Artif Intell Rev (2015) 43:381–416
DOI 10.1007/s10462-012-9383-6

Application of reinforcement learning to routing in distributed wireless networks: a review

Hasan A. A. Al-Rawi · Ming Ann Ng · Kok-Lim Alvin Yau

Published online: 8 January 2013

Abstract The dynamicity of distributed wireless networks caused by node mobility, dynamic network topology, and others has been a major challenge to routing in such networks. In the traditional routing schemes, routing decisions of a wireless node may solely depend on a predefined set of routing policies, which may only be suitable for a certain network circumstances. Reinforcement Learning (RL) has been shown to address this routing challenge by enabling wireless nodes to observe and gather information from their dynamic local operating environment, learn, and make efficient routing decisions on the fly. In this article, we focus on the application of the traditional, as well as the enhanced, RL models, to routing in wireless networks. The routing challenges associated with different types of distributed wireless networks, and the advantages brought about by the application of RL to routing are identified. In general, three types of RL models have been applied to routing schemes in order to improve network performance, namely Q-routing, multi-agent reinforcement learning, and partially observable Markov decision process. We provide an extensive review on new features in RL-based routing, and how various routing challenges and problems have been approached using RL. We also present a real hardware implementation of a RL-based routing scheme. Subsequently, we present performance enhancements achieved by the RL-based routing schemes. Finally, we discuss various open issues related to RL-based routing schemes in distributed wireless networks, which help to explore new research directions in this area. Discussions in this article are presented in a tutorial manner in order to establish a foundation for further research in this field.

Keywords Q-routing · Routing · Wireless networks · Q-learning · Reinforcement learning

H. A. A. Al-Rawi (✉)· M. A. Ng · K.-L. A. Yau
Department of Computer Science and Networked System,
Faculty of Science and Technology, Sunway University, No. 5 Jalan Universiti,
Bandar Sunway, 46150 Petaling Jaya, Selangor, Malaysia
e-mail: h.al-rawi@lancaster.ac.uk

1 Introduction

Compared to static wired networks, the dynamicity of various properties in distributed wireless networks, including mobility patterns, wireless channels and network topology, have imposed additional challenges to achieving network performance enhancement in routing. Traditionally, routing schemes use predefined sets of policies or rules. Hence, most of these schemes have been designed with specific applications in mind; specifically, each node possesses predefined sets of policies suitable for a certain network condition. Since the policies may not be optimal in other network conditions, the schemes may not achieve the optimal results most of the time due to the unpredictable nature of distributed wireless networks.

The application of Machine Learning (ML) algorithms to solve issues associated with the dynamicity of distributed wireless networks has gained a considerable attention (Forster 2007). ML algorithms help wireless nodes to achieve context awareness and intelligence for network performance enhancement. Context awareness enables a wireless node to observe its local operating environment; while intelligence enables the node to learn an optimal policy, which may be dynamic in nature, for decision making on its operating environment (Yau et al. 2012). In other words, context awareness and intelligence help a wireless node to take actions based on its observed operating environment in order to achieve optimal or near-optimal network performance.

Various ML techniques, such as Reinforcement Learning (RL) (Sutton and Barto 1998), swarm intelligence (Kennedy and Eberhart 1995), genetic algorithms (Gen and Cheng 1999), and neural networks (Forster 2007; Rojas 1996) have been applied to enhance network performance. The choice of a ML algorithm may be based on the characteristics of a distributed wireless network. Examples of distributed wireless networks are Wireless Sensor Networks (WSNs) (Akyildiz et al. 2002), wireless ad hoc networks (Toh 2001), cognitive radio networks (Akyildiz et al. 2009), and delay tolerant networks (Burleigh et al. 2003). In relation to the application of ML in distributed wireless networks, Forster (2007) provides a comparison of various ML algorithms. For instance, in Forster (2007), the application of RL is found to be more suitable for energy-constrained WSNs compared to the swarm intelligence approach. The rationale behind this is that, swarm intelligence usually incurs higher network overheads compared to RL, hence it may consume more energy. Meanwhile, genetic algorithm may be more suitable for centralized wireless networks because it requires global information (Forster 2007).

In distributed wireless networks, routing is a core component that enables a source node to find an optimal route to its destination node. Route selection may depend on the characteristics of the operating environment. Hence, the application of ML in routing schemes to achieve context awareness and intelligence has received a considerable research attention. For instance, in mobile networks, ML-based routing schemes are adaptive to the operating environment because it may be impractical for network designers to establish and develop routing policies for each movement in which the characteristics and parameters of the operating environment change with time and location (Ouzecki and Jevtic 2010; Chang et al. 2004).

This article provides an extensive survey on the application of various RL approaches to routing in distributed wireless networks. Our contributions are as follows. Section 2 presents an overview of RL. Section 3 presents an overview of routing in distributed wireless networks from the perspective of RL. Section 4 presents RL models for routing. Section 5 presents new RL features for routing. Section 6 presents an extensive survey on the application of RL to routing. Section 7 presents an implementation of a RL-based routing scheme in wireless platform. Section 8 presents performance enhancements brought about by the application of RL in various routing schemes. Section 9 presents open issues. Finally, we provide conclusions.

All discussions are presented in a tutorial manner in order to establish a foundation and to spark new interests in this research field.

2 Reinforcement learning

Reinforcement Learning (RL) is a biological-based ML approach that acquires knowledge by exploring its local operating environment without the need of external supervision (Xia et al. 2009; Santhi et al. 2011). A learner (or agent) explores the operating environment itself, and learns the optimal actions based on a trial-and-error concept. RL has been applied to keep track of the relevant factors that affect decision making of agents (Sutton and Barto 1998). In distributed wireless networks, RL has been applied to model the main goal(s), particularly network performance metric(s) such as end-to-end delay, rather than to model all the relevant factors in the operating environment that affect the performance metric(s) of interest. Through learning the optimal policy on the fly, the goal of the agent is to maximize the long-term rewards in order to achieve performance enhancement (Ouzecki and Jevtic 2010).

A RL task that fulfills the Markovian (or memoryless) property is called a Markov Decision Process (MDP) (Sutton and Barto 1998). The Markovian property implies that the action selection of an agent at time t is dependent on the state-action pairs at time $t - 1$ only, rather than the past history at time $t - 2, t - 3, \ldots$. In MDP, an agent is modeled as a four-tuple consisting of $\{S, A, T, R\}$, where S is a set of states, A is a set of actions, T is a state transition probability matrix that represents the probability of a switch from one state at time t to another state at time $t + 1$, and R is a reward function that represents a reward (or cost) r received from the operating environment. At time t, an agent observes state $s \in S$ and chooses action $a \in A$ based on its knowledge (or learned optimal policy). At time $t + 1$, the agent receives a reward r. As time goes by, the agent learns and associates each state-action pair with a reward. In other words, the reward indicates the appropriateness of taking action $a \in A$ in state $s \in S$. Note that, MDP requires an agent to construct and keep track of a model of its dynamic operating environment in order to estimate the state transition probability matrix T. By omitting T, RL learns knowledge through constant interaction with operating environment.

Q-learning is a popular RL approach, and it has been widely applied in distributed wireless networks (Yau et al. 2012). In RL, an agent is modeled as a three-tuple consisting of $\{S, A, R\}$ as described below:

- State. An agent has a set of states S that represent the decision making factors observed from its local operating environment. At any time instant t, agent i observes state $s_t^i \in S$. The state can be internal, such as buffer occupancy rate, or external, such as a destination node. The agent observes its state in order to learn about its operating environment. If the state is partially observable (i.e. operating environment with noise), an agent can estimate its state, which is commonly called the belief state, using Partially Observable Markov Decision Process (POMDP) (Sutton and Barto 1998).
- Action. An agent has a set of available actions A. Examples of actions are data transmission and next-hop node selection. Based on the continuous observation and interaction with the local operating environment, an agent i learns to select an action $a_t^i \in A$ that maximizes its current and future rewards.
- Reward. Whenever an agent i carries out an action $a_t^i \in A$, it receives a reward $r_{t+1}^i(s_{t+1}^i)$ from the operating environment. A reward $r_{t+1}^i(s_{t+1}^i)$ may represent a performance

metric, such as transmission delay, throughput, and channel congestion level. Weight factor may be used to estimate the reward if there are two or more different types of performance metrics. For instance, in Dong et al. (2007), $r^i_{t+1}(s^i_{t+1}) = \omega r^i_{a,t+1}(s^i_{t+1}) + (1-\omega) r^i_{b,t+1}(s^i_{t+1})$ where $r^i_{a,t+1}(s^i_{t+1})$ and $r^i_{b,t+1}(s^i_{t+1})$ indicate rewards for different performance metrics, respectively; and ω indicates the weight factor. There are two types of rewards, namely delayed rewards and discounted rewards (or accumulated and estimated future rewards). Consider that an action is taken at time t, the *delayed reward* represents the reward received from the operating environment at time $t+1$; whereas the *discounted reward* is the accumulated rewards expected to be received from the operating environment in the long run at time $t+1, t+2, \ldots$. The agent aims to learn how to maximize its total rewards comprised of delayed and discounted rewards (Sutton and Barto 1998).

2.1 Q-learning model

Q-learning defines a Q-function $Q^i_t(s^i_t, a^i_t)$, which is also called a state-action function. The Q-function estimates Q-values, which are the long-term rewards that an agent can expect to receive for each possible action $a^i_t \in A$ taken in state $s^i_t \in S$. An agent i maintains a Q-table that keeps track of Q-values for each possible state-action pair, so there are $|S| \times |A|$ entries. Subsequently, based on these Q-values, the agent derives an optimal policy π that defines the best-known action a^i_t, which has the maximum Q-value for each state s^i_t. For each state-action pair (s^i_t, a^i_t) at time t, the Q-value is updated using Q-function as follows:

$$Q^i_{t+1}\left(s^i_t, a^i_t\right) \leftarrow (1-\alpha)\, Q^i_t\left(s^i_t, a^i_t\right) + \alpha \left[r^i_{t+1}\left(s^i_{t+1}\right) + \gamma \max_{a \in A} Q^i_t\left(s^i_{t+1}, a\right)\right] \tag{1}$$

where $0 \leq \alpha \leq 1$ is learning rate, and $0 \leq \gamma \leq 1$ is discount factor. Higher learning rate α indicates higher speed of learning, and it is normally dependent on the level of dynamicity in the operating environment. Note that, too high a learning rate may cause fluctuations in Q-values. If $\alpha = 1$, the agent solely relies on its newly estimated Q-value $r^i_{t+1}(s^i_{t+1}) + \gamma \max_{a \in A} Q^i_t(s^i_{t+1}, a)$, and forgets its current Q-value $Q^i_t(s^i_t, a^i_t)$. On the other hand, γ enables the agent to adjust its preference on the long-term future rewards. Unless $\gamma = 1$ in which both delayed and discounted rewards are given the same weight, the agent always gives more preference to delayed rewards.

2.2 Action selection: exploitation or exploration

During action selection, there are two types of actions, namely, *exploitation* and *exploration*. Exploitation selects the best-known action $a^i_t = \text{argmax}_{a \in A} Q^i_t(s^i_t, a)$, which has the highest Q-value, in order to improve network performance. Exploration selects a random action $a^i_t \in A$ in order to improve knowledge, specifically, the estimation of the Q-values for all state-action pairs. A well-balanced tradeoff between exploitation and exploration helps to maximize accumulated rewards as time goes by. This tradeoff mainly depends on the accuracy of the Q-value estimation, and the level of dynamicity of the operating environment (Yau et al. 2012).

Upon convergence of Q-values, exploitation may be given higher priority because exploration may not discover better actions. A popular tradeoff mechanism is the ε-greedy approach in which the agent performs exploration with a small probability ε (e.g. $\varepsilon = 0.1$) and exploitation with probability $1 - \varepsilon$.

The ε-greedy approach may not be suitable in some scenarios because exploration selects non-optimal actions randomly with equal probability, hence the worst action with the lowest Q-value may be chosen (Sutton and Barto 1998). A popular softmax approach based on the Boltzmann distribution has been applied to choose non-optimal actions for exploration, and actions with higher Q-values are given higher priorities (Sutton and Barto 1998). For instance, in Dowling et al. (2005), a node i chooses its next-hop neighbor node $a_t^i \in A$ using Boltzmann distribution with probability:

$$P\left(s_t^i, a_t^i\right) = \frac{e^{-Q_t^i(s_t^i, a_t^i)/T}}{\sum_{a \in A} e^{-Q_t^i(s_t^i, a)/T}} \tag{2}$$

where A represents a set of node i's neighbor nodes; and T is the temperature factor that determines the level of exploration. Higher T value indicates higher possibility of exploring non-optimal routes, whereas lower T value indicates higher possibility of exploiting optimal routes.

In Bhorkar et al. (2012), a node i chooses its routing decision a_t^i based on the historical information as follows:

$$\varepsilon\left(s_t^i\right) = \frac{1}{c_t^i\left(s_t^i\right) + 1} \tag{3}$$

where $c_t^i(s_t^i)$ is a counter that represents the number of successful packet transmissions from node i to next-hop neighbor node $s_t^i \in S^i$ until time t. Subsequently, with probability $1 - \varepsilon(s_t^i)$, node i chooses its routing decision $a_t^i = \text{argmax}_{a \in A(s_t^i)} Q_t^i(s_t^i, a)$, while with a smaller probability $\varepsilon(s_t^i)$, node i chooses its routing decision $a_t^i \in A(s_t^i)$ equally with probability $\varepsilon(s_t^i)/|A(s_t^i)|$.

