SAPIENZA
UNIVERSITÀ DI ROMA

"Sapienza" University of Rome
Faculty of Information Engineering,
Informatics and Statistics
Department of Computer Science

RL-based protocols review

Author
Alessio Bandiera

November 29, 2025

Contents

Information and Contacts

Personal notes and summaries collected as part of the *RL-based protocols review* course offered by the degree in Computer Science of the University of Rome "La Sapienza".

Further information and notes can be found at the following link:
https://github.com/aflaag-notes. Anyone can feel free to report inaccuracies, improvements or requests through the Issue system provided by GitHub itself or by contacting the author privately:

- Email: **alessio.bandiera02@gmail.com**
- LinkedIn: **Alessio Bandiera**

The notes are constantly being updated, so please check if the changes have already been made in the most recent version.

Suggested prerequisites:

Basic concepts of Reinforcement Learning.

Licence:

1

Q-routing literature review

TODO intro

In RL based design, the following aspects are addressed:

- identification of the most appropriate states and actions of the agent
- definition of the reward function depending on the metrics to optimize
- identification of environment model when available

Given a target field of application, different design models may be elaborated, which differ in how they address each of the previous aspects.

In the last 25 years many RL-based routing protocol have been proposed, but most of them share the same high-level structure.

In literature, nodes are confused with agents, and in almost all protocols the reward is (at least partially) calculated by a node upon selecting an entire route to use for all packtes to transmit, or just a next hop to transmit the current data packet. Thus, a *node* should be considered to consist of an agent and optional *modules*:

- **local reward** module: it calculates reward based on local view, which reflects the cost of communication as seen by the packet sender
- **remote reward** module: it receives feedback sent by the next hop or by the destinatino node — if local and remote modules are both employed they are combined to form the reward return to the agent
- **link-state information maintenance** module: it collects useful link state information through periodi or on-demand *Hello packets*

Therefore, the neighboring nodes of a node define the environment of the agent representing a node.

TODO parlare dei tipi di reti?

1.1 Q-routing

Boyan and Littman [BL93] were the first to propose a hop-by-bop routing algorithm based on Q-learning, called Q-routing, and the most of exists RL-based routing protocols today are just extensions of this algorithm. The following is the algorithm that defines Q-routing in detail.

1: **function** QROUTING( )
2: Initialize Q_i matrix randomly
3: **while** termination condition holds **do**
4: **if** packet P is ready to be sent to d **then**
5: Determine node $j^* \leftarrow \arg\min_{j \in \mathcal{N}(i)} Q_i(d, j)$
6: Send packet to node j^*
7: Collect estimate $\theta_{j^*}(d)$ from node j^*
8: Update $Q_i(d, j^*) \leftarrow (1 - \alpha) \cdot Q_i(d, j^*) + \alpha \cdot [W_i^q(P) + T_{ij^*} + \theta_{j^*}(d)]$
9: **end if**
10: **end while**
11: **end function**

We will briefly explain the algorithm. First, let's present the notation:

- i is the node that is currently running the algorithm
- P is a packet that node i needs to forward to destination d
- $Q_i(d, j)$ is the *delivery delay* that i estimates it takes, for node j, to deliver the packet P at destination i
- $\mathcal{N}(j)$ is the set of j's neighbors
- $\theta_j(d)$ is j's estimate for the time remaining in the trip to destination d of packet P
- $W_i^q(P)$ is the time spent by packet P in node i's queue
- T_{ij} is the transmission time between nodes i and j

Each entry of the table Q_i is called **Q-value**, and when node i wants to send a packed, it selects the node j^* that minimizes the Q-value. Upon sendin packet P to node j^*, node i receives back from j^* the value

$$\theta_{j^*}(d) = \min_{k \in \mathcal{N}(j^*)} Q_{j^*}(d, k)$$

Then, node i has to update its estimate of $Q_i(d, j^*)$ based on $\theta_{j^*}(d)$, which can be performed utilizing the formula

$$Q(s_t, a_t) = (1 - \alpha) \cdot Q(s_t, a_t) + \alpha \cdot \left[R_{t+1} + \gamma \cdot \max_{a \in A} Q(s_{t+1}, a) \right]$$

by setting

- $R_{t+1} = W_i^q(P) + T_{ij^*}$ since it represents the *link cost*
- $\gamma = 1$

- $\max_{a \in A} Q(s_{t+1}, a) = \min_{k \in \mathcal{N}(j^*)} Q_{j^*}(d, k)$ since the "action to take" corresponds to choosing an neighbor in this context, however we seek to *minimize* the Q-value since the delay is clearly a decreasing metric

Despite the wide adoption of this protocol over the years, Q-routing is still far from perfect and it has its flaws. Some of the problems are inherent problems of Q-learning in general, such as *slow convergence* rate and high *parameter setting sensitivity*, but there are problem that arise from the protocol itself. For instance, to avoid frequen oscillations of the Q-values — in case of sudden variations of traffic in the network — and limit the overhead of the protocol, Nowe, Steenhaut, Fakir, et al. [NSF+98] proposed an extension in which j^* sends an average $\overline{\theta}_{j^*}(d)$ to i only after a certain number of exchanged packets. Another known problem is called **Q-value freshness**: the estimate $\theta_j(d)$ is evaluated upon packet transmission on a route, therefore if a route is not selected during a long period of time, the agent does not have an accurate estimate of the current condition of such route.

