IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 5, 2019, accepted April 23, 2019, date of publication April 29, 2019, date of current version May 9, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2913776

Reinforcement Learning Based Routing in Networks: Review and Classification of Approaches

ZOUBIR MAMMERI, (Senior Member, IEEE)
IRIT, Paul Sabatier University, 31062 Toulouse, France
e-mail: zoubir.mammeri@irit.fr

ABSTRACT Reinforcement learning (RL), which is a class of machine learning, provides a framework by which a system can learn from its previous interactions with its environment to efficiently select its actions in the future. RL has been used in a number of application fields, including game playing, robotics and control, networks, and telecommunications, for building autonomous systems that improve themselves with experience. It is commonly accepted that RL is suitable for solving optimization problems related to distributed systems in general and to routing in networks in particular. RL also has reasonable overhead—in terms of control packets, memory and computation—compared to other optimization techniques used to solve the same problems. Since the mid-1990s, over 60 protocols have been proposed, with major or minor contributions in the field of optimal route selection to convey packets in different types of communication networks under various user QoS requirements. This paper provides a comprehensive review of the literature on the topic. The review is structured in a way that shows how network characteristics and requirements were gradually considered over time. Classification criteria are proposed to present and qualitatively compare existing RL-based routing protocols.

INDEX TERMS Reinforcement learning, communication networks, routing protocols, path optimization, quality of service.

I. INTRODUCTION

Machine learning (ML) is a field of computer science and statistics that encompasses a set of algorithms and methods that learn from datasets and are capable of making predictions or helping to make them [1]–[3]. Nowadays, ML has a leading role in computerized societies. It is very likely that almost all future devices and machines will include ML-based components to improve their operation management and adapt themselves to their environment. ML is a powerful tool to address complex problems. Widely used in image and speech recognition, robot guidance, autonomous car guidance, telecommunication, and many other sectors, ML techniques proved their efficiency. For tasks such as classification and optimization, ML are known to (often) produce better results than human beings.

ML techniques are categorized in four classes based on how learning is carried out [1], [2]: supervised, unsupervised, semi-supervised, and reinforcement. In supervised learning—called learning with teacher—input and output variables are used to learn the mapping function from input to output; the goal is to approximate the mapping function in such a way that an output (also called label) can be accurately predicted from its associated input. In unsupervised learning—called learning without teacher—only input is used; the goal is to model the structure or distribution (e.g., data clustering) in the data in order to learn specific characteristics about data. Semi-supervised learning is similar to supervised one, but not all observations have labels (outputs). Finally, reinforcement learning is a technique inspired by the behavioral psychology and it provides system modeling based on agents interacting with their environment [4]. In the sequel, the paper only focuses on reinforcement learning (RL) application to routing in communication networks.

The complexity and the heterogeneity of modern networks, the end-users' QoS and security requirements, the economic aspects of telecommunication operators and service providers, and the social internetworking have significantly increased since the earlier communication networks. From

The associate editor coordinating the review of this manuscript and approving it for publication was Tariq Umer.

wired and manually configured networks, we moved to very dynamic and autonomic networks. The management of most of current networks became beyond the manual administration and configuration. Consequently, ML techniques have been applied to the networking field to address issues and challenges, including traffic classification and prediction, fault management, configuration management, congestion control, QoS monitoring, energy optimization, and security management [5]–[7]. The goals of ML applications to networks is to automatically learn the dynamics of networks, in particular, new flow arrivals, congestion points, topology changes, quality of links, and energy consumption to improve the service quality offered to end-users, while optimizing network resources and providers' revenues.

This paper reviews literature regarding application of RL to routing, which is a function of paramount importance in networks. When the routing is of concern, RL-based learning is prevailing in literature compared to supervised and unsupervised learning models, because of lack of datasets—representative in routing field—required by supervised and unsupervised learnings. Indeed, RL does not require availability of datasets collected through measurement campaigns.

In networks, routing is the problem of selecting paths to send packets from source(s) to destination(s), while meeting QoS requirements, if any, and optimizing network resources. The standard approach to routing is to consider the network as a weighted graph and to find paths with minimum cost in the graph and satisfying QoS requirements. The graph weights include a variety of link metrics (or factors), such as latency, reliability, stability, and energy. Whenever multiple metrics are required, routing problem becomes NP-complete [8]. Many heuristics have been applied to provide sub-optimal solutions. Literature on routing is plenteous and has considered different types of routing (hop-by-hop, source, unicast, multicast, unicast, opportunistic, and QoS-aware routing) applied to many network classes (wired, wireless, mobile, ad hoc networks, etc.).

Routing has been investigated since the earlier networks. One major concern in routing is the optimization of routes while considering dynamic topology changes. Most present networks are (very) dynamic in nature. For example, in vehicular and ad hoc networks, nodes frequently move, which results in topology changes. Likewise, energy consumption in wireless sensor networks results in nodes ceasing activity because of their battery expired. Traditional routing techniques, which are based on huge assumptions regarding traffic flows and network condition changes, are more and more perceived as inefficient to suit complex and highly changing conditions of mobile, wireless, and delay-tolerant networks. Indeed, in the event the underlying assumptions are not satisfied online, network performance may strongly deviate from those expected and (often) confirmed by simulation. RL is an efficient alternative to address network conditions as they appear in real world.

Applications of RL to solve routing problems started in 1994 with the seminal work of Boyan and Littman [9]. Then, tens of works followed the original idea of using RL to optimize routing, while gradually taking into account evolution of communication networks and users' requirements. The first objective of this paper is to provide a comprehensive presentation of the main characteristics of RL-based routing protocols. The second objective is to provide classification criteria to enable analysis and comparison of existing protocols and to help the design of new protocols for specific contexts of use. Indeed, our classification enables to address RL-based routing protocols from three complementary perspectives:

- Context of use: in which context the protocol may be used, including targeted network class, type of routing, selection of predefined routes or online discovery of optimal routes, QoS metrics to consider in optimization and in constraint satisfaction?
- Design characteristics: what are the main design characteristics of each protocol compared to others, including modeling of agent states, actions, action selection procedures, and reward function?
- Protocol performance: how significant is the overhead (in memory size and control packet amount) of routing protocols?

It should be noticed that all protocols presented in sequel differ from each other in their design—particularly in agent function design, including reward function—whereas, at first glance, some routing protocols seem similar in their modeling.

As far as we know, only [10] and [11] provided surveys regarding the early RL-based protocols—most surveyed protocols are prior to 2011. Their surveys were limited in terms of reviewed protocols (less than 20) and included very few algorithms features. In [10], the distinctive feature of surveyed protocols was the class of targeted networks. The objective of [11] was the survey of a dozen of RL-based algorithms and not the proposal of classification criteria. Furthermore, the application of RL to the design of routing protocols has made a significant progress in the current decade to address the characteristics of evolving networks, which resulted in the emergence of huge innovative and original RL-based routing protocols that have not been addressed in existing surveys. The aim of this paper is to go beyond those surveys and to provide the first comprehensive review and classification criteria to address over sixty RL-based protocols. Eighteen criteria are proposed to address different points of view in RL-based routing design. The proposed classification would serve as a framework to guide the selection of routing protocols, which fulfill given deployment requirements, or to help proposing new protocols.

The remainder of the paper is structured as follows. In section II, RL principles useful for modeling routing protocols are presented. The objectives and the context of use of existing RL-based routing protocols are summarized in section III. Section IV outlines the building blocks of RL-based protocols and the characteristics of targeted networks. In section V, criteria are proposed to

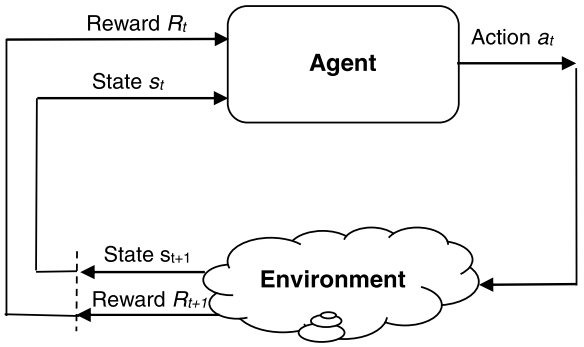

FIGURE 1. Reinforcement learning model.

classify protocols. Finally, some challenges are discussed in section VI, which concludes the paper.

II. OVERVIEW OF REINFORCEMENT LEARNING PRINCIPLES

The presentation below is limited to basic aspects of RL, which are fundamental to the analysis and comparison of building blocks of RL-based routing protocols. For more details, readers should refer to [4] and [12].

A. RL MODELING: STATE, ACTION, REWARD

The learner, also called *Agent*, interacts with its environment and selects its *actions* to be applied to *environment* according to its current state and the reinforcement it collects from the environment (Fig. 1). For example, a router interacts with its neighboring nodes to make routing decisions. In such a case, the agent is a router, the environment is router's neighborhood and actions are selections of next neighbor nodes to transmit data packets.

RL algorithms are based on *reward functions*. The role of the reward, which is returned by the environment to the agent, is to provide feedback to the learning algorithm about the effect of the recent taken action. Whereas a reward function indicates what is good (or bad) in an immediate sense, a *value function* indicates what is good (or bad) in the long-term.

Usually, an RL problem is modeled by a 4-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R})$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ the set of actions, $\mathcal{P}$ the matrix of state transition probabilities, and $\mathcal{R}$ the reward function. The environment model is described by $\mathcal{P}$ and $\mathcal{R}$. There two approaches to RL problems: model-based and model-free approaches. In the first approach, the agent learns the environment model and then improves its policy to reach optimality. Learning the environment model results in the computation of transition probability matrix. Those approaches are known to learn much faster than model-free approaches since they can reuse information stored in their internal models. However, model-based approaches are less popular because of their greater size storage cost and their dependence on the accuracy of the initial environment model. In model-free approaches, the agent improves its policy without a priori knowledge of the environment model, i.e. without requiring a transition probability matrix $\mathcal{P}$.

A policy π_t defines how the learning agent behaves at time t. $\pi_t(a \mid s)$ denotes the probability that: $a_t = a$ if $s_t = s$. a_t and s_t denote the action and state at time t, respectively. The probability to move, at time t, from state s to state s' by taking action a is:

$$\begin{aligned}&\mathcal{P}\left(s, a, s'\right)\\&\quad = Pr\left(s' \mid s, a\right) = Pr\left\{s_{t+1} = s' \mid s_t = s, a_t = a\right\}\\&\qquad \mid \sum_{s' \in \mathcal{S}} P\left(s' \mid s, a\right) = 1\end{aligned} \tag{1}$$

The reward received at time t is a real number denoted by R_t. The reward of being in state, s and taking action a also is denoted $\mathcal{R}(s, a)$. An agent selects an action at each *step* (also called *epoch* or *learning period*) of its lifetime. Consequently, time is discrete.

The objective of the agent is to take actions in order to maximize the global discounted reward, denoted by G_t, it receives over the future. Thus, the agent must be able to learn which of its actions are desirable based on the reward that can be received far in the future. There are three basic models to address optimality and to define G_t:

- *Finite-horizon* model in which the agent should optimize the reward for the next h steps:

$$G_t = \sum_{k=1}^{h} R_{t+k}$$

  Finite-horizon model is appropriate when the agent lifetime is known. For example, when routing problems are of concern, an agent may be associated with each packet and the number of states of agent is the number of hops to packet destination. When a packet arrives at destination, its agent is deleted. In such a case, agent's lifetime is known.
- *Infinite-horizon* model in which the agent should optimize the reward for the long-term run:

$$G_t = \sum\nolimits_{k=0}^{\infty} \gamma^k * R_{t+k+1}$$

  where $\gamma (0 \leq \gamma \leq 1)$ is called *discount rate*. If $\gamma = 0$ the agent is called 'myopic' and it is only concerned by maximizing the immediate reward. As γ approaches 1, the awareness to the future rewards is stronger. It is worth noticing that the infinite-horizon model is prevailing in literature regarding RL-based systems.
- *Average-reward* model in which the agent should take actions that optimize the long-run average reward:

$$G_t = \lim_{h \to \infty} \frac{1}{h} \sum_{k=0}^{h} R_{t+k+1}$$

If the environment of the agent has the Markov property (i.e., the next state depends only on the present state and the action to take), we have:

$$\begin{aligned}&Pr\left\{R_{t+1} = r, s_{t+1} = s' \mid (s_0, a_0, R_0), .., (s_t, a_t, R_t)\right\}\\&\qquad = Pr\left\{R_{t+1} = r, s_{t+1} = s' \mid s_t, a_t\right\}\end{aligned} \tag{2}$$

Notice that it is usually assumed in the field of RL applications that the environment state has Markov property or has an approximation of Markov state property. Under Markov property, the expected value of the next reward is independent of the past rewards.

RL algorithms involve two types of *Q-value functions*: *state-value* function, which estimates how good it is for the agent to be in a state and *action-value* function, which estimates how good it is to perform a given action in a given state. *State-value* of a state s under policy π, denoted $V_\pi(s)$, is the expected return in s and it is defined as: $V_\pi(s) = E[(G_t \mid s_t = s)]$. Similarly, the *action-value* of taking action a in state s under policy π, denoted $Q_\pi(s,a)$, is the expected return in starting from s and taking action a and it is defined as: $Q_\pi(s,a) = E[(G_t \mid s_t = s, a_t = a)]$.

Solving an RL problem means finding a policy to achieve a maximum reward over the long term. A policy π is better than a policy π' if its return is greater or equal to the one of π' for all states, i.e. $V_\pi(s) \geq V_{\pi'}(s), \forall s \in \mathcal{S}$. The optimal policy, denoted π_*, has a state-value function $V^*(.)$ and action-value function $Q^*(.)$ defined as follows:

$$\begin{aligned} V_{\pi*}(s) &= V^*(s) = \max_\pi V_\pi(s), \quad \forall s \in \mathcal{S} \\ Q_{\pi*}(s,a) &= Q^*(s,a) = \max_\pi Q_\pi(s,a), \quad \forall s \in \mathcal{S} \\ V^*(s) &= \max_{a \in \mathcal{A}} Q^*(s,a), \quad \forall s \in \mathcal{S} \end{aligned} \tag{3}$$

The last equation is called self-consistency condition, which simply means that the value of a state under an optimal policy must equal the expected return for the best action.

The optimal value function $V^*(s)$ can be defined as the solution to the following equation system:

$$V^*(s) = \max_{a \in \mathcal{A}} \left(\mathcal{R}(s,a) + \gamma \sum_{s' \in S} \left(\mathcal{P}\left(s,a,s'\right) * V^*\left(s'\right) \right) \right), \quad \forall s \in \mathcal{S} \tag{4}$$

Equation (4) can be solved mainly by using statistical techniques and dynamic programming methods. However, it should be noticed that the number of space solutions is at most $|\mathcal{A}|^{|\mathcal{S}|}$, which makes many problem solving techniques computationally intractable. $|X|$ is cardinality of set X. Many approaches have been proposed to learn an optimal policy at reasonable cost depending on the assumptions regarding the environment, i.e. depending on whether the learning is model-free or model-based. In the field of routing in networks, most proposed solutions are model-free. In the sequel (§II.B), our presentation is limited to the most used learning technique, called *Q-learning*, to solving routing problems.

B. Q-LEARNING

In [13], Watkins proposed an approach to estimate action-function $Q^*(.)$. Watkins's action-function is called *Q-function* and the resulting learning technique is called *Q-learning*. The latter is a model-free learning technique. Watkins's approximation of action function is independent of the policy followed by the agent, which makes Q-learning applicable in many situations and easy to implement. Thus, we have:

$$\begin{aligned} Q_{\pi*}(s_t, a_t) &\triangleq Q^*(s_t, a_t) \\ V^*(s) &= \max_{a \in \mathcal{A}} \{Q(s,a)\} \end{aligned} \tag{5}$$

In Q-learning, the agent learning consists in a sequence of stages, called epochs $(0, 1, \ldots, n \ldots)$. In epoch n, the agent is in state s_n, it performs action a_n, it receives a reward R_n, and it moves to state s_{n+1}. The action value is updated as follows:

$$\begin{aligned} Q_n(s_n, a_n) = (1-\alpha) * Q_{n-1}(s_n, a_n) \\ + \alpha * \left[R_n + \gamma * \max_{a \in \mathcal{A}} \{Q_{n-1}(s_{n+1}, a)\} \right] \end{aligned} \tag{6}$$

where α is called *learning factor*. The initial Q-values, $Q_O(s,a)$, for all states and actions are assumed given.

Q-function value updating also can be written in discrete time t (s_t and a_t are the state and action at time t) as follows to [4], which is often used in literature relating to Q-learning applications:

$$\begin{aligned} Q(s_t, a_t) = (1-\alpha) * Q(s_t, a_t) \\ + \alpha * \left[R_{t+1} + \gamma * \max_{a \in \mathcal{A}} \{Q(s_{t+1}, a)\} \right] \end{aligned} \tag{7}$$

Watkins showed that Q-learning converges to the optimum action-values with probability 1 as long as all actions are repeatedly sampled in all states [14]. For this reason, Q-learning is the most popular and most effective learning technique for learning from delayed reinforcement, i.e. learning based on reward that can be received far in the future. However, the speed of convergence remains an open issue.

C. EXPLOITATION AND EXPLORATION

In machine learning, the learner tries to improve the current solution while switching between exploration and exploitation of the solution space. Exploitation consists in considering a limited (but promising) region of the search space with the hope of improving the solutions already found. Exploitation is a local search, which has one major drawback; the search may be blocked around a local optimum. Exploration, on the other hand, consists in considering a much larger portion of the search space with the hope of finding other promising solutions that are yet to be refined. Exploration is related to global search and is more likely to lead to the global optimum. There is a wide variety of solutions to the exploitation vs exploration tradeoff problem. When RL is applied to large—in terms of solution space—systems, heuristics are commonly used, as they scale well at reasonable cost. Most known heuristics include:

- *Greedy strategies* in which the action with the highest Q-value is greedily selected. It should be noticed that greedy strategy alone may never converge to global optimum, because the action selection may never explore some actions whose initial Q-values are low.

- *ε-greedy strategies*(also called *randomized strategies*) in which actions with the highest Q-value are selected by default. However, with a probability ε, an action is randomly selected from the actions eligible in the current state whatever is its reward, in order to explore alternative actions.
- *Interval-based strategies* in which statistics (including number of trials and number of successes) for each action are stored. Then, an action is selected depending on its confidence interval on the action success probability.
- *Probability distribution based strategies* in which the decision is based on a chosen probability distribution. In routing protocols, Boltzmann probability distribution is sometimes used thanks to its effectiveness. It is defined as follows:

$$P(a_t|s_t) = \frac{e^{\beta * Q(s_t,a_t)}}{\sum_{a\in\mathcal{A}} e^{\beta * Q(s_t,a)}} \quad (8)$$

It should be noticed that the choice of exploration and exploitation techniques has a great impact on the speed of convergence to optimality of the learning process.

III. APPLICATION OF RL TO ROUTING PROTOCOLS

A. MAIN ISSUES AND COMPONENTS IN RL-BASED ROUTING DESIGN

In RL-based design, the following aspects are addressed: i) identification of the most appropriate states and actions of agent, ii) definition of the reward function depending on metrics to optimize, and iii) identification of environment model when available.

Given a target field of application, different design models may be elaborated. Those models differ in how they address each of the three previous aspects. It is the same, when routing in networks is of concern.

Many RL-based routing protocols have been proposed in last 25 years. Fig. 2 shows a high-level structure, which highlights components involved in RL-based routing, not all components are included in all existing routing protocols.

In basic reinforcement learning method, reward is received from the environment after action selection. Reward is positive when selected action is 'good' regarding the environment or negative when it is 'bad'. Revenue (in dollar) is an eloquent example of reward, as the agent may win or lose money. In RL-based routing protocols, reward is addressed as a 'cost' to send packets. From a networking point of view, cost may refer to delivery delay, loss rate, energy consumption, and so on.

In literature, nodes are confused with agents and in almost all protocols, the reward is calculated—at least partially—by a node upon selecting an entire route to use for all packets to transmit or just a next hop to transmit the current data packet. Consequently, to fully comply with RL principles, a node should be considered to consist of an agent and optional components:

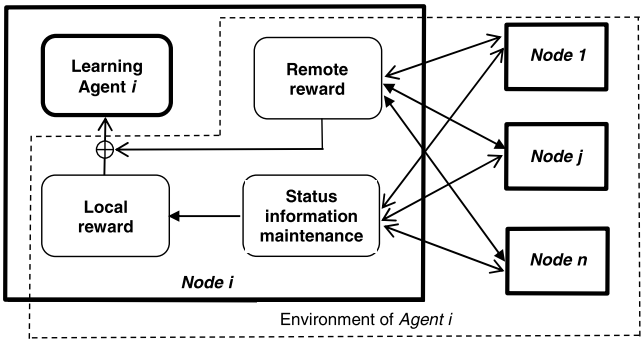

FIGURE 2. High-level structure of RL-based routing.

- *Local reward* module, which calculates reward based on local view; local reward reflects the cost of communication as seen by packet sender.
- *Remote reward* module, which receives feedback sent by the next hop or by the destination node. Whenever both local and remote rewards are used in routing protocol, they are combined to form the reward returned to agent.
- *Link-state information maintenance* module, which collects useful link state information (such as the location and the residual energy of neighboring nodes and the quality of links) through periodic or on-demand *Hello beacon packets*. By shortcut, the latter are commonly called *Hello packets*. They are used for link state advertising in networks and consequently provide a support for collaboration of agents in RL-based routing algorithms.

Thus, a part of the node hosting an agent and the neighboring nodes form the environment of agent.

B. NETWORK CLASSES AND CHARACTERISTICS ADDRESSED IN RL-BASED PROTOCOLS

Communication networks are commonly categorized into different classes depending on their characteristics regarding the type of medium (wired or wireless), energy constraints, mobility, and so on. Network characteristics have a strong impact on RL-based optimization of routes. In particular, the design of the components shown in Fig. 2 is guided by the characteristics of targeted network. Overall, distinctive characteristics of networks that impact the protocol design in general and the design of RL-based protocols in particular are mainly: infrastructure-less or infrastructure-based networks, centralized or distributed control, mobility of nodes, topology changes, energy consumption and lifetime, capacity and stability of links, duration of link disconnections, availability of resources (wavelengths, channels, bandwidth), cooperation between nodes (in load balancing, in relaying packets, in data retrieval…), and accuracy of link-state information disseminated in network.

