A Reinforcement Learning-based Bi-Objective Routing Algorithm for Energy Harvesting Mobile Ad-hoc Networks

Meisam Maleki
Department of computer engineering
Amirkabir University of Technology
Tehran, Iran
meisam.maleki@aut.ac.ir

Vesal Hakami
Department of computer engineering
Amirkabir University of Technology
Tehran, Iran
vhakami@aut.ac.ir

Mehdi Dehghan
Department of computer engineering
Amirkabir University of Technology
Tehran, Iran
dehghan@aut.ac.ir

***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 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 (MDP) which captures both the link state dynamics due to node mobility and energy state dynamics due to nodes' rechargeable energy sources. We propose a reinforcement learning-based algorithm to approximate the optimal routing policy in the absence of a priori knowledge of the system statistics. We compare the performance of the proposed scheme with that obtained from a value-iteration-based algorithm which assumes perfect statistics.**

Keywords—MANET; routing; Reinforcement Learning; MDP; end to end delay; network life time

I. 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. Besides the mobility of the nodes, the routing algorithms must face other specific and demanding challenges, such as energy limitations given by the battery of the connected devices. Recent technological advancements have enable energy harvesting capabilities for wireless nodes as a means to mitigate energy scarcity through recharging of a renewable energy source [1]. However, to fully exploit the benefits of this technology, one needs to make special design considerations.

In an energy-harvesting MANET, node mobility gives rise to the randomness of the link quality over time. Also, the energy level of a node is generally random due to the randomness of both the energy harvested from the environment as well as the amount of consumption. Therefore, a principled approach to achieve optimal routing in these networks is to model the problem as a stochastic optimization problem to optimize the system's long-term objectives (e.g., average end to end delay, average energy consumption, etc.). Within this perspective, many works have modeled the MANET routing problem as a Markov decision process (MDP). MDP can directly reflect the random changes of system states and enables stochastic optimization of system objectives. It also comes with an optimal analytical guarantee on the system's average performance measures. However, because the statistics of link qualities and the nodes' power sources are not known a priori, reinforcement learning (RL) is used to determine the optimal routing policy in most cases. RL allows nodes to autonomously sense the network environment, learn the network dynamics based on their local information, and properly adapt their routing decisions to changes [2].

Most of the studies in RL-based MANET routing have utilized the well-established Q-routing algorithm [7] as their underlying idea, albeit with some improvement [3,4,5,6]. Q-routing is based on the traditional Q-learning model in which each node makes its routing decision based on the local routing information. In terms of their objectives, the Q-routing-based algorithm in [5] reduces end-to-end delay. In [3], a routing algorithm is presented that uses Q-routing and couples it with on-policy Monte Carlo to reduce energy consumption and enhance the MANET lifetime. The work in [4] introduces a dynamic discount factor to the operation of Q-routing with the objective of reducing the number of route discovery processes following a link failure. A general objection against Q-routing-based methods is that it acts greedily in environment and optimizes routing metrics locally and does not guarantee network's global quality of service. On the other hand, in the routing algorithms based on multi-agent reinforcement learning (MARL), in order to achieve global optimization, each node in addition to local learning, cooperatively exchanges its local observations with its neighbors [8]. For example, in [9], SAMPLE routing algorithm is proposed based on MARL for MANET networks. In this paper authors use positive and negative feedback to increase the rate of convergence to the optimal policy. Their goal was to increase packet delivery ratio and global throughput of network. The MDP model used in SAMPLE is also used for SNL-Q routing protocol in [10]. Their objective has also been increasing the network throughput, and delay and energy consumption have not been targeted.

978-1-4799-5359-2/14/$31.00 ©2014 IEEE

In all previous studies, a single objective has been considered as the routing metric. Also, none of these works consider energy harvesting capabilities for nodes. Considering these shortcomings, in this paper, we provide an RL-based routing algorithm for MANETs which comes with the following innovations: Unlike previous works, we provide a bi-objective MDP formulation for the routing problem which simultaneously decreases the average end-to-end delay and increases the network lifetime. As mentioned earlier, the nodes in MANET suffer from energy constraints, so merely considering the delay measure is not appropriate. In fact, greedily utilizing a path that has less delay may discharge the nodes on that path and cause disruptions to the network. In the proposed formulation, the impact of node mobility on the link state has been considered explicitly and their quality is supposed to be random. In addition, the energy cost associated with using a node as relay is computed logistically according to its residual energy. The nodes are considered rechargeable and the harvested energy has a random pattern in environment. To learn the optimal routing policy, each node deploys a MARL algorithm. Using this algorithm, each node besides local learning, can exchange its local observations with its neighbors to enable global network-wide optimization. We conduct simulation experiments to demonstrate the convergence properties of our algorithm and its effectiveness in achieving a reasonable energy-delay trade-off.