In Liang et al. (2008), a node i adjusts its level of exploration according to the level of dynamicity in the operating environment, particularly node mobility. The node computes the exploration probability as follows:

$$\varepsilon_i = \frac{n_i^{a,T} + n_i^{d,T}}{n_i^T} \tag{4}$$

where $n_i^{a,T}$ and $n_i^{d,T}$ are the number of nodes that appear and disappear within node i's transmission range, respectively; n_i^T is the number of node i's neighbor nodes; and T is a time window. Hence, higher ε_i indicates a highly mobile network, and so RL requires more explorations.

2.3 Q-learning algorithm

Figure 1 shows the traditional Q-learning algorithm presented in Sects. 2.1 and 2.2.

3 Routing in distributed wireless networks

Routing is a key component in distributed wireless networks that enables a source node to search and establish route(s) to the destination node through a set of intermediate nodes. The objectives of the routing schemes are mainly dependent on the type of operating environment and the underlying network, particularly its characteristics and requirements. This section reviews the concepts of routing in various types of distributed wireless networks, and the

```
Initialize $s_t^i \in S$; $a_t^i \in A$; $\alpha = [0,1]$; $\gamma = [0,1]$; $Q_t^i(s_t^i, a_t^i) = [Q_{min}, Q_{max}]$
Repeat
  1. Observe the state $s_t^i$
  2. Select an exploitation or exploration action $a_t^i \in A$
     a. If exploitation, choose the best-known action $a_t^i = \text{argmax}_{a \in A} Q_t^i(s_t^i, a)$
     b. If exploration, choose a random action $a_t^i \in A$
  3. Perform the action $a_t^i$ on the operating environment
  4. Observe the next state $s_{t+1}^i$ and reward $r_{t+1}^i(s_{t+1}^i)$ at the next time instance
     $t + 1$
  5. Update Q-value $Q_t^i(s_t^i, a_t^i)$ using Equation (1) as follows:
     $Q_{t+1}^i(s_t^i, a_t^i) \leftarrow (1 - \alpha) Q_t^i(s_t^i, a_t^i) + \alpha \left[ r_{t+1}^i(s_{t+1}^i) + \gamma \max_{a \in A} Q_t^i(s_{t+1}^i, a) \right]$
```

Fig. 1 The traditional Q-learning algorithm at agent *i*

advantages brought about by RL to routing. With respect to routing, Sect. 3.1 reviews several major types of distributed wireless networks, particularly network characteristics, routing challenges, and the advantages brought about by RL to routing. Section 3.2 provides an overview of the application of RL to routing in distributed wireless networks, and a general formulation of the routing problem using RL.

3.1 Types of distributed wireless networks

This section presents four types of distributed wireless networks, namely wireless ad hoc networks, wireless sensor networks, cognitive radio networks, and delay tolerant networks. Table 1 summarizes each type of these networks.

3.1.1 Wireless ad hoc networks

A wireless ad hoc network is comprised of self-configuring static or mobile nodes. Two main types of wireless ad hoc networks are static ad hoc networks and Mobile Ad hoc NETworks (MANETs) (Toh 2001; Boukerche 2009).

Figure 2 shows a wireless ad hoc network scenario. Nodes within the range of each other (e.g. A and B) may communicate directly; and out-of-range nodes (e.g. A and F) may use a routing scheme to search for a route, comprised of intermediate nodes, from node (A) to node (F). The routing scheme uses a cost metric to compute the best possible route, such as the shortest route and route with the lowest end-to-end delay.

The main routing challenge in wireless ad hoc networks is the dynamic topology caused by nodes' mobility. For instance, in Fig. 2, source node (A) establishes a route (A-H-G-F) to destination node (F). Suppose, node (H) moves and becomes out-of-range from node (A) resulting in link breakage, then node (A) searches for another route to node (F). The packet end-to-end delay and packet loss rate are dependent on the effectiveness of the routing scheme. Furthermore, in link-state routing schemes, such as Optimized Link State Routing (OLSR) (Clausen and Jacquet 2003), each node maintains a route to every other nodes in the network.

Table 1 Characteristics of distributed wireless networks

	Wireless ad hoc networks	**Wireless sensor networks**	**Cognitive radio networks**	**Delay tolerant networks**
Examples of applications	Vehicular networks	Physical environment monitoring	Extension of broadband service to rural communities	High-speed vehicular networks
	Multiplayer games	Security surveillance	Emergency and rescue operations	Spacecraft communications
	Emergency and rescue operations	Health applications		Emergency and rescue operations
Main routing challenge(s)	High mobility	Limited energy and processing capabilities	Dynamicity of channel availability	Lack of end-to-end routes between any two nodes at most of the times
			Minimizing interference to licensed users	
Main advantage(s) brought about by RL	Adaptive to dynamic topology	Has lower computational cost	Adaptive to dynamic channel availability	Adaptive to dynamic topology
		Incurs lower routing overhead		

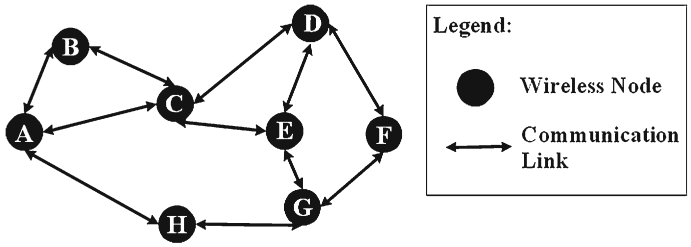

Fig. 2 Wireless ad hoc network scenario

In highly mobile networks, the OLSR constantly updates these routes due to link breakages, causing high computing cost and routing overhead.

RL-based routing schemes have been shown to be highly adaptive to topology changes (Forster 2007). For example, RL enables a node to observe its neighbor nodes' mobility characteristics, and to learn how to improve the end-to-end delay and throughput performances of routes. Subsequently, the node selects a next-hop node that can satisfy the Quality of Service (QoS) requirements imposed on the route.

3.1.2 Wireless sensor networks

Wireless Sensor Networks (WSNs) are comprised of sensor nodes with sensing, computing, storing, and short-range wireless communication capabilities commonly used for monitoring the operating environment. WSNs share similar characteristics with wireless ad hoc networks

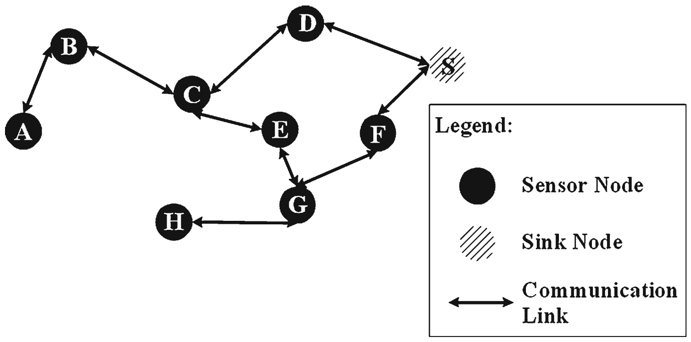

Fig. 3 WSN scenario

in that both are multi-hop networks. An intrinsic characteristic of WSNs is that the sensor nodes are highly energy constrained with limited processing capability (Akyildiz et al. 2002).

In WSNs, there is a special gateway called a sink node as shown in Fig. 3. The sink node monitors a WSN by sending control messages to the sensor nodes, and gathers sensing outcomes from them.

The main routing challenge in WSNs is the need to reduce energy consumption and computational cost at sensor nodes in order to prolong network lifetime. For instance, routing schemes for WSNs must avoid frequent flooding of routing information in order to reduce energy consumption.

Since RL incurs low computational cost and routing overhead (Forster 2007), RL-based routing is suitable for WSNs. For instance, RL enables a sensor node to observe and estimate energy consumption of nodes along a route based on local observations, so that each node can perform load-balancing and select routes with higher residual energy in order to prolong network lifetime.

3.1.3 Cognitive radio networks

Cognitive Radio (CR) is the next generation wireless communication systems that address issues associated with the efficiency of spectrum utilization (Akyildiz et al. 2009). In CR networks, unlicensed users (or Secondary Users, SUs) exploit and use underutilized licensed channels. A distributed CRN shares similar characteristics with wireless ad hoc networks in that both are multi-hop networks. An intrinsic characteristic of CRNs is that the SUs must prevent harmful interference to the licensed users (or Primary Users, PUs), who own the channels. Since the SUs must vacate their channels whenever any PU activity appears, the channel availability is dynamic in nature.

The main routing challenge in CRNs is that, since SUs must be adaptive to the dynamic changes in spectrum availability, routing in CRNs must be spectrum-aware (Al-Rawi and Yau 2012). Figure 4 shows a CRN scenario co-located with three PU Base Stations (BSs). Suppose, SU (A) wants to establish a route to SU BS. Using a traditional routing algorithm may provide a route with the minimum number of hops (A–C–E–G) to the SU BS. However, the SUs may suffer from poor network performance because the route passes through three PU BSs and their hosts (B, D, F, H), resulting in harmful interference to the PUs. On the other hand, CR-based spectrum-aware routing may provide a route with higher number of hops (A–C–I–K–L) that generates less interference to the PUs and their hosts (B, D, J), and so it provides better end-to-end SU performance.

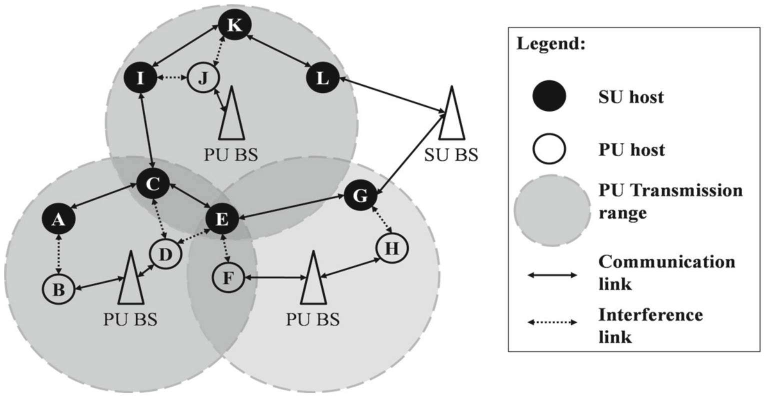

Fig. 4 CRN scenario

RL has been shown to improve network performance of CRNs. For instance, based on the local observations and received rewards, a SU node may learn the behavior and channel utilization of PUs. Subsequently, the SU selects a route that reduces end-to-end interference to PUs.

3.1.4 Delay tolerant networks

Delay Tolerant Networks (DTNs) interconnect highly heterogeneous nodes (Burleigh et al. 2003), and several common assumptions adopted by the traditional networks are relaxed. Examples of the assumptions are low end-to-end delay, low error rate, and the availability of end-to-end routes in the entire network. Nevertheless, these assumptions may not be practical due to various challenges, such as highly mobile networks, low-quality wireless links and small coverage, as well as low residual energy.

The aforementioned assumptions have imposed new challenges to routing. As an example, due to the lack of end-to-end routes between any two nodes for most of the times, the reactive and proactive routing schemes may not be applicable to DTNs (Elwhishi et al. 2010). Using traditional Ad hoc On-Demand Distance Vector (AODV) (Perkins and Royer 1999) routing scheme may constantly establish routes during frequent link breakages, and this increases energy consumption and routing overhead. Hence, routing in DTNs may need to follow a "store-and-forward" approach in which a node buffers its data packets until a link between itself and a next-hop node becomes available (Elwhishi et al. 2010).

RL-based routing schemes have been shown to be adaptive to link changes by choosing a next-hop node based on various local states (or conditions) that affect a link's availability. For instance, without using the global information, a node may observe and learn about its local operating environment, such as link congestion level and buffer utilization level of the next-hop node, so that it selects a route with higher link availability and lower buffer utilization in order to increase packet delivery rate (Elwhishi et al. 2010).

3.2 RL in the context of routing in distributed wireless networks

The RL-based routing schemes have seen most of their applications in four types of distributed wireless networks (see Sect. 3.1), namely ad hoc networks, wireless sensor networks, cognitive radio networks, and delay tolerant networks. Table 2 shows the application of RL to routing in various distributed wireless networks.