RL-based routing protocols have been extended and improved gradually as networks evolved. Thus, existing RL-based protocols addressed characteristics of almost all

network categories. Below is a brief presentation of networks, which focuses on the main characteristics considered to design RL-based routing.

1) WIRED NETWORKS (WiredN)

They were prevailing twenty years ago and are generally characterized by static network topology and stable link capacities compared to wireless networks.

2) WIRELESS NETWORKS (WNd)

WNs use the air as a medium for transmissions. They include all categories of wireless networks presented below.

3) WIRELESS MESH NETWORKS (WMNs)

In WMN, each mesh router is equipped with multiple radio interfaces and a subset of nodes serves as a gateway to the Internet [15]. Routing in WMN aims at establishing paths to enable regular nodes (also called user nodes) to access the Internet via gateways.

4) COOPERATIVE COMMUNICATION WIRELESS NETWORKS (CCWNs)

In CCWN, users work cooperatively by relaying data packets for each other, thus minimizing the antenna number to support multi-hop communications [16]. The basic idea of CCWN is that single-antenna mobiles in a multi-user scenario can share their antennas in a manner that creates a virtual MIMO system. Thus, each mobile user is assumed to transmit its data and acts as a cooperative agent for another user.

5) OPTICAL NETWORKS (ONs)

In WDM (wavelength division multiplexing) ON, the bandwidth is divided into many wavelength channels [17]. The function of routing in WDM ON is to dynamically find a path and to assign wavelengths to support traffic of requesting sources. In contrast to electrical switches and routers, optical switches do not have buffers to queue packets. Consequently, one main performance issue in optical networks is the minimization of dropped packets. In order to avoid packet discarding in case of contentions (i.e., when multiple packets request the same path), one commonly used mechanism is routing deflection, which consists in forwarding packets on alternate paths. Since deflection disturbs nodes along selected alternate paths (because those nodes have more traffic to relay), deflection routing should minimize deflection actions and select the best alternate paths, while minimizing packet discarding.

6) AD HOC NETWORKS (MANETs AND WANETs)

They are composed of wireless nodes without central control; they are characterized by frequent changes in topology and link capacities. When node mobility is considered, ANETs are categorized as MANETs (Mobile Ad hoc Networks) [18]. Mobility is an important issue when routing is of concern.

7) WIRELESS SENSOR NETWORKS (WSNs)

A WSN is composed of a set of nodes (sensors and sinks), which are (often) densely deployed in an area to supervise a given phenomenon (e.g., a building, a farm, a factory...). Nodes communicate wirelessly and (often) have battery limitations and also computation and memory limitations [19]. The main challenges, when routing is of concern, is to consider energy consumption to optimize network lifetime.

8) VEHICULAR NETWORKS (VANETs)

They are a category of MANETs where nodes are vehicles moving in an urban area or on highways. In present and future Intelligent Transportation Systems, moving vehicles need to acquire real-time traffic and road information from sensors, deployed along roads and forming a WSN. VANETs are used as a technology to enhance safety on roads through exchanges between vehicles to announce accidents, traffic congestion, obstacles, road distortion, and freezes [20]. Delay constraints associated with messages exchanged and dynamic changes in topology of VANETs are the main issues to consider in routing to on-time deliver a maximum of safety messages to vehicles on zone.

9) DELAY TOLERANT NETWORKS (DTNs)

They are characterized by their lack of connectivity, resulting in a lack of instantaneous paths to deliver packets at destination [21]. DTN routers follow a ''store and forward'' strategy, i.e. routers keep packets until they discover new neighbors, which are likely to deliver the packets. The main objective of DTNs is to maximize the delivery ratio (i.e., maximize the number of packets delivered to destination).

10) FLYING AD HOC NETWORKS (FANETs)

They are ad hoc wireless networks to support communication between Unmanned Autonomous Vehicles (UAV), such as drones and aircrafts. The high speed of UAVs and distances between UAVs make the routing in FANET a challenging issue [22].

11) SOCIAL DTNs (SDTNs)

They are a category of DTNs where nodes belong to people moving, for example, in a campus, in a city or on a highway. In SDTNs, the interest profiles of people and their social interactions as well as their visited locations are explored to route data packets.

12) COGNITIVE RADIO NETWORKS (CRNs)

They provide solutions to scarce spectrum resource problem [23]. They enable unlicensed users (also called secondary users, SU) to seek opportunities for transmission by exploiting the idle periods of licensed users. In CRN, each secondary node is equipped with a cognitive radio transceiver. CR nodes sense the spectrum to detect unused frequency bands and then decide to exploit them. As the availability of channels

depends on the licensed users in neighborhood, secondary nodes have a dynamic view of the spectrum utilization, which differ from node to node depending on their location. RL has been used to collect accurate view of spectrum availability at a reasonable cost and maximize spectrum availability to secondary users and minimize interference between primary and secondary users.

13) NAMED DATA NETWORKING (NDN)

NDN is one of the most attractive Information-Centric networking architectures. NDN focuses on the content itself—and not on node addresses—and follows the publisher/consumer model [24]. In NDN, some nodes store contents and reply to interest requests issued by end-users. When a node needs a content, it broadcasts an interest request including the name of the content. Nodes receiving an interest request may reply sending the content or forward the request. In NDN architectures, interest packets do not include data as in traditional networking, but an interest in receiving a content specified through a *data name*. In addition, nodes store content that is likely to be requested. The main issue in NDN is to optimize content storage at relaying nodes and data packet delivery delay, while taking into account the spatial and temporal distributions of interest requests.

14) PEER-TO-PEER NETWORKS (P2PNs)

P2P networking provides a wide range of Internet applications, such as content delivery, file sharing, and multimedia streaming [25]. Since the nodes relaying the queries do not have a complete knowledge of the network, query load balancing between relaying nodes is one of the most prevalent issues in P2P networks. Flooding-oriented (also called blind) query forwarding protocols do not optimize neither network resources nor content server resources. Using—through reinforcement learning—history information about relaying nodes results in a significant improvement of resources utilization and satisfaction of users requesting contents.

15) SOFTWARE DEFINED NETWORKING (SDN)

SDN has been recognized recently as a promising solution to simplify the management of network resources [26]. SDN is based on layered centralized architecture. In particular, SDN-controllers determine the best routes and send routing tables to switches, which follow the received tables to forward data packets.

IV. OVERVIEW OF RL-BASED ROUTING PROTOCOLS

As far as we know, Boyan and Littman were the first to propose a hop-by-hop routing algorithm based on Q-learning, called Q-routing [9]. Many of existing RL-based routing protocols are extensions to Q-routing. In the sequel, Q-routing is presented in detail and its principles will serve as a basis for introducing other protocols.

Algorithm 1 Q-Routing

1: $Q_i\,(*,*)$ is the Q-value matrix of node i.
   /*Q_i matrix may be randomly initialized. */
2: **Loop**
3: **if** (Packet to send is ready):
4: Select next hop j with the lowest Q-value
5: Send packet to node jj
6: Node i immediately gets back j's an estimate for the time remaining in the trip to destination d denoted calculated by formula (9)
7: Node i updates its delivery delay estimate using formula (10)
8: **end if**
8: **Until** Termination_condition /* which may be a number of iterations, a time-out or something else */

A. Q-ROUTING PROTOCOL

The name of the proposed algorithm comes from the notation of *Q-function* used in Q-learning method (§II.B). The algorithm may be summarized as follows:

Let ii denote the node holding a packet P to forward and $Q_i\,(d,j)$ denote the end-to-end delay (or simply delivery delay) that node ii estimates it takes, for node j, to deliver packet Pi at destination di. Node i maintains a table including the transfer delay estimates, called Q-values; a Q-value is associated with each neighbor of node i. When node i has a packet to send, it selects a node j with the lowest Q-value. Upon sending packet P to node j, node i immediately gets back j's an estimate for the time remaining in the trip to destination d denoted by $\theta_j(d)$:

$$\theta_j(d) = \min_{k \in Ng(j)} Q_j\,(d,k) \tag{9}$$

where $Ng(j)$ denotes the set of j's neighbors. Then node i updates its delivery delay estimate associated with neighbor j as follows:

$$Q_i\,(d,j) = (1-\alpha) * Q_i\,(d,j) + \alpha * \left(qt_i + TxT_{i,j} + \theta_j(d)\right) \tag{10}$$

where α is a learning parameter, qt_i is the time spent by packet, P in i's queue and $TxT_{i,j}$ is the transmission time between nodes i and j. The pseudo code of Q-routing is the following:

Q-routing complies with Q-learning as follows:

- In Q-routing, a state is a node and an action is ''select a neighbor to be the next forwarder to deliver packet to destination''.
- Regarding Q-learning Q-value updating rule (7), immediate reward of Q-routing, R_{t+1} is $qt_i + TxT_{i,j}$(this reward represents the link cost) and $\max_{a \in \mathcal{A}} \{Q\,(s_{t+1}, a)\}$ is $\min_{k \in Ng(j)} Q_i(d,k)$('min' is substituted to 'max', because delay is a decreasing metric, i.e. the lower the delay is the better the path). Discount factor γ is set to 1.

To avoid frequent oscillations in Q-values (in case of sudden variations of traffic in the network) and limit the overhead of Q-routing protocol, [27] proposed an extension in which the receiver does not send an immediate reward for each received packet, but after receiving a certain number of packets (i.e., the receiver returns the average delay for a group of packets). However, Q-routing still suffers from at least the following:

- *Q-value freshness.* The estimate of delay occurs upon packet transmission on a route. When a route is not selected during a long period of time, the agent has no accurate estimate about the current condition of such a route and its Q-value may become unreliable. When a node resumes its transmission activity on a given route, after a long idling period, its delay estimate values may result in non-optimal selection of next hop.
- *Slow convergence.* Q-learning requires a number of epochs (or tries) before being able to converge to the optimal solution.
- *Parameter setting sensitivity.* Small adjustments in learning parameter may result in serious fluctuation in routing performance.

B. OBJECTIVES AND MAIN SPECIFICITIES OF RL-BASED ROUTING PROTOCOLS

Literature of which we were aware regarding RL-based routing has been carefully addressed. Any paper, which provides a contribution beyond the state-of-the-art to application of RL to routing, is included in the presentation below. Before providing details about the design of each protocol in section VI, the main objectives and specificities of existing RL-based routing protocols are summarized. It should be noticed that protocols are presented in *their chronological order*, because many protocols are extensions or specialization of previously published ones. A few former routing protocols such as [28]–[30] focused on statistical learning in routing and will not be included in the sequel, as they do not follow RL paradigm (including explicit description of Agents, Q-value update rules, and reward).

In the sequel, protocols marked with 'NATG' are those protocols without title or acronym given by their authors. To provide a homogeneous presentation, we chose titles and acronyms for a few protocols.

PQ-R (Predictive Q-routing) – It is an extension to the original Q-Routing to consider Q-value freshness [31]. PQ-R keeps the best Q-values and reuses them by predicting the traffic trend. The idea of PQ-R is that when routes are considered congested, they should not be selected for packet transfer for a period of time to enable them recover from congestion. Those routes are called regulated routes. To check regulated route conditions and refresh delay estimate values, PQ-R probes them at a given frequency (i.e., regulated routes are occasionally selected for packet transmission). Notice that the probing frequency has an impact on the conditions of congested routes and probing should not make them worse. Compared to Q-Routing, PQ-R does not use a discount rate, but two additional learning parameters β_1 and β_2 are introduced to address the path delay variation.

ARL-R$^{(NATG)}$ (Ants and RL-based Routing) – It is a combination of RL and the ants optimization technique to address routing in networks subject to frequent topology changes due to link failures [32]. Unlike distance-vector algorithm, ARL-R is probabilistic in nature. Ants algorithm is used to explore the network. Each host (i.e., end device) s periodically generates a packet (referred to as ant) to another randomly chosen host d to collect path cost c from d to s. When a response message $ant(s, d, c)$ is sent backward to host s, routers on path update their routing tables (i.e., their Q-value tables).

CQ-R (Confidence-based Q-routing) – CQ-R is an extension to Q-Routing to address Q-value freshness [33]. In CQ-Routing, a confidence measure between 0 and 1, also called C-value, is associated with each Q-value. A C-value close to 1 means that the corresponding Q-value accurately represents the current state of the route, while a C-value close to 0 means that the corresponding Q-value is almost random. When a node j j sends its best delay estimate $\theta_j(d)$ to node i, it also sends its C-value $C_j(d)$ associated with such a Q-value. At each epoch, C-value associated with each neighbor is either updated using the feedback from neighbor or decayed with a constant factor $\lambda \in [0, 1]$. Setting of λ depends on how many learning epochs may elapse without selecting a node to consider that the Q-value associated with that node does not provide any help to select the best next hop. The learning should be high (i.e., the estimate of delay is more accurate in the current epoch than in the previous one) if either (or both): i) confidence in the old Q-value is low or ii) the confidence in the new Q-value is high.

CACR-RL (Connection Admission Control and Routing based on RL) – [34] proposed a solution to call admission control and routing on a list of paths (with limited bandwidth) fixed offline to optimize revenue. The network supports a set of service classes characterized by their bandwidth demands. A link can carry simultaneously any combination of calls, as long as the sum of bandwidth allocated to calls does not exceed the link capacity. When a new call arrives, the network provider either accepts or rejects the call. When the call is accepted, a predefined path, which can meet call QoS requirements, is assigned to the call. Whenever a call is accepted, network provider gets an immediate reward depending on the call service class. RL is used to optimize the long-term revenue of network operator, assuming that the average holding time is known for each service class.

Q-MAP (Q-learning Multi-Agent multicast routing Protocol) – It is a multicast routing with resource reservation to support flows with soft delay requirements in MANETs [35]. It is a mesh-based multicast scheme and it uses group forwarding concept to maintain group membership depending on the availability of resources on paths. When a node wants to join a group, it sends a join request (including its ID and a group ID) to find an optimal route. Whenever a node receives a join request, which is a non-duplicated request,

it generates a new join request (including its ID, the upstream ID, and the cost link) and sends it. Whenever a node receives a join response, it updates its routing table. Once multicast paths are established and resources reserved, they are periodically refreshed and used to send data packets. RL is used in the join phase to select the next hop and build an optimal path, which is part of the tree associated with the group ID.

GAPS-R (Gradient Ascent Policy Search based Routing) – It is an RL and gradient ascend based routing for wired networks [36]. Packets (without data and representing implicit Route requests) are periodically introduced in the network with uniformly nodes of origin and destination. Sending a packet on a link has a cost. Once a packet reaches its destination, it is dropped and an acknowledgement is sent backward, which enables nodes on path to update their path cost. Gradient ascent—relating to histories of interactions with other nodes during an epoch—is used to calculate transition probabilities. The idea is to adjust parameters used in transition probabilities in the direction of the empirically estimated gradient of the aggregate reward.

Q-LMRWA (Q-Learning Multicast Routing and Wavelength Assignment) – [37] proposed a multicast routing protocol for optical networks with blocked light-path optimization. In WDM (Wavelength-Division Multiplexing) networks, a light-path is composed of optical links. Connections between sources and destinations use light paths allocated to them to send their data packets. Whenever some connections share the same optical link, they cannot simultaneously use the same wavelength. Consequently, a connection is blocked if there are no available wavelengths. One performance metric of interest in WDM networks is ''blocking probability'', which is defined as the number of blocked connections divided by the total number of active connections. Whenever a connection request is issued, RL is used to find a route that optimizes the number of required wavelengths while minimizing blocking probability.

RL-AODV$^{(NATG)}$ (RL-based AODV) – [38] proposed a modification to AODV protocol [39] to make the next-hop selection dependent on the experience gained through reinforcement learning and more aware of network dynamics. Whenever a node receives a Route request, it selects the best one among its neighbors according to local state represented by Q-values. A special class of neural networks is used to store Q-values with a constant memory size. From RL point of view, RL-AODV is an adaptation of Q-learning.

MARL-R$^{(NATG)}$ (Mobility-Aware RL-based Routing) – In Q-routing, network topology is assumed static. To address changes in topology of MANETs due to node mobility, [40] proposed MARL-R, a light adaptation to Q-routing, which can be summarized as follows: when a node j j moves out of range of node i, the latter sets the Q-value associated with j to ∞. Consequently, node j will be no more selected by node i. When j is detected again in range of node i, j's Q-value is optimistically set to 0. This optimistic bias encourages exploration allowing node i try to send packets via the new discovered neighbor. After some tries, the estimated delay via node j will reflect its real capacity in the network.

MQR (Modified Q-learning Routing) – [41] proposed a modification to Q-routing to address routing in wired networks. The specificity of MQR is that agents (nodes) exchange their immediate rewards, which results in multi-agent solution for global optimization. Q-values are based only on link cost. In addition, MQR proposed to periodically devaluate Q-values in order to enhance the solution space exploration, i.e. to escape the local minima.

CRL-SAMPLE (Collaborative Reinforcement Learning SAMPLE routing) – [42] proposed a routing protocol based on a variation of RL to maximize delivery in MANETs, while minimizing transmissions per packet from source to destination. CRL-SAMPLE follows a model-based learning. Using activity history (including number of successes and failures of transmitted packets and number of unicast and broadcast received packets) of wireless link, each node builds a probability transition matrix, which reflects the probability of progress to destination when each neighbor is selected as next hop to forward the current packet. Then, the probability transition matrix is used to select the next hop in the forwarding process. Each node stores the last advertised link costs received from its neighbors. Links that have not been advertised for long time are devaluated, until eliminated from the forwarder selection.

RLGAMAN (RL Genetic Algorithm based routing for MANets)– [43] proposed a routing protocol for route discovery in MANETs with enough resources to provide QoS in terms of bandwidth and delivery delay. RLGANMAN integrates two key parts RL and GA (genetic algorithm). RL is used to find feasible QoS routes, based on local information. To avoid all packets travel on the same routes (which results in congestion) and to explore as much as possible the solution space, GA approach is used. Through RL, many feasible QoS routes are discovered by the source. GA population consists of individuals, which represent routes between a source and potential destinations. The fitness of a route is determined by the QoS measurements returned by the ACK received upon sending a data packet on that route. Crossover and mutation operations are introduced in RLGAMAN to optimize routes. The selection probability of a route to send a data packet is based on the rank (i.e., fitness value) of the route. While, in previous protocols, the data packets are sent on the best route discovered by the learning algorithm, in RLGAMAN, RL is used on a route-request-basis to discover routes and Genetic algorithm is used on a per-data-packet-basis to select route for data packet sending.

RLCF$^{(NATG)}$ (Reinforcement Learning based Constrained Flooding) – [44] proposed an adaption of Q-routing to optimize the number of packet transmissions to send a data packet to a sink in WSN using flooding; thus it enables energy saving. To reduce the cost of flooding, RLCF uses the Q-learning technique to learn the cost of packet sending (the cost is generically defined and may be adapted to consider hop count, delivery delay, and so on). Each data packet

includes a Q-value representing the estimated cost of the sender. Instead of selecting the next neighbor with the lowest cost as done in Q-routing, the receiver of a packet makes a decision regarding the packet broadcasting based on three mechanisms: constrained propagation (retransmit the packet if the difference between sender and receiver costs is below a threshold), differential delay (use the cost difference to yield a waiting time before retransmitting), and probabilistic retransmission (use a probability based on the number of receptions of the same packet to decide whether to retransmit or discard such a packet). Notice that no control packets are used in RLCF.

AdaR (Adaptive Routing) – [45] proposed AdaR protocol for routing in wireless sensor networks. As far as we know, AdaR is the first to consider multiple metrics instead of a single one (i.e., delay) as did the other protocols. More precisely, four QoS metrics are considered in path selection: number of hops, residual energy, link reliability, and number of routes crossing in a node. When a packet is transmitted, a QoS vector is appended to the packet. Whenever the packet arrives at base station, one can trace the QoS information along the whole routing path. AdaR uses LSPI (Least Squares Policy Iteration), a variant of reinforcement learning [46], which enables faster convergence to optimal solution without suffering initial parameter setting.

RLGR (Reinforcement Learning based Geographic Routing) – [47] proposed a routing protocol for WSN in which nodes relaying packets are equipped with Ultra-Wide Band (UWB) transceivers and can obtain their location. Each node knows its residual energy and those of its neighbors and their location, through Hello packets. To select a forwarder, each node uses residual energy and locations of its neighbors. RLGR aims at maximizing network lifetime.

Q-PR (Q-Probabilistic Routing) – [48] proposed a Q-learning based opportunistic routing protocol to broadcast packets in WSN. Originality of Q-PR is that it combines RL and Bayesian decision. In opportunistic routing, where each node decides to forward or discard a packet with a probability, a packet may never arrive at sink. Optimizing delivery ratio is a concern in opportunistic routing design. In addition, in WSN, optimizing energy is of paramount importance. In Q-PR, each node is aware of its geographic position as well as positions of its neighbors and the sink. Each node keeps an estimate of the delivery probability, for each neighbor; the estimate is defined only in function of distance between nodes. In Q-PR, whenever a node *i*i receives a packet, such a packet also is heard by its neighbors. If *i*'s neighbors with higher delivery probability forward the packet, then node *i* discards the packet. Otherwise, node *i* has to decide to forward or to discard the packet. To do so, it proceeds as follows: firstly, among neighbor nodes closer than itself to sink, it selects a neighbor that minimizes the ''expected number of retransmissions'' divided by ''how much distance to sink is decreased when choosing such a neighbor''. Then, iteratively it selects the best nodes among remaining neighbors that fulfill the constraint of being neighbors of the previously selected candidates (i.e., all selected nodes at this stage can communicate each with others and consequently, if one of them forwards a packet, the others will be aware of it). Secondly, Using the residual energy and locations of selected candidates, Bayesian decision is applied to infer whether node *i* should forward or discard the packet. It calculates the probability to transmit the packet depending on residual energy of selected candidates and the probability that all its neighbors with higher delivery probability will not transmit. In case node *i* confirms hypothesis that at least one of selected candidates will (re)forward packet, it broadcasts the packet, hoping that it will be (re)forwarded. In case node *i* confirms the reverse hypothesis (i.e., no candidate will forward the packet), it does not uselessly transmit to save energy. After transmission, if any, node *i* listens to transmission in its neighborhood. If no selected candidate (re)forwards the packet, node *i* retries once again to transmit or to discard the packet.