The rest of the paper is organized as follows: In section II, we discuss the system model and assumptions. In section III, the problem formulation and MDP framework for the routing problem is presented. In section IV, our MARL algorithm for learning the optimal routing policy is discussed. Section V provides simulation results and Section VI concludes the paper.

II. System Model

In this section, we describe the system model and our assumptions for the MANET routing problem. We assume that all nodes are homogeneous (e.g., in terms of transmission power and capacity of battery). Network nodes that reside within the effective transmission range of a node will be recognized as its neighbors. Nodes are capable of recharging, but their charge amount is not constant and follows a random pattern according to the environment they are operating in (see Fig. 1). More formally, in Fig. 1, Γ_i^t represents the amount of energy consumed by node i at time t.

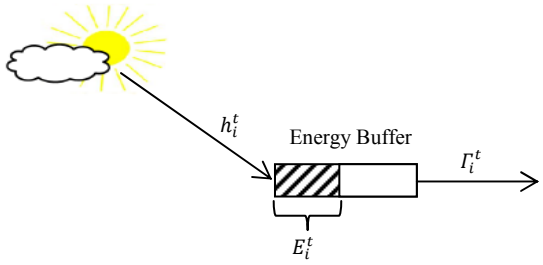

Fig. 1. Energy harvesting model in node i

E_i^t represents the node's energy level at time t and its amount of charge at time t is denoted by h_i^t . Indeed, $\{H_i^t\}_{t\geq 0}$ is an i.i.d stochastic process with overall distribution $\Pr\{H_i\}$ and mean $\bar{H}_i = \mathbb{E}[H_i]$. Also the $\{H_i\}_i$ process is independent with respect to node index *i*.

We consider a reference point group mobility (RPGM) model for node's movements [11]. RPGM model represents the random motion of a group of nodes as well as the random motion of each individual node within the group. Group movements are based upon the path traveled by a logical center for the group. The motion of the group center completely characterizes the movement of the corresponding group of nodes, including their direction and speed. Individual nodes randomly move about their own pre-defined reference points, whose movements depend on the group movement. This motion pattern is common in some important applications 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 [12] are other instances of RPGM.

We assume 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.

III. 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 III.A, and then we define the long-term system objective we seek to optimize in III.B.

A. Routing as a Markov Decision Problem

The infinite horizon MDP in the agent *i* is defined by tuple $< S_i, A_i, C_i, T_i >$ in discrete time *t*=0,1,2,…. S_i represents the set of possible states in node *i*, and A_i is a set of actions that node *i* can perform. $C_i: S_i \times A_i \to \mathbb{R}$ represents immediate cost function that node i receives in each time step for performing an action $a_i \in A_i$ in each state $s_i \in S_i$. $T_i: S_i \times A_i \times S_i \to [0,1]$ shows the state transition probabilities from $s_i \in S_i$ to $s'_i \in S_i$ for performing action $a_i \in A_i$. In the following, we describe each component in further detail:

1) state: Let $N(i)$ be the set of neighbor nodes of *i*. i.e., the nodes that are located within node *i*'s effective transmission range R_e^i. Then, 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 t as follows:

$$dis_{ij}^t = \sqrt{(x_i^t - x_j^t)^2 + (y_i^t - y_j^t)^2} \quad (1)$$

To discretize the distance, the effective transmission range of node i (R_e^i) is divided into k intervals of length ε meters (Fig. 2); i.e.,

$$k = \left\lceil \frac{R_e^i}{\varepsilon} \right\rceil \quad (2)$$

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

$$d_{ij}^t = \left\lceil \frac{dis_{ij}^t}{\varepsilon} \right\rceil \Leftrightarrow dis_{ij}^t \in \left[(d_{ij}^t - 1)\varepsilon, d_{ij}^t \varepsilon \right] \quad (3)$$