Table 2 RL-based routing schemes in various distributed wireless networks

Types of networks	Bhorkar et al. (2012)	Lin and Schaar (2011)	Santhi et al. (2011)	Nurmi (2007)	Dowling et al. (2005)	Chang et al. (2004)	Usaha (2004)	Naruephiphat and Usaha (2008)	Fu et al. (2005)	Chetret et al. (2004)
Static ad hoc networks	×	×								
Mobile Ad hoc NETworks (MANETs)			×	×	×	×	×	×	×	×
Wireless Sensor Networks (WSNs)										
Cognitive Radio Networks (CRNs)										
Delay Tolerant Networks (DTNs)										

Table 2 continued

Types of networks	Forster and Murphy (2007)	Hao and Wang (2006)	Baruah and Urgaonkar (2004)	Hu and Fei (2010)	Arroyo-Valles et al. (2007)	Liang et al. (2008)	Dong et al. (2007)	Zhang and Fromherz (2006)	Xia et al. (2009)	Elwhishi et al. (2010)
Static ad hoc networks										
Mobile Ad hoc NETworks (MANETs)										
Wireless Sensor Networks (WSNs)	×	×	×	×	×	×	×	×		
Cognitive Radio Networks (CRNs)									×	
Delay Tolerant Networks (DTNs)										×

Routing in distributed wireless networks has been approached using RL so that each node makes local decision, with regards to next-hop or link selection as part of a route, in order to optimize network performance. In routing, the RL approach enables a node to:

a. estimate the dynamic link cost. This characteristic allows a node to learn about and adapt to its dynamic local operating environment.
b. search for the best possible route using information observed from the local operating environment only.
c. incorporate a wide range of factors that affect the routing performance.

Table 3 shows a widely used RL model for the routing schemes in distributed wireless networks. The state represents all the possible destination nodes in the network. The action represents all the possible next-hop neighbor nodes, which may be selected to relay data packets to a given destination node. Each link within a route may be associated with different types of dynamic costs (Nurmi 2007), such as queuing delay, available bandwidth or congestion level, packet loss rate, energy consumption level, link reliability and changes in network topology, as a result of irregular node's movement speed and direction. Based on the objectives, a routing scheme computes its Q-values, which estimate the short-term and long-term rewards (or costs) received (or incurred) in transmitting packets along a route to a given destination node. Examples of rewards are throughput and packet delivery rate performances; and an example of cost is end-to-end delay. To maximize (minimize) the accumulated rewards (costs), an agent chooses an action with the maximum (minimum) Q-value.

Figure 5 shows how a RL model (see Table 3) can be incorporated into routing. All possible states are $S = \{1, 2, 3, 4, 5, 6\}$. All possible actions of node $i = 1$ are $a_t^i \in \mathrm{A} = \{2, 3, 4\}$. Suppose, node $i = 1$wants to establish a route to node $s_t^i = 6$, and the objective of the

Table 3 A widely used RL model of agent i for a routing scheme

State	$s_t^i \in S = \{1, 2, \ldots, N-1\}$, each state s_t^i represents a destination node n. N represents the number of nodes in the entire network
Action	$a_t^i \in A = \{1, 2, \ldots, J\}$, each action a_t^i represents the selection of a next-hop neighbor node j. J represents the number of node i's neighbor nodes
Reward	$r_t^i \left(s_t^i, a_t^i\right)$, which is dependent on the objective of the routing scheme, is a reward (or cost) that represents a performance metric, such as queuing delay, transmission delay, throughput, channel utilization level, and mobility factor

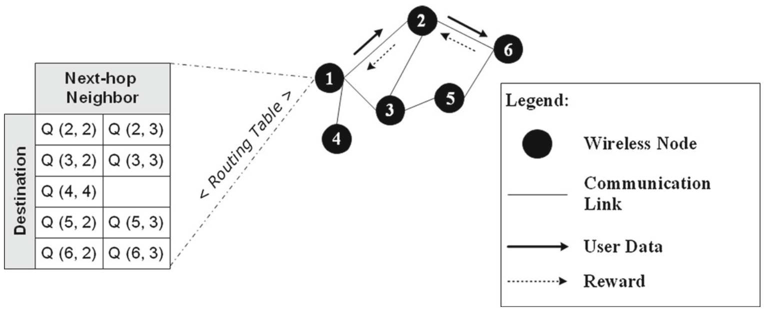

Fig. 5 RL-based routing scenario

routing scheme is to find a route that provides the highest throughput performance. Node i may choose to send its packets to its neighbor node $a_t^i = 2$, and receives a reward from node $a_t^i = 2$ that estimates throughput achieved by the route from upstream node $a_t^i = 2$ to destination node $s_t^i = 6$, specifically route (2–6). Subsequently, node i updates its Q-value for state $s_t^i = 6$ via action $a_t^i = 2$ using Q-function (see Eq. 1), specifically $Q_t^i \left(s_t^i = 6, a_t^i = 2\right)$. Likewise, when node i sends its data packets to node $a_t^i = 3$, it receives a reward that estimates the throughput from upstream node $a_t^i = 3$, which can be either (3–2–6) or (3–5–6), and updates $Q_t^i \left(s_t^i = 6, a_t^i = 3\right)$. Note that, whether route (3–2–6) or (3–5–6) is chosen by upstream node $a_t^i = 3$ is dependent on the Q-values of $Q_t^{i=3} \left(s_t^{i=3} = 6, a_t^{i=3} = 2\right)$ and $Q_t^{i=3} \left(s_t^{i=3} = 6, a_t^{i=3} = 5\right)$ at node $a_t^i = 3$, and the upstream node that provides the maximum Q-value is the exploitation action. The matrix in Fig. 5 shows an example of routing table (or Q-table) being constantly updated at node i. Node i keeps track of the Q-values of all possible destinations through its next-hop neighbor nodes in its Q-table.

4 Reinforcement learning models for routing

RL models have been applied to routing schemes in various distributed wireless networks. The RL models are: Q-routing, Multi-Agent Reinforcement Learning (MARL), and Partially Observable Markov Decision Process (POMDP). The rest of this section discusses the RL models.

4.1 Q-routing model

Boyan and Littman (1994) propose Q-routing, which is based on the traditional Q-learning model (Sutton and Barto 1998). In Q-routing, a node chooses a next-hop node, which has the minimum end-to-end delay, in order to mitigate link congestion (Ouzecki and Jevtic 2010; Chang et al. 2004). The traditional Q-routing approach has also been adopted as a general approach to improve network performance in Zhang and Fromherz (2006).

In Q-routing, the state s_t^i represents a destination node in the network. The action a_t^i represents the selection of a next-hop neighbor node to relay data to a destination node s_t^i. Each link of a route is associated with a dynamic delay cost comprised of queuing and transmission delays. Subsequently, for each state-action pair (or destination and next-hop neighbor node pair), a node computes its Q-value, which estimates the end-to-end delay for transmitting packets along a route to a destination node s_t^i. Specifically, at time instant $t+1$, a particular node i updates its Q-value $Q_t^i(s_t^i, j)$ to a destination node s_t^i via a next-hop neighbor node $a_t^i = j$. Hence, Eq. (1) is rewritten as follows:

$$Q_{t+1}^i \left(s_t^i, j\right) \leftarrow (1-\alpha)\, Q_t^i \left(s_t^i, j\right) + \alpha \left[r_{t+1}^i \left(s_{t+1}^i, j\right) + \min_{k \in a_t^j} Q_t^j \left(s_t^j, k\right) \right] \tag{5}$$

where $0 \leq \alpha \leq 1$ is the learning rate; $r_{t+1}^i(s_{t+1}^i, j) = d_{qu,t+1}^i + d_{tr,t+1}^{i,j}$ represents two types of delays, specifically $d_{qu,t+1}^i$ is the queuing delay at node i, and $d_{tr,t+1}^{i,j}$ is the transmission delay between node i and its next-hop neighbor node j; and $Q_t^j(s_t^j, k)$, which is a Q-value received from next-hop neighbor node j, is the estimated end-to-end delay along the route from node j's next-hop neighbor node $k \in a_t^j$ to the destination node.

Referring to Fig. 5, suppose node $i = 1$wants to establish a route to node $s_t^i = 6$ using the Q-routing model. Node $i = 1$ may choose its next-hop neighbor node $a_t^i = j = 2$ to forward its data packets. When node i sends its data packets to node $j = 2$, it receives from neighbor node $j = 2$ an estimate of $\min_{k \in a_t^j} Q_t^j(s_t^j, k)$ that represents the estimated minimum end-to-end delay from node j to destination nodes_t^i. Node $i = 1$ also measures its queuing delay $d_{qu,t+1}^i$ and transmission delay $d_{tr,t+1}^{i,j}$ between itself and neighbor node $j = 2$. Subsequently, using Eq. (5), node i updates its Q-value, $Q_t^i(s_t^i = 6, a_t^i = 2)$ that represents the end-to-end delay from itself to destination node $s_t^i = 6$ through the chosen next-hop neighbor node $a_t^i = 2$.

4.2 Multi-agent reinforcement learning model

The traditional RL model, which is greedy in nature, provides local optimizations regardless of the global performance; and so, it is not sufficient to achieve global optimizations or a network-wide QoS provisioning. This can be explained as follows: since nodes share a common operating environment in wireless networks, a node's neighbor nodes may take actions that affect its own performance due to channel contention. In Multi-Agent Reinforcement Learning (MARL), in addition to learning locally using the traditional RL model, each node exchanges locally observed information with neighboring nodes through collaboration in order to achieve global optimizations. This helps the nodes to consider not only their own performance, but also others' performance. Hence, the Multi-Agent Reinforcement Learning (MARL) model extends the traditional RL model through fostering collaboration among neighboring nodes so that a system-wide optimization problem can be decomposed into a set of distributed problems solved by individual nodes in a distributed manner.

Referring to Fig. 5, node $i = 1$ constantly exchanges knowledge (i.e. Q-values and rewards) with neighbor nodes $j = 2, 3$ and 4. As an example, in Dowling et al. (2005), the MARL-based routing scheme addresses a routing challenge in which a node i selects its next-hop neighbor node j with the objective of increasing network throughput and packet delivery rate in a heterogeneous mobile ad hoc network (see Sect. 3.1.1). Each node may possess different capabilities in solving the routing problem in a heterogeneous environment. Hence, the nodes share their knowledge (i.e. route cost) through message exchange. The exchanged route cost is subsequently applied by a node i to update its Q-values so that an action, which maximizes the rewards of itself and its neighboring nodes, is chosen.

4.3 Partially observable Markov decision process model

The Partial Observable Markov Decision Process (POMDP) model extends the Q-routing and Multi-agent RL model. In POMDP-based routing model, a node is not able to clearly observe its operating environment. Since the state is unknown, the node must estimate the state. For instance, the state of a node may incorporate its next-hop neighbor node's local parameters, such as the forwarding selfishness, residual energy, and congestion level.

As an example, in Nurmi (2007), the routing scheme addresses a routing challenge in which a node i selects its next-hop neighbor node j with the objective of minimizing energy consumption. Node j's forwarding decision, which is based on its local parameters (or states) such as the forwarding selfishness, residual energy and congestion level, is unclear to node i. Additionally, node j's decision is stochastic in nature. Hence, node j's information is unknown to node i. The routing scheme is formulated as a POMDP problem in which node i

estimates the probability distribution of the local parameters based on its previous estimation and its observed historical actions H_t^j of node j.

5 New features

This section presents the new features that have been incorporated into the traditional RL-based routing models in order to further enhance network performance.

5.1 Achieving balance between exploitation and exploration

Exploitation enables an agent to select the best-known action(s) in order to improve network performance; while exploration enables an agent to explore the random actions in order to improve the estimation of Q-value for each state-action pair. A well-balanced trade-off between exploitation and exploration in routing is important to maximize accumulated rewards as time goes by.

Hao and Wang (2006) propose a Bayesian exploration approach for exploration. The Bayesian approach (Dearden et al. 1999) constantly updates a belief state (also called an estimated state) based on the historical observations of the state in order to address the uncertainty of MDPs (see Sect. 2). This approach estimates two values. Firstly, it estimates the expected Q-values in the future, $E[Q_t^i\left(s_t^i, a_t^i\right)]$ based on a constantly updated model of transition probabilities and reward functions. Secondly, it estimates the expected reward $E[r_t^i(s_t^i, a_t^i)]$ for choosing an exploration action. This approach chooses the next action with the maximum value of $E[Q_t^i(s_t^i, a_t^i)] + E[r_t^i(s_t^i, a_t^i)]$ in order to make a balanced tradeoff between exploitation and exploration. This approach has been shown to provide higher accumulated rewards compared to the traditional Q-learning approach.

In Forster and Murphy (2007), routes are assigned with exploration probabilities such that routes with lower costs are initially assigned with higher exploration probabilities. Subsequently, during the learning process, the exploration probability of a route is adjusted by a constant factor f based on the selection frequency and the received rewards. There are three types of received rewards, namely positive, negative, and neutral rewards. The *positive* and *negative* rewards are received whenever there are any changes to the operating environment; and the *neutral* are received when learning has achieved convergence. For instance, for a route via next-hop node j, its exploration probability is decreased by a value of f each time it is being selected. This scheme has been shown to provide higher convergence rate compared to the traditional uniform exploration methods.

Fu et al. (2005) adopt a genetic-based approach for exploration. In the traditional RL-based routing scheme, data packets are routed using exploitation actions regardless of their respective service classes. Consequently, in this scheme, exploration is adjusted using a genetic-based approach in order to discover routes based on the QoS requirement(s) of packets (e.g. throughput and end-to-end delay). Based on the genetic algorithm, each gene represents a route between a source and a destination node pair. The length of a chromosomes length changes with the dynamicity of the operating environment. The fitness of a route is based on the delay and throughput of the route. Subsequently, routes are ranked and selected based on their fitness.

