6]. The SAMPLE's objective is global throughput maximization which is achieved through exchanging positive and negative feedbacks across the network. The system state in [6] consists of three components: the first component indicates that a packet is in a buffer waiting to be forwarded, the second component indicates the successful transmission to a neighbor node, and the third component indicates the successful reception from the neighbor node. Each action is also comprised of three components: the transmission of a packet to a neighbor, the delivery of a packet to a higher layer entity in protocol stack, and the broadcast of a neighbor discovery packet. The transition model in [6] corresponds to link reception probabilities. In order to build the state transition model, the number of different system events is sampled within a small time window into the past. The rate of these events is used to estimate the probability of state transitions. The proposed MDP model in SAMPLE is also utilized in SNQL routing protocol of [5]. Both SAMPLE and SNQL, however, target at throughput maximization as system-wide objective and do not account for delay and energy.

1.1 Contributions

In all foregoing works in RL-based MANET routing, a single objective is the target of system optimization. Also, the existing schemes are oblivious to energy harvesting capabilities of wireless nodes. With these two as motivation, in this paper, we propose a new RL-based MANET routing algorithm which features the following novelties:

- Unlike previous schemes, our MDP-based formulation for MANET routing problem accounts for both end-to-end delay and energy consumption. As mentioned earlier, the nodes in MANETs are faced with energy scarcity and a delay-centric formulation makes it vulnerable to reduced network longevity. This is because constantly relying on small delay paths eventually uses up the energy sources of the nodes on these paths and can even lead to network partitioning. In our proposed formulation, we explicitly

capture the stochastic dynamics induced by node mobility on link qualities, and derive a cost model for single-hop delay based on instantaneous link quality. Also, the single-hop energy costs are calculated based on the residual energy of the neighboring nodes. Finally, the total single-hop cost model is formed as a linear combination of delay and energy components. A distinguishing feature of our work is that we assume nodes are capable of harvesting energy from the environment and thus their energy levels vary with time randomly. This gives rise to yet another source of uncertainty which should be accounted for while making routing decisions. Accordingly, the routing policy calculated in this paper is adaptive to the nodes' energy states.

- We propose a model-based reinforcement learning algorithm, henceforth referred to as MRL-routing, to approximate the optimal routing policy for the adaptive selection of next-hop relays. The convergence speed of our algorithm is much higher in comparison with its model-free counterpart and can also lead to more efficient routing policies. Our model-based scheme draws on the proposed model for the expected costs and involves an online estimation of transition probabilities using a tabular maximum likelihood scheme. With the cost and transition functions at hand, a node is relieved of direct interaction with the environment to acquire sample costs and next state values for each single update. Instead, nodes can effectively make use of the feedback from a single interaction to carry out multiple updates in each time step. This way, our model-based updating scheme is particularly suited for MANETs which exhibit frequent changes and high dynamics.
- The proposed MRL-routing is an instance of multi-agent reinforcement learning process. Hence, each node, besides local learning, cooperatively exchanges its observations with its neighbors so that the overall process leads to network-wide optimization.
- We conduct a series of simulation experiments to investigate the efficiency and the convergence properties of the proposed method. Performance evaluation is done in terms of energy-delay trade-offs in various scenarios. MRL-routing is compared with standard offline and model-free online solutions and is shown to outperform the latter while also working without the informational assumptions of the former.

The rest of the paper is organized as follows: in Sect. 2, we discuss the system model together with our basic assumptions. In Sect. 3, we present our MDP-based formulation for the MANET routing problem. In Sect. 4, we introduce our MRL-routing algorithm for MANETs. Simulation results are presented in Sect. 5. We conclude the paper in Sect. 6.

2 System Model

In this section, we describe the system model and our assumptions for the MANET routing problem.

- *General setting*: We assume that all nodes are homogeneous (e.g., in terms of transmission power and battery capacity). Network nodes that reside within the effective transmission range of a node will be recognized as its neighbors. According to [13], we assume a radio channel is characterized by three components: path loss exponent (ξ), multi path fading (X^2) and shadowing (σ_s). Hence, the effective transmission range of each node is calculated as follows:

$$R_e = f(\xi, \sigma_s, x) = \frac{-2.33\sigma_s - 10\log_{10} E\{x^2\} + \eta}{10\xi} \quad (1)$$

where R_e is the maximum value of the transmission range R for which the received signal strength is greater than or equal to the received signal threshold with a very high probability. In other words, the received signal can be used if it is at least equal to the threshold. η is a constant (see [13] for more details).

- *Energy Harvesting model*: Nodes are capable of recharging, and the amount of harvested energy follows a random pattern according to the environment the nodes are operating in. It is assumed that the harvested power is stored in an energy buffer (see Fig. 1). In Fig. 1, Γ_i^t represents the amount of energy consumed by node i at time t. E_i^t represents the nodes' energy level at time t and its amount of charge at time t is denoted by h_i^t. Indeed, we view the energy charging in each node as a packet arrival process, and each energy packet is assumed to be an integer multiple of a basic energy unit (EU). Also, we assume that the energy packet arrival process $\{H_i^t\}_{t\geq 0}$ is an i.i.d stochastic process with general distribution $Pr\{H_i\}$ and mean $\bar{H}_i = \mathbb{E}[H_i]$. Also $\{H_i\}_i$ process is independent with respect to node index i.
- *Mobility model*: We consider a reference point group mobility (RPGM) model for nodes' movements [14]. The RPGM model captures the random movement of a group of nodes together with the motion of each individual node within the group. For each group, a logical center is assumed and group movements can be described in terms of the path traveled by the logical center for the group. The group displacement is determined via a group motion vector, denoted by $\overrightarrow{GM}$ in Fig. 2 [15]. Hence, the movement of the group center according to $\overrightarrow{GM}$ completely describes the direction and the speed of the group motion. As for individual node movements, each node is assumed to be associated with a reference point (RP in Fig. 2), whose movements depend on the group movement. Each node's RP at time $t+1$ is updated according to $\overrightarrow{GM}$; however, a node can also randomly roam around its RP. Such individual motions are characterized via a random motion vector $\overrightarrow{RM}$ (see Fig. 2). Overall, the updated position for each node is determined by summing the group motion vector $\overrightarrow{GM}$ and the individual nodes' random motion vector $\overrightarrow{RM}$. The length of $\overrightarrow{RM}$ is assumed to be uniformly distributed within a specified radius centered at $RP(t+1)$ and its direction is uniformly distributed between 0° and 360°. The motion pattern described by RPGM is