SERLR$^{(NATG)}$ (Selfishness and Energy aware RL based Routing) – [49] proposed a generic algorithm, which combines RL, stochastic approximation, and function approximation to select the next hop for packet forwarding in MANETs. It provides a generic framework to estimate energy consumption and selfishness of nodes. Then, a forwarding probability is dynamically associated with each neighbor node based on its energy and selfishness estimates and on its ratio of packet re-forwarding. The implementation of SERLR requires the definition of energy and selfishness functions, which are not addressed in the paper, because the proposed algorithm is generic and may be specialized according to the targeted routing protocol. Notice that SERLR is the only RL-based routing algorithm that uses selfishness as a metric in next hop selection.

FROMS (Feedback Routing for Optimizing Multiple Sinks), E-FROMS, and CLIQUE – The first RL-based protocol for multicast routing in WSN was proposed in [50]. FROMS protocol aimed at establishing efficient paths to enable a source sends its data to multiple mobile sinks, such as vehicles. Selected paths form a tree, like a Steiner tree. Routing to multiple destinations is defined as the minimum cost path starting at the source and reaching all destinations interested in the data from the source. The cost of a spanning tree is defined as the number of one-hop broadcasts to reach all sinks. Finding the minimum-cost tree is known to be NP-hard. Therefore, RL is used to approximate the optimal solution.

For estimating the initial Q-values, FROMS uses control packets sent by sinks to announce their interest to receive data collected by the source. Specifically, a sink announcement packet records the number of hops from sink to source. Since the announcement packets are sent independently by sinks, they don't provide the optimal spanning tree. Then, the recorded hop counts are used to calculate the optimal spanning tree from source to sinks. When a node *i* decides to forward a packet to its selected neighbor *j* with minimum Q-value, it includes in the packet routing information (which neighbor is selected to reach which sinks). Since the

wirelessly broadcasted packets are heart by all neighbors, when a node i forwards a packet to node j all i's neighbors can update their link-state information useful to calculate the reward. By repeating forwarding operations, accurate information propagates from the sinks to the source. Sink mobility also is addressed in FROMS. When a sink moves, it may become unreachable through the previously computed spanning tree. To consider such a problem, FROMS suggests to use periodic sink announcement signaling to detect broken links and learn news paths. In [51], the authors provided intensive simulation of FROMS.

FROMS authors proposed two extensions to FROMS: CLIQUE [52] to address dynamic clustering in WSN and E-FROMS [53] to address energy consumption. In CLIQUE, each node can decide on per-packet basis to act as a cluster and send the packet in sink direction or just forward the packet to one of its neighbors. RL is a powerful tool to avoid many of the assumptions (e.g., cluster head directly communicates with the sink, cluster heads are one-hop from each others, cluster head has a very-long-term battery capacity, etc.) made in traditional clustering approaches. In addition, CLIQUE minimizes the clustering overhead, which suffers most of proposed clustering approaches.

RL-QRP (RL-based QoS aware Routing Protocol) – [54] proposed a routing protocol to deliver packets (which contain patient body temperature, heart rate, and so on) to medical center reliably and on-time. Sensor nodes are GPS-equipped and exchange their local information through periodic Hello packets. With location information, nodes compute available paths according to QoS requirements of data packets and the link quality of available paths and then forward data packets. RL is used for QoS path computation and next hop selection. Each data packet contains the QoS requirements of transported data. Whenever a data packet arrives at a node, which cannot meet packet QoS requirements, the packet is discarded.

MRL-QRP (Multi-agent Reinforcement Learning based QoS Routing Protocol) – It is a routing protocol with QoS support in WSNs [55]. Local information is exchanged between neighbors through Hello packets. The distinctive characteristic of MRL-QRP is the collaboration between agents. Instead of independent agents—as used in almost all RL-based protocols—MRL-QRP relays on tight collaboration between agents when they compute their Q-values. Specifically, whenever a node computes Q-value associated with a neighbor node, it uses Q-values of all its neighbors and assigns a weight to each one of them. This approach is global optimization-based. However, assigning a weight to each neighbor makes MRL-QRP difficult to use in practice.

RLDRS (RL-based Deflection Routing Scheme) – In optical networks, switches are buffer-less, which results in packet loss whenever contentions occur at switching nodes, because multiple connections share the same wavelength on the same fiber link. Reference [56] proposed RLDRS protocol to provide routing solution to deflection in optical networks, while minimizing loss rate. RL is used to learn the best output links and the available wavelengths and select them in case of burst deflection. Link quality is measured in function of dropped packets and successfully sent packets. High quality link, which is selected in case of deflected, is the one with less dropped packets.

RL-BER$^{(NATG)}$ (RL-based Balanced Energy Routing) – RL-BER is a protocol aiming at balancing energy consumption among nodes and maximizing network lifetime in MANETs [57]. RL is used to estimate energy consumption of paths. Path selection is based on energy consumption associated with paths and on bottleneck link residual energy. Consequently, nodes with low residual energy (i.e., bottleneck links) have less chance to be selected, which minimizes their energy consumption.

SQ-R (Spectrum-aware Q-Routing) – It is an extension to Q-routing to optimizing channel sharing between primary users (PUs) and secondary users (SUs) in radio cognitive networks [58]. RL is used to learn availability of channels (i.e., channels temporarily unused by PUs) and temporarily allocate them to SUs, while minimizing interference PU-SU. Interference occurs when a PU resumes activity and its allocated channel is used by SUs. Q-values associated with paths reflect the availability of channels along paths. Packets incorporate Q-values (i.e., available channels), which enable nodes be aware of channel availability in their neighborhood.

QoS-RSCC (QoS support adaptive Relay Selection for Cooperative Communications) – It is an extension to AODV protocol to integrate cooperative communication (CC) in WSN [59]. When a path is established, following AODV principle (such a path is denoted AODV-path), a set a relaying candidates is associated with each pair of routers along AODV-path to support cooperative communication. Then, data packets are sent along AODV-paths when QoS requirements are satisfied along those paths. Whenever an AODV-path cannot meet QoS requirements, CC relaying nodes are invoked to cooperate and support data packets until the AODV-path becomes again able to meet QoS requirements. QoS-RSCC follows a multi-agent paradigm (i.e., an agent uses Q-values of all its neighbors to update its local information). Routers exchange ACK and NACK to provide feedback about data packet transmission along the path. RL is used to learn which CC relaying nodes are more likely to forward data packets, while satisfying QoS requirements.

RL-CAC (RL-based Call Admission Control) – [60] provided a CAC and priority-based routing for per-class services on a list of optical paths configured offline. RL-CAC objective is to reserve network resources to services with higher priority in order to maximize the long-term revenue of network operators. For each pair of source and destination nodes, a set of alternative paths are built offline. Online, each node collects information regarding availability of wavelengths in neighborhood. Whenever a new call arrives, RL is used to reject or accept the new call to maximize revenue. RL-CAC learning addresses optimality from infinite horizon point of view.

RRD-R$^{(NATG)}$ (RL and Russian Doll based Routing) – [61] proposed a Q-learning-based routing protocol for MANETs for optimizing multiple (generic) QoS metrics. It differs from other protocols in reward calculation and in action selection. While most of RL-based protocols calculate locally the reward, in RRD-R, the reward is computed at destination and sent backward to source, enabling nodes on path to update their Q-value tables. As RRD-R may be applied to consider multiple metrics, the reward is a vector. While almost all protocols use additive or multiplication functions where each metric has a weight assigned to it, RRD-R requires from the user to divide the multidimensional space of QoS metrics into boxes, where the outer box contains all the space and the inner one contains the best QoS for all metrics. QoS boxes look like ''Russian dolls''. Then, action selection is based on boxes depth, the inner the box, the better the QoS. It is worth noticing that RRD-R is difficult to deploy, unless the user has clear preferences among QoS metrics.

QDTR (Q-learning-based DTN Routing) – [62] proposed QDTR, a Q-Learning based protocol for routing in Delay Tolerant Networks (DTNs) in underwater context. QDTR addresses delivery time delay and energy optimization in DTNs. In addition, packet queues at each node are served according to deadlines assigned to packets. Thus, urgent packets are first served on the selected paths. When the deadline of a packet is exceeded, the packet is removed from the network. In an underwater environment, the network topology is modeled as a 3D layered topology depending on the covered water depth and volume. Only nodes in the same layer (i.e., at the same depth and at some distance) can communicate. The novelty of QDTR is in its reward function, which combines energy, distance, density, and discovery of mobile nodes.

ARBR (Adaptive Reinforcement Based Routing) – [63] proposed the ARBR protocol, which enables routing in DTNs. Nodes cooperate each with others to make forwarding decisions based on contact time statistics, network congestion, and node buffer occupancy sampled during previous contacts between nodes. ARBR follows the Collaborative RL paradigm, which means that the decisions are made based on contact tables (which include all statistics as seen by individual nodes) exchanged between nodes. A node selects the next forwarder based on its ability (learnt from previous contact table exchanges) to provide progress to the destination of the packet. A probability matrix of successful exchange between nodes is updated whenever packets are forwarded and used to select the next forwarder for the current packet. The storage capacity, in terms of number of packets, of each node is limited and any node should not be selected as forwarder whenever its capacity limit is reached.

QELAR (Q-learning based Energy-efficient and Lifetime-Aware Routing) – [64] proposed the QELAR protocol aiming at finding routing paths in underwater DTNs. As for QDTR, QELAR addresses delay and energy optimization. In addition, QELAR addresses the optimization of network lifetime. The main distinctive feature of QELAR is its reward function, which is designed to enable the protocol to be energy-efficient and lifetime-aware. As the calculation of the reward—and consequently the Q-values—depends on the energy of each node i and that of its successor and the average energy of their respective groups, when node i forwards a packet to its selected neighbor j, it includes in the packet its current Q-value, its current residual energy, the current average residual energy of its group, and j as next forwarder. Node j returns an Ack once it receives the packet. Nonselected neighbors drop the packet after checking the next forwarder field. To refresh Q-values when no data packets are transmitted, each node periodically broadcasts to its neighbors a Hello packet including its link-state information. When a node receives a packet (either a data packet or a Hello packet from its neighborhood), it updates information (Q-value, residual and group energy) associated with the sender. If the packet forwarding attempt from node i to node j fails (i.e., if node i does not receive an Ack from node j after a given number of retransmissions), a failure reward is applied, because energy has been consumed in (re)transmissions.

SNL-Q (Statistical Network Link based Q-learning routing) – It is an RL-based routing protocol for MANETs, which uses link quality as the main metric for searching optimal paths [65]. Link quality is calculated using statistics related to communication events (i.e., numbers of attempted unicast, successful unicast, received unicast, and received broadcast) occurring in node neighborhood. Then, link quality is used in Boltzmann probability distribution to explore solution space.

QLMAODV (Q-Learning Modified AODV) – It is an extension to AODV for routing in MANETs, which supports flows with soft QoS requirements [66]. Mobility estimate, available link bandwidth, and transmission power are used as metrics to update Q-values. Only the node, which directly delivers the packet to destination, receives a non-null reward. Local information is disseminated in neighborhood as in AODV.

RL-RPC$^{(NATG)}$ (RL-based Routing and Power Control) – [67] proposed an RL-based protocol for routing and power control in multi-hop wireless networks, while providing soft delay guarantees to delay-sensitive applications such as video streams. RL is used to learn channel conditions and packets waiting in queues. At each node, RL-RPC selects the best route and the best power to forward packets. When the deadline included in the packet can no more be satisfied, the packet is dropped. Optimizing the routes results in lower delivery delays and controlling the transmission power results in less interference between nodes and consequently in higher throughput.

CQLAODV (Cognitive Q-Learning AODV) – [68] proposed an RL-based routing protocol, which uses Bayesian Network (BN) for weight tuning. Route discovery procedure is the same as in AODV. CQLAODV requires each node to maintain two tables: Q-value table and BN table. The latter stores prior and conditional probabilities. When a node has a packet to forward, it searches its Q-values and BN tables to select the best forwarder.

R-CRS$^{(NATG)}$ *(RL-based Cooperative Relay Selection)* – It is an RL-based protocol to select relays in cooperative wireless networks [69]. As nodes (sources, relays, and base station) interfere the ones with the others, the number of active relays should be optimized. To achieve the best SER (Symbol Error Rate), learning is based on reward, which is defined as the improvement in SNR (Signal-to-Noise Ratio) when some relays are selected. Periodically, relays are selected among the set of relays in the network. At the end of each relay selection period, RL is used to evaluate the current relay subset and decide (or not) to change relays. Notice that there is a single agent in the system, which optimizes the selection of relays.

FQLAODV (Fuzzy constraint Q-Learning AODV) – [70] proposed a fuzzy and RL based routing protocol to consider link and topology changes in VANETs. Nodes include a localization module (e.g., GPS). Because of node mobility in VANETs, link quality fluctuates in time. By using fuzzy logic, FQLAODV can handle imprecise and uncertain link-state information of wireless links. Link-state information (including available bandwidth and relative speed of nodes) of nodes is periodically exchanged with neighborhood through Hello packets. Nodes also infer link quality from the strength of received signals. Offline, a fuzzy table is defined according to three metrics: available bandwidth (large, medium, small), mobility factor (slow, medium, fast), and link quality (bad, acceptable, good, very good). Example of entries in fuzzy table: ''When Bandwidth is Large, Mobility is Slow and Link quality is Medium THEN Rank is Good''. Online, whenever a neighbor j is selected by node i for forwarding, node ii estimates the three metrics (bandwidth, mobility factor, and link quality) associated with neighbor j and fetches the fuzzy table, which provides the rank associated with the 3-tuple and uses the returned rank to update its Q-values table.

FQ-R (Fault-tolerant Q-Routing) – [71] proposed FQ-R, a Q-routing extension, to consider node mobility in MANETs under space free propagation model. The proposed protocol is proactive and aims at anticipating link breaks due to the mobility of nodes. Quality of a link between two nodes depends on the distance between nodes, their velocity and direction, and their respective neighborhood. The proposed link availability metric aims at providing information on how link quality evolves with node moves, which enables to anticipate link breaks. Using link quality metric in reward function is the main contribution of FQ-R. Control packets including link-state information of nodes (i.e., position, velocity, direction) is periodically broadcast. In addition, any neighbor, which receives a packet, is requested to send back an ACK packet including its offered reward. Consequently, neighboring nodes know the link quality perceived by the packet sender.

d-AdaptOR (Adaptive Opportunistic Routing)– In [72], an opportunistic routing protocol is proposed to route data packets in ANETs using RL. d-AdoptOR can be specialized to consider different forms of link costs and metrics to optimize (e.g., hop count, delivery delay or energy). The basic idea of d-AdoptOR is similar to the one of [64]: first, the sender broadcasts its data packet, then the receiving neighbors acknowledge with an ACK packet including the estimated cost to deliver the packet at destination, and finally the sender selects the neighbor with the lowest cost and designates it as the next forwarder. The main contribution of [72] is the proof of convergence to optimality of the algorithm.

LR-WIV$^{(NATG)}$ *(RL-based Routing WSN Interacting with moving Vehicles)* – [73] proposed a RL-based protocol to consider routing from sensor nodes to moving sinks, which are vehicles. Vehicle groups on the roads are organized into VANETs and the leading vehicle of each group can disseminate information received from sensor nodes to other vehicles in its group. LR-WIV protocol aims at improving network performance including network lifetime, energy, delivery rate (i.e., reliability), and time delays. In LR-WIV protocol, when a vehicle (i.e., mobile sink) enters in a zone monitored with sensors, it announces itself with a control packet. The latter is rebroadcasted once by all nodes in the relevant zone of WSN. Each WSN node rebroadcasting a vehicle announcement packet includes its link-state information (i.e., remaining hop count to destination, residual energy, and perceived link quality) in the packet before broadcasting. Such a dissemination process of sink announcement enables all sensor nodes in the zone of interest to collect information and estimate the forwarding cost. In addition, whenever a sensor node forwards a data packet to its successor on path, it adds to packet its link-state information and the Q-value associated with the selected successor. Management of sink mobility is achieved through periodical sink announcements enabling sensor nodes to refresh their Q-values, and consequently their routing tables. RL modeling of LR-WIV protocol follows Q-routing principle, with an exception regarding its reward function and exchange of state information between nodes (i.e., during sink announcement phase and sensed data forwarding to sink as previously described).

PFQ-AODV (Portable Fuzzy constraint Q-learning AODV) – [74] proposed an extension to FQLAODV [70], which does not require GPS in VANETs. Both protocols have the same Q-value updating rule, but differ in how mobility and link quality are estimated. In FQ-AODV, link quality is associated with the received signal strength and Mobility factor is associated with the relative speed and distance between nodes. In PFQ-AODV, link quality is associated with the ratio of Hello packets received from each neighbor to the total received Hello packets. Mobility metric is associated with the change in node density in neighborhood at one and two hops.

RLBDR (Reinforcement Learning-based Distributed Routing) – [75] proposed a routing protocol to enable host nodes access the Internet through gateways in wireless mesh network (WMN). A WMN is composed of host nodes (i.e., sources and destinations of data), mesh routers (providing multi-hop forwarding to reach gateways), and gateways connected to the Internet. Gateways periodically broadcast advertisement messages including their estimated load enabling

source nodes to select the most appropriate gateway. To minimize the delivery delay and the loss ratio, the objective of RLBDR is to first select a gateway with low load. It should be noticed that in WMN context, path from the selected gateway to the final destination is not addressed as it depends on routing in the Internet, only paths to exit the WMN matter. Consequently, gateways are seen as destinations in RLBDR and learning is applied to optimize paths from sources to gateways. Reward function is designed to address link quality (including interference between neighbors and congestion).

PQDR (Predictive Q-learning based Deflection Routing) – [76] proposed an RL-based routing protocol, which is an extension to Q-routing, to optimizing the selection of alternate paths in case of contention at optical switches. It is similar to RLDRS protocol. Both protocols differ in their reward calculation and exchanges between a packet sender and the next hop. In PQDR, the reward is defined as a function of the number of remaining hops to destination as estimated by the sender, the number of hops to destination as estimated by next hop, and blocking probability of links. In each time window, link probability blocking table is updated by counting the number of successfully transmitted packets and the number of discarded packets for each outgoing link.

R-EAR$^{(NATG)}$ (RL-based Energy-Aware Routing) – [77] proposed another RL-based protocol routing to optimizing network lifetime and delivery delay in MANETs. Only feedback (i.e., cost) between a packet sender and the next forwarder is collected. Cost in selecting a next neighbor is based on link delay and energy consumption. Although there are only two metrics (i.e., link delay and energy consumption), R-EAR cost function is based on five parameters (two weights associated with metrics and three parameters associated with a logistic function, which is applied to energy consumption). Thus, R-EAR would not be easy to tune in practice.

EQR-RL (Energy-aware Qos routing RL-based) – It is a routing protocol for energy consumption optimization in WSNs, while providing soft delivery delay guarantees [78]. Periodic Hello packets are broadcast and each data packet in the network incorporate the delivery delay estimate and the residual energy of the sender. This information enables the sender neighbors to update their local information regarding the sender. In addition, each data packet includes its delivery delay requirements. Packets are discarded at any relaying node when delivery delay constraint cannot be satisfied anymore. Next hop selection is probability-based. Each neighbor is associated with a dynamic selection probability, which is calculated using three weighted metrics: link delay, ratio of packets between packet sender and the selected forwarder, and residual energy of selected forwarder.

QGrid (Q-learning-based Grid routing) – [79] proposed a special RL-based routing protocol composed of two phases: offline learning of Q-values and online use of the Q-value table to forward packets in VANETs. Network deployment region is divided into grids depending on the preferred granularity of the user. QGrid performs macroscopic (i.e., selection of grid) and microscopic (i.e., selection of vehicle in grid) routing. Q-values are associated with movements between neighboring grids, which reflect how vehicles enter and exit grids. From the history of inter-grid movements, one can infer optimal paths composed of grids to cross in order to reach a destination located in a given grid. Each vehicle stores a Q-values table calculated offline, but does not update the table. Whenever a vehicle has a packet (whose destination is not in the current grid) to forward, it uses the Q-values table to infer the next grid to use. Then, it forwards the packet to one neighbor, which belongs to the next grid; if no neighbor is in the next grid, it selects a neighbor which is closer than it to the destination and forwards the packet; if no neighbor is found, the packet is kept until the next forwarding opportunity (i.e., when new neighbors are discovered). QGrid protocol assumes that the online vehicle movements are close to the ones observed offline to build the Q-values table.

GR-PCCN$^{(NATG)}$ (Game theory and RL based Power Control in Cognitive Networks) – [80] proposed another protocol to optimizing transmission (TxT) power control in cooperative wireless networks and consequently save energy and reduce interferences between nodes. Periodically, after receiving link-state information from neighboring nodes, each node updates the probability of selection for each one of its TxT power levels taking into account TxT power levels of neighbors. GR-PCCN objective is that source nodes should adjust their TxT power levels to receive a bandwidth close to their minimum requirements, while relays should adjust their TxT power levels to maximize the amount of relayed traffic. First, game theory [81] is used to allocate transmission powers to cooperating relays. Then, RL is used to enable cooperating relays (agents) to converge to Nash equilibrium points, which result in optimal power allocation.

SMART (SpectruM-Aware cluster-based RouTing) – It is another routing protocol for cognitive radio networks. It is a cluster-based protocol [82]. Initially, each SU (Secondary User) scans available channels, while listing to other SUs, which may invite it to join their clusters. After scanning channel availability, SU decides either to join an existing cluster or to act as cluster head and invite other nodes to join its new cluster. Whenever a SU receives multiple invitations to join clusters, it selects the cluster with the highest number of available channels. Channel availability is the main metric used in RL to select actions (Join an existing cluster or Create a new cluster). Cluster maintenance is achieved through periodic join messages (to include new members) and merging clusters, which have a number of common channels satisfying a fixed threshold. Probability of OFF-state of channels is assumed known a priori.

