Reinforcement learning for traffic control system: Study of Exploration methods using Q-learning

Maha REZZAI[1], Wafaa DACHRY[2], Fouad MOUTAOUAKKIL[1] , Hicham MEDROMI[1]

[1]First Systems Architecture Team, Laboratory of Research in Engineering, University Hassan II Casablanca ENSEM Casablanca 20200, Morocco
[2]Laboratory of Mechanical Engineering, Industrial Management and Innovation (IMMII) Hassan I University, FST of Settat, B.P. 577, Settat, Morocco.

---***---

Abstract - *Road congestion problems are becoming more complicated. This is because of static management of the traffic lights. Reinforcement Learning RL is an artificial intelligence approach that enables adaptive real-time control at intersections. RL allows vehicles to cross faster and minimize waiting times in the roadways. The objective of this article is to examine and test the different action selection techniques using the Q-learning algorithm in a case study. We will also present the different signal control systems based on RL, as well as the theoretical framework and the design elements of a RL system. The tests are performed using the VB-EXCEL tool.*

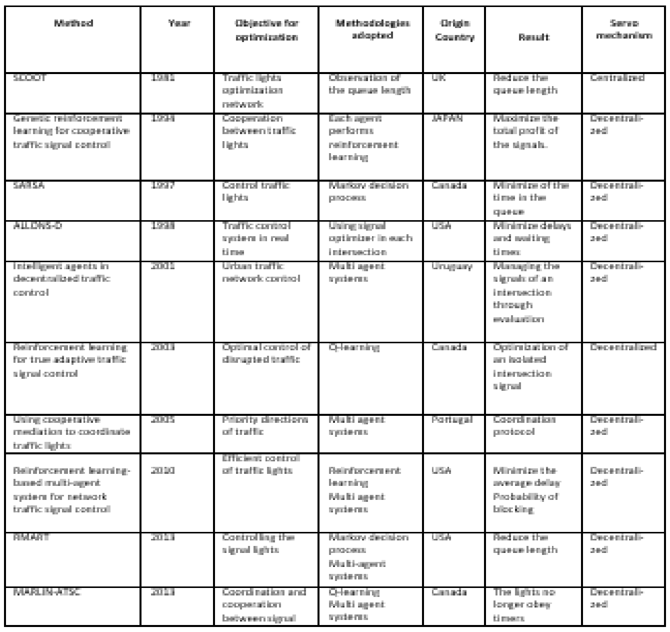
Key Words: Traffic control system, Reinforcement Learning, Softmax, Є-greedy, Q-learning.

1. INTRODUCTION

Traffic congestion is a new phenomenon that is overwhelming large cities around the world. It is due to a static management of traffic lights. Instead of considering them as a source of safety and efficiency, they are found as a source of delay, due to the high demand. In order to remedy this problem, we find Intelligent Transportation Systems (ITS). ITS are an advanced applications or services involving information technology and communication (ICT) in transport engineering [1]. They constitute a real challenge for the common transport policy and are considered as a relevant tool [2] to improve road infrastructure performance in terms of safety, traffic flow and reduction of CO2 emissions. Therefore, the most feasible option in the short term is to improve the use of existing roads by providing a new control in intersections to allow vehicles to cross faster and minimize waiting times in the roadways. In this context, the control of traffic lights has been studied by many researchers, a detailed description can be found in [3]. Their common goals are to reduce congestion at intersections.

The following table summarizes the different design programs for signal traffic control.