More specifically, d_{ij}^t represents the interval that distance between nodes i and $j \in N(i)$ resides in this interval at time t. It is noted 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 $\mathcal{D}_i = \left\{ \mathcal{D}_{ij} \right\}_{j \in N(i)}$.

b) neighbor node's amount of energy ($e_{ij}, j \in N(i)$)*:* the energy of node $j \in N(i)$ is calculated at any time as follows:

$$e_{ij}^{t+1} = max\left\{ \left[e_{ij}^t - \Gamma_j^t \right]^+ + h_j^t, E_{max} \right\} \quad (4)$$

In this equation, Γ_j^t is the amount of consumed energy by node j at time t, and h_j^t is charging amount of node j at time t (c.f., Section II). Symbol $[.]^+$ in (4) represents $\max(.,0)$ operation, E_{max} also is maximum energy of each node. We quantize the energy level of each node into three states: low, medium and high; more formally,

$$e_{ij}^t \epsilon \mathcal{E}_{ij} = \{ 'low' \stackrel{\text{def}}{=} \mathcal{L}, 'medium' \stackrel{\text{def}}{=} \mathcal{M}, 'high' \stackrel{\text{def}}{=} \mathcal{H} \} \quad (5)$$

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

$$s_i^t = \left(d_{ij}^t, e_{ij}^t \right)_{j \in N(i)} \epsilon S_i = \mathcal{D}_i \times \mathcal{E}_i \quad (6)$$

2) Action: the action that each node i performs at time t (a_i^t), is to select one of its neighbors (e.g., node j) as next hop node. In other words:

$$a_i^t \epsilon A_i = N(i) \quad (7)$$

3) Immediate cost function: in order to achive global optimization, the cost function of nodes should be calculated in an end-to-end fashion which involves the exchange of feedback information between nodes. Given that the goal is to reduce the avergae end-to-end delay and also increasing the network lifetime, the immediate cost function is composed of two parts: end-to-end transmission delay and path energy cost. Given that both energy and delay are additive measures, it is possible to decompose the end-to-end cost into two components: the single hop cost and the cumulative cost from the next hop towards the destination. By receiving the single-hop cost directly from neighbor nodes and by backing up the cost values from the destination node back to the source, the cumulative end-to-end cost can be obtained. Therefore, we first discuss the single hop cost $cost_i^t$ and then present the complete end-to-end immediate cost formula. The realized

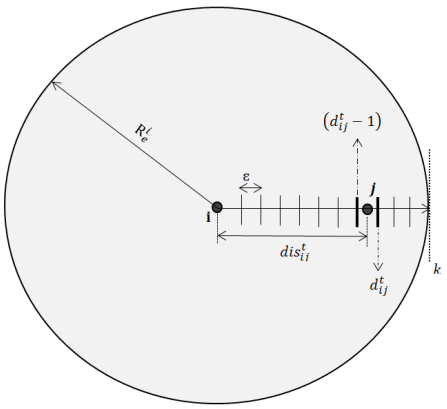

Fig. 2. Effective transmission range for Node i

transmission delay on link ij is denoted by $delay_{ij}^t$, which is directly measurable by node i. The energy cost function is determind logistically by the energy level of the chosen next hop neighbor as relay. The energy level of nodes has an inverse relation with cost value. This means that, the lower the energy level of a chosen relay, the higher would be the cost incurred by node i. Hence, similarly to [13], to model the cost function incurred by node i for selecting node $j \in N(i)$ with the energy level e_{ij}, we use the following logistic function:

$$\mathcal{P}(e_{ij}) = \frac{a}{1 - e^{-e_{ij}/b}} \quad (8)$$

In this equation, a is a scaling parameter and b is preferred coefficient. Overall, we use the following equation for describing the immediate single-hop cost incurred by choosing node j as relay (i.e. $a_i^t = j$) at time t:

$$cost_i^t = \alpha . delay_{ij} + \beta . \mathcal{P}\left(e_{ij}^t \right) . \lambda \quad (9)$$

$cost_i^t$ 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 factor.