QSGrd (Q-Smart Gradient based routing protocol) – [83] proposed a hybrid protocol, which combines Q-learning and transmission gradient, for optimizing energy consumption in WSNs. The probability of transmission success depends only on the distance between nodes and the maximum transmission range. Associating transmission success probabilities to neighbors yields in transmission gradient. Then, those

probabilities are used to update Q-values, which results in an RL-Gradient combination. RL is used to learn and select the best paths based on residual energy of the next hop and the average least number of transmissions to sink. Three types of packets are used: data packets, ACK packets, and Status packets. Status packet includes ENTxT (estimated number of transmissions) to reach the sink from the sender. Whenever a node receives a status packet, it updates its ENTxT value using its old ENTxT value, the received ENTxT value, and its highest probability of transmission success and broadcasts a status packet including its updated ENTxT.

QAR (Q-learning Adaptive Routing) – [84] proposed a Q-learning based protocol suitable for SDN architectures. Addressed architectures consist of a hierarchy of four levels: super controllers (at the top), domain controllers, slave controllers, and switches (at the bottom and are in charge of data forwarding). Each domain controller is in charge of routing inside its domain. Whenever a data packet has to travel through different domains, the super controller is in charge of the routing between domains. Slave controllers provide access to switches and receive port-status messages from them; they also can provide some simple control functions, such as traffic admission control, flow or congestion control, to share control workloads with domain controllers. To address QoS provisioning, QAR proposed a reward function based on multiple QoS metrics such as delay, loss rate, and available bandwidth. QAR protocol determines the path inside each domain for the respective domain controller and the global forwarding direction among domains for the super controller. Indeed, whenever a new flow arrives to a switch, the latter forwards the first packet of the flow to its domain controller and requests a forwarding path. Depending on the destination of the new flow, two cases are to be considered: i) source and destination nodes are in the same domain: the domain controller updates the current network state using state information from the salve controllers; it selects the best path using reinforcement learning to meet QoS requirements of the new flow, and requires modification of forwarding tables of switches along the selected path. ii) source and destination nodes are in different domains: the source domain controller sends the first packet to the super controller. The latter finds, using reinforcement learning, the forwarding direction among domains and sends the corresponding notifications to the domain controllers of involved domains. The same learning model is used by the super controller and the domain controllers. It is worth noticing that QAR provides a path on which all the packets of a flow will travel; i.e. there is no packet-per-packet forwarding decisions. Recall that in an SDN context, not all relaying nodes take forwarding decision, rather forwarding decisions are made only be controllers.

RLOR (RL Opportunistic Routing) – [85] proposed RLOR, which is a routing protocol for routing video streaming flows in wireless networks. To take into account the main requirement of video streaming, which is the delivery delay, RLOR proposed a rewarding function based on the delivery delay estimate. In RLOR protocol, a node, which has a data packet to forward, selects a subset of its neighbors and then broadcasts a packet including the data and a ranked list of selected neighbors; the list is ranked on the estimated delivery delay associated with each neighbor. Neighbors, which receive the packet, send an ACK to confirm their acceptation to forward. After receiving ACKs, the node holding the data packet selects one neighbor among responding nodes to forward the packet.

QGeo (Q-learning-based Geographic routing) – [86] proposed QGeo, which is an extension to Q-routing, to take into account mobility of unmanned robots. Hello packets are periodically broadcast to enable each node to refresh the GPS locations of other nodes. Then, nodes select next hop based on the geographic distance to destination. Consequently, distance metric is the main factor to guide routing.

QGR (Gain Q-learning based Routing) – In nowadays social networks and device-to-device communications, people may exchange commercial contents, such as coupons relating to products or services. A device may be interested in forwarding commercial content packets to receive some profit. Reference [87] proposed an approach to develop coupon dissemination mechanisms—through delay tolerant networks—that can maximize a coupon's economic gain in the presence of costs/rebates for Device-to-Device forwarding incentives. In particular, a gain metric is proposed to design RL-based routing protocol useful in commercial content distribution. In QGR protocol, the next hop to forward a commercial content (CC) packet is driven by the expected gain, which depends on the interest of the next hop in forwarding CC packet. Q-value in QGR refers to the global gain of a node and the reward refers to immediate gain. The selection of the next hop for forwarding depends on the interest of forwarders and their social interactions and on the packet content to advertise and its deadline.

IQ-L (Interest Q-Learning) protocol) – [88] proposed a Q-learning based protocol to forward 'interest' packets in named data networks, while overcoming the deficiency of interest packet flooding strategies in use. In IQ-L forwarding, Q-value refers to how delay-efficient is the selection of a node to forward an interest packet. Q-value design relays on the duration between the time a node sends an interest packet to one of its neighbors and the time its receives feedback from the selected neighbor. In addition, each node collects the results (success or failure in forwarding a packet interest) for each neighbor. Whenever requested data is sent by the node storing the content to the requesting user, a success ACK packet is forwarded upstream to inform the nodes which relayed the interest packet. In addition, each relaying node sets a timer (with a value equals to the interest expiry time), which enables to detect failures in delivering the content to requesting user. Maintained success and failure counts enable the nodes to derive the appropriateness probability in selecting neighbors to fetch a specific content.

AQFE (Adaptive Q-routing Full Echo) – [89] proposed an extension to Q-routing to improve exploration according to traffic load. Two learning parameters are used in AQFE: the

basic learning parameter α and an additional learning parameter, called α_2, which depends on average and maximum delivery delay. Each node i, which is ready to make a forwarding decision, updates the Q-value of selected neighbor j using the basic learning parameter α and the Q-values of the other neighboring nodes using the additional parameter α_2. Notice that Q-values are updated even though nodes are not selected; the objective is to adjust a balance between exploration and exploitation. The modified Q-value update rule provides faster reaction to changes in Q-value tables of neighbors; consequently, nodes are forced to select other possible paths especially when the average delivery delay increases.

DLTPC (Distributed Learning-Theoretic Power Control) – In [90], DLTPC RL-based algorithm is proposed for power control in energy harvesting (EH) networks. DLTPC considers two-hop networks where the sources broadcast their data to relays, which transmit them to a base station. EH relays (such as solar powered relays) power themselves from energy sources that are present in their environment. They accumulate the harvested energy for subsequent usage to transmit packets. DLTPC provides a RL and game theory based solution to decide when and how (i.e., adjust the transmission power) to use harvested power by relays in order to optimizing delivery delay. The challenge addressed by DLTPC is the game theory-based collaboration between relays in such a way that they collectively optimize transfer delays, while efficiently using the energy they accumulate. Notice that the objective of DLTPC is similar to the one of GR-PCCN [80], but DLTPC is more complex, because it considers energy harvesting, which is dynamic and random in nature.

RL-budget – In clustered networks, cluster heads close to the sink may require a higher energy consumption than cluster heads on the border. To balance energy consumption between cluster heads, [91] proposed a protocol to adjust the size (in number of members) of clusters in radio cognitive networks. Periodically, a cluster head broadcasts a message including the available channels (representing the budget) to invite nodes to join its cluster. Using RL, the cluster head adjusts its cluster size depending on the available radio channels and the energy consumption due to packet reception from cluster members and relaying to other cluster heads or to the sink.

GFRLR$^{(NATG)}$ (Game theory and Fuzzy logic based RL Routing) – [92] proposed an efficient algorithm mixing game theory, fuzzy logic, and reinforcement learning to optimize MAC collisions in VANETs when traffic information are exchanged between vehicles and RSU (roadside units). Hello packets broadcast by nodes include Q-values and mobility information (location, speed and direction). First, fuzzy logic is used to select cluster heads (CL) taking into account the relative mobility and movement patterns associated with vehicles. To reduce the number of sender nodes, and consequently reduce the collisions, when traffic information is disseminated to vehicles on road, nodes closer to RSUs are given preference to act as CLs. Second, game theory is used to stimulate cooperation of sender nodes to use the CLs for packet forwarding. Third, reinforcement leaning is used to evaluate multi-hop routes. Q-values are updated upon reception of Hello packets, which helps nodes to select the appropriate routes. Q-value update rule is based on signal quality factor (to give preference to nodes with better signal conditions), collision probability, and distance to RSUs.

RLSRP (RL based Self-Routing Protocol) – [93] proposed RLSRP to address rapid topology changes in FANETs composed of flying nodes, such as drones. Flying nodes exchange, periodically or on-demand, their link-state information including GPS position, speed, and direction. Then, RL is used to estimate future positions of flying nodes and to select forwarders to relay packets accordingly.

CCLBR (Congestion Control-based Load Balanced Routing) – [94] proposed a routing protocol to efficiently forward query packets in peer-to-peer networks. CCLBR objective is load balancing between peers under dynamic loads. In order to avoid sending queries to congested peers, RL is used to monitor the state—which includes the processing the capacity, number of queries being processed, and the number of monitored resources—of each peer and to decide which peers are most appropriate to relay queries.

Q2-R (Qos-aware Q-Routing)– [95] proposed an adaptation to Q-routing to optimize the overhead and the quality of discovered paths, while providing soft QoS in WANETs. Q2-R follows three steps: Bootstrapping, Learning, and Data routing. In Bootstrapping step, a reactive protocol, such as AODV, is used for path discovery. In Learning step, a second packet stream, called ''Q-Info'' packets, is used to optimize previously discovered paths through RL, as in Q-routing. Consequently, in Q2-R, learning does not start with random paths (as in Q-routing), but with paths providing connectivity with the destinations and then improves them. Paths are optimized regarding required QoS constraints such as delivery delay, jitter, and loss rate. Paths, which do not fulfill QoS constraints are penalized when their Q-values are updated. Unlike all the other QoS-aware protocols, Q2-R does not use any weights associated with QoS metrics. Finally, in data routing step, the learning traffic rate is reduced or increased depending on the observed variations in Q-values.

SRR (Smart Robust Routing)– [96] proposed an RL-based algorithm for routing in tactical networks (i.e., military battlefield networks) where applications require deterministic guarantees on network performance to meet mission requirements. Through RL, nodes learn stable paths and use them to forward unicast packets. When the topology of the network becomes unstable, packets are duplicated and broadcast to reliably reach the destination through multiple paths. Q-learning technique is used to learn the number of hops to reach the destination and a metric, called C-factor, representing the likelihood to reach the destination, is used to determine when packet duplication is required. The C-factor, which represents path information freshness, is close to 0 when the path is very likely to be broken and close to 1 when it is very likely to be stable. C-Factor is updated using ACKs and it is used to compute the learning rate α. C-Factor was

also used in *CQ-R* [33] and called C-value. SRR and CQ-R differ in how C-Factor is updated. In SRR, C-Factor is used both for adjusting the learning rate and measuring the loss rate. When a node has a packet to forward, it determines the next neighbor with the lowest (because shortest path is needed) product, which combines the Q-value and C-factor. Then, it decides, with a probability based on C-factor of the selected neighbor, to either forward to the selected neighbor or to broadcast.

SDCoR (Software Defined Cognitive Routing) – [97] proposed a solution to integrate SDN (software defined networking) approach and RL to improve routing in Internet of vehicles. In the proposed protocol, called SDCoR, an SDN-controller collects information about slave nodes (vehicles) and performs management tasks. In particular, an SDN-controller adapts routing to the conditions of the network (i.e., mobility and traffic demand). Instead of learning optimal routes through RL as the other RL-based routing protocols, SDCoR protocol suggested another approach, which is the selection of a routing protocol among a list of candidate and then calculates the routing tables depending of the selected routing protocol and send them to SDN-switches, which have to follow received forwarding tables until SDN-controller updates them. Using RL, SDN controller collects information from switches and selects the most appropriate routing protocol.

V. CLASSIFICATION OF RL-BASED ROUTING PROTOCOLS

To our knowledge, this paper is the first to propose classification criteria to help understanding and comparing the whole RL-based routing protocols. Proposed criteria are categorized into three groups (Fig. 3):

- *Context of use related criteria*, which describe the targeted applications and their characteristics and requirements;
- *Design characteristics related criteria*, which highlight how authors designed their protocols to make them efficient and different from other protocols;
- *Performance related criteria*, which provide a qualitative evaluation of the overhead of protocols and the metrics analyzed by authors through simulations.

A. CONTEXT OF USE

The proposed routing protocols aimed at providing efficient solutions to deploy in specific contexts, which may be characterized by five aspects.

1) ADDRESSED NETWORK CLASSES

As previously mentioned in subsection III.2, RL-based routing protocols have been extended and improved gradually as networks evolved, spanning wired, wireless mesh, wireless cooperative communication, optical, mobile ad hoc, wireless sensor, vehicular, delay tolerant, flying ad hoc, social, cognitive radio, software defined, named data Networking, and peer-to-peer networks. In addition to network class, some protocols made strong assumptions about traffic (e.g., distribution function of traffic is known a priori) or about the network (e.g., localization service is available to nodes or transmission errors may occur).

2) ROUTING OPTIMIZATION CONTEXT

From users' perspective, routing protocols should select the best (optimal) paths to convey data from sources to destinations. There are different ways to reach, partially or totally, such a goal depending on roles assigned to data sources and to relaying nodes and on the initial assumptions about routing. RL-based contributions to routing addressed six routing optimization contexts:

Data-packet driven optimization. Sources transmit their data packets and each node on path to destination, which receives a data packet, selects the best next forwarder depending on its local view and then sends backward a feedback. After a given amount of forwarded data packets, the routing process converges to the selection of (sub)optimal paths. This framework is prevailing due to its full distribution of roles and flexibility. However, its efficiency (speed of convergence to stable optimal paths) and overhead depend on traffic generated by sources.

Route request driven optimization. A node, which has data to send or which requests data from sources, sends a Route Request (RR). Then the RR is disseminated in the network. Each node receiving an RR decides to participate (or not) and selects the next node to continue the process of route request forwarding until the final destination is reached. Once a path is found, all packets from source to destination are routed on this path. Then, when data packets are transmitted, final destinations (and maybe intermediate nodes) send feedback regarding the performance of current routes. Then, routes are optimized until their optimum is reached or approximated. It is worth noticing that most RL-based routing protocols proposed in this category ([32], [35], [36]–[38], [43], [59], [66], [68], [70], [74]) are extensions to the well-known AODV protocol [39].

Content request driven optimization. In peer-to-peer systems and named data networks, nodes interested in a content send their requests to receive data packets from the nodes possessing the requested content. Nodes on path forward (or not) the requests to locate the requested content. Then, when data packet containing the requested content are forwarded, relaying nodes receive feedback and adapt their paths to keep only paths that enable to return the maximum content at a minimum cost [87], [88], [94].

Predefined routes driven optimization. Offline, each source builds a list of paths to reach the targeted destinations. Online (i.e., during data packet transmission), when a source has packets, it selects a path among predefined ones and sends its data packets; all relaying nodes must use the selected route. In the event a source detects a link break on the selected path, it switches to another predefined path. Periodically or on-demand, a feedback is sent backward to the source, which will adapt (i.e., optimize) its path selection among the predefined path list. Notice that in such a framework,

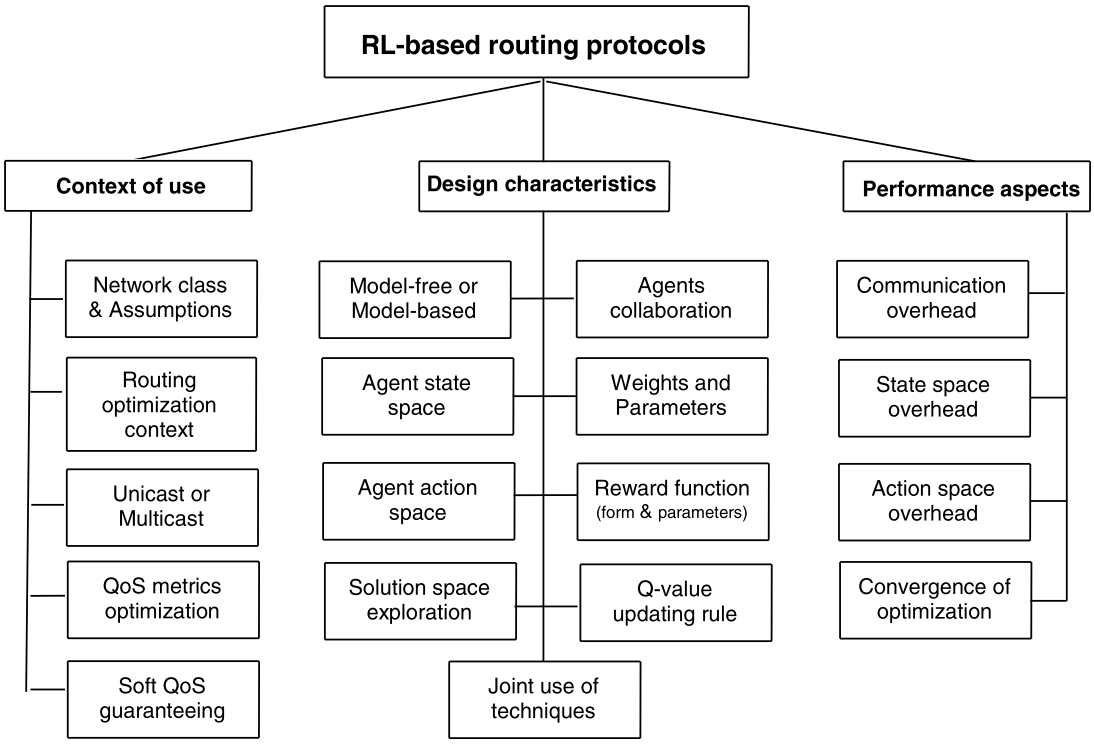

FIGURE 3. Classification criteria.

the cost optimization role is assigned to source nodes. Consequently, the efficiency of learning depends on the quality of predefined path lists (i.e., number of hops, path and link redundancy...) [34], [57], [60].

Cluster driven optimization. Clustering is defined as the process of partitioning the set of nodes into groups, with one cluster head per group. Clusters heads also may be divided into groups of cluster heads with one cluster head representing the group. Data packets are transmitted from sources to destination following a clustered hierarchy of the network (i.e., nodes, which are members of a cluster, send their data packets to their cluster head, which in turn sends the packets to a higher level cluster head or to destination). Depending on its available resources, a cluster head determines the number of members that can join the cluster. Then, following the transmission of data packets from members, cluster heads receive feedback and adjusts (i.e., optimizes) their cluster size accordingly [82], [91].

Routing protocol driven optimization. It is a very recent application of RL to improve routing performance. In this framework, a central node (an SDN-controller), has a set of routing protocols candidates (e.g., AODV, DSDV, DSR, OLSR, GPSR...) that can be used for forwarding packets by slave nodes (i.e., SDN-switches). In each time interval, the central node selects a routing protocol and calculates the routing tables and sends them to slave nodes to configure their forwarding tables. Then, a feedback is collected by the central node regarding the performance of computed routing tables followed by the slave nodes. The central node may change the current routing protocol for the next time interval depending on the observed performance. After some time intervals, the system converges to the most adequate routing protocol [97].

3) UNICAST OR MULTICAST

RL has been applied to select and optimize either Unicast or Multicast paths. It is worth noticing that optimization of multicast trees requires much more time and communications to reach optimal trees than for unicast paths. In addition, in case some links on paths are not sufficiently stable (because of congestions or wireless link instability, for example), the convergence to optimal trees would never happen. RL should be applied for multicasting when links are sufficiently stable and/or when partial delivery is allowed (i.e., when only a fraction of potential destinations may receive the broadcast packets).

4) QOS METRICS FOR OPTIMIZATION

Broadly speaking, routing problems in networks are typically multicriteria decision making (MCDM) problems. The well-known difficulty of MCDM problem solving comes from the heterogeneity natures of metrics. To search routing solutions, the user must decide—depending on the specificities of his/her application—to assign an importance level to each metric. Consequently, the MCDM solving approaches are based on parameters (or weights) to express the relative importance of metrics. Manifold QoS metrics have been addressed as objectives for optimization in RL-based routing protocols, including:

- *Delivery delay.* It also is called *Delivery time* or *End-to-end delivery delay* and represents the average time to deliver a packet at destination.
- *Delivery ratio*. It also is called *Delivery rate* and it is the proportion of packets successfully delivered at destination.
- *Hop count*. It is the average number of hops from source to destination. Protocols proposed in [50], [53], [56] use 'hop' as unit for Delivery delay instead of time unit (in sec).
- *Loss ratio.* It also is called (*Packet*) *Loss rate* and it is the proportion of packets not delivered at destination.
- *Symbol error rate*. It is similar to *bit error rate*. It considers transmission symbols instead of bits.
- *Light-Path Blocking probability*. It is the percentage of the blocked light-paths of all requests in optical network. When a light-path is blocked, arriving transmission requests are rejected. Light-path blocking probability metric is similar to Loss ratio.
- *Bandwidth*. It is the average bandwidth (in bits/sec) provided to sources.
- *Throughput*. It is the average amount of bytes (or packets) delivered in the entire network per time unit.
- *Path stability*. It indicates how a path between source and destination changes over time. Reference [71] selects links such that paths remain stable as long as possible, because discovering new paths takes time and results in packet loss while new paths are being searched.
- *Energy consumption*. It is the average energy consumption due to transmissions, receptions, and processing. Energy consumption may be related to a packet, to a node, to a group of nodes or to a network as a whole.
- *Network lifetime*. This metric is of paramount importance in wireless sensor networks. It indicates the average time over which the network is alive, with multiple meanings: all the nodes are alive, a certain ratio of nodes are alive, at least one node is alive, all node batteries are above a given threshold... [45], [47], [48] [57], [64], [73], [77].
- *Transmission power.* In this case, optimization of transmission means selecting the lowest power for transmission while providing acceptable performance regarding other metrics (e.g., bandwidth or delivery delay). It results in energy saving and interference reduction [80].
- *Load balancing*. It is used mainly in peer-to-peer networks and mesh networks to balance the load between nodes relaying traffic [75], 94].
- *PU-SU interference* (*ratio*). It is a metric used in cognitive radio networks and it indicates how Primary users (PU) are prevented from transmitting by secondary users (SU) [75], [82], [91].
- *Interest Satisfaction delay*. It also is called *Hit delay* and it is the average delay to return requested data in peer-to-peer and named data networks [88].
- *Interest Satisfaction ratio*. It also is called *Hit rate* and it is the proportion of satisfied requests in peer-to-peer and named data networks [87].
- *Gain* or *Revenue*. It is the average revenue (in $, ...) received by the agent when routing is seen from a business point of view and routing should result in a profit [34], [60].
- *Overhead*. It represents the average cost (in terms of control packets, retransmissions...) to deliver data packets at destination.
- *Generic metrics*. Some protocols ([35], [41], [55], [61], [78], [80], [84], [85], [95]) do not address specific metrics.