Table -1 Summary of different design programs for the control of traffic signal



Method	Year	Objective for optimization	Methodologies adopted	Origin Country	Result	Servo mechanism
SCOOT	1981	Traffic lights optimization network	Observation of the queue length	UK	Reduce the queue length	Centralized
Genetic reinforcement learning for cooperative traffic signal control	1994	Cooperation between traffic lights	Each agent performs reinforcement learning	JAPAN	Maximize the total profit of the signals.	Decentrali-zed
SARSA	1997	Control traffic lights	Markov decision process	Canada	Minimize of the time in the queue	Decentrali-zed
ALLONS-D	1998	Traffic control system in real time	Using signal optimizer in each intersection	USA	Minimize delays and waiting times	Decentrali-zed
Intelligent agents in decentralized traffic control	2001	Urban traffic network control	Multi agent systems	Uruguay	Managing the signals of an intersection through evaluation	Decentrali-zed
Reinforcement learning for true adaptive traffic signal control	2003	Optimal control of disrupted traffic	Q-learning	Canada	Optimization of an isolated intersection signal	Decentralized
Using cooperative mediation to coordinate traffic lights	2005	Priority directions of traffic	Multi agent systems	Portugal	Coordination protocol	Decentrali-zed
Reinforcement learning-based multi-agent system for network traffic signal control	2010	Efficient control of traffic lights	Reinforcement learning Multi agent systems	USA	Minimize the average delay Probability of blocking	Decentrali-zed
RMART	2013	Controlling the signal lights	Markov decision process Multi-agent systems	USA	Reduce the queue length	Decentrali-zed
MARLIN-ATSC	2013	Coordination and cooperation between signal	Q-learning Multi agent systems	Canada	The lights no longer obey timers	Decentrali-zed

We note that the recent designs programs for traffic control signal is based on reinforcement learning RL. RL is one of the most recently used control optimization techniques to resolve traffic light control [4]–[7], and the most appropriate when the environment is stochastic. In addition, the key element of RL is to improve efficiency and efficiency in the intersections. Thus, our approach adopts reinforcement learning techniques.

Indeed, in any system based on RL, an agent (in our case, the signal controller) interacts with the environment that is modeled as a Markov decision process (MDP) to make good actions. We further explain the interactions between the agent and the environment in the following sections. The main objective of this paper is to examine the different approaches for action selection by applying the Q-learning algorithm. In addition, this paper introduces in a novel way the concept of reinforcement learning, and the relationship between dynamic programming and RL. Our article is organized as follows. In Section 2, we present the different traffic control systems based on RL. The theoretical framework and design elements of RL system will be interpreted in the following section. Then, section 4 will describe our case study, followed by the results that will be developed and discussed. Finally, Section 5 contains the conclusion of this study and the scope for future research.

2. State of art of RL in traffic management system

The implementation of RL in traffic management system has been studied by many researchers, in order to reduce traffic congestion, we quote:

Moghaddam et al. [8] proposed a real-time traffic control system. This system contains two phases to adjust the duration of the lights according to the traffic flows.

Zhu et al. [9] developed a new reinforcement learning algorithm for signal control. Traffic lights are modeled as intelligent agents that interact with the stochastic traffic environment. The results show that the algorithm outperforms autonomous learning (Q-learning), and real-time adaptive learning, and fixed synchronization plans, in terms of average delay, number of stops, and vehicle emissions at the network level.

Stevanovic et al. [10] presented a methodology where the Pareto optimum has 3 signal synchronization dimensions (mobility, safety, and environmental factors) are optimized by the use of an evolutionary algorithm .

Le et al [11] proposed a decentralized traffic control policy for urban road networks. This strategy allows controlling the flow. The numerical results suggest that the proposed policy can outperform other policies in terms of network throughput and congestion.

Cong et al. [12] adopted a co-design approach in order to find the optimal network topology and the optimal parameters of traffic control laws at the same time by solving a co-optimization problem. The results show a higher performance compared to separate optimization.

Cesme and Furth [13]explored a new paradigm for controlling traffic lights "self-organizing signals". It is based on local activated control, but with some additional rules that create coordination mechanisms. In this system, the green phase can be prolonged or truncated relative to the speed of a section. The results show a reduction of transit time of about 60%, or 12 to 14 s per intersection with little impact on the delay of traffic.

Pescaru et al [14] proposed a methodology for the adaptive control of traffic lights in an area. It is integrated on a set of classifiers that intelligently process input data measured by a small number of sensors. These sensors are placed only on main roads entering this zone. The methodology, implemented in a generic way. It can be customized according to the needs of any given urban area with a radius of less than 4 km.