2

Classification criteria

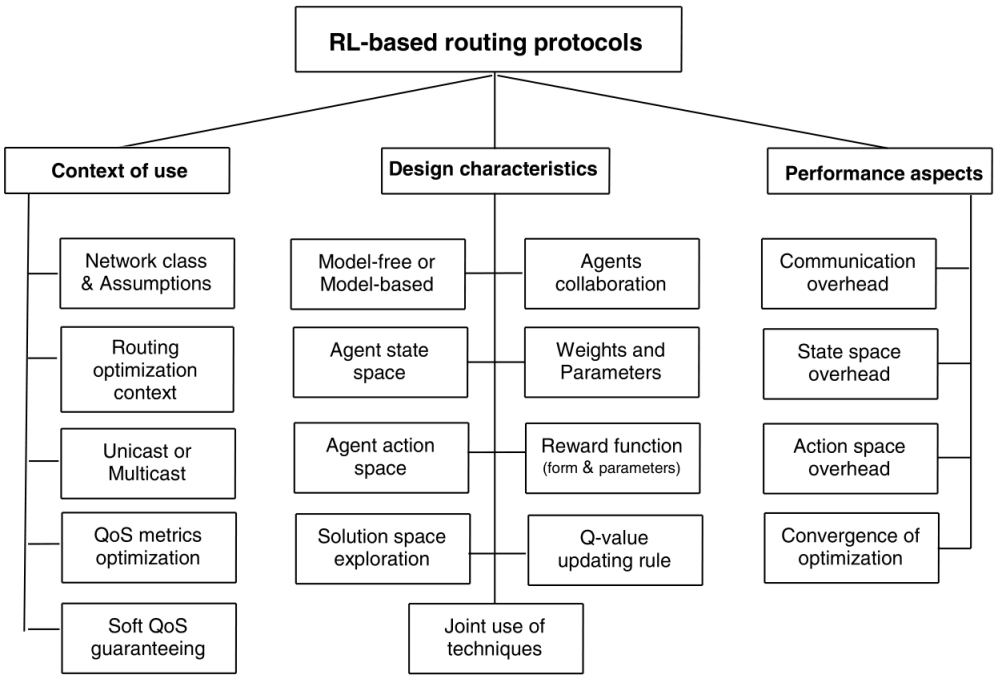

Figure 2.1: TODO

The authors underline that, to their knowledge, their work is the first in the literature that proposes classification criteria to help understanding and comparing all the available RL-based routing protocols. These criteria are divided into 3 groups:

- **context of use**: these are criteria that describe the targeted applications and their characteristics and requirements
- **design characteristics**: criteria in this group highlight how authors designed their protocols to make them efficient and different from the others

- **performance**: in this last category, criteria provide a qualitative evaluation of the overhead of protocols and the metrics used by the authors

We will cover each criteria in the

2.1 Context of use

2.1.1 Network class and Assumptions

TODO

non ho capito

2.1.2 Routing Optimization Context

From users' perspective, routing protocols should always be able to determine and select the optimal paths to convey data from sources to destinations. There are different ways to achieve such goal, that depend on

- roles assigned to data sources
- roles assigned to relaying nodes
- initial assumptions about routing

The authors outlined 6 different *routing optimization contexts*, which we will briefly explain one by one.

1. **Data-packet driven optimazion**: in this context the transmission of packets happens hop-by-bop from source s to destination d, and upon receival d sends back a feedback. After a given amount of forwarded packets, the routing process converges to the selectoin of optimal paths.
2. **Route requrest driven optimization**: a source s that has data to send to d, first sends a Route Request (RR) packet. The latter is then disseminated in the network, and each node that receives the RR packet can decide to partecipate or not — if it agrees to partecipate, it selects the next node to forward the RR packet to, and this process continues until d is reached. Once a path is found, all packets from s to d are routed through this path. Then, at the end of each transmission a feedback is sent back to the sender regarding the performance of current nodes. Most protocols in this category are extensions to the **AODV protocol**
3. **Context request driven optimization**: this is a setting that describes peer-to-peer systems and named data networks, in which a node s that is interested in some content C sends its requrest to receive data packets from the nodes d possessing C. Nodes on the path from s to d can then decide to forward the request to locate the requested content, and when data packets containing C are forwarded the relay nodes receive feedback and adapt their paths accordingly.
4. **Predefined routes driven optimization**: each source builds *offline* a list of paths of reachability for any target destination. Hence, when a source has packets to send it selects a path amonde the predefined ones. If a link break on the selected

path is detected, the source switches to another predefined path. Periodically, a feedback is sent backward to the source, which will adapt its path selection among the predefined list.