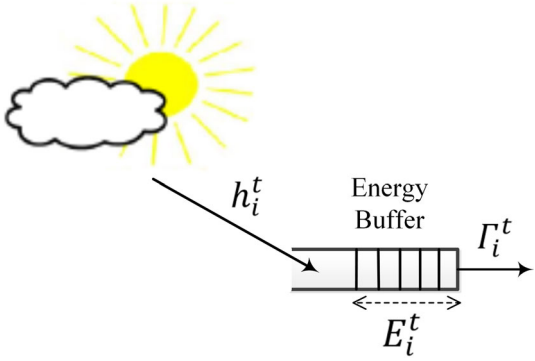


Fig. 1 Energy harvesting model in node i

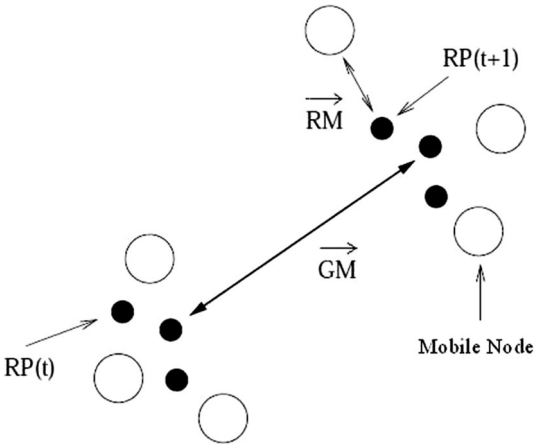


Fig. 2 RPGM model

common in some important applications of MANTEs in real world. For example, in many military applications, mobile agents (e.g., soldiers, tanks, drones, etc.) follow a group mobility model. The mobility of a rescue team in disaster relief scenarios or that of cyclists in a BikeNet [16] are other instances of RPGM.

- *Information assumptions*: We assume that the nodes are equipped with GPS modules and are aware of their current position. Nodes also locally and periodically exchange hello messages with each other, which contain their current position and energy level information.

3 Problem Formulation

As argued earlier in the Introduction, a successful routing policy in an energy harvesting MANET should account for the stochastic dynamics associated with the node mobility and renewable energy sources. In this section, we first describe our MDP formulation for the routing problem in Sect. 3.1, and then we define the long-term system-wide objective we seek to optimize in Sect. 3.2.

3.1 Routing as a Markov Decision Problem

The infinite horizon MDP in node i is defined by the tuple $< S_i, A_i, C_i, \mathcal{T}_i >$ in discrete time $t = 0, 1, 2, \ldots$. The symbol S_i represents the set of possible states in node i, and A_i is a set of actions that node i can perform. $C_i : S_i \times A_i \rightarrow \mathbb{R}$ represents the immediate cost function of node i, which is a mapping from the Cartesian product of its action and state spaces to real numbers. $\mathcal{T}_i : S_i \times A_i \times S_i \rightarrow [0, 1]$ denotes the state transition probabilities from $s_i \in S_i$ to $\acute{s}_i \in S_i$ for performing action $a_i \in A_i$. In the following, we describe each component in further detail:

- *state:* Let $N(i)$ be the set of neighbors nodes of node i. in other terms, $N(i)$ denotes the set of nodes located within node i's effective transmission range R_e^i. State $s_i \in S_i$ in node i is composed of two parts: (a) the distance d_{ij} to $\forall j \in N(i)$ and (b) energy level e_{ij} of $\forall j \in N(i)$.

(a) *Distance to neighbor nodes* $(d_{ij}, j \in N(i))$: The Euclidean distance between two nodes i and $j \in N(i)$ is calculated at any time as follows:

$$dis_{ij}^{t} = \sqrt{\left(x_i^t - x_j^t\right)^2 + \left(y_i^t - y_j^t\right)^2} \tag{2}$$

To discretize the distance, the effective transmission range of node i, R_e^i is divided into k intervals of length ε meters (Fig. 3); i.e.,

$$k = \lceil \frac{R_e^i}{\varepsilon} \rceil \tag{3}$$

Based on this division, each distance state d_{ij}^t between node i and its neighbor j is an integer within $\mathcal{D}_{ij} = \{1, \ldots, k\}$ that can be calculated based on ε and the real distance dis_{ij}^t as follows:

$$d_{ij}^t = \lceil \frac{dis_{ij}^t}{\varepsilon} \rceil \Leftrightarrow dis_{ij}^t \in \left[\left(d_{ij}^t - 1\right)\varepsilon, d_{ij}^t \varepsilon\right] \tag{4}$$

More specifically, d_{ij}^t represents the interval that the distance between nodes i and $j \in N(i)$ resides in this interval at time t. Note that for each node i, ε is a unit of distance and the number of states k depends on R_e^i. Finally, node i's set of distance states to all $j \in N(i)$ is denoted by $\mathcal{D}_i = \{\mathcal{D}_{ij}\}_{j \in N(i)}$.

(b) *neighbor nodes residual energy* $(e_{ij}, j \in N(i))$: the energy of node $j \in N(i)$ is calculated at any time $t + 1$ as follows:

$$e_{ij}^{t+1} = min\left\{\left[e_{ij}^t - \Gamma_j^t\right]^+ + h_j^t, E_{max}\right\} \tag{5}$$

In (5), Γ_j^t is the amount of consumed energy by node j at time t, and h_j^t is the random harvested energy by node j at time t (c.f., Sect. 2). Symbol $[\cdot]^+$ in (5) represents a shorthand notation for $max(., 0)$, Also E_{max} is the maximum energy capacity of each node j. We quantize the energy level of each node into three states: *low*, *medium* and *high*; more formally,

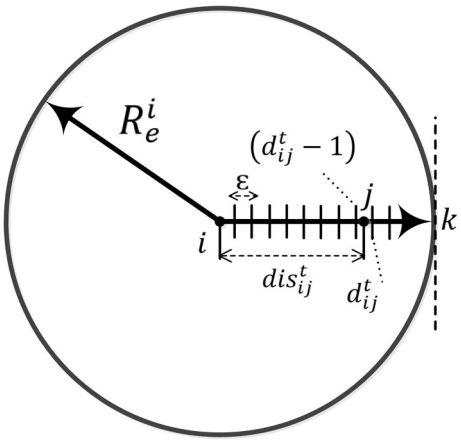


Fig. 3 Effective transmission range for node i

$$e_{ij}^t \in \mathcal{E}_{ij} = \{'low' \equiv \mathcal{L}, 'medium' \equiv \mathcal{M}, 'high' \equiv \mathcal{H}\} \quad (6)$$

The set of energy states of node i's neighbors is indicated by $\mathcal{E}_i = \{\mathcal{E}_{ij}\}_{j \in N(i)}$. According to these definitions, the complete state in each mobile node i is as follows:

$$s_i^t = \left(d_{ij}^t, e_{ij}^t\right)_{j \in N(i)} \in S_i = \mathcal{D}_i \times \mathcal{E}_i \quad (7)$$