5) QOS GUARANTEEING

Few routing protocols aimed at providing QoS guarantees, mainly regarding delivery delay to meet the requirements of delay-sensitive applications, such as multimedia applications, [54], [62], [67], [78], [84], [85], [95].

It is worth noticing that only soft QoS guarantees may be provided, because:

- In the beginning of learning, any path (i.e., random path) may be selected to send data packets and consequently it is unlikely to provide the required QoS levels. Many data transfers are required before acceptable QoS level is reached through learning. In QoS-constraint-aware RL-based protocols, when the required level of QoS is not met, packets are dropped at any forwarding node, which detects QoS violation.
- Even when learning has reached stable paths, some packets may be dropped, unless a mechanism of permanent resource reservation is deployed.

Table 1 categorizes RL-based protocols from context of use perspective.

B. DESIGN CHARACTERISTICS

1) LEARNING MODEL

In reinforcement learning, two classes of learning may be used, model-based and model-free. In model-based learning, the agent has a model of the environment, which guides state transitions. In the field of routing, a few algorithms are model-based [32], [36], [42], [48], [62], [63], [64], [80], [82], [87], [88]. Some of them use offline-collected information regarding environment model, while others calculate and improve the environment model online. For example, QGR and IQ-L protocols [87], [88] use the nodes' level of interest collected offline to select paths. SMART [82] uses the probability of channel availability to select paths. Model-based learning is more efficient, in term of convergence to optimal solution, than model-free learning. However, the required knowledge about the environment is hard (or intractable) to collect in many categories of networks and applications.

2) AGENT STATES AND ACTIONS SPACES

Applying RL to any optimization problem requires the definition of states and actions spaces. Both features are

TABLE 1. Context of use.

Protocol (year) ref.	Network Class & Assumptions	Routing optimization context	Routing: Unicast/ Multicast	QoS metrics optimization	Soft Constraint guaranteeing
Q-routing (1994) [9]	WiredN	DP ([1])	Unicast	Delivery delay	–
PQ-R (1996) [31]	WiredN	DP	Unicast	Delivery delay	–
ARL-R (1997) [32]	WiredN	RR ([2])	Unicast	Delivery delay	–
CQ-R (1998) [33]	WiredN	DP	Unicast	Delivery delay	–
CACR-RL (1998) [34]	WiredN	SR ([3])	Unicast	Operator revenue	Link capacity
Q-MAP (2002) [35]	ANET	RR	Multicast	Generic metric	–
GAPS-R (2002) [36]	WiredN	RR	Unicast	Delivery delay	–
Q-LMRWA (2003) [37]	ON	RR	Multicast	Light-Path Blocking	Light-Path blocking
RL-AODV (2004) [38]	MANET	RR	Unicast	Delivery delay	–
MARL-R (2004) [40]	MANET	DP	Unicast	Delivery delay	–
MQR (2005) [41]	WiredN	DP	Unicast	Generic metric	–
CRL-SAMPLE (2005) [42]	MANET	DP	Unicast	Delivery ratio, Throughput, Overhead	–
RLGAMAN (2005) [43]	MANET	RR + DP ([4])	Unicast	Bandwidth, Delivery delay	–
RLCF (2006) [44]	WSN	DP	Unicast	Energy	–
AdaR (2006) [45]	WSN	DP	Unicast	Network lifetime([5])	–
RLGR (2007) [47]	WSN ([6])	DP	Unicast	Delivery ratio, Network lifetime([7])	–
Q-PR (2007) [48]	WSN ([8])	DP	Unicast	Delivery ratio, Network lifetime([9])	–
SERLR (2007) [49]	ANET	DP	Unicast	Energy([10])	–
FROMS (2007) [50], [51]	WSN ([11])	DP	Multicast	Hop count	–
E-FROMS (2008) [53]	WSN ([12])	DP	Multicast	Energy, Hop count	–
RL-QRP (2008) [54]	WSN ([13])	DP	Unicast	Delivery delay, Delivery ratio	Delivery delay
MRL-QRP (2008) [55]	WSN	DP	Unicast	Generic metrics	Generic QoS
RLDRS (2008) [56]	ON	DP	Unicast	Loss ratio, Hop count	–
RL-BER (2008) [57]	MANET	SR	Unicast	Network lifetime ([14])	–
SQ-R (2009) [58]	CRN	DP	Unicast	Delivery delay	–
QoS-RSCC (2009) [59]	WN	RR	Unicast	Delivery ratio	Generic QoS
RL-CAC (2010) [60]	ON	SR	Unicast	Revenue	–
RRD-R (2010) [61]	WN	DP	Unicast	Generic metrics	–
QDTR (2010) [62]	DTN ([15])	DP	Unicast	Delivery delay, Energy	Delivery delay
ARBR (2011) [63]	DTN	DP	Unicast	Delivery delay and ratio, number of TxT	–
QELAR (2010) [64]	WSN ([16])	DP	Unicast	Deliv. delay, Energy, Net. lifetime ([17])	–
SNL-Q (2010) [65]	MANET	DP	Unicast	Delivery ratio	–
QLMAODV (2011) [66]	MANET ([18])	RR	Unicast	Delivery delay, Delivery ratio, Energy	–
RL-RPC (2011) [67]	WN	DP	Unicast	Delivery delay, Delivery ratio	Delivery delay

[1] DP: Data Packet-driven routing optimization.
[2] RR: Route Request-driven routing optimization.
[3] SR: Static Route-driven routing optimization.
[4] In RLGAMAN, RL is used on a route-request-basis, while Genetic algorithm is used on a per-data-packet-basis.
[5] Lifetime is not explicitly defined.
[6] Localization service is available. Sink has unlimited energy capacity.
[7] Network lifetime is defined as "the time when the energy of one node falls below a threshold".
[8] Localization service is available.
[9] Network lifetime is defined as the time when there is no route to reach the sink node.
[10] Energy optimization is addressed as an example of metric, but the proposed algorithm is generic and may be applied to any metric.
[11] Initial hop counts to all sinks are known in advance.
[12] Same assumption as in FROMS.
[13] Localization service is available. Application domain targeted is biomedical.
[14] Lifetime means "until the first node dies or its battery is below a given threshold".
[15] Nodes are deployed in underwater environment suffering signal attenuation. As nodes move at low speed, path disconnection times are longer than those experienced in terrestrial networks. Network is represented by a layered topology in a 3-D area.
[16] All nodes are static in an underwater environment.
[17] Two Network lifetime definitions are used: "time when the first sensor node dies" or "time when the network gets partitioned".
[18] Transmission environment follows a Space free propagation model.

TABLE 1. *(Continued.)* **Context of use.**

Protocol (year) ref.	Network Class & Assumptions	Routing optimization context	Routing: Unicast/ Multicast	QoS metrics optimization	Soft Constraint guaranteeing
CQLAODV (2012) [68]	MANET	RR	Unicast	Delivery delay, Delivery ratio	–
R-CRS (2012) [69]	CCWN	DP	Unicast	Symbol error rate	–
FQLAODV (2012) [70]	VANET	RR	Unicast	Delivery delay, Delivery ratio	–
FQ-R (2012) [71]	MANET ([19])	DP	Unicast	Path stability	–
d-AdaptOR (2012) [72]	ANET	DP	Unicast	Generic cost	–
LR-WIV (2013) [73]	WSN, VANET ([20])	DP	Unicast	Delivery delay, Delivery ratio, Network lifetime ([21]), Energy	–
PFQ-AODV (2013) [74]	VANET ([22])	RR	Unicast	Delivery delay, Delivery ratio	–
RLBDR (2013) [75]	WMN	DP	Unicast	Deliv. delay, Loss ratio, Load balancing	–
PQDR (2013) [76]	ON	DP	Unicast	Loss ratio	–
R-EAR (2014) [77]	MANET ([23])	DP	Unicast	Delivery delay, Network lifetime ([24])	–
EQR-RL (2014) [78]	WSN	DP	Unicast	Energy	Delivery Delay
QGrid (2014) [79]	VANET ([25])	DP	Unicast	Delivery delay	–
GR-PCCN (2015) [80]	CCWN ([26])	DP	Unicast	Transmission Power	Bandwidth
SMART (2015) [82]	CRN ([27])	CL ([28])	Unicast	Delivery ratio	–
QSGrd (2016) [83]	WSN	DP	Unicast	Delivery delay, Energy	–
QAR (2016) [84]	SDN ([29])	IRR ([30])	Unicast	Delivery delay, Loss ratio	Delivery delay, Loss ratio
RLOR (2017) [85]	WN	DP	Unicast	Delivery delay, Loss ratio	Delivery delay
QGeo (2017) [86]	MANET ([31])	DP	Unicast	Delivery delay, Delivery ratio	–
QGR (2017) [87]	SDTN ([32])	CD ([33])	Multicast	Gain	–
IQ-L (2017) [88]	NDN	CD	Unicast	Interest Satisfaction delay	–
AQFE (2017) [89]	MANET	DP	Unicast	Delivery delay	–
DLTPC (2017) [90]	CCWN ([34])	DP	Unicast	Delivery delay	–
RL-budget (2018) [91]	CRN ([35])	CL	Unicast	PU-SU interference	–
GFRLR (2018) [92]	VANET	DP	Unicast	MAC layer contentions	–
RLSRP (2018) [93]	FANET	DP	Unicast	Delivery delay, Delivery ratio	–
CCLBR (2018) [94]	P2PN ([36])	CD	Unicast	Load balancing	–
Q2-R (2018) [95]	ANET	DP ([37])	Unicast	Delivery delay, Jitter, Delivery ratio	Delivery delay, Jitter, Loss rate
SRR (2018) [96]	MANET ([38])	DP	Unicast	Loss rate, Hop count	–
SDCoR (2018) [97]	SDN ([39])	SP ([40])	([41])	Delivery delay, Delivery ratio	–

[19] No localization service is available.
[20] Sinks are mobile.
[21] Lifetime means "until the first node dies or its battery is below a given threshold".
[22] No localization service is available.
[23] Localization service is available. All nodes are homogeneous in terms of transmission power and battery capacity.
[24] Lifetime not explicitly defined.
[25] Localization service is available. Message destination does not move during the process of message delivery.
[26] There are multiple sources and a single destination. The network setup is such that all sources are able to meet their requirements.
[27] All Primary Users using a channel follow the same ON and OFF durations of the respective channel. Channels are noiseless.
[28] CL: CLuster-driven routing optimization.
[29] The network is structured into a hierarchy of SDN-controllers (Central controller, domain controllers…).
[30] IRR: Implicit Route Request driven routing optimization. After the first data packet is sent by a switch to SDN-controller, a route is established and all upcoming packets follow the established route.
[31] Localization service, such as GPS, is available.
[32] All nodes contribute to routing in an unselfish manner. Node interest profiles and their underlying social interactions are known.
[33] CD: Content Delivery-driven routing optimization.
[34] There are multiple sources and a single destination interconnected via a set of relays. All relays are one-hop from the destination.
[35] Perfect channel sensing, synchronization among secondary users, and availability of a common control channel are assumed.
[36] Time interval between two successive queries is the same for all peers in the network.
[37] In Q2-R, learning is not done for each sent data packet as for all other DP oriented protocols, but it only applies to a subset of data packets called Q-Info to reduce the overhead of the protocol.
[38] MANETs used in battlefield (i.e., tactical networks).
[39] Nodes at the bottom of SDN hierarchy are vehicles operating in Internet-of-vehicle context.
[40] SP: Standard Protocol-driven routing optimization.
[41] SDCoR does not select nodes or paths; rather, it selects a routing protocol (such as AODV or GPSR) that nodes must follow. Paths pre-established by the SDN-controller may be Unicast, Multicast or Broadcast depending on the selected routing protocol.

fundamental to specify the role(s) and dynamic behavior of learning agents. In the reviewed RL-based routing protocols, multiple forms have been proposed for modeling agent's states and actions. More specifically, state space may be:

- Set of nodes (i.e., the current state of agent is the index of node holding the packet); notice that this form of state modeling is the most popular in RL-based routing protocols.
- Set of grids (i.e., the current state of agent is the number of grid holding the packet) in grid-organized networks [79].
- Set of couples relating to the dynamics of nodes, for example, in VANETs, a couple is a vehicle speed class and context of move (urban, highway…), [49], [97].
- Set of paths and their characteristics (queue length, average delay, average delivery ratio, average energy consumption…) [34], [50], [53], [57], [67], [90].
- Set of QoS levels required by flows [59].
- Set of transmission power levels [80].
- Set of available channels at cluster head [91].
- Set of available wavelengths in optical networks [60].
- Set of packet states (i.e., Packet in buffer, Packet delivered, Packet broadcast…) [63], [65].

Action space is a set of single-type actions or a set of actions of different types. Cardinality of action space is the same as the one of entities associated with actions, e.g., when actions are associated with node selection, action space cardinality equals the number of nodes. Single-type actions take one of the following forms:

- "Select node *j* as next hop and forward packet", which is the most popular form of action. In this case, the action space is the set of node Ids.
- "Select a subset of neighbors *sn* and broadcast packet" [69], [85]. In this case, the action space is the set of partitions of Node Id set.
- "Select output link *l* and transmit packet" [86]. In this case, the action space is the set of links.
- "Select grid *g* and send packet to one of nodes in *g*" [79]. In this case, the action space is the set of grids.
- "Select predefined path *pth* and send packet along *pth*" [57], [60]. In this case, the action space is the set of predefined paths.
- "Allocate *m* free channels" [91]. In this case, the action space is the set of channels.
- "Select a transmission power *Pw*" [80]. In this case, the action space is the set of transmission power levels.
- "Select a protocol *Prt* among a list of (standard) routing protocols and configure the network with *Prt*", which is an action form used in SDN networks. In this case, the action space is the set of standard protocols.

Multiple-type action sets take one of the following forms:

- "Select node *j* as next hop and forward packet", "Broadcast packet", "Deliver packet" or "Keep packet". Keep packet is used only in DTNs [42]. In this case, the action space is the set of node Ids plus three special actions.
- "Accept call on predefined path *pth*" or "Reject call" [34]. In this case, the action space is the set of node Ids plus three special actions.

3) SOLUTION SPACE EXPLORATION

In machine learning, the learner tries to improve the current solution while switching between exploration and exploitation of the solution space. Consequently, the speed of convergence to optimal solution is directly dependent on how the exploitation and exploration are designed. In RL-based routing algorithms, six selection forms are used to handle the solution space:

- *Greedy selection.* Only the highest Q-value is used for selection; it is the simplest method to implement. Unfortunately, it is known that greedy selection alone may take a long time before convergence or never converge.
- *ε-greedy selection*. In addition to greedy selection, the learner uses a small amount of randomness (with ε probability) to explore new solutions. ε-greedy selection is the most used form of selection in the reviewed protocols.
- *Probability based selection*. A probability calculated from the history of learning is used to guide selection [32], [50], 53], [64], [78], [80], [82], [83], [88]. Some RL-based routing protocols use Boltzmann probability distribution [38], [42], [65], [67], [68], [77].
- *Bayesian network decision selection*. Action selection uses a powerful approach, which is Bayesian nets, to better explore the solution space 70].
- *Devaluation of solutions based selection*. Q-values are either periodically decayed or a confidence level (which decreases in time) is associated with Q-values in order to enforce exploration [41].
- *New neighbors first selection.* New discovered nodes are favored in next hop selection. Such a selection approach is particularly useful in mobile networks where mobility results in break of old paths, which have high Q-values, and new neighbors may quickly provide more efficient paths [40].

4) AGENTS COLLABORATION

In basic RL, each agent is independent and interacts with its environment. Using its link-state information, the agent applies an action to the environment, it receives a reward, and then it changes its state. In applications of RL to routing, almost all proposed protocols are based on collaborating agents, which involves not only reward but also exchanges of link-state information without actions undertaken from RL point of view. Indeed, in addition to reward, agents exchange link-state information (such as estimated end-to-end delay, node location, link quality, and residual energy) with their neighbors to select actions. From an RL point of view, selecting a next hop is an RL action, while receiving periodic Hello packets is not. Agent collaboration is useful to categorize routing protocols. More specifically, reviewed

RL-based routing protocols may be categorized into three classes:

- *No collaboration*. Either there is a single agent (for example located on an SDN controller) or there are multiple agents, which make decisions only based on their local view. No collaborative agent model has been used mainly in the earlier protocols such as [34], [57] and in centralized algorithms [69], [84], [91], [97].
- *Reactive collaboration*. When the selected neighbor receives a data packet, either it directly returns its feedback (i.e., its Q-value or reward, depending on protocols) in the ACK packet or it includes its link-state information in each data packet when it forwards it, thus providing feedback to previous sender. As shown on Table 2, the half of reviewed protocols are reactive.
- *Proactive collaboration*. In addition to sending (directly or indirectly) their feedback upon reception of a packet, nodes periodically (or on demand) broadcast their link-state information in Hello packets (which may include Q-values, distances, locations, residual energy and so on, depending on considered protocols) to their neighbors. Link-state information broadcasting enables agents to update their information used in metrics calculation and/or update their routing tables (i.e., their Q-value table) without taking RL actions. As shown on Table 2, the third of reviewed protocols are proactive.

5) HYBRIDATION WITH OTHER OPTIMIZATION TECHNIQUES

Most of RL-based routing algorithms are pure reinforcement learning. Some algorithms combine RL and other optimization techniques to speed up convergence. In particular, Gradient method [83], Game theory [80], [90], [92], Fuzzy logic [70], [74], [92], Bayesian networks [48], [68], Least square policy iteration [45], Neural networks [38], Genetic algorithms [43], and Ants optimization [32] have been used to improve exploration, thus providing powerful RL-based routing algorithms.

6) NUMBER OF PARAMETERS TO TUNE

Broadly speaking, users prefer tools (including optimization tools), which are easy to tune, while providing high performance. When RL is used, the values of two parameters—learning factor and discount rate—are frequently provided to the learning system. In addition, when QoS metrics and network-related metrics (such as distance between nodes, frequency of node moves, and network density) are of concern, weights are associated with each metric or with each group of metrics. Thus, most RL-based routing protocols require setting of multiple parameters and weights.

From the user perspective, tuning the weights used in reward functions may be (very) difficult. Without a clear understanding of the proposed reward function, the setting of learning algorithms to converge quickly to optimal would be impossible. In addition, the user should fix a trade-off between the variety of metrics to consider and the quality (in terms of optimality) of solutions (i.e., paths to route packets). Examples of protocols requiring much setting effort include [55], [59], [68], [77], [80], [84].

Table 2 categorizes RL-based routing protocol and highlight their design characteristics. The notation "$n_1|n_2|n_3$" is used as follows: $n_1 = 1$ means only RL learning factor (i.e., α) is used, $n_1 = 2$ means both RL learning factor and discount rate (i.e., γ) parameters are used; n_2 denotes the number of additional parameters relating to the weights of (QoS) metrics used in RL model, and n_3 denotes the number of parameters used in space exploration.

7) REWARD FUNCTIONS

It is worth noticing that, as conclusion from our review, the most distinctive feature of existing RL-based routing protocols is their reward function. Arguments of reward functions are metrics, which mainly include hop count, distance, mobility factor, residual energy of node, average energy of neighborhood, number of available channels, available link bandwidth, link delay, path delay, congestion level, signal strength, density of neighborhood, and transmission success rate. Metrics values are often estimated. Some RL-based routing algorithms only include generic reward functions with generic arguments [35], [36], [61], [93]. To be used in real context, such algorithms must be specialized depending on metrics of interest. Reward functions may be categorized into three classes: Test-based, linear and nonlinear functions.

Test-based reward functions– They are the simplest form of reward. Reward value takes a constant value depending on test [37], [45], [66], [70], [71], [72], [74], [79]. The most common test is "is the packet delivered to destination?". For example, reward equals 1 when a packet is delivered to destination and 0 otherwise.

Linear reward functions– They have the following form:

$$R = fc() + \sum_{k=1}^{H} \omega_k * M_k$$

where HH denotes the number of metrics; M_k and ω_k denote the k^{th} metric value and weight, respectively. Some reward functions include a function fc returning a constant (for example $fc() = 1$, if Ack received, $fc() = 0$, if neither Ack nor NAck received, and $fc() = -1$, if NAck received). There are more linear reward-function based protocols than nonlinear reward-function based protocols (see Table 3).

Nonlinear reward functions – They are designed with different forms of combinations of metrics. The following reward functions are representative examples of the state-of-the-art:

- Reward function used in E-FROMS [53], which is an energy-aware multicast routing protocol, is: $Rwd = HC(s') * E(s')$. s' represents the next forwarder node. $HC(s')$ denotes the sum of hop counts from node s' to destinations reachable through node s' and $E(s')$ denotes the minimum battery cost for routes cross-

TABLE 2. Solution design characteristics - Model, States, Actions, Collaboration, and Parameters.