5.2 Achieving higher convergence rate

Convergence to an optimal policy can be achieved after some learning time. Nevertheless, the speed of convergence is unpredictable and may be dependent on the dynamic operating environment. Traditionally, the *learning rate* α is used to adjust the speed of convergence. Higher learning rate may increase the convergence speed; however, the Q-value may fluctuate, particularly when the dynamicity of the operating environment is high because the Q-value is now dependent more on its recent estimates, rather than its previous experience.

Nurmi (2007) applies a stochastic learning algorithm, namely Win-or-Learn-Fast Policy Hill Climbing (WoLF-PHC) (Bowling and Veloso 2002), to adjust the learning rate dynamically based on the dynamicity of the operating environment in distributed wireless networks. The algorithm defines *Winning* and *Losing* as receiving higher and lower rewards than its expectations, respectively. When the algorithm is winning, the learning rate is set to a lower value, and vice-versa. The reason is that, when a node is winning, it should be cautious in changing its policy because more time should be given to the other nodes to adjust their own policies in favor of this winning. On the other hand, when a node is losing, it should adapt faster to any changes in the operating environment because its performance (or rewards) is lower than expected.

Kumar and Miikkulainen (1997) apply a dual RL-based approach to speed up the convergence rate by updating the Q-values of previous state (i.e. source node) and next state (i.e. destination node) simultaneously although this may increase the routing overhead. The traditional Q-routing model updates the Q-values in regards to the destination node (see Sect. 4.1). On the other hand, dual RL-based Q-routing model updates Q-values in regards to destination and source nodes. Since the dual RL-based approach updates Q-values of a route in both directions, it enables nodes along a route to make decisions on next-hop selection for both source and destination nodes while increasing the speed of convergence.

Hu and Fei (2010) use a system model to estimate the converged Q-values of all actions so that it may not be necessary to update Q-values only after taking the corresponding actions. This is achieved through running virtual experiment or simulation to update Q-values using a system model. The system model is comprised of a state transition probability matrix T (see Sect. 2), which is estimated using historical data of each link's successful and unsuccessful transmission rate based on the outgoing traffic of next-hop neighbor nodes.

5.3 Detecting the convergence of Q-values

When the Q-values have achieved convergence, further exploration may not change the Q-values, and so an exploitation action should be chosen. Hence, the detection of the convergence of Q-values helps to enhance network-wide performance.

Forster and Murphy (2007) propose two techniques to detect the convergence of Q-values for each route with the objective of enhancing the learning process so that the knowledge is sufficiently comprehensive. Specifically, the first technique ensures the convergence of each route at the *unit* level, while the second technique enhances the convergence of N different routes at the *system* level. The first technique assumes that the Q-value of a route has achieved convergence when the route receives M static (or unchanged) rewards. On the other hand, the second technique requires at least N routes have been explored, which ensures that convergence is achieved at the system level. Combining both techniques, the convergence is achieved when there are N routes receiving M static rewards.

5.4 Storing Q-values efficiently

When the number of states (e.g. destination nodes) increases, memory requirement to store the Q-values for all state-action pairs may increase exponentially. Storing the Q-values efficiently may reduce the memory requirement.

Chetret et al. (2004) adopt an approach called Cerebellar Model Articulation Controller (CMAC) (Albus 1975) in neural network to store the Q-values, which represent the end-to-end delay of routes. The advantage of CMAC is that, it uses a constant memory requirement to store the Q-values. CMAC stores smaller values with higher accuracy compared to larger values. This can helpful for routing schemes that aim to achieve low end-to-end delay because low values of end-to-end delay are stored (Chetret et al. 2004). CMAC computes a function using inputs, which can be represented by multiple dimensions, to store the values. Each value is represented by a set of points in an input space, which is a hypercube. Each of these points represents a memory cell in which the data is stored. Hence, a value is partitioned and stored in multiple memory cells. In order to retrieve a value, the corresponding points (or memory cells) of a hypercube must be activated to retrieve all portions of the original value. The combined portions are the stored Q-value. Furthermore, Least-Mean Square (LMS) algorithm (Yin and Krishnamurthy 2005) is adopted to update the weights, which are the learning parameters, in CMAC. The LMS is a stochastic gradient descent approach (Snyman 2005) that can minimize errors in representing the Q-values in each dimension of the hypercube.

5.5 Application of rules

Rules can be incorporated into the traditional Q-learning approach in order to fulfill network requirements, such as minimum end-to-end delay and number of hops to the destination node, of a routing scheme (Yau et al. 2012). Rules can be applied to exclude actions if their respective Q-values are higher (or lower) than a certain threshold.

Yu et al. (2008) calculate a ratio of the number of times an action a_t^i violates a rule to the total number of times the action a_t^i is executed. When the ratio exceeds a certain threshold, the action a_t^i is excluded from action selection in the future. In Liang et al. (2008), Lin and Schaar (2011), a node reads the QoS requirements, particularly end-to-end delay, encapsulated in the data packets. If the estimated amount of time to be incurred in the remaining route to the destination node will not fulfill the end-to-end delay requirement, the node will not forward the packet to a next-hop node.

Further research could be pursued to investigate the application of rules to address other issues in routing. For instance, rules may be applied to detect the malicious nodes, which may advertise the manipulated Q-values with a bad intention to adversely affect the routing decisions of other nodes.

5.6 Approximation of the initial Q-values

The traditional RL approach initializes Q-values with random values, which may not represent the real estimations, and so non-optimal actions are taken during the initial stage. Subsequently, learning takes place to update these values until the Q-values have achieved convergence. The initial random values may reduce the convergence rate, and may cause fluctuations in network performance, which occur especially at the beginning of the learning process. As a consequence, it is necessary to initialize Q-values to approximate values, rather than random values.

In Forster and Murphy (2007), Q-values are initialized based on the number of hops to each destination sink node in WSNs (see Sect. 3.1.2). Specifically, the sink node broadcasts request packets; and each sensor node initiates its Q-values as the sink node's request packets passing through it, hence each sensor node has estimation on the number of hops to the sink node. Subsequently, learning updates values using the received rewards.

6 Application of reinforcement learning to routing in distributed wireless networks

This section presents how various routing schemes have been approached using RL to provide network performance enhancement. We describe the main purpose of each scheme, and its RL model.

6.1 Q-routing model

This section discusses the routing schemes that adopt the Q-routing model (see Sect. 4.1).

6.1.1 Q-routing approach with forward and backward exploration

Dual RL-based Q-routing (Kumar and Miikkulainen 1997), which is an extension of the traditional Q-routing model, enhances network performance and convergence speed (see Sect. 5.2). Xia et al. (2009) also apply a Dual RL-based Q-routing approach in CRNs (see Sect. 3.1.3), and it has been shown to reduce end-to-end delay. In CRNs, the availability of a channel is dynamic, and it is dependent on the PU activity level. The purpose of the routing scheme is to enable a node to select a next-hop neighbor node with higher number of available channels. Higher number of available channels reduces channel contention, and hence reduces the MAC layer delay.

Table 4 shows the Q-routing model for the routing scheme at node i. Note that, the state and action representations are not shown, and they are similar to the general RL model in Table 3. The state s_t^i represents a destination node n. The action a_t^i represents the selection of a next-hop neighbor node j. The reward $r_t^i(s_t^i, a_t^i)$ represents the number of available common channels between node i and node $a_t^i = j$. The Q-routing model is embedded in each SU node.

Node i's Q-value indicates the total number of available channels at each link along a route to destination node s_t^i through a next-hop neighbor node $a_t^i = j$. Node i chooses a next-hop neighbor node $a_t^i = j$ that has the maximum $Q_t^i(s_t^i, j)$. Hence, Eq. (5) is rewritten as follows:

$$Q_{t+1}^i\left(s_t^i, j\right) \leftarrow (1-\alpha)\, Q_t^i\left(s_t^i, j\right) + \alpha \left[r_{t+1}^i\left(s_{t+1}^i, j\right) + \max_{k \in a_t^j} Q_t^j\left(s_t^j, k\right) \right] \tag{6}$$

where k is the next-hop neighbor node of $a_t^i = j$.

Traditionally, Q-routing performs *forward exploration* by updating the Q-value $Q_t^i(s_t^i, j)$ of node i whenever a feedback, specifically $\max_{k \in a_t^j} Q_t^j(s_t^j, k)$, is received from a next-hop neighbor node j for each packet sent to destination node n through node j. Xia et al. (2009) extend Q-routing with *backward exploration* (see Sect. 5.2) (Kumar and Miikkulainen 1997) in which Q-values are updated for the previous and next states simultaneously. This means that Q-values at node i and node j are updated for each packet sent from a source node $s \in N$ to a destination node s_t^i passing through node i and node j. Specifically, in addition to updating the Q-value of node i whenever it receives a feedback from node j, node j

Table 4 Q-routing model for the routing scheme at node i (Xia et al. 2009)

Reward	$r_t^i\left(s_t^i, a_t^i\right)$ represents the number of available common channels at node i and j

also updates its Q-value whenever it receives forwarded packets from node i. Note that, the packets are piggybacked with Q-values of node i into the forwarded packets to neighbor node j. Using this approach, node i has an updated Q-value to destination node s_t^i through neighbor node j; and node j has an updated Q-value to source node s through neighbor node i. Hence, nodes along a route have updated Q-values of the route in both directions. The dual RL-based Q-routing approach has been shown to minimize end-to-end delay.

6.1.2 Q-routing approach with dynamic discount factor

The traditional RL approach has a static discount factor γ, which indicates the preference on the future long-term rewards. This enhanced Q-routing approach with dynamic discount factor calculates the discount factor of each next-hop neighbor node, hence each of them may have different values of discount factor. This may provide a more accurate estimation on the Q-values of different next-hop neighbor nodes, which may have different characteristics and capabilities.

Santhi et al. (2011) propose a Q-routing approach with dynamic discount factor to reduce the frequency of triggering the route discovery process due to link breakage in MANETs. The proposed routing scheme aims to establish routes with high robustness, which are less likely to fail, and it has been shown to reduce end-to-end delay and increase packet delivery rate. This is achieved by selecting a reliable next-hop neighbor node based on three factors, which are considered in the estimation of discount factor γ, namely link stability, bandwidth efficiency, and node's residual energy. Note that, the link stability is dependent on the node mobility.

Table 5 shows the Q-routing model for the routing scheme at node i. Note that, the state and action representations are not shown, and they are similar to the general RL model in Table 3. The state s_t^i represents a destination node n. The action a_t^i represents the selection of a next-hop neighbor node j. The reward $r_t^i(a_t^i)$ indicates whether node i's packet has been successfully delivered to destination node n through node j. The Q-routing model is embedded in each mobile node.

Node i's Q-value, which indicates the possibility of a successful packet delivery to its destination node n through a next-hop neighbor node $a_t^i = j$, is updated at time $t+1$ as follows:

$$Q_{t+1}^i\left(s_t^i, j\right) \leftarrow (1-\alpha)\, Q_t^i\left(s_t^i, j\right) + \alpha\left[r_{t+1}^i\left(s_{t+1}^i, j\right) + \gamma_{i,j} \max_{k \in a_t^j} Q_t^j\left(s_t^j, k\right)\right] \quad (7)$$

where k is the next-hop neighbor node of $a_t^i = j$. The uniqueness of this approach is that the discount factor $0 \leq \gamma_{i,j} \leq 1$ is a variable, and so Q-values are discounted according to the three factors affecting the discount factors. These factors are estimated and piggybacked into *Hello* messages, and exchanged periodically among the neighbor nodes. Specifically, when node i receives a Hello message from its neighbor node j, it calculates its discount factor for node j, $\gamma_{i,j}$ as follows:

Table 5 Q-routing model for the routing scheme at node i (Santhi et al. 2011)

Reward	$r_t^i\left(a_t^i\right) = \{0, 1\}$. Node i receives a reward value of 1 if the packet it forwards has reached the destination node n through its next-hop neighbor node j; otherwise, it receives a reward value of 0

$$\gamma_{i,j} = \omega\sqrt{MF_j.BF_j.PF_j}, \quad j \in J \tag{8}$$

where ω is a pre-defined constant; MF_j represents the mobility factor, which indicates the estimated link lifetime between node $i$and node j; BF_j represents the available bandwidth at node j; and PF_j represents the residual energy of node j. The Q-routing model chooses the next-hop neighbor node $a_t^i = j$ with the maximum $Q_t^i(s_t^i, j)$ value. Based on Eq. (7), higher value of $\gamma_{i,j}$ generates a higher value of Q-value, which makes the action $a_t^i = j$ more likely to be selected.

6.1.3 Q-routing approach with learning rate adjustment

Bhorkar et al. (2012) propose a Q-routing approach with learning rate adjustment to minimize the average per-packet routing cost in wireless ad hoc networks (see Sect. 3.1.1). The learning rate is adjusted using a counter that keeps track of a node's observation (e.g. number of packets received from a neighbor node) from the operating environment.

Table 6 shows the Q-routing model to select a next-hop node for node i. The state s_t^i represents a set of node i's next-hop neighbor nodes that have successfully received a packet from node i. The action a_{rtx}^i represents retransmitting a packet by node i, a_f^i represents forwarding a packet to neighbor node $j \in s_t^i$, and a_d^i represents terminating/dropping of packet which can be either by the packet's destination node or an intermediate node. The reward represents a positive value only if the packet has reached its destination node. The Q-routing model is embedded in each mobile node.