- *Action:* the action that each node i performs at time t is denoted by (a_i^t), which represents node i's choice of next-hop relay from one of its neighbors (e.g., node j) as next hop node. In other words:

$$a_i^t \in A_i = N(i) \quad (8)$$

- *Immediate cost function model:* In RL, each agent interacts with its environment via feedbacks. More specifically, by taking an action, the agent receives a numerical value of its immediate cost from the environment. In model-free RL, an agent is not informed of its cost unless it actually performs an action in the environment. In model-based RL, on the other hand, the agent has access to a model of its cost function. This model relieves the agent from obtaining sample costs directly from the environment; instead, it can utilize the cost model to compute the expected cost associated with all potential actions. Using this model, fewer samples of the environment cost function are needed for the convergence of the learning process, and thus the sample complexity of the learning algorithm will decrease. Our proposed MRL-routing algorithm in Sect. 4.2 is an instance of model-based RL. Accordingly, here, we describe the derivation of a cost model for the nodes. Given that our goal is to reduce both the average end-to-end delay and energy consumption, the proposed cost model is also composed of two parts: end-to-end transmission delay and path energy cost. As already mentioned, the proposed routing algorithm is based on MARL. This means that in order to achieve global optimization, the cost function of the nodes should be defined in an end-to-end fashion. In what follows, we consider a given node i, and first derive a formula for computing the single-hop transmission delay of node i. Next, we show how a node can calculate its complete immediate path-wise cost by receiving succinct feedbacks from its successors. In fact, given that both energy and delay are aggregate measures (delay is an additive metric and the energy cost can be considered as a concave metric), we may compute the path-wise cost of node i by backing up low-cost aggregate quantities (sums for delays and mins for energy) in the reverse path from the destination node back to node i.

  (a) *Single-hop transmission delay cost function model:* The transmission delay of a packet on a link is determined according to link quality. Link quality (e.g. in terms of SNR) varies in a MANET due to nodes' mobility which causes random changes in distance between nodes. Naturally, the smaller the distance between two nodes, the higher would be the perceived SNR, and vice versa. Here, we propose a function to calculate the single-hop delay based on distance state d_{ij}^t of node i from a neighboring node j. To this end, we first compute the link SNR in terms of distance, and then determine the link error probability. Finally we derive the transmission delay formula given the packet length and effective bit

rate. According to [13], the received power (in dB) at the reference distance of 1 meter $\tilde{P}$ is calculated as follows:

$$\tilde{P} = P_{TX} - \bar{L}_0 - 10\log_{10} E\{x^2\} \tag{9}$$

In this equation, P_{TX} is the transmission power in transmitter, $\bar{L}_0$ is average path loss at reference distance of 1 meter and $E\{X^2\}$ captures the average of multi path fading (in dB). Given the received power at reference distance of 1 meter, each node i can calculate the received power at node j at time t as follows:

$$P(d_{ij}^t) = \tilde{P} - 10\xi \log_{10}\left(d_{ij}^t.\varepsilon\right) - \sigma_s, \tag{10}$$

where ξ and σ_s denote the path loss exponent and shadowing, respectively. With the received power at hand, the signal to noise ratio (SNR) can be calculated for each link as follows:

$$SNR(d_{ij}^t) = \frac{P(d_{ij}^t)}{N_0} \tag{11}$$

where N_0 represents the channel noise. According to [18], we can calculate the error rate Pl_{ij} at time t for link ij in terms of SNR as follow:

$$Pl_{ij}(d_{ij}^t) = \frac{1}{1 + e^{\zeta\left(SNR\left(d_{ij}^t\right) - \delta\right)}} \tag{12}$$

where ζ and δ are constants and are related to coding and modulation characteristics of a packet with a specific length. Now, given the link error rate, the transmission delay of the link ij can be calculated as a function of the distance state d_{ij}^t as follows:

$$delay_{ij}^t(d_{ij}^t) = \frac{L}{B_{ij} \times \left(1 - Pl_{ij}\left(d_{ij}^t\right)\right)} \tag{13}$$

where L is the packet length. and B_{ij} is the maximum transmission rate of link ij (in bit/s) at time t.

(b) *Immediate end-to-end cost function model:* We may now proceed to show how each node can calculate its immediate end-to-end cost in terms of the feedback it receives from its next-hop. In other words, when node i selects a neighbor node j as relay (i.e., $a_i^t = j$), it receives $F_j^t = (F_{j,delay}^t, F_{j,energy}^t)$ as feedback from node j. This feedback consists of two components: I) the delay feedback $F_{j,delay}^t$ which expresses the cumulative delay of transmitting a packet from node j to destination, and II) the energy feedback $F_{j,energy}^t$ which expresses the minimum energy level among the nodes on the partial route $j \rightsquigarrow D$ (excluding j's energy level itself). By receiving this feedback, node i computes its end-to-end immediate cost function $C_i(s_i^t, a_i^t)$ as:

$$C_i(s_i^t, a_i^t) = \alpha.\left(F_{j,delay}^t + delay_{ij}^t(d_{ij}^t)\right) + \beta.\lambda.\mathcal{P}\left(min\left(F_{j,energy}^t, e_{ij}^t\right)\right), \tag{14}$$

The cost $C_i(s_i^t, a_i^t)$ is, in fact, a weighted sum, where α and β are weight factors that should be determined according to the importance of energy and delay factors in the intended application. Also, λ is a normalization constant. Finally,

the function $\mathcal{P}(\cdot)$ in 14 is defined as a logistic function of the given energy level [17]; e.g., $\mathcal{P}(x) = \frac{\phi}{1-e^{\frac{\rho}{x}}}$, in which ϕ is a scaling parameter; higher values for ϕ amplifies the impact of the residual energy on cost. ρ is a preferred coefficient; higher values for ρ results in a lower growth rate for $\mathcal{P}(\cdot)$, and thus gives rise to a smoother cost function with less impulsive behavior. This way, the energy level of nodes will have an inverse relation with cost value. In other terms, the cost incurred by node i will increase as the energy status of some node(s) along the path $i \rightsquigarrow D$ becomes more critical. In fact, defining the energy cost function in this way, results in a more balanced distribution of network traffic across all nodes, as well as in increasing the minimum lifetime of nodes in the network. Node i then forms and sends its own feedback to the previous hop as follows:

$$F^t_{i,delay} = F^t_{j,delay} + delay^t_{ij}(d^t_{ij}), \tag{15}$$

$$F^t_{i,energy} = min\left(F^t_{j,energy}, e^t_{ij}\right). \tag{16}$$

- *State transition probabilities:* The state transition probabilities in node i's MDP is denoted by $\mathcal{T}_i$ which represents the transition from state (d^t_{ij}, e^t_{ij}) at time t to state $(d^{t+1}_{ij}, e^{t+1}_{ij})$ at time $t+1$. In general, calculating the exact value of state transition probabilities is often impossible in real networks due to the lack of prior knowledge about the system stochastic parameters. In our proposed model-based reinforcement learning method in Sect. 4.2, each node approximates its transition probabilities in an online manner using a tabular maximum likelihood method.