Protocol	Model Based/ Free	Agent states	Agent actions	Solution Space exploration	Agents collaboration	Other technique used	Weights/ parameters
Q-routing [9]	Free	SNS ([42])	ANS ([43])	G ([44])	Reactive	–	1 | 0 | 0
PQ-R [31]	Free	SNS	ANS	G	Reactive	–	1 | 2 | 0
ARL-R [32]	Based	SNS	ANS	BP ([45]) + ε-G	Proactive	Ants opt.	1 | 0 | 1
CQ-R [33]	Free	SNS	ANS	([46])	Reactive	–	1([47]) | 0 | 0
CACR-RL [34]	Free	([48])	([49])	G	No ([50])	–	2 | 0 | 0
Q-MAP [35]	Free	SNS	ANS	G	Proactive	–	2 | 0 | 0
GAPS-R [36]	Based	SNS	ANS	Gradient	Reactive ([51])	–	NS | 0 | 1
Q-LMRWA [37]	Free	SNS	ANS	ε-G	Reactive	–	2 | 0 | 1
RL-AODV [38]	Free	SNS	ANS	BzPD ([52])	Reactive	Neural nets	1 | 0 | 1
MARL-R [40]	Free	SNS	ANS	G + NN (53)	Reactive	–	1 | 0 | 0
MQR [41]	Free	SNS	SNH&FP (54)	G + Dev(55)	Reactive	–	2 | 0 | 1
CRL-SAMPLE [42]	Based	(56)	SNH&FP	BzPD	Proactive (57)	–	2 | 2 | 1
RLGAMAN [43]	Free	SNS	ANS	G	Reactive	Genetic algo.	1 | 0 | * ([58])
RLCF [44]	Free	NS ([59])	RD ([60])	([61])	No ([62])	–	1 | 0 | 3
AdaR [45]	Free	Node	SNH&FP	ε-G	Reactive	LSPI ([63])	2 | k | 1
RLGR [47]	Free	SNS	ANS	ε-G	Proactive	–	2 | 4 | 1
Q-PR [48]	Based	SNS	{SNH&FP, KP([64])}	ε-G	Reactive	Bayesian decision	1 | 1 | 1
SERLR [49]	Free	([65])	SNH&FP	ε-G	([66])	Gradient, Game theory	1 | k | * ([67])
FROMS [50], [51]	Free	([68])	Set of SNH&FP	PB	Reactive	–	1 | 0 | 1
E-FROMS [53]	Free	See FROMS	See FROMS	See FROMS	Reactive	–	1 | 1 | 1
RL-QRP ([54]	Free	Node	SNH&FP	G	Proactive	–	2 | 0 | 0

[42] SNS: state not explicitly specified by authors. Implicitly, a state is a node number.
[43] ANS: action not explicitly specified by authors. Implicitly, an action is SNH&FP.
[44] G: Greedy.
[45] PB: Probability-Based.
[46] Level of confidence in Q-values is used to explore new subspaces.
[47] A decay function is used to periodically change (i.e., decay) Q-values instead of using discount rate γ.
[48] A state is a list of paths and, for each path, the number of admitted calls and their service classes.
[49] Actions are "Accept a new call on a predefined path" and "Reject a new call".
[50] Connection admission control is achieved by a centralized solution (i.e., there is a single agent).
[51] Reward comes in the form of an ACK distributed in the network once the packet has reached its destination.
[52] BzPD: Boltzmann Probability Distribution.
[53] NN: Encourage selection of Newly discovered Neighbors.
[54] SNH&FP: action is "Select Next Hop and Forward Packet".
[55] All Q-values are periodically devaluated, using a forgetting factor, to reinforce exploration.
[56] State space is 3 states = {Packet in queue, Packet successfully sent, Packet delivered to destination)
[57] The solution is based on a genuine multi-agent paradigm (with collaboration in decisions).
[58] Other parameters are used by Generic algorithm.
[59] NS: state space not explicitly defined. The learning power relies on the mechanisms of action selection and not on the agent state.
[60] RD: action is "Retransmit" or "Drop" packet.
[61] Action selection is based on three mechanisms: Constrained propagation, Differential delay, and Probabilistic retransmission.
[62] Decision is made using only local information and received data packet.
[63] LSP: Least square policy iteration.

TABLE 2. *(Continued.)* **Solution design characteristics - Model, States, Actions, Collaboration, and Parameters.**

Protocol	Model B/F	Agent states	Agent actions	Sol. Space exploration	Agents collaboration	Other technique used	Weights/ parameters
MRL-QRP [55]	Free	Node	SNH&FP	ε-G ([69])	Proactive	–	2 | ([70]) | 1
RLDRS [56]	Free	SNS	ANS	ε-G	Reactive	–	1 | 0 | 1
RL-BER [57]	Free	([71])	([72])	G	No ([73])	–	0 | 3 | 0
SQ-R [58]	Free	SNS	SNH&FP	G	Reactive	–	1 | 0 | 0
QoS-RSCC [59]	Free	LQ ([74])	RS ([75])	ε-G	Reactive	–	2 | 2 + ([76]) | 1
RL-CAC [60]	Free	([77])	([78])	ε-G	Reactive	–	2 | 0 | 1
RRD-R [61]	Free	SNS	ANS	RDM ([79])	Reactive ([80])	RDM	1 | 0 | 0
QDTR [62]	Based	Node	{SNH&FP, KP}	G	Reactive	–	2 | 2 | 0
ARBR [63]	Based	(81)	SNH (82)	PB	Proactive (83)	–	(84)
QELAR [64]	Based	Node	{SNH&FP, KP}	PB	Proactive	–	2 | 4 | 0
SNL-Q [65]	Free	(85)	{F, B, D} (86)	BzPD	Proactive	–	2 | 2 | 1
QLMAODV [66]	Free	SNS	ANS	G	Proactive	–	2 | 0 | 0
RL-RPC [67]	Free	(87)	SNH&TP (88)	G	Proactive	–	(89)
CQLAODV [68]	Free	SNS	ANS	BzPD	Proactive	Bayesian nets	2 | 3 | 1
R-CRS [69]	Free	(90)	(91)	G	(92)	–	1(93) | 0 | 0
FQLAODV [70]	Free	SNS	ANS	BN	Proactive	Fuzzy logic	2 | 3 | 0
FQ-R [71]	Free	Node	SNH&FP	ε-G	Proactive	–	2 | 0 | 1
d-AdaptOR [72]	Free	Node	SNH (94)	ε-G (95)	Reactive	–	1 | 0 | 1
LR-WIV [73]	Free	Node	SNH&FP	G	Proactive	–	2 | 3 | 0
PFQ-AODV [74]	Free	SNS	ANS	FB	Proactive	Fuzzy logic	2 | 3 | 0
RLBDR [75]	Free	SNS	ANS	ε-G	Proactive	–	1 | 3 | 1
PQDR [76]	Free	SNS	SOL&TP (96)	G	Reactive	–	1 | 1 | 0

[64] KP: action is Keep Packet (and do not forward it).
[65] A state is a set of values of estimated metrics associated with each neighbor.
[66] No explicit exchanges between nodes are provided in the paper, which only focuses on generic estimators.
[67] k denotes the number of metrics and many other parameters are used in stochastic estimation of parameters and functions.
[68] A state is a set of sinks and, for each sink, the set of paths finishing at that sink.
[69] In exploration, ε is not a fixed parameter; it is locally calculated by each node depending on change in number of its neighbors.
[70] Number of weights associated with Q-values of neighbors.
[71] A state is a path with, for each link, the energy consumption and battery level.
[72] Action is "Select a path among multiple paths".
[73] It is a source routing protocol. All needed link information for routing are collected by the source.
[74] State space is a set of QoS levels (QL) associated with data sessions.
[75] RS (Relay Selection): action is "Select a relay among a set of candidates".
[76] Plus weights assigned to neighbors (see [55]).
[77] A state is a list of couples <Number of available wavelengths for each optical fiber, Service class of new call>.
[78] Action is "Select a path among predefined paths & forward packet".
[79] RDM: Russian dolls method.
[80] Only the reward is returned backward from destination to source along the path.
[81] State space is 3 states = {Packet in buffer, Packet successfully unicast to a neighbor, Packet delivered at destination}
[82] Action set of "Delegate the forwarding of the packet to one of the current neighbors".
[83] The solution is based on a genuine multi-agent paradigm (with collaboration in decisions).
[84] ARBR does not use learning and discount rates. It uses parameter for weighting inter-encounter time between nodes.
[85] State space is 3 states = {Packet in buffer, Packet successfully unicast to a neighbor, Packet delivered at destination}
[86] Action set is {select next neighbor and forward packet (F), Broadcast packet (B), Deliver packet to destination (D)}.
[87] A state is a set of the waiting queue length at all nodes and a set of status of all channels.
[88] SNH&TP: An action is a couple "Select next hop and select a transmission power".
[89] RL-RPC uses two parameters and weights associated with nodes on path to destination.
[90] State is "Total number of relays forming the relaying backbone in the network".
[91] Action is "Select a subset of relays among the set of relay candidates".
[92] There is a single agent.
[93] Only a discount rate is used.
[94] After reception of ACK from neighbor nodes, one node is designated as the next forwarder.
[95] ε is variable and depends on the number of packets received by the neighbors of the node making the action selection.
[96] SOL&TP: Action is "Select an Output Link and Transmit Packet".

TABLE 2. *(Continued.)* **Solution design characteristics - Model, States, Actions, Collaboration, and Parameters.**

Protocol	Model B/F	Agent states	Agent actions	Sol. Space exploration	Agents collaboration	Other technique used	Weights/ parameters
R-EAR [77]	Free	(97)	SNH&FP	BzPD	Reactive	–	2 | 5 | 1
EQR-RL [78]	Free	SNS	ANS	PB	Proactive	–	1 | 0 | 3
QGrid [79]	Free	Grid	SG&TP (98)	(99)	–	–	1+1(100) | 0 | 0
GR-PCCN [80]	Based	(101)	STP (102)	PB	Proactive	Fuzzy logic	(103)
SMART [82]	Based	Node	SNH&FP	PB	Reactive	–	1 | 0 | 0
QSGrd [83]	Free	SNS	ANS	PB	Reactive	Gradient	2 | 3 | 0
QAR [84]	Free	SNS	ANS	ε-G	No (104)	–	2 | 8 | 1
RLOR [85]	Free	Node	SSN&BP (105)	G	Reactive	–	1 | 0 | 0
QGeo [86]	Free	Node	SNH&FP	G	Proactive	–	2 | 0 | 0
QGR [87]	Based	Node	SNH&FP	G	Reactive	–	2 | 0 | 0
IQ-L [88]	Based	SNS	ANS	PB + (106)	Reactive	–	1 | 0 | 2
AQFE [89]	Free	SNS	ANS	G	Reactive	–	1+1(107) | 0 | 0
DLTPC [90]	Free	(108)	STP	G	No	Game theory	(109)
RL-budget [91]	Free	(110)	(111)	G	No (112)	–	1 | 0 | 0
GFRLR [92]	Free	Node	SNH&FP	G	Proactive	Game theory + Fuzzy logic	2 | 0 | 0
RLSRP [93]	Free	SNS	ANS	G	Proactive	–	2 | 0 | 0
CCLBR [94]	Free	Node	SNH&FP	G	Reactive	–	2 | 1 | 0
Q2-R [95]	Free	SNS	SNH&FP	ε-G	Reactive	–	1 | 0 | 1
SRR [96]	Free	SNS	SNHB(113)	ε-G + PB (114)	Reactive	–	1 | 1 | 1
SDCoR [97]	Free	(115)	SP (116)	ε-G	No (117)	–	2 | 2 | 1

[97] A state is a list of neighbors and with each neighbor are associated Distance and Battery level.
[98] SG&TP: Action is "Select a grid and transmit packet".
[99] Selection of action is dependent on the offline learning process (e.g., use of history of nodes movements to select actions).
[100] Discount rate γ is dynamic.
[101] A state is a TxT (Transmission) power level.
[102] STP: Action is "Select a transmission power level".
[103] High number of parameters because of variable learning and discount rates and other parameters due to game theory use.
[104] No Q-value exchanged between agents, because of centralization of SDN.
[105] SSN&BP: Select a Subset of Neighbors and Broadcast Packet.
[106] Probability-based plus heuristic "give more chance to New Neighbors to enhance Exploration".
[107] A second learning rate (α_2) is used.
[108] A state in DLTPC agent (i.e., a relay) is 3-tuple <Buffer state, Channel state, Energy level>.
[109] High number of parameters because of variable learning and discount rates and other parameters due to game theory use.
[110] A state is the number of available channels at a cluster head.
[111] An action is an "allocation of *m* available channels to cluster members".
[112] Each cluster head acts as an independent agent and only invites nodes (which are not agents) to join its cluster.
[113] SNHB: actions are "Select Next Hop and transmit" or "Broadcast".
[114] SRR combines ε-greedy and probability (based of confidence level) to select actions.
[115] A state is represented by a couple Vehicle density and vehicle speed range (e.g., s1 = "urban density, 0-40 kmph").
[116] SP (Select Protocol): an action to perform by SDCoR agent is the selection of a routing protocol (such as AODV).
[117] There is a single agent on the SDN controller, which interacts with its environment composed of moving vehicles.

TABLE 3. Solution design characteristics - Reward and Q-value updating functions.

Protocol	Reward function	Parameters of reward function	Q-Learning Compliance	Q-value update rule of non-Q-learning compliant algorithms ($Ng(j)$ denotes the neighbor set of node j)
Q-routing [9]	Linear	TxT delay, Local queueing delay	Yes (γ=1)	
PQ-R [31]	Linear	As Q-routing + Delay variation	Yes (γ=1)	
ARL-R [32]	Linear	Generic link cost	Modified Q-L	$Q_i(d,j) = Q_i(d,j) + \alpha * f(Rwd)$
CQ-R [33]	Linear	As Q-routing + Confidence level	Yes (γ=1) ([118])	
CACR-RL [34]	Generic	Revenue	No	This algorithm combines temporal difference TD(0) learning and decomposition to solve optimization problem.
Q-MAP [35]	Generic	Generic	Yes	
GAPS-R [36]	Generic	Generic	No	No Q-value updating rule included in paper.
Q-LMRWA [37]	Test-based	Next hop address	Yes	
RL-AODV [38]	Linear	Delivery delay	Yes	
MARL-R [40]	Linear	As Q-routing	Yes	
MQR [41]	Linear	Generic	Modified Q-L	$Q_i(d,j) = (1-\alpha) * Q_i(d,j) + Rwd + \gamma * \max_{k \in Ng(j)} Q_j(d,k)$
CRL-SAMPLE [42]	Nonlinear	TxT success	No	$Q_i(s,a) = f(arguments)$ ([119])
RLGAMAN [43]	Generic	Generic	Yes (α =1)	
RLCF [44]	Generic	Generic	Yes (γ=1)	
AdaR [45]	Test-based	Next hop address	No	This algorithm uses Least Squares Policy Iteration learning method to find the optimal Q-values.
RLGR [47]	Nonlinear	Hop count, Residual energy	Yes	
Q-PR [48]	Linear	Number of TxT	Yes (γ=1)	
SERLR [49]	Nonlinear	Generic	Yes (γ=0)	
FROMS [50], [51]	Linear	Hop count	Yes (γ=0)	
E-FROMS [53]	Nonlinear	Hop count, Residual energy	Yes (γ=0)	
RL-QRP [54]	Nonlinear	Distance, Link delay	Yes	
MRL-QRP [55]	Nonlinear	Transmission delay, Link delay	Modified Q-L	$Q_i(d,j) = (1-\alpha) * Q_i(d,j) + \alpha * [Rwd + \gamma * (weight * \max_{k \in Ng(j)} Q_j(d,k) + ClbFact)$ ([120])
RLDRS [56]	Nonlinear	Hop count, Loss probability	Modified Q-L	$Q_i(j,d) = (1-\alpha) * Q_i(j,d) + \alpha * f(arguments) * Rwd$
RL-BER [57]	Nonlinear	Energy	Modified Q-L	$Q(s,a) = Average(Rwd\ of\ pathP)$ ([121])
SQ-R [58]	Linear	Available channels	Yes (γ=0)	
QoS-RSCC [59]	Nonlinear	([122])	Modified Q-L	Q-value update rule is identical to the one of [55]
RL-CAC [60]	Generic	Revenue	Modified Q-L	$Q(s,a) = (1-\alpha) * Q(s,a) + \alpha * [Rwd - AvgRwd + \max_{b \in \mathcal{A}}\{Q(s',b)\}]$ ([123])
RRD-R [61]	Generic	Generic	Yes (γ=0)	
QDTR [62]	Linear	([124])	Yes	
ARBR [63]	Nonlinear	Channel and buffer availability times	No	Q-value update rule is not reported because of its complexity (five formulas are used)
QELAR [64]	Nonlinear	([125])	Yes	
SNL-Q [65]	Nonlinear	Transmission success	No	SNL-Q is based on the general RL paradigm. Authors did not describe how RL optimization problem has been solved.
QLMAODV [66]	Test-based	Next hop address	Modified Q-L	$Q_i(d,j) = (1-\alpha) * Q_i(d,j) + \alpha * \left(Rwd + \gamma * f(arguments) * \max_{k \in Ng(j)} Q_j(d,k)\right)$
RL-RPC [67]	Linear	([126])	No	RL-RPC is based on the general RL paradigm.

[118] α is function of confidence level associated with Q-values.
[119] Delivery-at-destination reward, transmission success probability, Q-value of selected neighbor, and node selection probability are arguments of $f(\ )$.
[120] *ClbFact* (collaboration factor) is a weighted sum of Q-values of all neighbor nodes in current epoch.
[121] An action means selection of a path among a list of predefined paths.
[122] Parameters are Transmission success, ACK and NACK reception times, and Average transfer between relay and destination.
[123] *AvgR* is the average of received rewards. An action means "Selection of a path among a list of predefined paths".
[124] Parameters are Distance, Residual energy of node, and Residual energy of node group.
[125] Parameters are Residual energy of node and node group, Probability of forwarding, and Probability of packet holding.

TABLE 3. ***(Continued.)*** **Solution design characteristics - Reward and Q-value updating functions.**

Protocol	Reward function	Parameters of reward function	Q-Learning Compliance	Q-value update rule of non-Q-learning compliant algorithms
CQLAODV [68]	Linear	Delivery delay, SINR, bandwidth	Yes	
R-CRS [69]	Nonlinear	SNR, Hop count	Yes (α=0)	
FQLAODV [70]	Test-based	Next hop address	Modified Q-L	$(Q_i(d,j) = (1-\alpha) * Q_i(d,j) + \alpha * f(arguments) * [Rwd + \gamma * \max_{k \in Ng(j)} Q_j(k,d)]$
FQ-R [71]	Test-based	Next hop address	Yes	
d-AdaptOR [72]	Test-based	Next hop address	Yes (γ=1)	
LR-WIV [73]	Linear	Hop count, Energy, Signal strength	Yes	
PFQ-AODV [74]	Test-based	Next hop address	Modified Q-L	$Q_i(d,j) = (1-\alpha) * Q_i(d,j) + \alpha * f(arguments) * [Rwd + \gamma * \max_{k \in Ng(j)} Q_j(d,k)]$
RLBDR [75]	Nonlinear	Interference and congestion levels	Yes (γ = 0)	
PQDR [76]	Nonlinear	([127])	Yes (γ=0)	
R-EAR [77]	Linear	([128])	Yes ([129])	
EQR-RL [78]	Linear	Delivery delay, Link delay	Yes (γ=0)	
QGrid [79]	Test-based	Next hop address	Yes	
GR-PCCN [80]	Nonlinear	([130])	Yes ([131])	
SMART [82]	Linear	Available Channels	Modified Q-L	$Q_i(d,j) = (1-\alpha) * Q_i(d,j) + \alpha * [\min(P_{OFF}(i,j,d), \max_{k \in Ng(j)} Q_j(d,k))]$ ([132])
QSGrd [83]	Nonlinear	Residual energy, Progress to destination	Modified Q-L	$Q_i(d,j) = (1-\alpha) * Q_i(d,j) + \alpha * [\gamma * [\max_{k \in Ng(j)} Q_j(d,k) + (1-\gamma) * Rwd * Psucc_i(d,j)]$ ([133])
QAR [84]	Linear	([134])	Yes	
RLOR [85]	Nonlinear	([135])	Yes (γ=0)	
QGeo [86]	Nonlinear	Distance, Link delay	Yes	
QGR [87]	Nonlinear	Economic factors, Interest in contents	Modified Q-L	$Q_i(d,j) = (1-\alpha) * Q_i(d,j) + \alpha * \max[Rwd, \gamma * \max_{k \in NS(j)} (Prob(j,k) * Q_j(d,k)]$ ([136]), ([137])
IQ-L [88]	Linear	Response delay	Yes (γ=1)	
AQFE [89]	Linear	As Q-routing	Adapted Q-L (γ=0)	Two learning rates, α and α_2, are used; α for updating Q-value associated with selected next hop and α_2 for non-selected nodes.
DLTPC [90]	Linear	Buffer size	Modified Q-L	Q-value update rule is based on both RL and gradient
RL-budget [91]	Nonlinear	Available channels	Yes (γ=0)	
GFRLR [92]	Test-based	Next hop address	Modified Q-L	$Q_i(d,j) = (1-\alpha) * Q_i(d,j) + \alpha * f(arguments) * [Rwd + \gamma * \max_{k \in Ng(j)} Q_j(d,k)]$
RLSRP [93]	Generic	Generic	No	RLSRP is based on the general RL paradigm. Authors did not describe how RL optimization problem has been solved.
CCLBR [94]	Nonlinear	Node connectedness and congestion	Modified Q-L	$Q_i(d,j) = (1-\alpha) * Q_i(d,j) + \alpha * [Rwd + \gamma * \max_{k \in Ng(j)} Q_j(d,k)] + f(arguments)$
Q2-R [95]	Linear	As Q-routing	Modified Q-L	$Q_i(j,d) = f(arguments) * (1-\alpha) Q_i(j,d) + \alpha * Rwd$
SRR [96]	Linear	Hop count	Yes (γ=0) ([138])	
SDCoR [97]	Linear	Delivery delay, delivery ratio	Yes	

[126] The immediate reward is the fraction of packets delivered at destination within delay deadline.
[127] Parameters are Hop count, Selection probability, and Delivery delay.
[128] Parameters are Delivery delay, Link delay, Energy consumption, and Battery level.
[129] A variable learning ratio $\alpha(s, s', t)$ is used instead of a constant one, α.
[130] Parameters are Channel capacity and Transmission power.
[131] Learning rate and discount factor are variable in time.
[132] $P_{OFF}(i,j,d)$ is the OFF-state probability of the bottleneck channel on route to d via j.
[133] $Psucc_i(j,d)$ is the selection success probability of node j by node i.
[134] Parameters are Link delay, Link loss ratio, and Available link bandwidth.
[135] Parameters are Link delay and Node selection probability.
[136] $Psucc(b)$ denotes the probability of success of action b, which means the probability that next forwarder will meet another node.
[137] $NS(j)$ denotes the set of nodes that node j has seen so far and has the possibility to meet again.
[138] α is function of C-Factor (i.e., confidence level) associated with Q-values.

ing node s' and finishing at destinations reachable via node s'.