Node i's Q-value, which indicates the appropriateness of transmitting a packet from node i to node $a_t^i = j$, is updated at time$t + 1$as follows:

$$Q_{t+1}^i\left(s_t^i, j\right) \leftarrow \left(1-\alpha_{c_t^i(s_t^i,j)}\right) Q_t^i\left(s_t^i, j\right) + \alpha_{c_t^i(s_t^i,j)}\left[r_{t+1}^i\left(s_{t+1}^i, j\right)+\max_{k\in a_t^j} Q_t^j\left(s_t^j, k\right)\right] \tag{9}$$

where $c_t^i(s_t^i, a_t^i)$ is a counter that keeps track of the number of times a set of nodes s_t^i have received a packet from node i using action a_t^i until time t; k is the next-hop neighbor node of node j, $k \in A(s_t^j)$. The learning rate $\alpha_{c_t^i(s_t^i,j)}$ is adjusted based on the counter $c_t^i(s_t^i, a_t^i)$, and so it is dependent on the exploration of state-action pairs. Hence, higher (lower) value of $\alpha_{c_t^i(s_t^i,j)}$ indicates a higher (lower) convergence rate at the expense of greater (lesser)

Table 6 Q-routing model for the routing scheme at node i (Bhorkar et al. 2012)

State	$s_t^i \subseteq S = \{1, 2, \ldots, J\}$, s_t^i represents a set of node i's neighbor nodes that have successfully received a packet from node i. J represents the total number of node i's neighbor nodes
Action	$a_t^i \in A(s_t^i) = \{a_{rtx}, a_f, a_d\}$, $A\left(s_t^i\right)$ represents the set of actions available to node i upon receiving a packet from its neighbor nodes $s_t^i \subseteq S$. Specifically, a_{rtx}^i represents retransmitting a packet; a_f^i represents forwarding a packet to a next-hop neighbor node $j \in s_t^i$; and a_d^i represents a packet arriving at destination (only if node i is the destination node) or dropping a packet
Reward	$r_t^i\left(s_t^i, a_t^i\right)$ represents a positive value for a packet arriving at the destination node s_t^i; a reward of 0 value is given if the packet fails to arrive at the destination node; and $-c_{a_t}$ represents the cost (e.g. required energy and hop count) of transmitting the packet

fluctuations of Q-values. Further research could be pursued to investigate the optimal value of $\alpha_{c_t^i(s_t^i,j)}$ and the effects of $c_t^i(s_t^i, j)$ on $\alpha_{c_t^i(s_t^i,j)}$.

6.1.4 Q-routing approach with Q-values equivalent to rewards

Baruah and Urgaonkar (2004) use rewards to reduce route cost and energy consumption in WSNs with mobile sink nodes. A similar Q-routing approach has also been applied in Forster and Murphy (2007), and the discussion in this section is based on Baruah and Urgaonkar (2004). Moles, which are located within a single hop from a sink node, use RL to learn a sink node's movement pattern, and send their received packets to the sink node. Each mole characterizes a sink node's movement pattern using a *goodness* value, G. Specifically, the goodness value is a probability that indicates the presence or absence of a sink node within a mole's one-hop vicinity. The purpose of the routing scheme is to enable a sensor node to select a next-hop node with the higher likelihood of reaching the mole, which subsequently sends packets to the sink node.

There are two main types of Q-routing models. Table 7 shows the Q-routing model for the routing scheme at node i using Multiplicative Increase Multiplicative Decrease (MIMD). Note that, the state and action representations are not shown, and they are similar to the general RL model in Table 3. Also note that, this routing scheme applies $Q_{t+1}^i(s_t^i, a_t^i) = r_t^i(s_t^i, a_t^i)$. In other words, action selection is based on reward $r_t^i(s_t^i, a_t^i)$. The state s_t^i represents a sink node n. The action a_t^i represents the selection of a next-hop neighbor node j. The rewards $r_t^i(s_t^i, a_t^i)$ represent the likelihood of reaching the sink node $s_t^i = n$ through node $a_t^i = j$. Note that, the reward is chosen based on the goodness value G, where $G_{TH,H}$ and $G_{TH,L}$ are thresholds for positive and negative reinforcements, respectively. Higher goodness value G indicates a higher probability of the presence of a sink node within a mole's one-hop vicinity. The Q-routing model is embedded in each sensor node. In MIMD, the reward value $r_t^i(s_t^i, a_t^i)$ is doubled when $G_{TH,H} < G \le 1$, and it is halved when $0 \le G < G_{TH,L}$.

Table 8 shows the Q-routing model for the routing scheme at node i using Distance Biased Multiplicative Update (DBMU). In DBMU, the reward value $r_t^i(s_t^i, a_t^i)$ is increased with a factor $\sqrt{(d+3)}$ when $G_{TH,H} < G \le 1$, and it is decreased with the similar factor $\sqrt{(d+3)}$ when $0 \le G < G_{TH,L}$, where d indicates the number of hop counts from the source sensor node. Hence, sensor nodes further away from the source sensor node receive higher rewards;

Table 7 Q-routing model for the routing scheme at node i using MIMD (Baruah and Urgaonkar 2004)

Reward	$r_{t+1}^i\left(s_t^i, a_t^i\right) = \begin{cases} 2 \times r_t^i\left(s_t^i, a_t^i\right) & if\ G_{TH,H} < G \le 1 \\ r_t^i\left(s_t^i, a_t^i\right) & if\ G_{TH,L} \le G \le G_{TH,H} \\ r_t^i\left(s_t^i, a_t^i\right) \big/ 2 & if\ 0 \le G < G_{TH,L} \end{cases}$

Table 8 Q-routing model for the DBMU-based routing scheme at node i (Baruah and Urgaonkar 2004)

Reward	$r_{t+1}^i\left(s_t^i, a_t^i\right) = \begin{cases} \sqrt{(d+3)} \times r_t^i\left(s_t^i, a_t^i\right) & if\ G_{TH,H} < G \le 1 \\ r_t^i\left(s_t^i, a_t^i\right) & if\ G_{TH,L} \le G \le G_{TH,H} \\ r_t^i\left(s_t^i, a_t^i\right) \big/ \sqrt{(d+3)} & if 0 \le G < G_{TH,L} \end{cases}$

Table 9 Q-routing model for the routing scheme at node i (Arroyo-Valles et al. 2007)

Action	$a_t^i \in A = \{1, 2, \ldots, J\}$, each action a_t^i represents the selection of a next-hop neighbor node j. J represents the number of node i's neighbor nodes
Reward	$r_t^i\left(a_t^i\right)$ represents the estimated number of retransmissions for each packet transmission from node i to next-hop neighbor node j

in other words, sensor nodes nearer to the mole (or closer to the sink node) are better indicators of the goodness value.

6.1.5 Q-routing approach with average Q-values

Arroyo-Valles et al. (2007) propose a geographic routing scheme that applies RL to increase packet delivery rate and network lifetime in WSNs (see Sect. 3.1.2). A node chooses its next-hop neighbor node towards a sink node by taking into account the expected number of retransmissions throughout a route.

Table 9 shows the Q-routing model for the routing scheme at node i. There is no state representation. The action a_t^i represents the selection of a next-hop neighbor node j, which is physically located nearer to the sink node. The reward $r_t^i(s_t^i, a_t^i)$ represents the estimated number of retransmissions for a single-hop transmission.

Node i's Q-value indicates the total number of retransmissions along a route to the sink node through a next-hop neighbor node $a_t^i = j$. The routing scheme of node i chooses a next-hop neighbor node $a_t^i = j$ that has the minimum $Q_t^i(j)$ value. In Arroyo-Valles et al. (2007), Eq. (5) is rewritten as follows:

$$Q_{t+1}^i(j) \leftarrow (1-\alpha)\, Q_t^i(j) + \alpha \left[r_{t+1}^i(j) + \bar{Q}_t^j(k)\right] \tag{10}$$

where k is the next-hop neighbor of node j, and the average Q-value $\bar{Q}_t^j(k)$ is as follows:

$$\bar{Q}_t^j(k) = \frac{\sum_{k \in K} P_k^j Q_t^j(k)}{\sum_{k \in K} P_k^j} \tag{11}$$

where K is a set of node j's neighbor nodes located nearer to the sink node, and P_k^j is the probability that node j will forward the packet to node k, and it is dependent on the local information, such as transmission and reception energies, message priority and neighbor node's profile (i.e. tendency to forward packets to the next-hop node).

6.1.6 Q-routing approach based on on-policy Monte Carlo

The traditional Q-routing approach updates Q-values upon receiving a reward for each action taken, and so the Q-values may fluctuate. In the On-Policy Monte Carlo (ONMC) approach, the Q-values are updated on an episode-by-episode basis in order to reduce fluctuation. An episode refers to a window of timeslots in which a number of actions can be taken.

Naruephiphat and Usaha (2008) propose an energy-efficient routing scheme using ONMC-based Q-routing in order to reduce energy consumption and to increase network lifetime in MANETs. The routing scheme aims to achieve a balanced selection of two types of routes: (1) routes that incur lower energy consumption; and (2) routes that are comprised of nodes with higher residual energy.

Table 10 ONMC-based Q-routing model for the routing scheme at node i (Naruephiphat and Usaha 2008)

State	$s_t^i = \{E_{j,t}^i, e_{k,t}^i\} \in S_1 \times S_2$. The substate $E_{j,t}^i \in S_1$ represents a quantized energy consumption level of an entire route $j \in M$, where M is a set of available routes. The substate $e_{k,t}^i \in S_2$ represents the least quantized residual energy level of a node within the entire route $k \in M$
Action	$a_t^i \in M$, each action a_t^i represents a route from the set of available routes M
Reward	$r_t^i\left(s_t^i, a_t^i\right) = \left(E_{a_t^i,t}^i\right)^{\omega_1}\left(e_{a_t^i,t}^i\right)^{-\omega_2}\left(e_{init}^i\right)^{\omega_3}$ represents the cost of a route a_t^i, where e_{init}^i is the initial residual energy of node i, while ω_1, ω_2 and ω_3 are weight factors

Table 10 shows the ONMC-based Q-routing model for the routing scheme at node i. The state s_t^i represents a two-tuple information, namely energy consumption level of an entire route and the least residual energy level of a node within an entire route. An action a_t^i represents the selection of a route. The reward $r_t^i(s_t^i, a_t^i)$ represents the estimation of the cost (or energy consumption) of a route.

At the end of each episode, the average cost of actions (or route selections) taken within the episode of a window duration of T_e is calculated as follows:

$$r_{avg,t}^i\left(s_t^i, a_t^i\right) = \frac{\sum_{T_e} r_t^i\left(s_t^i, a_t^i\right)}{T_e} \tag{12}$$

6.1.7 Q-routing approach based on model construction and update

Hu and Fei (2010) use a model-based Q-routing approach, which is based on the MDP model (see Sect. 2), to provide higher convergence rate (see Sect. 5.2) in order to reduce a route cost and energy consumption, and it is applied in underwater WSNs (see Sect. 3.1.2). The purpose of the routing scheme is to enable a sensor node to select a next-hop neighbor node with higher residual energy, which subsequently sends packets towards the sink node. Table 11 shows the Q-routing model for the routing scheme, and it shall be noted that the model is a representation for a particular packet. The state s_t^i represents the node in which a particular packet resides (or node i). The action a_t^i represents the selection of a next-hop neighbor node j. The reward $r_t^i(s_t^i, a_t^i)$, which is a dynamic value, represents various types of energies, including transmission energy and residual energy, incurred for forwarding the packet to node $a_t^i = j$. Taking into account the residual energy helps to avoid highly utilized routes (or hot spots) in order to achieve a balanced energy distribution among routes. The Q-routing model is embedded in each packet.

Node i's Q-function, which indicates the appropriateness of transmitting a packet from node i to node $a_t^i = j$, is updated at time $t+1$ as follows:

$$Q_{t+1}^i\left(s_t^i, j\right) = r\left(s_t^i, a_t^i\right) + \gamma\left[P_{s_t^i s_t^i}^{a_t^i} \max_{k \in a_t^i} Q_t^i\left(s_t^i, k\right) + P_{s_t^i s_t^j}^{a_t^i} \max_{k \in a_t^j} Q_t^j\left(s_t^j, k\right)\right] \tag{13}$$

where $P_{s_t^i s_t^i}^{a_t^i}$ is the transition probability of an unsuccessful transmission from s_t^i (or node i) after taking action a_t^i, while $P_{s_t^i s_t^j}^{a_t^i}$ is the transition probability of a successful transmission from s_t^i to s_t^j (or node j) after taking action a_t^i. Further explanation about the transition probability models to estimate the $P_{s_t^i s_t^i}^{a_t^i}$ and $P_{s_t^i s_t^j}^{a_t^i}$ are given in Sect. 5.2.