3.2 Optimization Objective

In our proposed formulation, we aim to compute a routing policy $\pi_i: S_i \rightarrow A_i$ using which each node takes a routing decision a_i at each state s_i in such a way that it leads to minimizing its average discounted cost in the long-run. Such a routing policy at the source node chooses a route with the lowest average discounted end-to-end delay and total end-to-end energy cost. More formally, each node computes its optimal relay selection policy π^*_i as follows:

$$\pi^*_i \in argmin_{\pi_i} \bar{C}^{\pi_i}(s_i) \equiv E^{\pi_i}\left[\sum_{t=0}^{\infty} \gamma^t C_i\left(s^t_i, a^t_i\right) | s^0_i = s_i\right], \forall s_i \in S_i \tag{17}$$

In (17), $0 < \gamma < 1$ is a discount factor that decays the impact of the cost over successive decision periods. In other words by applying γ, the importance of short-term costs increases with respect to the long-term costs. The smaller the discount factor, the sooner a node prefers to minimize its average routing cost.

4 Computing the Optimal Routing Policy

In this section, we compute the optimal routing policy π^*_i. To this end, we first discuss the pros and cons of standard value iteration and conventional model-free reinforcement

learning in Sect. 4.1. Standard value iteration requires a priori knowledge of the network statistics (transition probabilities $\mathcal{T}_i$ and average cost function $C_i(\cdot)$) for calculating π_i^*. Having this knowledge, this method can converge to the optimal value of the average discounted cost over a few iterations. However, in practice, such accurate knowledge is not always available. In the absence of such knowledge, model-free reinforcement learning methods such as *Q-learning* can converge to the optimal value of the average discounted cost through frequent interactions with the environment and sampling from the transition and immediate cost functions. However, this convergence can be very slow due to the need for a large number of samples. In Sect. 4.2, we introduce our MRL-routing algorithm which, similarly to *Q-learning*, is also independent of prior statistical knowledge. However, MRL-routing is a model-based RL procedure; i.e., we utilize the proposed cost function in (16) as a model of the expected cost, and also approximate the transition probabilities $\mathcal{T}_i$ using a tabular maximum likelihood method. Hence, unlike *Q-learning*, MRL-routing will have low sample complexity and short convergence time.

4.1 Conventional Methods

4.1.1 Standard Value Iteration

For calculating the optimal routing policy π_i^* in node i, standard *value iteration* can be used as follows:

$$V_i^{t+1}(s_i^t) = \min_{a_i^t \in A_i} \left\{ C_i(s_i^t, a_i^t) + \gamma \sum_{s_i^{t+1}} \mathcal{T}_i(s_i^t, a_i^t, s_i^{t+1}) V_i^t(s_i^{t+1}) \right\} \tag{18}$$

where $V_i^t(s)$ is the estimated long-term cost of state $s \in S$ at tth iteration. The sequence of estimates $\{V_i^t(s)\}_t$ for $\forall s \in S_i$ is provably convergent to $V_i^*(s)$, i.e., the lowest average long-term discounted cost of state s (e.g., see [19]). Having V_i^* , π_i^* can be calculated as follows:

$$\pi_i^*(s) = argmin_{a_i \in A_i} V_i^*(s), \forall s \in S_i \tag{19}$$

4.1.2 Q-Learning

The *Q-learning* algorithm [19] exploits realized samples of transitions and costs obtained from actual action implementation along with stochastic averaging for learning the average discounted cost. The *Q-learning* algorithm gradually learns an optimal decision policy without knowing the transition probabilities $\mathcal{T}_i$ and average cost function $C_i(\cdot)$. Let $Q_i^*(s,a)$ denote the sum of the immediate cost resulting from performing action a in state s together with the value of the average discounted cost associated with following the optimal policy π_i^* in all future states. Accordingly, $Q_i^t(s_i^t, a_i^t), \forall s_i^t \in S_i$ denotes the time t estimate of $Q_i^*(s,a)$. We define an asynchronous counter $v^t(s_i, a_i)$ given by (20). Also let $\alpha(t)$ be a sequence of step-sizes satisfying the conditions in (21).

$$v^t(s_i, a_i) := \sum_{\tau=1}^{t} \mathbb{I}_{\{(s_i^\tau, a_i^\tau) = (s_i, a_i)\}}. \tag{20}$$

$$\alpha(t) \to 0 \; as \; t \to \infty, \sum_t \alpha(t) = \infty, \sum_t \alpha(t)^2 < \infty. \tag{21}$$

The Q-learning update rule in (22) guarantees the convergence of the sequence $\{Q_i^t(s,a)\}_t$ to $Q_i^*(s,a)$ for $\forall s \in S_i$ and $\forall a \in A_i$ [20]:

$$Q_i^{t+1}(s_i,a_i) - Q_i^t(s_i,a_i) = \alpha(v^t(s_i,a_i)).\mathbb{I}_{\{(s_i,a_i)=(s_i^t,a_i^t)\}}.\left[cost_i^t + \gamma.\min_{a_i} Q_i^t(s_i^{t+1},a_i)\right] \tag{22}$$

where $cost_i^t$ denotes the realized cost of performing action a_i^t in state s_i^t, and s_i^{t+1} is the realized next state in node i after performing action a_i^t.

The action selection is performed using a Greedy in the Limit with Infinite Exploration (GLIE) policy in each iteration of *Q-learning*. This means that actions are selected greedily based on Q values after convergence; however, to ensure the theoretical convergence of the algorithm, any action should have a non-zero chance of selection so that all state-action pairs are visited infinitely often. To ensure this, we carry out our action selection based on Boltzmann distribution in each iteration (soft- min policy) [19]:

$$\pi_i^{t+1}(s_i^{t+1},a_i) = \frac{exp\left(\frac{-Q_i^{t+1}\left(s_i^{t+1},a_i\right)}{\tau}\right)}{\sum_{\forall a_i \in A_i} exp\left(\frac{-Q_i^{t+1}\left(s_i^{t+1},a_i\right)}{\tau}\right)} \tag{23}$$

In (23), τ is a so-called temperature parameter. High values for τ result in all actions having almost equal probabilities. While low values for τ give rise to greater differences in the selection probability of actions as their estimated Q-values become more distant. In the limit as $\tau \to 0$, soft-min action selection reduces to the greedy action selection scheme. Once the Q-values converge, π_i^t also converges to π_i^*.

The most important advantage of *Q-learning* is its low informational assumptions and low computational complexity. In each time interval, the *Q-learning* needs to update the Q-values of $\forall s_i^t \in S_i$ and for each state, $Q_i^t(s_i^{t+1},a_i)$ is calculated over $\forall a_i \in A_i$. Hence, the computational complexity is $O(|A_i||S_i|)$.