- Reward function used in R-CRS [69], which minimizes the number of active relays, is: $Rwd = \frac{SNR_{s'} - SNR_s}{|NR_{s'} - NR_s|} SNR_x$ and NR_x denote Signal-to-Noise Ratio and number of active relays when relay x is activated, respectively.
- Reward function used in RL-BER [57], which is an energy-aware protocol, is:

$$Rwd = \left(E_{min}(s')\right)^{\omega_1} * \left(BC(s')\right)^{-\omega_2} * (B_{init})^{\omega_3}.$$

In RL-BER, a state is a path. $E_{min}(s')$, $C(s')$, and B_{init} are the minimum battery level and the energy consumption on path s', and is the initial battery level, which is identical for all nodes, respectively. ω_1, ω_2, and ω_3 are weights.

- Reward function used in RL-QRP [54], which provide soft end-to-end delay guarantees, is:

$$Rwd = \left(\frac{Dist_{s',d} - Dist_{s,d}}{Dist_{s,d}}\right) \Big/ \left(\frac{T_{s,s'}}{DelReq}\right).$$

s, s', and ds denote the node holding packet, the next forwarder, the final destination, respectively. $Dist_{x,y}$ and $T_{x,y}$ denote distance and transfer delay between nodes x and y, respectively. $DelReq$ denotes end-to-end delay requirement.

8) Q-VALUE UPDATING RULE FORMS

Recall that learning consists in updating Q-values associated with couples of < state, action > until optimal Q-value is reached. When routing is of concern, optimal solution means optimal path regarding metrics of interest. Over half of proposed RL-based routing algorithms are direct applications of Q-learning (QL) proposed by Watkins [13] to learn optimal solution. In some QL-compliant routing algorithms, discount factor is set to 0, which results in myopic learning, or set to 1, which results in full-future-aware learning. Often, when routing is of concern, nodes represent states and actions are next forwarder selections. Consequently, the original QL Q-value updating rule (7) is rewritten in routing-related literature as follows:

$$Q_i(d,j) = (1-\alpha) * Q_i(d,j) + \alpha * [Rwd + \gamma * \max_{k \in Ng(j)} Q_j(d,k)] \quad (11)$$

In (11), i, j, and Rwd represent s_t, s_{t+1}, and R_{t+1} in (7), respectively. Action a_t is "select node j to be next forwarder to destination d", which is simply written "d,j". $Ng(j)$ denotes the neighbor set of node j.

Whenever actions are not node selections (e.g., selection of a transmission range or a cluster size), rule (7) remains unchanged. The second half of proposed RL-based routing algorithms either use a modified QL Q-value updating rule or do not rely on Q-learning but on the general paradigm of RL [4], [12]. Whenever QL Q-value updating rule is modified, often a variable factor based on metrics of interest or on specific probabilities—which are denoted *f(arguments)* in Table 3—are added to QL Q-value updating rule [32], [42], [56], [66], [70], [74], [94]. The objective is to make the learning more sensitive to the added factor.

Table 3 categorizes RL-based routing protocols from reward function and Q-value updating rule perspectives.

C. PERFORMANCE ASPECTS

Performance includes protocol overhead and provided QoS. In networks, two overhead factors are generally addressed as they have effects on scalability of protocols: space overhead and communication overhead (i.e., control packet overhead).

1) COMMUNICATION OVERHEAD

From a qualitative point of view, communication overhead is categorized as:

- *Null*, when no exchange is required between agents as in [34], [57], [69], [79].
- *Low*, when the selected next hop returns a feedback (which may include some of its link-state information) in an explicit ACK packet or it includes its feedback when, in turn, it (re)forwards the packet. The half of reviewed protocols have a low or medium communication overhead (see Table 4).
- *Medium*, when the feedback from the destination is propagated to all hops through an explicit ACK packet.
- *High*, when nodes periodically exchange link-state information (such as Q-values, energy consumption, locations, so on). The amount of control packet needed depends on the period of Hello packets as in [32], [35], [42], [47], [53], [54], [55], [63], [64], [65], [66], [72], [91], [93], [96].

Notice that routing protocols with high communication overhead may be inefficient under some network conditions. For example, high frequency of Hello packets may result in interferences and collisions, which deteriorates network performance, and consequently the QoS provided by RL-based routing protocols.

2) STATE SPACE OVERHEAD

RL-based algorithms require memory to store the states of agents. In some RL-based applications, the number of states may be very high (or even infinite), thus jeopardizing the use of RL. Fortunately, space overhead has not been seen as a barrier to apply reinforcement learning in routing field. From a qualitative point of view, the state space overhead is categorized as:

- *Very low*, when state space is states of a packet as in [42], [44], [65].
- *Low* (or *medium*), when state space is node IDs. Most of reviewed protocols have a low or medium space overhead (see Table 4).
- *Limited*, when the state space depends on factor such that number of transmission power levels, maximum number of available channels, number of grids, number of available wavelengths, etc. as in [79], [80], [91].

TABLE 4. Performance of protocols.

Protocol	Overhead (packets)	Overhead (state space)	Overhead (action space)	Metrics in simulation (addressed by authors)
Q-routing [9]	Low	Low	Low-Med [139]	Delivery delay
PQ-R [31]	Low	Low	Low-Med	Delivery delay
ARL-R [32]	High	Low	Low-Med	Delivery delay, Hop count, Link throughput
CQ-R [33]	Low	Low	Low-Med	Delivery delay
CACR-RL [34]	Null [140]	High [141]	High [142]	Average revenue, Call rejection rate.
Q-MAP [35]	High	Low	Low-Med	[143]
GAPS-R [36]	Medium [144]	Low	Low-Med	Delivery delay
Q-LMRWA [37]	Low	Low	Low-Med	LP-Blocking probability
RL-AODV [38]	Low	Low	Low-Med	Delivery delay, Energy, Delivery ratio
MARL-R [40]	Low	Low	Low-Med	Delivery ratio
MQR [41]	Low	Low	Low-Med	Delivery cost, Overhead
CRL-SAMPLE [42]	High	Very low [145]	Low-Med	Delivery ratio, Overhead, Throughput
RLGAMAN [43]	High	Low	Low-Med	Delivery ratio, Throughput
RLCF [44]	Null [146]	Very low	Very low [147]	Energy, Delivery ratio
AdaR [45]	Low	Low	Low-Med	Delivery ratio
RLGR [47]	High	Low	Low-Med	Delivery delay, Delivery ratio, Net. lifetime
Q-PR [48]	Medium [148]	Low	Low-Med	Overhead
SERLR [49]	[149]	Very High [150]	Low-Med	Delivery ratio, Number of alive nodes
FROMS [50], [51]	Low	High [151]	Very high [152]	Delivery ratio, Energy, Overhead
E-FROMS [53]	Low	See FROMS	as FROMS	Hop count, Residual energy
RL-QRP [54]	High	Low	Low-Med	Delivery delay, Delivery ratio
MRL-QRP [55]	High	Low	Low-Med	Delivery delay, Delivery ratio
RLDRS [56]	Low	Low	Low-Med	Loss probability, Hop count
RL-BER [57]	Null [153]	Very High [154]	High [155]	Delivery ratio, Network lifetime, Energy
SQ-R [58]	Low	Low	Low-Med	Delivery delay
QoS-RSCC [59]	Low	Low [156]	Very high [157]	Delivery delay, Delivery ratio
RL-CAC [60]	Low	Medium [158]	High [159]	Call rejection ratio, Revenue
RRD-R [61]	Medium [160]	Low	Low-Med	Delivery ratio, Delivery delay
QDTR [62]	Low	Low	Low-Med	Delivery delay, Delivery ratio, Energy
ARBR [63]	Very high [161]	Very low [162]	Low-Med	Delivery delay, Delivery rate, Number of transmissions
QELAR [64]	High	Low	Low-Med	Delivery delay, Energy, Delivery ratio, Overhead

[139] Low-Med: Low to Medium complexity.
[140] Connection admission control is achieved by a centralized solution (i.e., there is a single agent).
[141] A state is a list of paths and, for each path, the number of admitted calls and their service classes.
[142] Action space depends on the number of predefined paths.
[143] Authors only analyzed Q-values convergence to optimal value by simulation.
[144] Feedback comes in the form of an ACK distributed in the network once the packet has reached its destination.
[145] State space = {Packet in queue, Packet successfully sent, Packet delivered to destination)
[146] No control packets are used.
[147] There are two actions Broadcast or Drop packet.
[148] Protocol includes two phases: invite a subset of neighbors and then designate a node to forward the packet.
[149] The paper does not address the protocol issues including control packet exchanges.
[150] State space is a set of couples <neighbor, set of metric estimate values associated with neighbor>.
[151] A state is a set of sinks and, for each sink, the set of paths terminating at that sink.
[152] Action space is a set of combinations of paths to multiple destinations.
[153] It is a source routing protocol where the source decides, based only on its link-state information, which path to use.
[154] State space is a set of paths and each link is represented by a couple <energy consumption, battery level>.
[155] Action space depends on the number of paths.
[156] State space is a set of QoS levels, which is usually limited.
[157] Action space depends on the number combinations of relay set partitioning.
[158] State space is a set of couples <Number of available wavelengths for each optical fiber, Service class of new call>.
[159] Action space depends on the number of predefined paths.
[160] Cost due to sending reward by all destinations backward to sources.
[161] In each period, agents exchange their contact tables and Q-values.
[162] State space = {Packet in queue, Packet successfully sent, Packet delivered to destination)

TABLE 4. *(Continued.)* **Performance of protocols.**

Protocol	Overhead (packets)	Overhead (state space)	Overhead (action space)	Metrics in simulation (addressed by authors)
SNL-Q [65]	High	Very Low ([163])	Low-Med	Delivery ratio, Delivery cost
QLMAODV [66]	High	Low	Low-Med	Delivery Delay, Packet loss, Overhead
RL-RPC [67]	High	Very high ([164])	High ([165])	Delivery ratio (of packets with deadlines)
CQLAODV [68]	High	Low	Low-Med	Delivery delay, Delivery ratio, Overhead
R-CRS [69]	Null ([166])	High ([167])	High ([168])	Symbol error rate, Hop count
FQLAODV [70]	High	Low	Low-Med	Delivery delay, Delivery ratio
FQ-R [71]	High	Low	Low-Med	Throughput
d-AdaptOR [72]	Very High ([169])	Low	Low-Med	Transmissions per packet, Delivery ratio
LR-WIV [73]	High	Low	Low-Med	Hop count, Delivery ratio, Energy
PFQ-AODV [74]	High	Low	Low-Med	Delivery delay, Delivery ratio, Hop count
RLBDR [75]	High	Low	Low-Med	Delivery delay, Loss ratio, Throughput
PQDR [76]	Low	Low	Low-Med ([170])	Delivery delay, Loss ratio, Deflection ratio
R-EAR [77]	Low	High ([171])	Low-Med	Delivery delay, Energy
EQR-RL [78]	High	Low	Low-Med	Delivery delay, Delivery ratio, Residual energy
QGrid [79]	Null	Limited ([172])	High ([173])	Delivery delay, Delivery ratio, Hop count
GR-PCCN [80]	Very high ([174])	Limited ([175])	Low ([176])	Energy, Throughput
SMART [82]	Low	Low	Low-Med	Delivery ratio, PU-SU Interference ratio
QSGrd [83]	Low	Low	Low-Med	Delivery delay, Energy
QAR [84]	Medium ([177])	Low	Low-Med	Delivery delay, Loss ratio, Hop count, Bandwidth
RLOR [85]	Medium ([178])	Low	Very high ([179])	Delivery delay, Throughput
QGeo [86]	High	Low	Low-Med	Delivery delay, Delivery ratio, Overhead
QGR [87]	Low	Low	Low-Med	Gain
IQ-L [88]	Low	Low	Low-Med	Interest satisfaction delay, Interest satisfaction ratio, Overhead
AQFE [89]	Low	Low	Low-Med	Delivery delay
DLTPC [90]	Null ([180])	Very High	Low-Med	Buffer length at source nodes
RL-budget [91]	High	Limited ([181])	Very high ([182])	Number of clusters, Hop count, PU-SU Interference ratio
GFRLR [92]	High	Medium	Low-Med	Delivery delay, Delivery ratio, Throughput
RLSRP [93]	High	Low	Low-Med	Delivery delay and ratio, Path lifetime, Hop count
CCLBR [94]	Low	Low	Low-Med	Search time, Hit rate, Hop count
Q2-R [95]	High ([183])	Low	Low-Med	Delivery delay, Delivery ratio
SRR [96]	Very High ([184])	Low	Low-Med	Throughput
SDCoR [97]	([185])	([186])	Low ([187])	Delivery ratio

[163] State space is very low, because each packet has only three states.
[164] State space is very high because each state reflects the waiting queues at all nodes and the conditions of all channels.
[165] Action space is high because each action is a couple <next neighbor, TxT power>.
[166] There is a single agent.
[167] A state is a set of nodes, which are selected as relaying nodes. State space is all combinations of subsets of nodes.
[168] Action space depends on the number of combinations of relay set partitioning.
[169] Sender node receive ACK from all its neighbors before deciding which neighbor is designated as the next forwarder.
[170] Action space depends on the number of links per node.
[171] State space may be high, because a state is represented by a list of neighbors and a couple <distance, battery level> is associated with each neighbor.
[172] State space depends on the number of grids composing the network deployment area.
[173] Action space depends on the number of grids forming the network.
[174] In each period, agents exchange their Q-values and matrices of transition probabilities.
[175] State space depends on the number of power levels.
[176] Action space depends on the number of transmission power levels.
[177] Overhead is due to packet control exchange between SDN-controller and its 'slave" switches.
[178] Invited forwarders among neighbors confirm their participation in forwarding and send their Q-values.
[179] Action space depends on the number of combinations of neighbor set partitioning.
[180] There is no explicit message exchange between the relays.
[181] Complexity depends on the maximum number of available channels at cluster head.
[182] Action space depends on combinations of available channels.
[183] Q2-R uses both reactive routing, at bootstrap step to discover paths, and RL-based proactive routing to optimize the paths.
[184] SRR uses ACK and flooding when no stable paths are available.
[185] Only the SDN-controller is an agent. Overhead is due to control packets between SDN-controller and its SDN-switches.
[186] Complexity depends on the granularity of vehicle density and speed.
[187] Action space depends on the number of standard protocols that can be deployed by the SDN-controller.

- *High* (or *very high*), when state space is a list of paths with their current characteristics (energy, distance…) as in 34], [49], [50], [53], [57], [67], [69], [77], [90].

3) ACTION SPACE OVERHEAD

In addition to the space required to store states, RL-based algorithms require memory to store actions that can be selected by agents. From a qualitative point of view, action space overhead is categorized as:

- *Low*, when action space depends on factor such as number of available channels, number of transmission power levels. or number of protocols to deploy by SDN-controller. Most of reviewed protocols have a low to medium action space overhead (see Table 4).
- *Medium*, when action space depends on number of nodes in neighborhood.
- *High*, when action space depends on number of dynamic or predefined paths or on number of grids in network [34], [57], [60], [67], [69], [79].
- *Very high*, when state space depends on combinations of channel subsets or paths [50], [53], [59], [85], [91].

4) PROOF OF CONVERGENCE

In optimization field, the convergence to optimal solutions is one of the expected properties. Many existing techniques to solve multicriteria optimization problems are known to be sub-optimal. RL-based solutions would be widely deployed if their convergence to optimal could be (formally) demonstrated. The proof of convergence of RL-based routing protocols, which are direct applications of Q-learning, can be derived from the proof provided by Watkins and Dayan [14] as long as some assumptions are satisfied. Regarding protocols, which are not Q-learning compliant, convergence proof is an issue. Authors of this category of protocols did not formally consider (with the exception of [80] and [32]) the convergence of their algorithms. Rather, they addressed convergence from simulation point of view or just stating that convergence may be reached.

5) PROROCOL PERFORMANCE (SIMULATION)

RL-based routing algorithms are expected to provide optimal paths (i.e., paths with low delivery delay and delivery ratio, low energy consumption, high network lifetime…). Given the number of protocols and the variety in reward function design and in metric weights, even for the same network class, we cannot provide qualitative evaluation of existing protocols or compare their performance. Thus, we only include the metrics evaluated by authors through simulation.

Table 4 highlights performance aspects of RL-based routing protocols.

VI. CONCLUSIONS AND CHALLENGES

For a quarter century, Reinforcement learning is applied to routing in different classes of networks ranging from wired and static networks to very dynamic wireless and ad hoc networks. Tens of RL-based protocols have been proposed. This paper aims at providing a comprehensive review of literature on the field of research. A classification approach is proposed to highlight how RL-based routing protocols are designed, which would help adapting them to specific environments or improving them. RL is an efficient alternative to enforce online-awareness of routing protocols to their environment changes, so they can provide good levels of QoS, while optimizing resource utilization. However, some challenges still remain and should be investigated to provide evidence on applicability of RL-based protocols at large scale; they include the following:

Proof of optimality – Almost all reviewed papers did not convincingly address proof of convergence. However, it is obvious that RL would be definitely adopted in next generation networks when proof of convergence question has been answered. Q-learning author, Watkins, proved convergence under specific assumptions. It is not clear how the latter apply to routing. In addition, when Reward functions are based on multiple metrics, the proof of optimality becomes a harder problem.

Speed of convergence – Huge authors proposed heuristics to explore space of solutions (i.e., actions to select in states). Whenever large networks are considered, space exploration may take a (very) long time before optimal paths would be discovered, resulting in poor end-to-end performance of the network. Speed of convergence should be investigated further to provide bounds of delay regarding transient regime of routing optimization process and let users know when the network is ready to provide acceptable QoS levels.

Link-state information dissemination – In most of protocols, link-state information is used to calculate metrics, which are parameters of reward functions. Consequently, quality of routes as well as convergence of routing algorithms depend on freshness of disseminated link-state information. Frequency of Hello packets should be addressed in such a way to find compromise between protocol overhead and values of reward, which result in faster convergence. Relationships between both aspects should be investigated through analytical models, thus avoiding users to randomly set the period for Hello packets.

Weights associated with metrics and learning parameters – Whenever a metric is used in a reward function, a weight is associated with it. Also, Q-value update is based on two parameters – learning factor and discount rate. Authors of routing protocols addressed weights and learning parameters only from simulation point of view. However, learning parameters and weights have a significant impact on the quality of paths and on the speed of convergence. Development of a methodology to address the learning parameter setting and the weight assignment would make the deployment of RL-based routing protocols easier and more efficient.

Hybridization – A few routing protocols have sufficiently addressed the reduction of search space to make more efficient optimal-path search. RL should be used jointly with other techniques including classification, Bayesian networks,

Neural networks, Genetic algorithms, Ant colony, Swarm optimization, and Game theory to provide more (soft) guarantees on how solution space is wholly and quickly tested [98]. Recently, deep reinforcement learning has been proposed to enable RL to scale to complex problems [99], [100]. Deep RL would help designing efficient routing algorithms [101], [102], [103].

Predicting traffic demands – Learning in previously presented RL-based protocols is mainly based on network-oriented metrics (i.e., delays, loss rate, transmission success, mobility of nodes, etc.). Predicting traffic from sources to destinations would result in more efficient selection of forwarders and less congestion of nodes. Indeed, traffic prediction, through supervised learning, enables agents to avoid selection of some routes, if their selection would lead to performance degradation in the future.

Collaboration (*cooperative learning*) – Almost all proposed protocols are independent-agent-based. To face complexity of future networks (including heterogeneity of user traffics and QoS requirements as well as intelligence in networks), collaboration would help to design more robust and efficient learning to solve routing problems.