Samah et al [15] proposed a system called MARLIN-ATSC, where agents manage to coordinate and cooperate with each other.

Arora et al. [16] measured traffic density on the road using morphological edge detection and fuzzy logic technique. The method of morphology is very complicated to put in place, and it does not work well during the night.

Abdul Aziz et al. [6]proposed the RMART technique that controls the signal lights using the Markov decision process in a multi-agent framework.

Keyarsalan et al [17] applied a fuzzy ontology to control the light of traffic at isolated intersections through the use of vision techniques by software and neural networks to extract traffic data. The proposed system is more efficient compared to other similar approaches (the average delay is much lower for each vehicle in all traffic conditions).

Abdoos et al [18] used Q-learning for a non-stationary environment. The estimated states are based on the average queue length. The results show that the proposed Q-learning has outperformed the fixed time method in different traffic requests.

Dujardin et al. [19] applied a mixed integer linear programming for the multimodal control of traffic lights based on the optimization of three criteria (people's total delay, public vehicles total delay, number of stops for private vehicles to capture aspects of pollution).

Arel et al [20] introduced a multi-agent reinforcement learning system to obtain an effective traffic control policy, which aims to minimize the average delay, congestion and the probability of blocking in an intersection.

Shakeri et al. [21] introduced a new fuzzy method to optimize the control of traffic lights based on cellular automata. This method contains three layers: the first level of the street is calculated based on fuzzy rules. The second level calculates the actual speed of the vehicles. The third level makes it possible to decide the change of the status of the traffic light.

Tari et al.[22] presented an approach to control very complex traffic intersections with fuzzy hierarchical rules.
Oliveira and Bazzan [23] proposed an intersection control approach based on agents who engage in a coordination protocol to collectively decide which direction should be prioritized.

Abdulhai et al. [5] applied the Q-learning algorithm to optimize the control signal of a single intersection. The objective involves the optimal control of traffic severely disrupted.

Ferreira et al. [24] proposed a multi-agent strategy to control a network of urban traffic. Each agent is responsible for managing the signals of an intersection through current traffic queue lengths.

Bingham [25] proposed rules based on fuzzy logic that allocates green time based on the number of vehicles. The goal of learning is to minimize the delay caused by vehicles of signal control policy.

Thorpe [26] applied a RL named SARSA for control of traffic signals. SARSA minimizes total travel time and vehicle waiting times.

Mikami and Kakazu [27] proposed a method for the cooperation of traffic lights. The idea is that each agent performs reinforcement learning and declares its evaluation; combinatorial optimization is performed simultaneously to find appropriate parameters for long-term learning that maximize the total benefit of the signals.

3. Theoretical framework

Considering an observer who counts the number of vehicles stopped at a stop line of an intersection at a given time; the data collected by the observer over a period represents a stochastic process. The random variable is the number of vehicles. The observed number of vehicles can take different values at different times. In addition, a stochastic process may represent the variation in the number of vehicles waiting in a queue over time. This number changes when a new vehicle joins the queue or when a vehicle leaves the queue. At any time, the number of vehicles in the queue describes the state of the system. The prediction of the number of vehicles in a queue at the next moment will depend only on the current state of the system (i.e. the number of vehicles in the queue). Therefore, the traffic management problem can be regarded as a Markov decision process MDP. The MDP provides a mathematical framework for modeling the decision-making process in Markov processes. A MDP is defined by:

- A set of states s;
- A set of actions A (s) when I am in state s;
- A transition model T (s', (s, a)) where aεA (s);
- A reward function R (s) that informs about the utility of being in state s;
- The value function

$V(s)=R(s)+\gamma\Sigma s'\epsilon SP(s'(s,\pi(s))V(s'))$.

Indeed, the main objective of the MDP is to find an optimal policy to adopt. In this sensé, there are several ways to solve the CDM: dynamic programming and reinforcement learning.

In what follows, we popularize the notions of these methods to derive the relation between them.