Despite these advantages, the number of iterations required for the convergence of *Q-learning* can be very high. In other words, it slowly converges to the optimal policy due to its high sample complexity [21]. Therefore, in MANETs where node conditions are changing in a faster rate, *Q-learning* can be inefficient. In the following, we represent our MRL-routing algorithm which is based on model-based reinforcement learning. With a slight increase in computational complexity, MRL routing is convergent with fewer iterations and to better average costs of energy and delay.

4.2 Proposed Algorithm: MRL-Routing

In this section, we present MRL-routing, our model-based reinforcement learning algorithm, for computing the optimal routing policy π_i^*. MRL-routing exploits the cost function formula $C_i(\cdot)$ in (16) as a model of the expected costs. It also approximates the transition probabilities $\mathcal{T}_i$ in an online fashion to more efficiently use the experience obtained from interactions with the environment.

Let $n^t_{s_i^t s_i'}(a_i^t)$ represent the observed number of times before time step t that the action a_i^t is taken when the state was in s_i^t and made a transition to s_i', and let $n^t_{s_i^t}(a_i^t)$ represent the

observed number of times before time step t that the action a_i^t is taken when the state was in s_i^t, i.e.,

$$n_{s_i^t}^t(a_i^t) = \sum_{s_i' \in S_i} n_{s_i^t s_i'}^t(a_i^t) \tag{24}$$

The state transition probabilities can be approximated using the following empirical distribution:

$$\hat{\mathcal{T}}_i^t(s_i^t, a_i^t, s_i') = \frac{n_{s_i^t s_i'}^t(a_i^t)}{n_{s_i^t}^t(a_i^t)} \tag{25}$$

It has been shown in [22] that $\hat{\mathcal{T}}_i^t$ is in fact a maximum likelihood estimator of true transition probabilities $\mathcal{T}_i$ of a given Markov chain; also, a central limit theorem is given in [22] which proves that $\hat{\mathcal{T}}_i^t$ converges (in distribution) to true transitions $\mathcal{T}_i$. By estimation of $\mathcal{T}_i$ in (25), each node i updates its Q-value in MRL-routing as follows:

$$Q_i^{t+1}(s_i^t, a_i^t) = (1 - \alpha_t) Q_i^t(s_i^t, a_i^t) + \alpha_t \{C_i(s_i^t, a_i^t) + \gamma \min_{a_i^t \in A_i} \sum_{s_i'} \hat{\mathcal{T}}_i^t\left(s_i^t, a_i^t, s_i'\right) Q_i^t(s_i', a_i^t)\} \tag{26}$$

where $0 < \alpha_t < 1$ is a learning rate that satisfies the conditions in (21). Also, similar to *Q-learning*, action selection policies is adjusted in GLIE fashion (see (23)).

Q-learning (22) can only update the Q-value of the current state-action pair (s_i^t, a_i^t). This is because the resulting cost cannot be obtained unless the associated action is actually taken in current state. In our MRL-routing, however, using the cost function model and transition probabilities, it is possible to update Q-values for all states $\forall s_i \in S_i$ at each time step. As a result, our model-based method can converge much faster. This advantage can be useful and very important for highly dynamic MANET scenarios with rapidly changing conditions. In terms of computational complexity, in MRL-routing, the Q-values of $\forall s_i^t \in S_i$ need to be updated in each time step, and for each state, the term $\min_{a_i^t \in A_i} \sum_{s_i'} \hat{\mathcal{T}}_i^t(s_i^t, a_i^t, s_i') Q_i^t(s_i', a_i^t)$ in (26) is calculated over $\forall a_i^t \in A_i$. Hence, the computational complexity is $O(|A_i||S_i|^2)$. As we show in next section, despite this slight increase in computational complexity, MRL-routing has the advantage of much faster convergence as well as near-optimal performance. The complete procedure of our proposed MRL-routing algorithm is given in Table 1.

5 Simulation

In this section, we evaluate numerically the performance of our MRL routing algorithm. The performance metrics of interest are average end-to-end delay and energy cost. We compare the performance of our algorithm with the solution obtained from the standard value-iteration algorithm for MDPs [19] and model-free reinforcement learning algorithm of *Q-learning*. The value-iteration algorithm assumes prefect knowledge of the system dynamics (link state and node energy state dynamics). Using this knowledge speeds up the convergence to optimal policy and also results in near-optimal routing performance. In contrast, *Q-learning* converges with slower rate due to the lack of knowledge of the system dynamics, and its performance is far from optimal.

Table 1 MRL-routing algorithm in node i

Initialization:	$t \leftarrow 0,\ s_i^0,\ F_{a_i^i}^0,\ Q_i^0(s_i^t, a_i^t) = 0,\ \forall s_i^t \in S_i, \forall a_i^t \in A_i$
1:	//Select an action a_i^t based on policy π_i^t:
	Randomly select the action a_i^t according to the $[\pi_i^{t+1}(s_i^{t+1}, a_i), \forall a_i^t \in A_i]$
2:	Transmit the packet and observe the new state $s_i^{'}$
3:	//Update state transition probability:
	update the number of state transition $n_{s_i^t s_i^{'}}^{t+1}(a_i^t) = n_{s_i^t s_i^{'}}^{t}(a_i^t) + 1$, and
	update state transition probability $\mathcal{T}_i(s_i^t, a_i^t, s_i^{'})$ using (25)
4:	Receive $F_{a_i^t}^t$ from next hop neighbor
5:	//Evaluate immediate cost:
	Calculate $C_i(s_i^t, a_i^t)$ using (14)
6:	//Update the Q-value:
	For each state $s_i^t, \forall s_i^t \in S_i$ and action a_i^t, update Q-value $Q_i^{t+1}(s_i^t, a_i^t)$ using (26)
7:	//Update the policy:
	For each state $s_i^t, \forall s_i^t \in S_i$, update the policy $\pi_i^{t+1}(s_i^t)$
8:	//Update the feedback values:
	Update F_i^t Using (16)
9:	$t \leftarrow t + 1$, go back step 1

Table 2 Simulation parameters

Parameter	Value
Network size $(x \times y)$	1000 m × 1000 m
Number of network nodes	25, 50
Simulation time	4000 iterations
Effective transmission range (R_e)	239 m
Packet size	512 byte
Band with	512 kbps
Energy buffer size	50 energy packet
Energy unit (EU)	2.5 μJ

The network environment is assumed to be a square region of 1000 m × 1000 m. Node mobility is assumed to follow an RPGM model (c.f., Sect. 2). Both the movement of the logical center for each group, and the random motion of each individual mobile node within the group, are implemented via the Random Waypoint Mobility Model. One difference, however, is that individual mobile nodes do not use pause times while the group is moving. Pause times are only used when the group reference point reaches a destination and all group nodes pause for the same period of time.

The vector $\overrightarrow{GM}$ is determined by a constant speed 10 m/s and random direction at each time. The pause time of 5 s is considered for the group motion. Each individual mobile node roams around its reference point with a predefined average speed.