With the single-hop cost now fully specified, each node can calculate its immediate end-to-end cost using the feedback it receives from its next-hop. In other words, when node i selects a neighbor node j as relay ($a_i^t = j$), it receives F_j^t as feedback from node j. This feedback represents the cumulative delay of transmitting a packet from node j to destination and also the total energy cost of intermediate nodes from node j to destination. To compute F_i^t, each node i adds its $cost_i^t$ to the feedback F_j^t it receives from relay node j and then sends this total cost as its feedback to the previous hop. Hence, F_i^t is calculated recursively as:

$$F_i^t = F_j^t + cost_i^t \quad (10)$$

4) State transition probabilities: the state transition probabilities in node i's MDP is indicated with function T_i which denotes the transition from state (d_{ij}^t, e_{ij}^t) at time t to state $(d_{ij}^{t+1}, e_{ij}^{t+1})$ 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 RL method in Section IV, each node learns its optimal routing policy only

through interactions with environment and without requiring explicit knowledge of transition probabilities.

B. 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:

$$\pi_i^* \in \arg\min_{\pi_i} \bar{C}^{\pi_i}(s_i) \stackrel{\text{def}}{=} E^{\pi_i}[\sum_{t=0}^{\infty} \gamma^t F_i^t \,|s_i^0 = s_i], \quad \forall s_i \in S_i \tag{11}$$

In (11), $0 < \gamma < 1$ is a discount factor that decays the impact of the cost over successive decision periods.

IV. Reinforcement Learning For Computing the Optimal Routing Policy

The RL based methods [15] converge to the optimal value of a discounted cost function such as (11) by using frequent interactions with the environment and incremental updates. These methods exploit sample realized values of transitions and costs obtained from actual action implementation along with stochastic averaging for learning the average discounted cost. RL algorithms gradually learn an optimal decision policy without knowing the transition probabilities and the closed-form expression of the immediate cost function. 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, $\mathrm{Q_i^t}(s_i^t, a_i^t), \forall s_i^t \in S_i$ denotes the time t estimate of $Q_i^*(s,a)$. We define an asynchronous counter $\upsilon^t(s_i, a_i)$ given by (12). Also let $\alpha(t)$ be a sequence of step-sizes satisfying the conditions in (13).

$$\upsilon^{\mathrm{t}}(\mathrm{s_i}, \mathrm{a_i}) := \sum_{\tau=1}^{\mathrm{t}} \mathbb{I}_{\{(\mathrm{s_i^\tau}, \mathrm{a_i^\tau}) = (\mathrm{s_i}, \mathrm{a_i})\}}. \tag{12}$$

$$\alpha(\mathrm{t}) \rightarrow 0 \text{ as } \mathrm{t} \rightarrow \infty, \sum_{\mathrm{t}} \alpha(\mathrm{t}) = \infty, \sum_{\mathrm{t}} \alpha(\mathrm{t})^2 < \infty. \tag{13}$$

The Q-learning updating rule in (14) guarantees the convergence of the sequence of $\{\mathrm{Q_i^t(s,a)}\}_\mathrm{t}$ estimates to $\mathrm{Q_i^*(s,a)}$ for $\forall \mathrm{s} \in \mathrm{S_i}$ and $\forall \mathrm{a} \in \mathrm{A_i}$ [14].

$$\mathrm{Q_i^{t+1}(s_i, a_i)} - \mathrm{Q_i^t(s_i, a_i)} = \alpha\big(\upsilon^{\mathrm{t}}(\mathrm{s_i}, \mathrm{a_i})\big). \mathbb{I}_{\{(\mathrm{s_i}, \mathrm{a_i}) = (\mathrm{s_i^t}, \mathrm{a_i^t})\}}. \big[\mathrm{cost_i^t} + \mathrm{F_{a_i}^t} + \gamma \min_{\mathrm{a_i}} \mathrm{Q_i^t(s_i^{t+1}, a_i)}\big] \tag{14}$$

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) [15]:

$$\pi_i^{t+1}(s_i^{t+1}, a_i) = \frac{\exp\left(\frac{-\mathrm{Q_i^{t+1}}(s_i^{t+1}, a_i)}{\tau}\right)}{\sum_{\forall a_i \in A_i} \exp\left(\frac{-\mathrm{Q_i^{t+1}}(s_i^{t+1}, a_i)}{\tau}\right)} \tag{15}$$

In (15), 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 $\tau \rightarrow 0$, soft-min action selection reduces to the greedy action selection scheme, and $\pi_\mathrm{i}^\mathrm{t}(.)$ converges to π_i^*.