Illustrative example

Usually, each specie reacts in their own environment by taking an action. In order to react in the environment we have two approaches. The first one is to choose from the beginning a better action, in other words, select an optimal action defined as:

$$\begin{tikzcd} S \arrow[r] & A \\ v(s) \arrow[r] & a^* \end{tikzcd}$$

Where S is the set of states and A the set of actions.

To find this optimum, there is the dynamic programming algorithm (see next section). Among the disadvantages of this algorithm is that it assumes the existence of a perfect model of the environment.

The second approach is not to choose from the beginning an optimal action, but a random action. After tests and experiments, we can reach an optimal action or one close to the optimality f (s, a). The operation of this approach is that the agent memorizes the action taken and then he improves it. In another way, the agent learns from experiences. Therefore, we will have the following equation:

Experience (t) = Experience (t-1) + 1/c * Learning (1)

c is the number of visits of the pair (s, a).

Over time the variable c increases, so the term 1 / c * Learning tends to 0. This means that the agent no longer learns, but he exploits his experiences and knowledge. Where the reinforcement learning is inspired.

The figure below illustrates the evolution of the agent by adopting this approach.

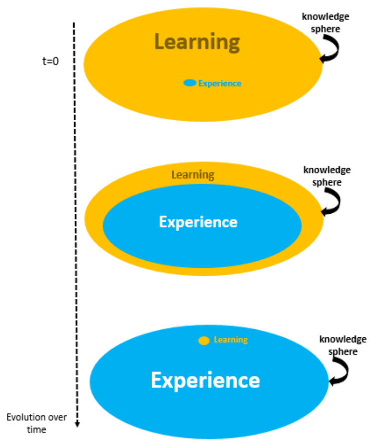


Fig -1: Agent's evolution over time

The figure above shows the importance of learning to accumulate experience, and reflects the intelligence of the agent over time. In addition, the success of this approach lies in the way of learning, which depends on the choice of action in each step. As a result, the finding that we can rise is that the learning is essential for the success of this approach, because it depends on the choice of action in each step and the quality of memorization of the experience. In other words, does the agent benefit from the last experience? Or does he look at a learning horizon? For more details, refer to the next section.

The relation between the two approaches mentioned above, is that the first equals the maximum of the second:

$$\mathbf{v\ (s) = max\ f\ (s, a)} \qquad \mathbf{(2)}$$

3.1 Dynamic Programming DP

- Value iteration algorithm

The value iteration algorithm is a dynamic programming technique that used to find the optimal policy. It is an iterative method that acts on the optimal policy by finding the first optimal value function, and subsequently the corresponding optimal policy. The basic idea is to set up a series of functions, such that this sequence converges on an optimal function of the state-value [28]. One of the drawbacks of this algorithm is that it presupposes the existence of a perfect model of the environment, represented by the matrices of transition probabilities and rewards, which is not the case for traffic networks[15]. In addition, this algorithm cannot be practical for large-scale problems, since it involves calculations on all the states of the MDP at each iteration, and it requires a large amount of memory to store large matrices (probabilities transition and rewards).

Policy iteration is another dynamic programming technique that is used to find the optimal policy [29].

3.2 Reinforcement Learning RL

The reinforcement learning RL [30] as its name suggests, is to learn from experiences what to do in different situations (called policy) based on the success or failure observed (reinforcements or rewards) to optimize quantitative rewards over time.

The RL takes a few simple key concepts based on the fact that the intelligent agent:

- Observes the effects of its actions ;
- Inferred from his observations the quality of his actions ;
- Improves future actions.

In reinforcement learning RL [31], the learning agent interacts firstly with an unknown environment and modifies its action policies to maximize its cumulative gains. Thus, RL provides an effective framework for solving control-learning problems that are difficult or even impossible. Indeed, the "intelligent" agent decides to perform an action based on its status to interact with its environment. The environment sends back a reward in the form of a positive or negative value and then it executes the action. The figure below illustrates the different existing interactions between the agent and the environment.