Traffic load corresponds to an interactive environment. For each source-destination pair, data packets are assumed to be generated at the source following a Poisson process with

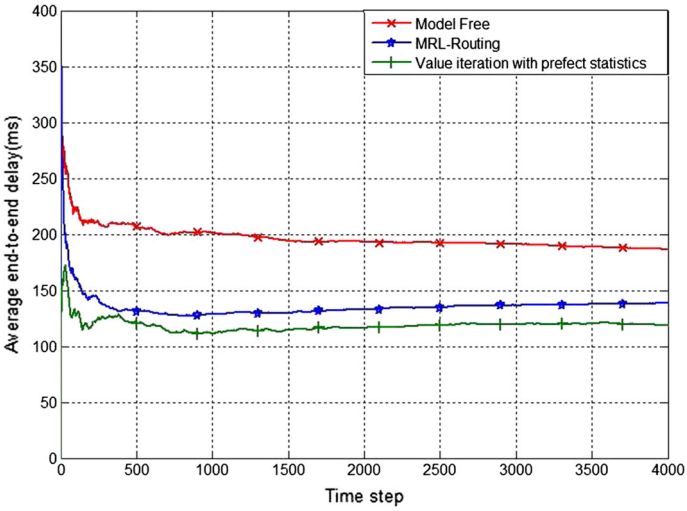


Fig. 4 Avearge end-to-end delay in a network with 50 nodes

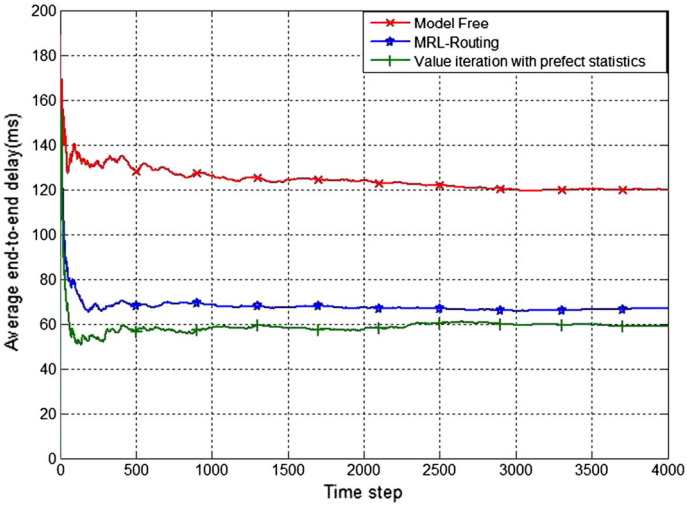


Fig. 5 Average end-to-end delay in a network with 25 nodes

average interval of 0.5 s. This amounts to a traffic volume of 8 kbps per source/destination pair, assuming that data packet length is 512 bytes. The energy buffer size for each node is taken to be 50 energy packets. Each energy packet is taken to be the size of two energy units. Also, as discussed in Sect. 3.1, we uniformly quantize the energy buffer space into three equal segments: denoted by $\mathcal{L}$, $\mathcal{M}$, and $\mathcal{H}$. Furthermore, each energy unit amounts to 2.5 μJ, and we consider Poisson energy arrival with average arrival rate of 1 energy packet per time step. Table 2. lists the parameters used in simulations.

First, we investigate the impact of the number of network nodes on average end-to-end delay and energy costs. As can be seen, these performance metrics are directly related to

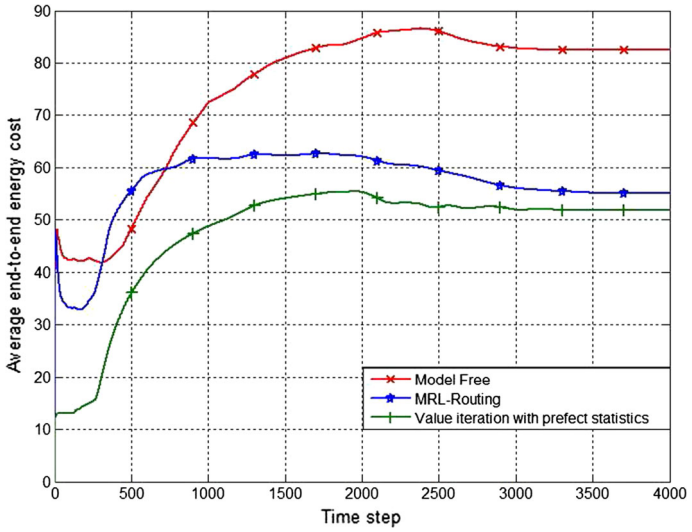


Fig. 6 Average end-to-end energy cost in a network with 50 nodes

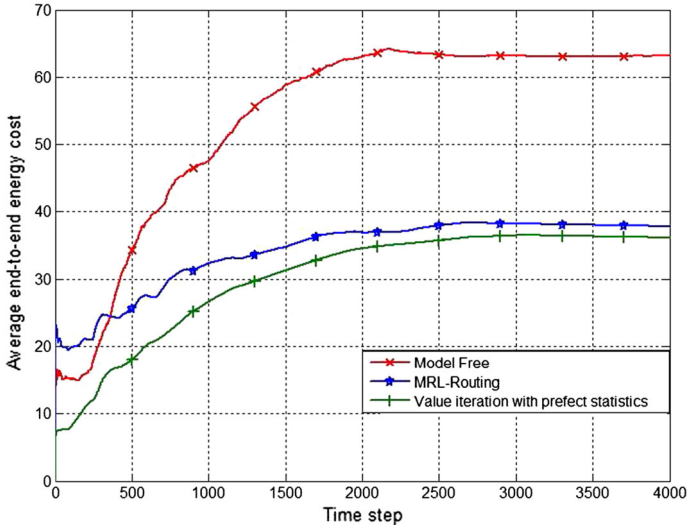


Fig. 7 Average end-to-end energy cost in a network with 25 nodes

the number of nodes. We conduct the experiment with 25 and 50 nodes. In each diagram, the performance of the proposed MRL-routing algorithm is compared with *Q-learning* and standard value-iteration. Figure 4 plots the average end-to-end delay for a source-destination pair in a network with 50 nodes, while Fig. 5 depicts the same for a network of 25 nodes.

In the network with 50 nodes, average end-to-end delay converges to 138.71 ms, while in the network with 25 nodes, this value is 66.97 ms. These values indicate a direct correlation between the number of nodes in the network and the average end-to-end delay.