The most important advantage of our proposed algorithm is its low informational assumptions and low computational complexity. In each time interval, the proposed routing algorithm needs to update the Q-values of $\forall s_i^t \in S_i$ and for each state, $\mathrm{Q_i^t}(s_i^{t+1}, a_i^t)$ over $\forall a_i^t \in A_i$ is calculated. Hence, the computational complexity is $O(|A_i||S_i|)$. The complete steps of our proposed RL-based routing scheme is given in TABLE I.

TABLE I. RL-BASED ROUTING ALGORITHM IN NODE I

Initialization:	t←0, $\mathbf{s_i^0}, \boldsymbol{F_{a_i^t}^0}$, $\mathbf{Q_i^0}(\boldsymbol{s_i^t}, \boldsymbol{a_i^t})$= 0 , $\forall \boldsymbol{s_i^t} \in \boldsymbol{S_i}$, $\forall \boldsymbol{a_i^t} \in \boldsymbol{A_i}$
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 current cost $cost_i^t$ and the new state s_i^{t+1}
3:	Receive $F_{a_i^t}^t$ from next hop neighbor
4:	// Update the Q-value: For state s_i^t and action a_i^t ,update Q-value $\mathrm{Q_i^{t+1}}(s_i^t, a_i^t)$ using equation (14)
5 :	// Update the policy: For s_i^t, update the policy $\pi_i^{t+1}(s_i^t)$
6 :	// Update the feedback values: Update F_i^t Using (10)
7:	t←t+1, go back step 1;

V. Simulation

In this section we evaluate numerically the performance of our RL-based routing algorithm. The performance metrics of interested are 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 [15]. 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. We consider a 1000 m × 1000 m network environment in which a total of 30/60 nodes are deployed. The nodes movement follow a group mobility model with absolute group speed of 10 m/s toward random directions. Also each individual node is assumed to roam around its corresponding

TABLE II. SIMULATION PARAMETERS

Parameter	Value
Network Size(x×y)	1000m ×1000 m
Simulation Time	3000 iteration
Effective transmission range	240m
Packet size(L)	512 byte
Band With	512 kbps
Energy Buffer Size	30

reference point with a predefined average speed. For each source-destination pair, data packets are assumed to be generated at the source following a Poisson process with average interval of 0.5 s. The energy buffer size for each node is taken to be 30. Furthermore, we consider Poisson energy arrival with average arrival rate 1 unit. TABLE II. lists the parameters used in simulations.

In this simulation average end-to-end delay and energy cost are measured and the convergence of the proposed RL algorithm is investigated. First, we investigate the impact of the number of network nodes. We conduct the experiment with 30 and 60 nodes. In each diagram, the performance of the proposed RL-based routing algorithm is compared with the near-optimal solution derived from the standard value-iteration algorithm.

Fig. 3 plots the average end-to-end delay for a source-destination pair in a network with 60 nodes, while Fig. 4 depicts the same for a network of 30 nodes. The main observation in these plots is that our RL-based algorithm converges and that it has acceptable performance compared with the near-optimal solution. As can be seen, the average of end-to-end delay in a network with 60 nodes is higher compared to that of a network with 30 nodes. These values indicate a direct correlation between the number of nodes in the network and the average end-to-end delay. In the following, the impact of the number of network nodes on the end-to-end energy cost of the path is investigated. Fig. 5 plots the average sum of the energy costs of the nodes on the path for a source-destination pair in a network with 60 nodes. Fig. 6 reports the results for a network with 30 nodes.