Table 11 Q-routing model for the routing scheme for a packet at node i (Hu and Fei 2010)

State	$s_t^i \in S = \{1, 2, \dots, N\}$, each state s_t^i represents a node i in which a particular packet resides. N represents the number of nodes in the entire network
Action	$a_t^i \in A = \{1, 2, \dots, J\}$, each action a_t^i a next-hop neighbor node j. J represents the number of node i's neighbor nodes
Reward	$r\left(s_t^i, a_t^i\right) = -g - \omega_1\left[c_r\left(s_t^i\right) + c_r\left(s_{t+1}^j\right)\right] + \omega_2\left[r_a\left(s_t^i\right) + r_a\left(s_{t+1}^j\right)\right]$ represents the cost incurred by forwarding a packet from node i to node j. g represents resource consumption, particularly transmission and reception energy, associated with each packet forwarding. $c_r\left(s_t^i\right)$ and $c_r\left(s_{t+1}^j\right)$ indicates the amount of residual energy at node i and node j; and lower $c_r\left(s_t^i\right) + c_r\left(s_{t+1}^j\right)$ indicates higher amount of residual energy. $r_a\left(s_t^i\right)$ and $r_a\left(s_{t+1}^j\right)$ indicates the comparison of the amount of residual energy at node i and node j compared to average amount of residual energy in a group of neighboring nodes. Higher $r_a\left(s_t^i\right) + r_a\left(s_{t+1}^j\right)$ indicates higher amount of residual energy. ω_1 and ω_2 are weight factors

Table 12 Q-routing model for the routing scheme for hth hop nodes (Lin and Schaar 2011)

State	$s_t^h \in S = \left(\mathbf{G}_t^h, \mathbf{L}_t^h\right)$. Sub-state $\mathbf{G}_t^h$ represents the current channel state, specifically, the propagation gain of links between hth and $h+1$th hop nodes. Sub-state $\mathbf{L}_t^h$ represents the current queue size of hth hop nodes; and each queue contains packets with the same remaining lifetime within a certain deadline
Action	$a_t^h \in A = \left(a_{n,t}^h, a_{p,t}^h\right)$. The joint action of hth hop nodes, a_t^h is comprised of $a_{n,t}^h$ and $a_{p,t}^h$. Action $a_{n,t}^h$ represents the selection of an $h+1$th hop node. Action $a_{p,t}^h$ represents a transmission power
Reward	$r\left(s_t, a_t\right) = r\left(s_t^H, a_t^H\right)$ represents the number of received packets at the destination node, which is located at the Hth hop, within a certain deadline

6.1.8 Q-routing approach with actor-critic method

Lin and Schaar (2011) reduce message exchange in the traditional Q-routing approach without considerably jeopardizing the convergence rate in a joint routing and power control scheme for delay-sensitive applications in wireless ad hoc networks (see Sect. 3.1.1). The purpose of the routing scheme is to enable a node to select a next-hop neighbor node that provides a higher successful transmission rate.

In Lin and Schaar (2011), nodes are organized in hops such that any node is located h hops from a base station. Note that, the base station is located at the Hth hop. Table 12 shows the Q-routing model for the routing scheme for hth hop nodes. The state s_t^h represents a two-tuple information, namely the channel state and queue size of hth hop nodes. Note that, s_t^h is locally available to all hth hop nodes. The investigation is limited to a single destination node in Lin and Schaar (2011), and so the destination node is not represented in the state. The action a_t^h represents two types of actions: $a_{n,t}^h$ represents the selection of an hth hop node, and $a_{p,t}^h$ represents a transmission power. The reward $r(s_t, a_t)$ represents the number of received packets. Note that, the reward is only a function of the state and action at Hth hop destination node; however, it represents successful receptions of packets throughout the entire network. Hence, $r\ (s_t, a_t)$ also indicates the successful transmission rate of all links in the network.

Instead of using the traditional Q-routing-based Q-function (5), Lin and Schaar (2011) apply an actor-critic approach (Sutton and Barto 1998) in which the value function $V_t^h(s_t^h)$ (or critic) and policy $\rho_t^h(s_t^h, a_t^h)$ (or actor) updates are separated. The critic is used to strengthen or weaken the tendency of choosing a certain action, while the actor indicates the tendency of choosing an action-state pair (Sutton and Barto 1998). Denote the difference in the value function between hth hop (or the current hop) and $h+1$th hop (or the next hop) by $\delta_t^h(s_t^h) = V_t^{h+1}(s_t^{h+1}) - V_t^h(s_t^h)$, the value function at hth hop nodes is updated using $V_{t+1}^h(s_{t+1}^h) = (1-\alpha)V_t^h(s_t^h) + \alpha[\delta_t^h(s_t^h)]$, and so it is updated using value functions received from $h+1$th hop nodes. Denote $\delta_t^H(s_t^H) = r(s_t, a_t) + \gamma V_t(s_t) - V_t^H(s_t^H)$, the value function at Hth hop nodes (or the destination node) is $V_{t+1}^H(s_{t+1}^H) = (1-\alpha)V_t^H(s_t^H) + \alpha[\delta_t^H(s_t^H)]$, and so it is updated using value function $V_t(s_t)$, which is received from 1th hop nodes (or all the source nodes), and reward $r(s_t, a_t)$. Therefore, the reward $r(s_t, a_t)$ is distributed throughout the network in the form of value function $V_t^h(s_t^h)$. The routing scheme of an hth hop node chooses a next-hop neighbor node and transmission power a_t^h that provide the maximum $\rho_t^h(s_t^h, a_t^h)$ value, which is updated by $\rho_{t+1}^h(s_{t+1}^h, a_{t+1}^h) = \rho_t^h(s_t^h, a_t^h) + \beta\delta_t^h(s_t^h)$, where β is the learning rate. This routing scheme has been shown to increase the number of received packets at the destination node within a certain deadline (or goodput).

Lin and Schaar (2011) also propose two approaches to reduce message exchange. Firstly, only a subset of nodes within an hth hop is involved in estimating the value function $V_t^h\left(s_t^h\right)$. Secondly, message exchanges are only carried out after a certain number of time slots.

6.2 Multi-agent reinforcement learning (MARL) model

Multi-Agent Reinforcement Learning (MARL) model decomposes a system-wide optimization problem into sets of local optimization problems, which are solved through collaboration without using global information. The collaboration may take the form of exchanging local information including knowledge (i.e Q-value), observations (or states) and decisions (or actions). For instance, MARL enables a node to collaborate with its neighbor nodes, and subsequently make local decisions independently in order to achieve network performance enhancement.

6.2.1 MARL approach with reward exchange

Elwhishi et al. (2010) propose a MARL-based routing scheme for delay tolerant networks (see Sect. 3.1.4), and it has been shown to increase packet delivery rate, as well as to decrease transmission delay. Routing schemes for delay tolerant networks are characterized by the lack of end-to-end aspect, and each node explores network connectivity through finding a new link to a next-hop neighbor node when a new packet arrives, which must be kept in the buffer while a link is formed. The purpose of this routing scheme is to select a reliable next-hop neighbor node, and it takes into account three main factors. Specifically, two factors that are relevant to the channel availability (node mobility and congestion level) and a single factor that is relevant to the buffer utilization (remaining space in the buffer).

Table 13 shows the MARL model for node i to select a reliable next-hop neighbor node. The state s_t^i includes three events $\{B^i, U^{i,j}, D^{j,i}\}$. The action a_t^i represents a transmission from node i to a next-hop neighbor node j. The reward represents the amount of time in which a mobile node i and a node j are neighbor nodes. The MARL model is embedded in each node.

Table 13 MARL model for the routing scheme at node i (Elwhishi et al. 2010)

State	$s_t^i \in S = \{B^i, U^{i,j}, D^{j,i}\}$, state B^i represents an event of nonempty buffer; state $U^{i,j}$ represents a successful transmission from node i to a next-hop neighbor node j; and state $D^{j,i}$ represents a successful transmission from a next-hop neighbor node j to node i
Action	$a_t^i \in A = \{1, 2, \ldots, J\}$, each action a_t^i represents the selection of a next-hop neighbor node j. J represents the number of node i's neighbor nodes
Reward	$r_t^i \left(a_t^i\right)$ represents the amount of time in which node i and node $a_t^i = j$ are neighbor nodes

Node i's Q-value, which indicates the appropriateness of transmitting a packet from node i to a node $a_t^i = j$, is updated at time instant $t+1$ as follows:

$$Q_{t+1}^i \left(s_t^i, a_t^i\right) \leftarrow r_t^i \left(a_t^i\right) + T_{free} - \frac{P_{unsuccess}^{i,j}}{P_{success}^{i,j}} T_{busy} \tag{14}$$

where T_{free} and T_{busy} are the amount of time a channel is free (or successful transmission) and busy (or unsuccessful transmission) during a time window interval T_{window}, respectively; $P_{success}^{i,j} = T_{free}/T_{window}$ and $P_{unsuccess}^{i,j} = T_{busy}/T_{window}$ are the probability of successful and unsuccessful transmissions from node i to a next-hop neighbor node j, respectively. The probability $P_{unsuccess}^{i,j}$ covers two types of events that may cause unsuccessful transmissions: channel congestion and buffer overflow at the next-hop neighbor node j. The MARL chooses a next-hop neighbor node $a_j^{i,j}$ that has the maximum $Q_t^i(s_t^i, a_t^i)$ value.

Collaboration among the agents involve the exchanges of reward $r_t^i(a_t^i)$. The next-hop node is chosen from a selected set of nodes towards the destination node, which may be multiple hops away. For instance, agent i calculates $r_t^i(a_t^{i,k}) = r_t^i(a_t^{i,j}) + r_t^j(a_t^{j,k})$, which is the amount of time in which node i can communicate with two-hop neighbor node k though neighbor node j. The $r_t^i(a_t^{i,k})$ is exchanged with one-hop neighbor nodes.

6.2.2 MARL approach with Q-value exchange

Liang et al. (2008) propose a MARL-based approach called Distributed Value Function-Distributed Reinforcement Learning (DVF-DRL) for routing in WSNs (see Sect. 3.1.2); and it has been shown to increase packet delivery rate, as well as to decrease end-to-end delay. The purpose of this routing scheme is to select a next-hop neighbor node that provides lower end-to-end delay; and it takes into account the Q-values, and hence the performance, of its neighboring nodes.

Table 14 shows the MARL model for the routing scheme at node i to select a next-hop neighbor node. The state s_t^i represents a two-tuple information, namely S_n represents a set of node i's neighbor nodes, and S_p represents a set of packets, which are encapsulated with QoS requirements, to be sent or forwarded. The action a_t^i represents either A_f, which is a packet transmission from node i to a next-hop neighbor node j; or A_d, which is a packet drop if the end-to-end delay of a packet fails to fulfill the QoS requirement encapsulated in the packet itself. Denote the average one-hop link delay (i.e. queuing, transmission, processing and channel contention delays) between nodes i and j by $T_{P_{ij}}$, and the average one-hop transmission delay by $T_{P_{avr}}$. When a data packet transmission is successful, the reward value $r_t^i(s_t^i, a_t^i) = T_{P_{avr}}/T_{P_{ij}}$; and when a data packet transmission is unsuccessful, the reward value $r_t^i(s_t^i, a_t^i) = -1$. The MARL model is embedded in each sensor node.

Table 14 MARL model for the routing scheme at node i (Liang et al. 2008)

State	$s_t^i \in S = \{S_n, S_p\}$, S_n represents a set of node i's neighbor nodes; while S_p represents a set of packets, which are encapsulated with QoS requirements, to be sent or forwarded
Action	$a_t^i \in A = \{A_f, A_d\}$, $A_f = \{1, 2, \ldots, J\}$ represents the selection of a next-hop neighbor node j. J represents the number of node i's neighbor nodes. A_d represents a packet drop
Reward	$r_t^i\left(s_t^i, a_t^i\right) = \begin{cases} TP_{avr}/TP_{ij} & if \text{ ACK} \, received \\ -1 & if \text{ NACK} \, received \\ -1.5 & if \; neither \text{ ACK} \; nor \text{ NACK} \; is \; received \end{cases}$

Node i's Q-value, which indicates the appropriateness of transmitting a packet from node i to node $a_t^i = j$, is updated at time instant $t+1$ as follows:

$$Q_{t+1}^i\left(s_t^i, j\right) \leftarrow (1-\alpha)\, Q_t^i\left(s_t^i, j\right) + \alpha\left[r_{t+1}^i\left(s_{t+1}^i\right) + \gamma\omega(i, j) \max_{k \in a_t^j} Q_t^j\left(s_t^j, k\right) + \gamma \sum_{j' \in a_t^i, j' \neq j} \omega\left(i, j'\right) \max_{k \in a_t^{j'}} Q_t^{j'}\left(s_t^{j'}, k\right)\right] \tag{15}$$

where $\omega(i, j)$ indicates node i's weight on Q-values received from node j; $j' \in a_t^i$ and $j' \neq j$ indicate node i's neighbor nodes excluding its chosen next-hop neighbor node j. Generally speaking, $Q_{t+1}^i(s_t^i, j)$ in Eq. (15) allows node i to keep track of its Q-value, the immediate reward, the maximum Q-value of its chosen next-hop neighbor node j, and the maximum Q-values of all its neighbor nodes (except its chosen next-hop node j).