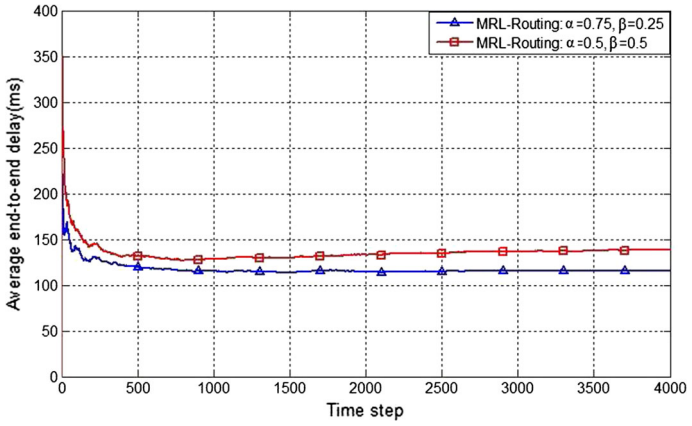


Fig. 8 Average end-to-end delay with weight factors $\alpha = 0.75, \beta = 0.25$

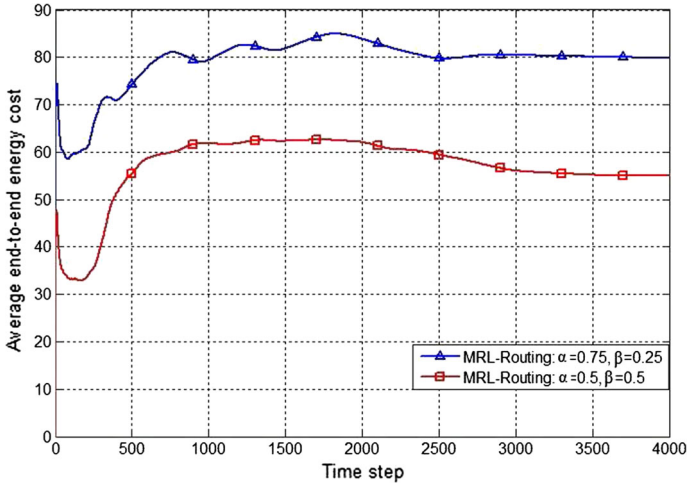


Fig. 9 Average end-to-end energy cost with weight factors $\alpha = 0.75, \beta = 0.25$

Next, we investigate the impact of the number of network nodes on the end-to-end energy cost. Figure 6 plots the average end-to-end energy cost for a source-destination pair in a network with 50 nodes. Figure 7 shows the results for a network with 25 nodes.

As with the case of delay, the energy diagrams indicate a direct correlation between the number of nodes in the network and the average end-to-end energy cost. The proposed MRL-routing algorithm converges much faster than *Q-learning* and its performance is much closer to near-optimal values.

Next, we explore the impact of the weight factors α and β on the performance measures. As mentioned earlier, these parameters can be exploited to strike different energy-delay trade-offs. The average end-to-end delay and energy costs are measured by setting $\alpha = 0.75$ and $\beta = 0.25$ in a network with 50 nodes. The results are compared to the baseline case in which the weight factors are set to 0.5. A higher value for α puts more weight on the

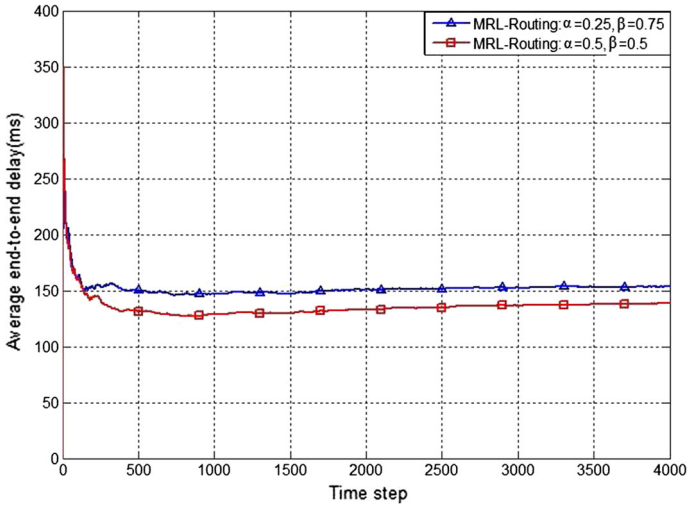


Fig. 10 Average end-to-end delay with weight factors $\alpha = 0.25, \beta = 0.75$

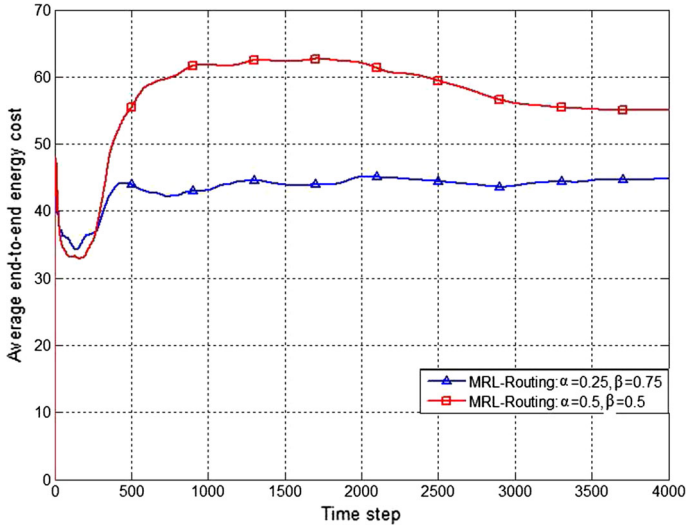


Fig. 11 Average end-to-end energy cost with weight factors $\alpha = 0.25, \beta = 0.75$

end-to-end delay of a source-destination pair, and treats the energy cost as a lower importance criterion.

Figure 8 depicts the average end-to-end delay for both cases of equal and unequal weight factors. As can be observed, for $\alpha = 0.75$ and $\beta = 0.25$, the average end-to-end delay converges to a lower value by trading against the energy criterion. Figure 9 plots the average end-to-end energy cost for $\alpha = 0.75$ and $\beta = 0.25$. As expected, the algorithm converges to a higher energy cost by trading for the delay criterion.

We repeat the same experiment, but this time favoring the energy criterion. Therefore, we interchange α and β values, i.e., we set $\alpha = 0.25$ and $\beta = 0.75$ As it turns out, the

energy cost converges to a lower value; in contrast, the average delay increases accordingly. Figures 10 and 11 show the average end-to-end delay and energy cost for the case $\alpha = 0.25$ and $\beta = 0.75$.