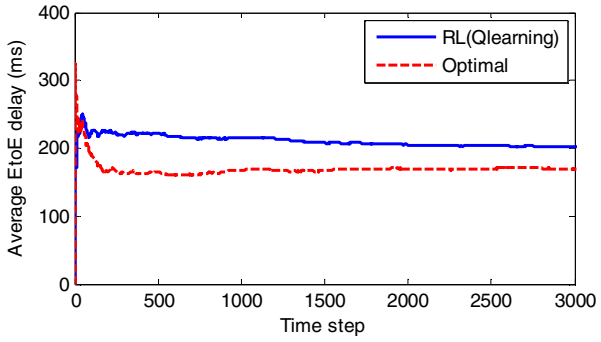

Fig. 3. Avearge end-to-end delay in a network with 60 nodes

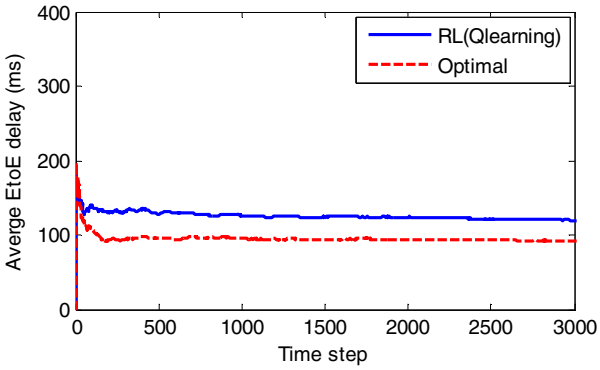

Fig. 4. Average end-to-end delay in a network with 30 nodes

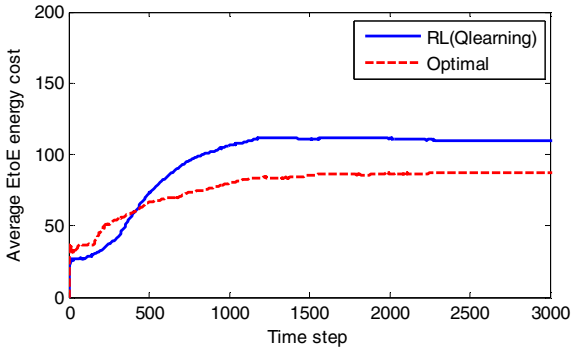

Fig. 5. Average end-to-end energy in a network with 60 nodes

As with the case of delay, our algorithm converges and has acceptable margin from the near-optimal performance.

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 cost are measured by setting $\alpha = 0.75$ and $\beta = 0.25$ in a network with 60 nodes. The results are compared to the baseline case in which the weight factors are set to 0.5. Setting $\alpha > 0.5$ puts more weight on the end-to-end delay of a source-destination pair, and treats the energy cost as a lower importance criterion. Fig. 7 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.

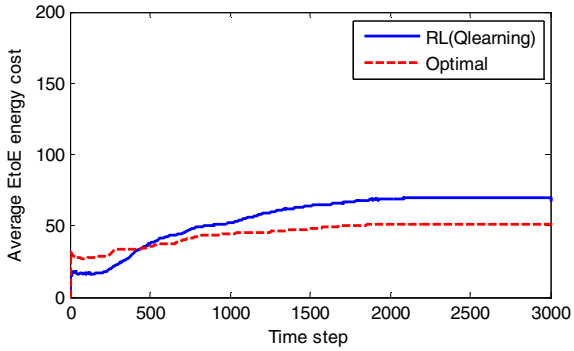

Fig. 6. Average end-to-end energy cost in a network with 30 nodes

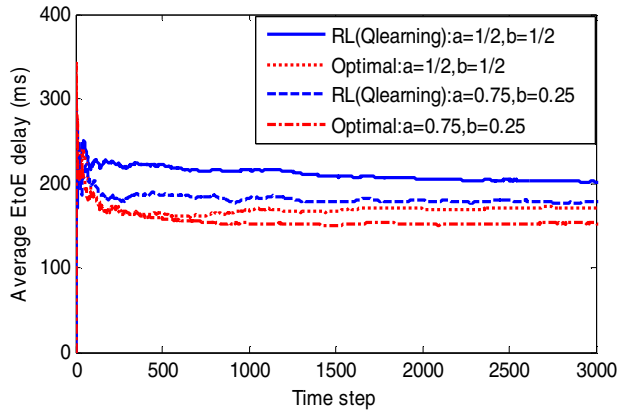

Fig. 7. Average end-to-end delay with weight factors α =0.75,β =0.25

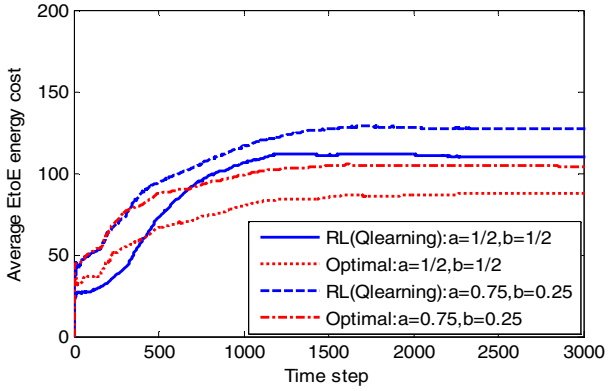

Fig. 8. Average end-to-end energy cost with weight factors α =0.75,β =0.25

Fig. 8 plots the average end-to-end energy cost for α = 0.75 and β = 0.25. As expected, the algorithm converges to a higher energy cost by trading for the delay criterion.