REFERENCES

[1] T. M. Mitchell, *Machine Learning*. New York, NY, USA: McGraw-Hill, 2017.
[2] K. V. Murphy, *Machine Learning: A Probabilistic Perspective*. Cambridge, MA, USA: MIT Press, 2012.
[3] D. Witten, R. Tibshirani, and T. Hastie, *An Introduction to Statistical Learning: With Applications in R*. New York, NY, USA: Springer, 2013.
[4] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. 2nd ed. Cambridge, MA, USA: MIT Press, 2018.
[5] S. Ayoubi *et al.*, "Machine learning for cognitive network management," *IEEE Commun. Mag.*, vol. 56, no. 1, pp. 158–165, Jan. 2018.
[6] R. V. Kulkarni, A. Forster, and G. K. Venayagamoorthy, "Computational intelligence in wireless sensor networks: A survey," *IEEE Commun. Surveys Tuts.*, vol. 13, no. 1, pp. 68–96, 1st Quart., 2011.
[7] M. Wang, Y. Cui, X. Wang, S. Xiao, and J. Jiang, "Machine learning for networking: Workflow, advances and opportunities," *IEEE Netw.*, vol. 32, no. 2, pp. 92–99, Mar./Apr. 2018.
[8] Z. Wang and J. Crowcroft, "Quality-of-service routing for supporting multimedia applications," *IEEE J. Sel. Areas Commun.*, vol. 14, no. 7, pp. 1228–1234, Sep. 1996.
[9] J. A. Boyan and M. L. Littman, "Packet routing in dynamically changing networks: A reinforcement learning approach," in *Proc. Neural Inf. Process. Syst. Conf. (NIPS)*, 1994, pp. 671–678.
[10] H. Al-Rawi, M. Ng, and K. Yau, "Application of reinforcement learning to routing in distributed wireless networks: A review," *Artif. Intell. Rev.*, vol. 43, no. 3, pp. 381–416, 2015.
[11] S. Chettibi and S. Chikhi, "A survey of reinforcement learning based routing protocols for mobile ad-hoc networks," in *Recent Trends in Wireless and Mobile Networks*. Ankara, Turkey: CoNeCo, 2011.
[12] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *J. Artif. Intell. Res.*, vol. 4, no. 1, pp. 237–285, Jan. 1996.
[13] C. J. Watkins, "Learning from delayed rewards," Ph.D. dissertation, King's College, London, U.K., 1989.
[14] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, nos. 3–4, pp. 279–292, 1992.
[15] I. F. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: A survey," *Comput. Netw.*, vol. 47, no. 4, pp. 445–487, Mar. 2005.
[16] A. Nosratinia, T. E. Hunter, and A. Hedayat, "Cooperative communication in wireless networks," *IEEE Commun. Mag.*, vol. 42, no. 10, pp. 74–80, Oct. 2004.
[17] R. Ramaswami, K. N. Sivarajan, and G. H. Sasaki, *Optical Networks: A Practical Perspective*. Amsterdam, The Netherlands: Elsevier, 2010.
[18] S. Basagni, M. Conti, S. Giordano, and I. Stojmenovic, *Mobile Ad Hoc Networking*. Hoboken, NJ, USA: Wiley, 2004.
[19] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: A survey," *Elsevier Comput. Netw.*, vol. 38, no. 4, pp. 393–422, 2002.
[20] S. Olariu and M. Weigle, *Vehicular Networks: From Theory to Practice*. London, U.K.: Chapman & Hall, 2009.
[21] K. Fall, "A Delay-tolerant network architecture for challenged internets," in *Proc. ACM SIGCOMM*, 2003, pp. 27–34.
[22] I. Bekmezci, O. K. Sahingoz, and . Temel, "Flying ad-hoc networks (FANETs): A survey," *Ad Hoc Netw.*, vol. 11, no. 3, pp. 1254–1270, 2013.
[23] K.-C. Chen and R. Prasad, *Cognitive Radio Networks*. Hoboken, NJ, USA: Wiley, 2009.
[24] L. Zhang *et al.*, "Named data networking," *SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 3, pp. 66–73, Jul. 2014.
[25] E. K. Lua, J. Crowcroft, and M. Pias, "A survey and comparison of peer-to-peer overlay network schemes," *IEEE Commun. Surveys Tuts.*, vol. 7, no. 2, pp. 72–93, 2nd Quart., 2005.
[26] D. Kreutz, F. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proc. IEEE*, vol. 103, no. 1, pp. 14–76, Jan. 2015.
[27] A. Nowe, K. Steenhaut, M. Fakir, and K. Verbeeck, "Q-learning for adaptive load based routing," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, San Diego, CA, USA, 1998, pp. 3965–3970.
[28] B. Krishnamachari, C. Zhou, and B. Shademan, "LEQS: Learning-based efficient querying for sensor networks," Univ. Southern California, Tech. Rep., Los Angeles, CA, USA, 2003.
[29] W. Usaha, "A reinforcement learning approach for path discovery in MANETs with path caching strategy," in *Proc. 1st Int. Symp. Wireless Commun. Syst.*, Mauritius, 2004, pp. 220–224.
[30] P. Baruah and R. Urgaonkar, "Learning-enforced time domain routing to mobile sinks in wireless sensor fields," *Proc. 29th Annu. IEEE Int. Conf. Local Comput. Netw.*, Tampa, FL, USA, 2004, pp. 525–532.
[31] S. P. Choi and D.-Y. Yeung, "Predictive Q-routing: A memory-based reinforcement learning approach to adaptive control," in *Proc. Neural Inf. Process. Syst. Conf. (NIPS)*, 1996, pp. 946–951.
[32] D. Subramanian, P. Druschel, and J. Chen, "Ants and reinforcement learning: A case study in routing in dynamic networks," in *Proc. 15th Int. Joint Conf. Artif. Intell.*, 1997, pp. 832–839.
[33] S. Kumar and R. Miikkulainen, "Confidence based Q-routing: An online network routing algorithm," in *Proc. in 16th Conf. Artif. Neural Netw. Eng.*, San Francisco, CA, USA, 1998, pp. 758–763.
[34] P. Marbach, O. Mihatsch, and J. N. Tsitsiklis, "Call admission control and routing in integrated services networks using reinforcement learning," in *Proc. 37th IEEE Conf. Decis. Control*, 1998, pp. 563–568.
[35] R. Sun, S. Tatsumi, and G. Zhao, "Application of multiagent reinforcement learning—To multicast routing in wireless ad hoc networks ensuring resource reservation," in *Proc. IEEE Int. Conf. Syst., Man Cybern.*, Oct. 2002, pp. 409–414.
[36] L. Peshkin and V. Savova, "Reinforcement learning for adaptive routing," in *Proc. Int. Joint Conf. Neural Netw.*, 2002, pp. 1825–1830.
[37] P. Garcia, A. Zsigri, and A. Guitton, "A multicast reinforcement learning algorithm for WDM optical networks," in *Proc. 7th Int. Conf. Telecommun.*, 2003, pp. 419–426.
[38] D. Chetret, C. K. Tham, and L. Wong, "Reinforcement learning and CMAC-based adaptive routing for MANETs," in *Proc. 12th IEEE Int. Conf. Netw. (ICON)*, Nov. 2004, pp. 540–544.
[39] C. Perkins, E. Belding-Royer, and S. Das, *Ad Hoc On-Demand Distance Vector (AODV) Routing*, document RFC 3561, IETF, 2003.
[40] Y. H. Chang, T. Ho, and L. P. Kaelbling, "Mobilized ad-hoc networks: A reinforcement learning approach," in *Proc. 1st Int. Conf. Autonomic Comput.*, May 2004, pp. 240–247.
[41] R. Rudek, L. Koszalka, and I. Pozniak-Koszalka, "Introduction to multi-agent modified Q-learning routing for computer networks," in *Proc. Adv. Ind. Conf. Telecommun.*, 2005, pp. 408–413.
[42] J. Dowling, E. Curran, R. Cunningham, and V. Cahill, "Using feedback in collaborative reinforcement learning to adaptively optimize MANET routing," *IEEE Trans. Syst., Man, Cybern., A, Syst. Humans*, vol. 35, no. 3, pp. 360–372, May 2005.
[43] P. Fu, J. Li, and D. Zhang, "Heuristic and distributed QoS route discovery for mobile ad hoc networks," in *Proc. 5th Int. Conf. Comput. Inf. Technol.*, Shanghai, China, 2005, pp. 512–516.

[44] Y. Zhang and M. Fromherz, "Constrained flooding: A robust and efficient routing framework for wireless sensor networks," in *Proc. 20th Int. Conf. Adv. Inf. Netw. Appl.*, Vienna, Austria, 2006, p. 392.

[45] P. Wang and T. Wang, "Adaptive routing for sensor networks using reinforcement learning," in *Proc. 6th IEEE Int. Conf. Comput. Inf. Technol.*, Sep. 2006, p. 219.

[46] M. G. Lagoudakis and R. Parr, "Least-squares policy iteration," *J. Mach. Learn. Res.*, vol. 4, pp. 1107–1149, Dec. 2003.

[47] S. Dong, P. Agrawal, and K. Sivalingam, "Reinforcement learning based geographic routing protocol for UWB wireless sensor network," in *Proc. IEEE Global Telecommun. Conf.*, Nov. 2007, pp. 652–656.

[48] R. Arroyo-Valles, R. Alaiz-Rodríguez, A. Guerrero-Curieses, and J. Cid-Sueiro, "Q-probabilistic routing in wireless sensor networks," in *Proc. 3rd Int. Conf. Intell. Sensors, Sensor Netw. Inf.*, Dec. 2007, pp. 1–6.

[49] P. Nurmi, "Reinforcement learning for routing in ad hoc networks," in *Proc. 5th Int. Symp. Modeling Optim. Mobile, Ad Hoc Wireless Netw.*, Limassol, Cyprus, 2007, pp. 1–8.

[50] A. Forster and A. L. Murphy, "FROMS: Feedback routing for optimizing multiple sinks in WSN with reinforcement learning," in *Proc. 3rd Int. Conf. Intell. Sensors, Netw. Inf.*, 2007, pp. 371–376.

[51] A. Forster and A. L. Murphy, "FROMS: A failure tolerant and mobility enabled multicast routing paradigm with reinforcement learning for WSNs," *Ad Hoc Netw., Elsevier*, vol. 9, no. 5, pp. 940–965, 2011.

[52] A. Forster and A. L. Murphy, "CLIQUE: Role-free clustering with Q-learning for wireless sensor networks," in *Proc. IEEE Int. Conf. Distrib. Comput. Syst.*, Jun. 2009, pp. 441–449.

[53] A. Forster and A. L. Murphy, "Balancing energy expenditure in WSNs through reinforcement learning: A study," in *Proc. 1st Int. Workshop Energy Wireless Sensor Netw. (WEWSN)*, 2008, p. 7.

[54] X. Liang, I. Balasingham, and S. S. Byun, "A reinforcement learning based routing protocol with QoS support for biomedical sensor networks," in *Proc. 1st Int. Symp. Appl. Sci. Biomed. Commun. Technol.*, 2008, pp. 1–5.

[55] X. Liang, I. Balasingham, and S. S. Byun, "A multi-agent reinforcement learning based routing protocol for wireless sensor networks," in *Proc. IEEE Int. Symp. Wireless Commun. Syst.*, 2008, pp. 552–557.

[56] A. Belbekkouche, A. Hafid, and M. Gendreau, "A reinforcement learning-based deflection routing scheme for buffer-less OBS networks," in *Proc. IEEE Global Telecommun. Conf.*, 2008, pp. 1–6.

[57] W. Naruephiphat and W. Usaha, "Balancing tradeoffs for energy-efficient routing in MANETs based on reinforcement learning," in *Proc. IEEE Veh. Technol. Conf.*, May 2008, pp. 2361–2365.

[58] B. Xia, M. H. Wahab, Y. Yang, Z. Fan, and M. Sooriyabandara, "Reinforcement learning based spectrum-aware routing in multi-hop cognitive radio networks," in *Proc. 4th Int. Conf. Cogn. Radio Oriented Wireless Netw. Commun.*, 2009, pp. 1–5.

[59] X. Liang, I. Balasingham, and V. C. Leung, "Cooperative communications with relay selection for QoS provisioning in wireless sensor networks," in *Proc. IEEE Global Telecommun. Conf.*, Nov. 2009, pp. 1–9.

[60] C.-L. Chang and S.-J. Kang, "Using reinforcement learning to the priority-based routing and call admission control in WDM networks," in *Proc. 5th Int. Multi-Conf. Comput. Global Inf. Technol.*, 2010, pp. 126–130.

[61] A. Petrowski, F. Aissanou, I. Benyahia, and S. Houcke, "Multicriteria reinforcement learning based on a Russian doll method for network routing," in *Proc. 5th IEEE Int. Conf. Intell. Syst.*, Jul. 2010, pp. 321–326.

[62] T. Hu and Y. Fei, "An adaptive and energy-efficient routing protocol based on machine learning for underwater delay tolerant networks," in *Proc. 18th IEEE/ACM Int. Symp. Modeling, Anal. Simulation Comput. Telecommun. Syst.*, Aug. 2010, pp. 381–384.

[63] A. Elwhishi, P.-H. Ho, K. Naik, and B. Shihada, "ARBR: Adaptive reinforcement-based routing for DTN," in *Proc. 6th IEEE Int. Conf. Wireless Mobile Comput., Netw. Commun.*, Niagara Falls, ON, Canada, Oct. 2010, pp. 376–385.

[64] T. Hu and Y. Fei, "QELAR: A machine-learning-based adaptive routing protocol for energy-efficient and lifetime-extended underwater sensor networks," *IEEE Trans. Mobile Comput.*, vol. 9, no. 6, pp. 796–809, Jun. 2010.

[65] Z. Binbin, L. Quan, and Z. Shouling, "Using statistical network link model for routing in ad hoc networks with multi-agent reinforcement learning," in *Proc. 2nd Int. Conf. Adv. Comput. Control*, 2010, pp. 462–466.

[66] G. Santhi, A. Nachiappan, M. Zaid Ibrahime, R. Raghunadhane, and M. K. Favas, "Q-learning based adaptive QoS routing protocol for MANETs," in *Proc. Int. Conf. Recent Trends Inf. Technol.*, 2011, pp. 1233–1238.

[67] Z. Lin and M. van der Schaar, "Autonomic and distributed joint routing and power control for delay-sensitive applications in multi-hop wireless networks," *IEEE Trans. Wireless Commun.*, vol. 10, no. 1, pp. 102–113, Jan. 2011.

[68] K. Wang, W.-C. Wong, and T. Y. Chai, "A MANET routing protocol using Q-learning method integrated with Bayesian network," in *Proc. IEEE Int. Conf. Commun. Syst.*, Nov. 2012, pp. 270–274.

[69] H. Jung, K. Kim, J. Kim, O.-S. Shin, and Y. Shin, "A relay selection scheme using Q-learning algorithm in cooperative wireless communications," in *Proc. 18th Asia–Pacific Conf. Commun.*, 2012, pp. 7–11.

[70] C. Wu, S. Ohzahata, and T. Kato, "Routing in VANETs: A fuzzy constraint Q-Learning approach," in *Proc. IEEE Global Commun. Conf.*, 2012, pp. 195–200.

[71] G. Oddi, D. Macone, A. Pietrabissa, and F. Liberati, "A proactive link-failure resilient routing protocol for MANETs based on reinforcement learning," in *Proc. 20th Medit. Conf. Control, Automat.*, 2012, pp. 1259–1264.

[72] A. A. Bhorkar, M. Naghshvar, T. Javidi, and B. D. Rao, "Adaptive opportunistic routing for wireless ad hoc networks," *IEEE/ACM Trans. Netw.*, vol. 20, no. 1, pp. 243–256, Feb. 2011.

[73] J. Yang, H. Zhang, C. Pan, and W. Sun, "Learning-based routing approach for direct interactions between wireless sensor network and moving vehicles," in *Proc. 16th Int. IEEE Conf. Intell. Transp. Syst.*, Oct. 2013, pp. 1971–1976.

[74] C. Wu, S. Ohzahata, and T. Kato, "Flexible, portable, and practicable solution for routing in VANETs: A fuzzy constraint Q-learning approach," *IEEE Trans. Veh. Technol.*, vol. 62, no. 9, pp. 4251–4263, Nov. 2013.

[75] M. Boushaba, A. Hafid, A. Belbekhouche, and M. Gendreau, "Reinforcement learning based routing in wireless mesh networks," *Wireless Netw*, vol. 19, no. 8, pp. 2079–2091, 2013.

[76] S. Haeri, M. Arianezhad, and L. Trajkovic, "A predictive Q-learning algorithm for deflection routing in buffer-less networks," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Oct. 2013, pp. 764–769.

[77] M. Maleki, V. Hakami, and M. Dehghan, "A reinforcement learning-based Bi-objective routing algorithm for energy harvesting mobile ad-hoc networks," in *Proc. 7th IEEE Int. Symp. Telecommun.*, Sep. 2014, pp. 1082–1087.

[78] S. V. Jafarzadeh and M. H. Moghaddam, "Design of energy-aware QoS routing algorithm in wireless sensor networks using reinforcement learning," in *Proc. 4th Int. Conf. Comput. Knowl. Eng.*, 2014, pp. 722–727.

[79] R. Li, F. Li, X. Li, and Y. Wang, "QGrid: Q-learning based routing protocol for vehicular ad hoc networks," in *Proc. 33rd IEEE Int. Perform. Comput. Commun. Conf.*, Dec. 2014, pp. 1–8.

[80] F. Shams, G. Bacci, and M. Luise, "Energy aware power control for multi-relay cooperative network using *Q*-learning," *IEEE Trans. Wireless Commun.*, vol. 14, no. 3, pp. 1567–1580, Mar. 2015.

[81] M. J. Osborne, *An Introduction to Game Theory*. London, U.K.: Oxford Univ. Press, 2000.

[82] Y. Saleem, K. A. Yau, H. Mohamad, N. Ramli, and M. H. Rehmani, "Joint channel selection and cluster-based routing scheme based on reinforcement learning for cognitive radio networks," in *Proc. Int. Conf. Comput., Commun., Control Technol.*, 2015, pp. 21–25.

[83] B. Debowski, P. Spachos, and S. Areibi, "Q-learning enhanced gradient based routing for balancing energy consumption in WSNs," in *Proc. 21st IEEE Int. Workshop Comput. Aided Modelling Design Commun. Links Netw. (CAMAD)*, Oct. 2016, pp. 18–23.

[84] S.-C. Lin, I. F. Akyildiz, P. Wang, and M. Luo, "QoS-aware adaptive routing in multi-layer hierarchical software defined networks: A reinforcement learning approach," in *Proc. IEEE Int. Conf. Services Comput.*, Jun. 2016, pp. 25–33.

[85] K. Tang, C. Li, H. Xiong, J. Zou, and P. Frossard, "Reinforcement learning-based opportunistic routing for live video streaming over multi-hop wireless networks," in *Proc. IEEE 19th Int. Workshop Multimedia Signal Process.*, Oct. 2017, pp. 1–6.

[86] W.-S. Jung, J. Yim, and Y.-B. Ko, "QGeo: Q-learning-based geographic *Ad Hoc* routing protocol for unmanned robotic networks," *IEEE Commun. Lett.*, vol. 21, no. 10, pp. 2258–2261, Oct. 2017.

[87] F. Hajiaghajani and S. Biswas, "Learning based gain-aware content dissemination in delay tolerant networks," in *Proc. 9th Int. Conf. Commun. Syst. Netw.*, 2017, pp. 198–205.

[88] B. Fu, L. Qian, Y. Zhu, and L. Wang, "Reinforcement learning-based algorithm for efficient and adaptive forwarding in named data networking," in *Proc. IEEE/CIC Int. Conf. Commun. China*, Oct. 2017, pp. 1–6.

[89] M. Kavalerov, Y. Likhacheva, and Y. Shilova, "A reinforcement learning approach to network routing based on adaptive learning rates and route memory," in *Proc. SoutheastCon*, 2017, pp. 1–6.
[90] V. Hakami and M. Dehghan, "Distributed power control for delay optimization in energy harvesting cooperative relay networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 6, pp. 4742–4755, Jun. 2017.
[91] Z. Javed, K. A. Yau, H. Mohamad, N. Ramli, J. Qadir, and Q. Ni, "RL-budget: A learning-based cluster size adjustment scheme for cognitive radio networks," *IEEE Access*, vol. 6, pp. 1055–1072, 2018.
[92] C. Wu, T. Yoshinaga, Y. Ji, and Y. Zhang, "Computational intelligence inspired data delivery for vehicle-to-roadside communications," *IEEE Trans. Veh. Technol.*, vol. 67, no. 12, pp. 12038–12048, Dec. 2018.
[93] Z. Zheng, A. K. Sangaiah, and T. Wang, "Adaptive communication protocols in flying ad hoc network," *IEEE Commun. Mag.*, vol. 56, no. 1, pp. 136–142, Jan. 2018.
[94] X.-J. Shen, Q. Chang, L. Liu, J. Panneerselvam, and Z.-J. Zha, "CCLBR: Congestion control-based load balanced routing in unstructured P2P systems," *IEEE Syst. J.*, vol. 12, no. 1, pp. 802–813, Mar. 2018.
[95] T. Hendriks, M. Camelo, and S. Latre, "Q^2-routing: A Qos-aware Q-routing algorithm for wireless ad hoc networks," in *Proc. 5th Int. Workshop Cooperat. Wireless Netw.*, Cartagena, Spain, 2018.
[96] M. Johnston, C. Danilov, and K. Larson, "A reinforcement learning approach to adaptive redundancy for routing in tactical networks," in *Proc. IEEE Military Commun. Conf.*, Los Angeles, CA, USA, Oct. 2018, pp. 267–272.
[97] C. Wang, L. Zhang, Z. Li, and C. Jiang, "SDCoR: Software defined cognitive routing for Internet of vehicles," *IEEE Internet Things J.*, vol. 5, no. 5, pp. 3513–3520, Oct. 2018.
[98] L. Calvet, J. D. Armas, D. Masip, and A. A. Juan, "Learnheuristics: Hybridizing metaheuristics with machine learning for optimization with dynamic inputs," *Open Math*, vol. 15, no. 1, pp. 261–280, 2017.
[99] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, 2015.
[100] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 26–38, Nov. 2017.
[101] C. Yu, J. Lan, Z. Guo, and Y. Hu, "DROM: Optimizing the routing in software-defined networks with deep reinforcement learning," *IEEE Access*, vol. 6, no. 18, pp. 54539–64533, 2018.
[102] A. Valadarsky, M. Schapira, D. Shahaf, and A. Tamar, "Learning to route," in *Proc. 16th ACM Workshop Hot Topics Netw. (HotNets)*, 2017, pp. 185–191.
[103] Q. Mao, F. Hu, and Q. Hao, "Deep learning for intelligent wireless networks: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 2595–2621, 4th Quart., 2018.

ZOUBIR MAMMERI received the Ph.D. degree in real-time distributed systems and the habilitation to supervise researches from the National Polytechnic Institute of Lorraine (INPL), Nancy, France, in 1985 and 1995, respectively.

He has been a Full Professor of computer science and engineering with Paul Sabatier University, Toulouse, France, since 1998. He has coauthored six books and over 200 papers in refereed journals and conferences in the fields of real-time systems and communication networks. His research interests include the quality of service in networks, QoS-based routing, application of reinforcement learning to networking, packet scheduling, mobile ad hoc networks, wireless sensor networks, security in sensor and vehicular networks, real-time systems, and task scheduling. He is a member of IFIP TC6 WG.8. So far, he has been the General Chair or Program Chair of 16 international conferences and workshops. He served on the TPC for over 210 international conferences. He is a member of the Editorial Board of four journals.