6.2.3 MARL approach with decay function

Dowling et al. (2005) propose a MARL-based routing scheme for MANETs, and it has been shown to increase network throughput, packet delivery rate, and to reduce the number of transmissions required for each packet. Routing in ad hoc networks can be challenging due to two main reasons. Firstly, node mobility causes frequent changes in network topology. Secondly, the imperfect underlying radio links causes congestions and deteriorates network performance. Using the MARL model, the purpose of this scheme is to enable the routing agent to adapt to the varying network conditions, and choose stable links in order to meet requirements on network performance. By sharing the current optimal policies among the neighboring nodes, the convergence rate to the optimal policies is expected to increase. A positive feedback contributes towards the convergence of the optimal policies, whereas a negative feedback results in the policy update of an agent due to congestions of some links, or deteriorating of a routing decision through some neighbor nodes.

Table 15 shows the MARL model to select a reliable next-hop node for node i. The state s_t^i includes three kinds of events $\{B^i, U^{i,j}, D^{j,i}\}$. The action a_t^i includes three kinds of actions $\{a_f, a_D, a_B\}$. The reward represents link stability between node i and node j, which is calculated based on the ratio between successful and unsuccessful transmissions. The MARL model is embedded in each mobile node.

Collaboration in MARL enables agents to exchange route cost advertisements. Note that, given state B^i, a node i updates and calculates its route cost $V_t^i\left(B^i\right) = \max_{a \in a_f^{i,j}} Q(B^i, a)$, which is advertised and sent to its neighbor nodes. The route cost (or negative reward) is maximized and it is calculated as follows:

Table 15 MARL model for the routing scheme at node i (Dowling et al. 2005)

State	$s^i \in S = \{B^i, U^{i,j}, D^{j,i}\}$, state B^i indicates that there is a packet waiting in the buffer to be sent; state $U^{i,j}$ represents an event of successful transmission from node i to neighbor node j; and state $D^{j,i}$ represents an event of successful transmission from neighbor node j to node i
Action	$a_t^i \in A = \left\{a_f^{i,j}, a_D^{i,j}, a_B^{i,j}\right\}$, action $a_f^{i,j}$ is to forward (or delegate) a packet from node i to neighbor node j; action a_D^i is to deliver a packet locally to the upper layer at node i; and action a_B^i is to broadcast a packet by node i to discover new neighbor nodes. Note that, $a_f^{i,j} = \{1, 2, \ldots, J\}$, and J represents the number of node i's neighbor nodes
Reward	There are two types of rewards. Firstly, the $r_s^i\left(a_f^{i,j}\right) = -1$ represents a reward for successful transmission, and $r_f^i\left(a_f^{i,j}\right) = -7$ represents a link cost for unsuccessful transmission. Secondly, the $r_{tc}^i(s_t^i, a_t^i)$ represents the cost of executing actions locally at node i, specifically cost of -7 for action a_B^i, 0 for action $a_f^{i,j}$ and 0 for action a_D^i

$$V_{t+1}^i\left(B^i\right) = r_{tc}^i\left(s_{t+1}^i, a_{t+1}^i\right) + \max_{a \in a_f^{i,j}}\left[\text{Decay}_{t+1}^i\left(V_t^j\left(U^{i,j}\right)\right) + r_s^i(a) + \frac{P_{unsuccess}^{i,j}}{P_{success}^{i,j}} r_f^i(a)\right] \quad (16)$$

where $\text{Decay}_{t+1}^i(V_t^j(U^{i,j}))$ is a decay function at node i that deteriorates the advertised routing cost $V_t^j(U^{i,j})$ by neighbor node j over time; $r_s^i(a_f^{i,j})$ and $r_f^i(a_f^{i,j})$ are link reward and cost for successful and unsuccessful transmissions to neighbor node j, respectively; $P_{success}^{i,j} = P^i(U^{i,j}|B^i, a_f^{i,j})$ and $P_{unsuccess}^{i,j} = 1 - P^i(U^{i,j}|B^i, a_f^{i,j})$ are the probability of successful and unsuccessful transmissions from node i to neighbor node j, respectively, which are calculated using a statistical model that samples the transmission and reception events of a link.

Finally, as it can be seen, using the feedback model, node i updates its routing behavior (i.e policy) whenever one of three events occurs. Firstly, changes in the link quality (i.e. $P_{success}^{i,j}$ and $P_{unsuccess}^{i,j}$). Secondly, changes in the advertised route cost ($V_t^j(U^{i,j})$) by a neighbor node j. Thirdly, if the routing cost $V_t^j(U^{i,j})$ is not updated within a time window (i.e. in the absence of new advertisements from neighbor node j), then node i deteriorates $V_t^j(U^{i,j})$ so that possible future performance degradation is taken into account as follows:

$$\text{Decay}_{t+1}^i\left(V_t^j\left(U^{i,j}\right)\right) = V_t^j(U^{i,j}).p^{t_U} \quad (17)$$

where t_U is the time since the last update; and p is the deteriorating factor.

The MARL-based routing scheme makes use of feedbacks while learning the optimal routing policy. The use of feedbacks increases the convergence rate compared to the traditional Q-routing approach in which the policy is only updated whenever an agent executes its action.

6.3 Partial observable Markov decision process (POMDP) model

Nurmi (2007) proposes a POMDP-based routing scheme that estimates its state comprised of its neighbor node's local parameters (i.e. forwarding selfishness and energy consumption)

Table 16 POMDP model for the routing scheme at node i (Nurmi 2007)

State	$s_t^i = \left(\theta_t^{i,j}\left(H_t^{i,j}, \theta_{t-1}^{i,j}\right), \ldots, \theta_t^{i,J}\left(H_t^{i,J}, \theta_{t-1}^{i,J^n}\right)\right)$, where $\theta_t^{i,j}$ represents the estimated probability distribution of neighbor node j's forwarding selfishness and energy consumption parameters. J^n represents set of node$i$'s one-hop neighbor nodes that have a valid route to a destination node$n$. Each state $\theta_t^{i,j}\left(H_t^{i,j}, \theta_{t-1}^{i,j}\right)$ is estimated based on the previously observed historical actions H_t^j and previously estimated probability distribution value θ_{t-1}^j
Action	$a_t^i \in A = \{1, 2, \ldots, J^n\}$, each action a_t^irepresents the number of packets generated by node i to each of the neighbor node $j \in J^n$ at time instant t
Reward	$r_t^i\left(s_t^i, a_t^i\right)$ represents the gain or cost (e.g. a constant value) when node i's packets are forwarded or dropped by neighbor node j, respectively

in ad hoc networks, and it has been shown to reduce energy consumption. Note that, to the best of our knowledge, this is the only routing scheme that applies the POMDP model.

Table 16 shows the POMDP model for the routing scheme at node i. The state $\theta_t^{i,j}(H_t^{i,j}, \theta_{t-1}^{i,j})$ represents node i's estimates on neighbor node j's forwarding selfishness and energy consumption parameters, which indicate node j's capability to forward packets to destination. Note that, neighbor node $j \in J^n$, where J^n is the set of one-hop neighbor nodes that have a valid route to a destination node n. An action a_t^i represents the number of packets generated by node i to a neighbor node j. The reward represents the gain or cost when node i's packets are forwarded or dropped by next-hop neighbor node j, respectively. The POMDP model is embedded in each node.

In order to estimate the parameter of next-hop neighbor nodes, an algorithm that mainly consists of two parts is proposed. Firstly, it uses a stochastic approximation algorithm, namely Win-or-Learn Fast Policy Hill Climbing (WoLF-PHC), to estimate the local parameters $\theta_t^{i,j}$ of its next-hop neighbor node $j \in J^n$. WoLF-PHC uses a variable learning rate to adapt to the level of dynamicity in the operating environment (see Sect. 5.2). Secondly, a function approximation technique is proposed to learn a control function, which estimates the forwarding probability through neighbor node $j \in J^n$. The forwarding probability function f_{t+1}^i is optimized using a stochastic gradient descent algorithm (Snyman 2005) as follows:

$$f_{t+1}^i = f_t^i + \eta \frac{\partial E}{\partial \theta_t^j} \tag{18}$$

where η is the step size of the gradient; E is a loss function that indicates the error between the observed and the estimated probability.

7 Implementation of routing using reinforcement learning in wireless platform

Forster et al. (2008) implement a RL-based routing protocol in mobile WSNs on a ScatterWeb platform comprised of MSB530 sensor nodes and ChipCon 1020 transceivers. The aim of this platform implementation is to evaluate the feasibility of implementing a RL-based routing protocol (Forster and Murphy 2007) in real hardware and to benchmark its performance against a traditional routing protocol, namely Directed Diffusion (Intanagonwiwat et al. 2003). The RL-based routing protocol aims to select a next-hop neighbor node with the lowest route cost (i.e. lower number of hops and packet retransmissions along a route) leading to multiple sink nodes (Forster and Murphy 2007). Additionally, a random backoff delay was introduced to reduce channel contention.

Compared to the traditional Directed Diffusion (Intanagonwiwat et al. 2003) routing protocol, the findings of the platform implementation is that, the RL-based routing protocol:

- Provides lower routing cost.
- Provides higher packet delivery rate or lower packet loss rate attributed to the optimal routes.
- Requires higher memory requirement due to the complex data structures required to store the Q-values.
- Causes higher route discovery delay.

8 Performance enhancements

Table 17 shows the performance enhancements brought about by the application of RL in various routing schemes. The RL approach has been shown to achieve the following performance enhancements.

a. *Lower end-to-end delay*.
b. *Higher throughput*.
c. *Higher packet delivery rate* or *lower packet loss rate*.
d. *Lower routing overhead*. Lower routing overhead may indicate more stable routes (or higher route robustness), and so it may indicate a lower number of packet retransmissions (Dowling et al. 2005). Usaha (2004) reduces the number of route discovery messages by incorporating a cache, which stores routing information, at each mobile node. This reduces the need to invoke the route discovery process for each connection request.
e. *Longer network lifetime*. Longer network lifetime indicates lower energy consumption. For instance, Nurmi (2007) takes account of the residual energy of each node in routing in order to increase network lifetime.
f. *Higher reward value*. Bhorkar et al. (2012) achieves higher reward value, which indicates lower average cost incurred by routing.

9 Open issues

This section discusses open issues that can be pursued in this area. Additionally, further research could be pursued to investigate the new features discussed in Sect. 5.

9.1 Suitability and comparisons of action selection approaches

Action selection techniques, such as ε *-greedy* and *softmax* (see Sect. 2.2) have been widely applied in the literature. The ε-greedy approach performs exploration with small probability ε (e.g. $\varepsilon = 0.1$), and exploitation with probability $1 - \varepsilon$. On the other hand, the softmax approach ranks the exploration actions so that it does not explore actions that are far away from the optimal action. Nevertheless, each action selection technique has its merits and demerits in regards to the respective applications. Hence, further research could be pursued to investigate the suitability and network performance brought about by each of the techniques for routing in distributed wireless networks.

Table 17 Performance enhancements achieved by the RL-based routing schemes

Performance enhancements compared to existing approaches	**Q-routing model**									**MARL model**			**POMDP model**
	Xia et al. (2009)	Santhi et al. (2011)	Bhorkar et al. (2012)	Forster and Murphy (2007)	Naruephiphat and Usaha (2008)	Baruah and Urgaonkar (2004)	Hu and Fei (2010)	Lin and Schaar (2011)	Arroyo-Valles et al. (2007)	Dowling et al. (2005)	Elwhishi et al. (2010)	Liang et al. (2008)	Nurmi (2007)
Lower end-to-end delay	×	×					×				×	×	
Higher throughput								×		×			
Higher packet delivery rate		×		×			×		×	×	×	×	
Lower routing overhead				×				×		×			
Longer network lifetime				×	×	×	×		×				×
Higher reward value			×										

9.2 Degradation of optimal route due to exploitation

Exploitation may deliver all data packets along the same optimal (or near-optimal) route, which may cause congestion, resulting in higher energy consumption, routing overhead, as well as the degradation of network performance. Various traditional approaches may be used to ameliorate this issue as described below:

- Increasing learning rate. This increases the responsiveness of nodes to the dynamic conditions of the operating environment. When an optimal route becomes congested, nodes may change their respective routes. However, a new route may be suboptimal compared to the existing one. For instance, some intermediate nodes may have lower residual energy. Hence, achieving a better balance in energy consumption among various routes, including the optimal route, is necessary.
- Increasing exploration rate. This reduces the traffic load on the optimal route. Nevertheless, this may affect the network performance because the traditional exploration techniques may choose the non-optimal routes, which may not meet the requirements on network performance.
- Using different QoS classes. This enables nodes to achieve a balance among the available routes based on the QoS requirements. For instance, high priority traffics may use an optimal route (i.e exploitation), while lower priority traffics may use a suboptimal route (i.e exploration). For instance, Fu et al. (2005) adopt a genetic-based approach for exploration in route discovery, in which routes are discovered based on a predefined set of QoS goals (e.g. the throughput and end-to-end delay requirements). This approach has been shown to improve the network performance. Further research could be pursued to investigate the effectiveness and amount of overhead of this technique, the coordination among nodes for QoS provisioning along a route, as well as to ameliorate the selfishness of intermediate nodes.