VI. CONCLUSION

In this paper, we presented a systematic way to minimize the average end-to-end delay and energy costs incurred by routing decisions in an energy harvesting MANET. To this end, we formulated the routing problem as a Markov decision problem and proposed a multi-agent reinforcement learning scheme to calculate the optimal routing policy. Simulation results demonstrate that our algorithm converges properly and has an acceptable performance in comparison with the full-knowledge optimal case.

REFERENCES

[1] V. Sharma, U. Mukherji, V. Joseph, and S. Gupta., "Optimal energy management policieis for energy harvesting sensor nodes, " IEEE Trans. Autom. Control, vol. 9, pp. 1326–1336, Apr. 2010.

[2] H. P. Shiang and M. Van Der Schaar, "Online learning in autonomic multi-hop wireless networks for transmitting mission-critical applications," IEEE Journal on Selected Areas in Communications, vol. 28, pp. 728-741, 2010.

[3] W. Naruephiphat and W. Usaha, "Balanced Energy-Efficient Routing in MANETs using Reinforcement Learning," in International Conference on Information Networking, ICOIN, 2008, pp. 1-5.

[4] D. Macone, G. Oddi, and A. Pietrabissa, "MQ-Routing: Mobility-, GPS- and energy-aware routing protocol in MANETs for disaster relief scenarios," *Ad Hoc Networks,* 2012.

[5] Y.-H. Chang, T. Ho, and L. P. Kaelbling, "Mobilized ad-hoc networks: A reinforcement learning approach," in International Conference on Autonomic Computing. Proceedings, 2004, pp. 240-247.

[6] Tao, T., Tagashira, S., Fujita, S.: LQ-Routing Protocol for Mobile Ad-Hoc Networks. In: Proceedings of the Fourth Annual ACIS International Conference on Computer and Information Science, 2005.

[7] J. A. Boyan and M. L. Littman, "Packet routing in dynamically changing networks: A reinforcement learning approach," Proceedings of NIPS Adv neural information processing systems, pp. 671-678, 1994.

[8] H. A. Al-Rawi, M. A. Ng, and K.-L. A. Yau, "Application of reinforcement learning to routing in distributed wireless networks: a review," Artificial Intelligence Review, pp. 1-36, 2013.

[9] J. Dowling, E. Curran, R. Cunningham, and V. Cahill, "Using feedback in collaborative reinforcement learning to adaptively optimize MANET routing," IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans, vol. 35, pp. 360-372, 2005.

[10] B. Zhang, Q. Liu ,and S. Zhao, "Using statistical network link model for routing in ad hoc networks with multi-agent reinforcement learning," International Conference on Advanced Computer Control, pp. 462-466, 2010.

[11] T. Camp, J. Boleng, and V. Davies, "Mobility models for ad hoc network simulations," in Wireless Communication and Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications, vol. 2, pp. 483-502, 2002.

[12] S. B. Eisenman, E. Miluzzo, N. D. Lane, R. A. Peterson, G.-S. Ahn, and A. T. Campbell, "BikeNet: A mobile sensing system for cyclist experience mapping," ACM Transactions on Sensor Networks (TOSN), vol. 6, p. 6, 2009.

[13] N. Edalat, C.-K. Tham, and W. Xiao, "An auction-based strategy for distributed task allocation in wireless sensor networks," Computer Communications, vol. 35, pp. 916-928, 2012.

[14] J. N. Tsitsiklis, "Asynchronous Stochastic Approximation and Q-Learning," Machine Learning, Vol. 16, pp. 185–202, 1994.

[15] R. S. Sutton and A. G. Barto, Reinforcement learning: An introduction vol. 1, Cambridge Univ Press, 1998.