6 Conclusion

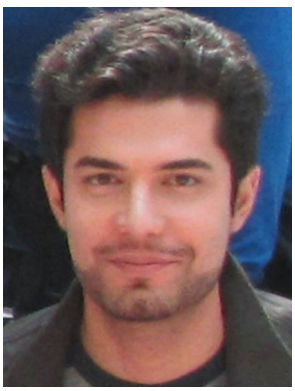
In this paper, we addressed the bi-objective problem of delay and energy efficient routing in energy harvesting MANETs. We showed how a node should choose the next-hop relay in the presence of link and energy dynamics so that in the long run, both the end-to-end delay and network lifetime is optimized. In order to explicitly account for the stochastic dynamics of the network environment, we modeled the routing problem as a Markov decision process (MDP). We also proposed a model-based reinforcement learning (RL)-based algorithm to approximate the optimal routing policy of the formulated MDP. In particular, we have modeled each node's cost function by deriving an expression for the expected value of end-to-end costs. Also, the transition probabilities are estimated online, which enables a node to perform multiple updates following a single interaction with the system. Also, the multi-agent basis of our proposed RL method allows for global system optimization. As evidenced by simulations, compared to a model-free solution, our proposed scheme converges much faster and to better values of system objectives.

References

1. Sharma, V., Mukherji, U., Joseph, V., & Gupta, S. (2010). Optimal energy management policies for energy harvesting sensor nodes. *IEEE Transactions on Wireless Communications*, *9*(4), 1326–1336.
2. Gozalvez, J. (2010). Green radio technologies [mobile radio]. *IEEE Vehicular Technology Magazine*, *5*(1), 9–14.
3. Naruephiphat, W., & Usaha, W. (2008). Balanced energy-efficient routing in MANETs using reinforcement learning. In *International conference on information networking, 2008 (ICOIN 2008)* (pp. 1–5). IEEE.
4. Tao, T., Tagashira, S., & Fujita., S. (2005). LQ-routing protocol for mobile ad-hoc networks. In *Fourth annual ACIS international conference on computer and information science, 2005* (pp. 441–446). IEEE.
5. Binbin, Z., Quan, L., & Shouling, Z. (2010). Using statistical network link model for routing in ad hoc networks with multi-agent reinforcement learning. In *2010 2nd International conference on advanced computer control (ICACC)* (pp. 462–466). IEEE.
6. Dowling, J., Curran, E., Cunningham, R., & Cahill, V. (2005). Using feedback in collaborative reinforcement learning to adaptively optimize MANET routing. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, *35*(3), 360–372.
7. Macone, D., Oddi, G., & Pietrabissa, A. (2013). MQ-routing: Mobility-, GPS-and energy-aware routing protocol in MANETs for disaster relief scenarios. *Ad Hoc Networks*, *11*(3), 861–878.
8. Chang, Y.-H., Ho, T., & Kaelbling, L. P. (2004). Mobilized ad-hoc networks: A reinforcement learning approach. In *Proceedings of the international conference on autonomic computing, 2004* (pp. 240–247). IEEE.
9. Santhi, G., Nachiappan, A., Ibrahime, M. Z., Raghunadhane, R., & Favas, M. (2011). Q-learning based adaptive QoS routing protocol for MANETs. In *2011 International conference on recent trends in information technology (ICRTIT)* (pp. 1233–1238). IEEE.
10. Shiang, H.-P., & van der Schaar, M. (2010). Online learning in autonomic multi-hop wireless networks for transmitting mission-critical applications. *IEEE Journal on Selected Areas in Communications*, *28*(5), 728–741.
11. Boyan, J. A., & Littman, M. L. (1994). Packet routing in dynamically changing networks: A reinforcement learning approach. In J. D. Cowan, G. Tesauro, & J. Alspector (Eds.), *Advances in Neural Information Processing Systems* (Vol. 6, pp. 671–678). Morgan Kaufmann.

12. Al-Rawi, H. A., Ng, M. A., & Yau, K.-L.A. (2013). Application of reinforcement learning to routing in distributed wireless networks: A review. *Artificial Intelligence Review*, *43*(3), 381–416.
13. Zhao, M., Li, Y., & Wang, W. (2012). Modeling and analytical study of link properties in multihop wireless networks. *IEEE Transactions on Communications*, *60*(2), 445–455.
14. Hong, X., Gerla, M., Pei, G., & Chiang, C.-C. (1999). A group mobility model for ad hoc wireless networks. In *Proceedings of the 2nd ACM international workshop on modeling, analysis and simulation of wireless and mobile systems* (pp. 53–60). ACM.
15. Camp, T., Boleng, J., & Davies, V. (2002). A survey of mobility models for ad hoc network research. *Wireless Communications and Mobile Computing*, *2*(5), 483–502.
16. Eisenman, S. B., Miluzzo, E., Lane, N. D., Peterson, R. A., Ahn, G.-S., & Campbell, A. T. (2009). BikeNet: A mobile sensing system for cyclist experience mapping. *ACM Transactions on Sensor Networks (TOSN)*, *6*(1), 6.
17. Edalat, N., Tham, C.-K., & Xiao, W. (2012). An auction-based strategy for distributed task allocation in wireless sensor networks. *Computer Communications*, *35*(8), 916–928.
18. Krishnaswamy, D. (2002). Network-assisted link adaptation with power control and channel reassignment in wireless networks. In *3G wireless conference*.
19. Sutton, R. S., & Barto, A. G. (1998). Reinforcement learning: An introduction. *IEEE Transactions on Neural Networks*, *9*(5), 1054–1054.
20. Tsitsiklis, J. N. (1994). Asynchronous stochastic approximation and Q-learning. *Machine Learning*, *16*(3), 185–202.
21. Barto, A. G., Bradtke, S. J., & Singh, S. P. (1995). Learning to act using real-time dynamic programming. *Artificial Intelligence*, *72*(1), 81–138.
22. Billingsley, P. (1961). *Statistical inference for Markov processes*. Chicago: The University of Chicago Press.

Meisam Maleki received his B.Sc. (Summa Cum Laude) in Information Technology from University of Kurdistan, Sanandaj, Iran in 2011, and his M.Sc. in Information Technology from Amirkabir University of Technology (AUT), Tehran, Iran in 2014. Now his research is focused on computer networking, with special interest in wireless and mobile communications.



Vesal Hakami received his B.S. degree in computer engineering (software) and his M.S. and Ph.D. degrees in information technology (computer networking), all from Amirkabir University of Technology (AUT), Tehran, Iran, in 2004, 2008 and 2015, respectively. In 2016, He joined as an assistant professor to the Department of Computer Engineering, Iran University of Science and Technology (IUST), Tehran, Iran. Prior to joining IUST, he has been a research consultant in Iran Telecommunications Research Center (ITRC). His current research mainly focuses on cognitive control of computer networks using stochastic control theory and game-theoretic learning.

Mehdi Dehghan received his B.Sc. in Computer Engineering from Iran University of Science and Technology, Tehran, Iran in 1992, his M.Sc. in Computer Engineering from Amirkabir University of Technology (AUT), Tehran, Iran in 1995, and his Ph.D. in Electrical Engineering from AUT in 2001. Now, as a Professor, he is the Director of Wireless Networks Lab at the Department of Computer Engineering and Information Technology in AUT. His research entails multimedia networking, distributed systems, and wireless networks.