9.3 Multi-agent RL approaches

Most routing schemes in the literature have been applying the traditional single-agent RL approach. These schemes have been shown to improve network performance. Nevertheless, the performance may be further enhanced using the multi-agent RL model (see Sect. 4.2) (Dowling et al. 2005; Elwhishi et al. 2010; Liang et al. 2008). The multi-agent RL model aims to coordinate agents in order to achieve the optimal network-wide performance, and its application in the literature has been limited. Nevertheless, the multi-agent RL approach requires additional overhead and complexity (Di Felice et al. 2010; Dowling et al. 2005), and so further research could be pursued to address these issues. In regards to the challenges of each type of distributed wireless network (see Table 1), some of the expected advantages brought about by coordination through the application of multi-agent RL approach are as follows:

- Mobile Ad Hoc Networks (see Sect. 3.1.1). The application of multi-agent RL approach may improve the coordination of nodes moving in groups, which may be an important factor in nodes' routing decisions. Consequently, the routing decision of nodes in each group can be further enhanced and coordinated in real-time.
- Wireless Sensor Networks (see Sect. 3.1.2). The application of the multi-agent RL approach may prolong network lifetime of a WSN. For instance, the neighboring sensor nodes may cooperate with each other to choose a sensor node with higher residual energy as clusterhead so that it is capable of collecting and forwarding data to the sink node.

- Cognitive Radio Networks (see Sect. 3.1.3). The application of multi-agent RL approach may be helpful to solve many issues that require coordination among SUs, as well as PUs. For instance, the neighboring SUs can share their respective estimations of PU utilization level, which is location-dependent. Subsequently, they can coordinate their respective routing decisions in order to reduce the interference to PUs.
- Delay Tolerant Networks (see Sect. 3.1.4). The application of multi-agent RL has been shown to improve the network-wide performance. For instance, Elwhishi et al. (2010) propose a MARL-based routing scheme that enables neighboring nodes to collaborate among themselves so that they can respectively select a reliable next-hop based on three factors, namely node mobility, the congestion level, and the buffer utilization level.

9.4 Exploration with stability enhancement

Exploration helps a routing scheme to converge to an optimal route selection policy. However, route discovery using non-optimal actions may increase network instability. Consequently, network performance (e.g. throughput and end-to-end delay) may fluctuate. While various approaches have been proposed to provide a balanced tradeoff between exploitation and exploration, further research could be pursued to investigate the possibility of achieving route exploration, while minimizing network instability. For instance, using rules, exploration may be triggered based on the required network performance level. Exploration of non-optimal routes may only be granted if network performance is unsatisfactory.

9.5 Application of events to routing

Another feature that can be incorporated into the traditional RL-based routing scheme is called *event*. Traditionally, *state* monitors the conditions of the operating environment at all times. In contrast, *event*, such as node failure and handoff, is detected occasionally whenever it occurs (Yau et al. 2012). In a highly dynamic operating environment, Q-values may change rapidly and fluctuate. The event may capture and detect the fluctuation of Q-values so that appropriate actions can be taken in order to improve the convergence rate and network performance.

Rules (see Sect. 5.5) and events may be also applied together. For instance, rules and events can improve the estimation of Q-values, and avoid invalid routes. For example, when mobile nodes move out of range, their respective Q-values may become invalid, and so making routing decisions based on these Q-values may degrade the network performance. An event may be applied to detect whether a node is out-of-range due to node movement; subsequently rules may be applied to set Q-values of neighbor nodes to infinity (Yau et al. 2012; Chang et al. 2004).

9.6 Lack of the implementation of RL-based routing schemes on wireless platform

To the best of our knowledge, there is only a single implementation of RL-based routing scheme in wireless platform by Forster et al. (2008) (see Sect. 7), and most of the existing routing schemes have been evaluated using simulation. Real implementation on wireless platform is important to validate the correctness and feasibility of RL-based schemes. Further research can be pursued to investigate the implementation and challenges of the RL-based routing schemes on wireless platforms.

10 Conclusions

Reinforcement Learning (RL) has been applied to various routing schemes for distributed wireless networks, including wireless ad hoc networks, wireless sensor networks, cognitive radio networks and delay tolerant networks, and it has been shown to improve network performance, such as higher throughput and lower end-to-end delay. RL enables a wireless node to observe its local operating environment, and subsequently learn to make global routing decisions efficiently. The advantages brought about by RL to routing are foreseen to draw significant research interests in the near future. This article has provided an extensive review on the existing RL-based routing schemes in distributed wireless networks. Firstly, this article presents Q-learning, which is a popular RL approach. Secondly, it identifies the challenges and advantages brought about by RL in various types of distributed wireless networks. A general RL model for routing is also presented. Thirdly, it presents three types of RL models for routing, namely Q-routing, multi-agent RL, and partially observable Markov decision process. Fourthly, it provides an extensive review on how various routing schemes have been formulated and modeled using RL in order to improve network performance. Fifthly, new features aiming to enhance the traditional RL approach for routing are presented. Sixthly, it presents an implementation of RL-based routing scheme on wireless platform. Seventhly, it presents performance enhancements achieved by the RL-based routing schemes. Lastly, it discusses the open issues associated with the application of RL in the routing schemes. Certainly, there is a substantial room for future work in the application of RL to routing.

References

Akyildiz IF, Su W, Sankarasubramaniam Y, Cayirci E (2002) Wireless sensor networks: a survey. Comput Netw 38(4):393–422

Akyildiz IF, Lee WY, Chowdhury KR (2009) Cognitive radio ad hoc networks. Ad Hoc Netw 7(5):810–836

Al-Rawi HAA, Yau K-LA (2012) Routing in distributed cognitive radio networks: a survey. Wirel Pers Commun Int J. doi:10.1007/s11277-012-0674-7

Albus JS (1975) A new approach to manipulator control: the cerebellar model articulation controller. J Dyn Syst Meas Control 97:220–227

Arroyo-Valles R, Alaiz-Rodriquez R, Guerrero-Curieses A, Cid-Sueiro J (2007) Q-probabilistic routing in wireless sensor networks. In: Proceedings of ISSNIP 3rd international conference intelligent sensors, sensor network and information processing, pp. 1–6

Baruah P, Urgaonkar R (2004) Learning-enforced time domain routing to mobile sinks in wireless sensor fields. In: Proceedings of LCN 29th annals IEEE international conference local computer networks, pp. 525–532

Bhorkar AA, Naghshvar M, Javidi T, Rao BD (2012) Adaptive opportunistic routing for wireless ad hoc networks. IEEE ACM Trans Netw 20(1):243–256

Boyan J, Littman ML (1994) Packet routing in dynamically changing networks: a reinforcement learning approach. In: Proceedings of NIPS Adv neural information processing systems, pp 671–678

Boukerche A (2009) Algorithms and protocols for wireless, mobile and ad hoc networks. Wiley, New Jersey

Burleigh S, Hooke A, Torgerson L, Fall K, Cerf V, Durst B, Scott K, Weiss H (2003) Delay-tolerant networking: an approach to interplanetary internet. IEEE Commun Mag 41(6):128–136

Bowling M, Veloso M (2002) Multiagent learning using a variable learning rate. Artif Intell 136(2):215–250

Chang Y-H, Ho T, Kaelbling LP (2004) Mobilized ad-hoc networks: a reinforcement learning approach. In: Proceedings of ICAC international conference autonomic computer, pp 240–247

Chetret D, Tham C-K, Wong LWC (2004) Reinforcement learning and CMAC-based adaptive routing for MANETs. In: Proceedings of ICON 12th IEEE international conference networks, pp. 540–544

Clausen T, Jacquet P (2003) Optimized link state routing protocol (OLSR). IETF RFC 3626

Dearden R, Friedman N, Andre D (1999) Model based Bayesian exploration. In: Proceedings of UAI 15th conference uncertainty, artificial intelligence, pp 150–159

Di Felice M, Chowdhury KR, Wu C, Bononi L, Meleis W (2010) Learning-based spectrum selection in cognitive radio ad hoc networks. In: Proceedings of WWIC 8th international conference wired wireless internet communications, pp 133–145

Dong S, Agrawal P, Sivalingam K (2007) Reinforcement learning based geographic routing protocol for UWB wireless sensor network. In: Proceedings of GLOBECOM IEEE global telecommunications conference, pp 652–656

Dowling J, Curran E, Cunningham R, Cahill V (2005) Using feedback in collaborative reinforcement learning to adaptively optimize MANET routing. IEEE Trans Syst Man Cybern Part A Syst Hum 35(3):360–372

Elwhishi A, Ho P-H, Naik K, Shihada B (2010) ARBR: Adaptive reinforcement-based routing for DTN. In: Proceedings of WIMOB IEEE 6th international conference wireless and mobile computes, networks and communications, pp. 376–385

Forster A (2007) Machine learning techniques applied to wireless ad-hoc networks: guide and survey. In: Proceedings of ISSNIP 3rd international conference intelligent sensors, sensor Networks and information, pp. 365–370

Forster A, Murphy AL (2007) FROMS: Feedback routing for optimizing multiple sinks in WSN with reinforcement learning. In: Proceedings of ISSNIP 3rd international conference intelligent sensors, sensor Networks and, informations, pp. 371–376

Forster A, Murphy AL, Schiller J, Terfloth K (2008) An efficient implementation of reinforcement learning based routing on real WSN hardware. In: Proceedings of WIMOB IEEE international conference wireless and mobile computers, networks and communcations, pp 247–252

Fu P, Li J, Zhang D (2005) Heuristic and distributed QoS route discovery for mobile ad hoc networks. In: Proceedings of the CIT 5th international conference on computer and information technology, pp. 512–516

Gen M, Cheng R (1999) Genetic algorithms and engineering optimization. Wiley, NY

Hao S, Wang T (2006) Sensor networks routing via Bayesian exploration. In: Proceedings of LCN 31th annals of IEEE international conference local computing Networks, pp. 954–955

Hu T, Fei Y (2010) QELAR: a machine-learning-based adaptive routing protocol for energy-efficient and lifetime-extended underwater sensor networks. IEEE Trans Mobile Comput 9(6):796–809

Intanagonwiwat C, Govindan R, Estrin D, Heidemann J, Silva F (2003) Directed diffusion for wireless sensor networking. IEEE ACM Trans Netw 11(1):2–16

Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of IEEE international conference neural networks. pp 1942–1948

Kumar S, Miikkulainen R (1997) Dual reinforcement Q-routing: an on-line adaptive routing algorithm. In: Proceedings of ANNIE artificial neural networks in engineering conference. pp 231–238

Liang X, Balasingham I, Byun S-S (2008) A multi-agent reinforcement learning based routing protocol for wireless sensor networks. In: Proceedings of ISWCS IEEE international symposium Wireless communications systems. pp 552–557

Lin Z, Schaar Mvd (2011) Autonomic and distributed joint routing and power control for delay-sensitive applications in multi-hop wireless networks. IEEE Tran Wirel Commun 10(1):102–113

Naruephiphat W, Usaha W (2008) Balancing tradeoffs for energy-efficient routing MANETs based on reinforcement learning. In: Proceedings of VTC spring IEEE vehicular techmology conference. pp 2361–2365

Nurmi P (2007) Reinforcement learning for routing in ad hoc networks. In: Proceedings of WiOpt 5th international symposium modeling and optimization in mobile, ad hoc and wireless network and workshops, pp 1–8

Ouzecki D, Jevtic D (2010) Reinforcement learning as adaptive network routing of mobile agents. In: Proceedings of MIPRO 33rd international convention, pp 479–484

Perkins CE, Royer EM (1999) Ad-hoc on-demand distance vector routing. In: Proceedings of WMCSA mobile computers systems and applications, pp 90–100

Rojas R (1996) Neural networks: a systematic introduction. Springer, NY

Santhi G, Nachiappan A, Ibrahime MZ, Raghunadhane R, Favas MK (2011) Q-learning based adaptive QoS routing protocol for MANETs. In: Proceedings of ICRTIT international conference recent trends in information technology, pp 1233–1238

Sutton RS, Barto AG (1998) Reinforcement learning: an introduction. MIT Press, Cambridge

Snyman A (2005) Practical mathematical optimization: an introduction to basic optimization theory and classical and new gradient-based algorithms. Springer, NY

Toh CK (2001) Ad hoc mobile wireless networks: protocols and systems. Prentice Hall, New Jersey

Usaha W (2004) A reinforcement learning approach for path discovery in MANETs with path caching strategy. In: Proceedings of ISWCS 1st international symposium wireless communications systems, pp 220–224

Xia B, Wahab MH, Yang Y, Fan Z, Sooriyabandara M (2009) Reinforcement learning based spectrum-aware routing in multi-hop cognitive radio networks. In: Proceedings of CROWNCOM 4th international conference cognitive radio oriented wireless networks and communications, pp 1–5

Yau K-LA, Komisarczuk P, Teal PD (2012) Reinforcement learning for context awareness and intelligence in wireless networks: review, new features and open issues. J Netw Comput Appl 35(1):253–267
Yin GG, Krishnamurthy V (2005) Least mean square algorithms with markov regime-switching limit. IEEE Trans Autom Control 50(5):577–593
Yu FR, Wong VWS, Leong VCM (2008) A new QoS provisioning method for adaptive multimedia in wireless networks. IEEE Trans Veh Technol 57(3):1899–1909
Zhang Y, Fromherz M (2006) Constrained flooding: a robust and efficient routing framework for wireless sensor networks. In: Proceedings of AINA 20th international conference advanced information networking and applications