5. **Cluster driven optimization**:
6. **Routing protocol driven optimization**:

me so rotto

me so rotto

2.1.3 Unicast or Multicast

Unicast and Multicast routing strategies are vastly different in terms of optimization, so it comes natural to define this criteria in order to categorize the algorithm proposed in the literature. The difference between the two approaches lies in the overhead that Multicast trees requires, both in terms of times and communications, in order to reach optimal trees. Additionally, when some links are not sufficiently stable, the convergence to optimal trees is outright *impossible*. Indeed, RL shoule be applied for multicasting scenarios only when links are sufficiently stable and/or when partial delivery is allowed — for instance, it is greatly discouraged by the authors on wireless networks.

2.1.4 QoS metrics for optimization

In general, routing problems in networks are **multicriteria decision making (MCDM)** problems, which are notoriously difficult to solve because of the heterogeneity nature of the metrics utilized. In fact, the choice of the metrics is one of the most important aspects of a user, which depends on the specificities of their application. Consequently, MCDM solving approaches are based on *weights* that express the relative importance of each metrics. **Quality of Service (QoS)** metrics that have been addressed as objectives for RL-based routing include:

- **delivery rate**: the average time to deliver a packet at destination
- **delivery ratio**: the proportion of packets successfully delivered at destination
- **hop count**: the average number of hops from source to destination
- **loss ratio**: the proportion of packets not delivered at destination
- **symbol error rate**: the proportion of *symbols* incorrectly transmitted
- **light-path blocking probability**: the percentage of the blocked light-paths of all requrests in optical networks — it is similar to the *loss ratio*
- **bandwidth**: the average bandwidth provided to sources
- **throughput**: the average amount of bytes delivered in the entire network per time unit
- **path stability**: it indicates how a path between source and destination changes over time
- **energy consumption**: the average energy consumption due to transmissions, receptions and processing

- **network lifetime**: the average time over which the network is still alive — this is essential in wireless sensor networks (WSNs)
- **transmission power**: the power for performing transmission — usually results in energy saving and interference reduction
- **PU-SU interference (ratio)**: it indicates how Primary users (PU) are prevented from transmissing by secondary users (SU)
- **hit delay**: the average delay to return requested data in peer-to-peer and named data networks
- **hit ratio**: the proportion of statisfied requrests in peer-to-peer and named data networks
- **gain** or **revenue**: the average revenue (in $ or any other currency) received by the agent when routing is seen from a business point of view, and routing should result in profit
- **overhead**: the average *cost* to deliver data packets at destinatino — the definition of *cost* may vary depending on the application

2.1.5 QoS guaranteeing

Lastly, there are a few routing protocols aimed at providing QoS guarantees, regarding delivery delay to meet requirements of some **delay-sensitive applications** – such as multimedia applications.

2.2 Design characteristics

2.2.1 Learning model

In RL there are two classes of learning strategies, namely **model-free** and **model-based**. Even if the vast majority of RL-based routing algorithms are model-free, since constructing a model requires knowledge about the enviroment that can be very hard to collect, it is worth mentioning that a few algorithms are actually model-based. Some of them use offline-collected information, regarding the environment model, while other caluclate and improve the environment model online. Model-based learning can offer an interesting opportunity when the the speed of convergence is a crucial requirement, as model-based approaches are known to have a faster convergence rate.

2.2.2 Agent states and Action spaces

In order to apply RL to any optimization problems we obviously need some definitions of **agent states** and **action spaces**. Let's discuss the former first. The following is a brief list of possible *agent state spaces* utilized in the reviewd literature:

non ho capito che ho scritto in questa lista

- *set of nodes*, which is the most popular in RL-based routing protocols
- *set of grids*, used in grid-organized networks

- *set of couples* relating to the dynamics of nodes, for instance in VANETSs a *couple* is a vehicle speed class and context of move (urban, highway...)
- *set of paths* and their characteristics
- *set of QoS levels required by flows*
- *set of transmission power levels*
- *set of available wavelengths*, in optical networks
- *set of packet states*

Next, we need to outline the possible *action spaces*. Broadly speaking, an action space is a set of single-type actions, or a set of actions of different types. The following is a table containing the possible *single-type actions* and corresponding action state spaces:

Action selection	**Action space**
Select node j as next hop and forward packet	Set of node IDs
Select a subset of neighbors S and broadcast packet	Set of partitions of node IDs
Select output link l and transmit packet	Set of links
Select grid g and send packet to one of the nodes in g	Set of grids
Select predefined path p and send packet along p	Set of predefined paths
Allocate m free channels	Set of channels
Select a transmission power pw	Set of transmission power levels
Select a protocol prt among a list of routing protocols and configure the network with prt	Set of standard protocols

2.2.3 Solution space exploration

Bibliography

[BL93] Justin Boyan and Michael Littman. "Packet routing in dynamically changing networks: A reinforcement learning approach". In: *Advances in neural information processing systems* 6 (1993).

[NSF+98] Ann Nowe, Kris Steenhaut, Mohamed Fakir, et al. "Q-learning for adaptive load based routing". In: *SMC'98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 98CH36218)*. Vol. 4. IEEE. 1998, pp. 3965–3970.