Note that the RL agent obtains only the evaluation of the actions implemented recently since it is modeled by a MDP.

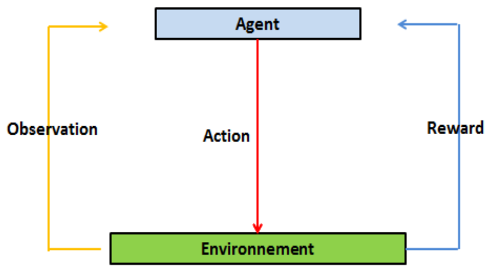


Fig -2: Interactions between agent and environment

3.3 RL system's components

Reinforcement learning ‘components are the state, the action and the reward. A microscopic traffic simulator is considered as the traffic environment. In addition, RL’s elements are modeled through this simulator. Note that there are several simulators in the literature. A description and comparison of microscopic simulators for road traffic are described in the article [32]. In our study, we will propose our own environment.

System’s state

In what follows, we define the system’s state across several sub-states to minimize the average vehicle delay, minimize vehicle waiting times at intersections, and reduce the occupancy rate on lanes (number of vehicles compared to the size of the network). These sub-states consider one of the following representations that are studied separately in the literature as follows:

- State Definition 1: Queue length

This is the most common definition in the literature for signal control based RL[5], [15], [33]. This definition is represented by the following equation:

$$s_i^t = max_{l \in L_i} q_l^t \qquad (3)$$

Where Li is the length of the way i.

- State Definition 2: Queue length at the red phase or arrival of the vehicles at the green phase.

In this definition, one of the key elements of the state is the maximum of the queue length q_i^t the red phase, and the maximum arrivals in the green phase Ar_i^t. The authors [23], [26], [34]–[36] adopt a similar definition. This definition is represented by the following equation:

$$S_i^t = \begin{cases} \mathrm{Max}_{i\in Li} q_i^t & \text{if i = red phase} \\ \mathrm{Max}_{i\in Li} A_i^t & \text{if i = green phase} \end{cases} \quad (4)$$

Where Li is the length of the way i .

- State Definition 3: Time interval between vehicles "GAP”

The time interval between two consecutive vehicles will be reduced when the traffic flow increases. A similar definition is defined by [37]. This definition is represented by the following equation:

$$\mathrm{GAP} = \begin{cases} 1 & \text{if } \mathrm{GAP} \le \mathrm{GAP}_{th} \\ 0 & \text{else} \end{cases} \quad (5)$$

- State Definition 4: Occupancy status of the way i "Occ".

It indicates the occupation ‘state on the way i, if Occi = 1, it means that the way i is saturated. This definition is represented by the following equation:

$$\mathrm{Occ} = \begin{cases} 1 & \text{if vehicles are getting closer to the detector} \\ 0 & \text{else} \end{cases} \quad (6)$$

Actions definition

Action’s choice must appropriate to the current situation at each learning step. For the moment, we have focused on the following actions: The first concerns the extension of the minimum green time. This action is performed when the occupancy status of the vehicle arrival lane is superior to that of the red phase, and the second is to switch the green phase to red. In fact, the entries of this module are the state of the environment, the current policy of the agent and the selection strategy. The output is the next action of the agent. Indeed, the agent needs good experiences to learn the best policy. It is in the sense that we find reinforcement learning algorithms that require a balance between exploitation and exploration in action selection strategies for [29]. The role of exploration is to enable the agent to learn enough to make good decisions about optimal actions. The simplest action rule is to select the action (or one of the actions) with the highest estimated state-action value (greedy behavior). This is why we find several strategies to balance significantly the exploration and exploitation compromise. In what follows, we show the strengths and weaknesses of each strategy to understand how it works. There are several strategies for updating the action. We cite:

- Greedy approach

The common goal of all reinforcement learning algorithms is to maximize the reward over time. A simple approach to ensuring that the agent takes optimal action at all times is to choose the action that yields the highest reward. Where does the name "Greedy" come from. As a result, the agent uses his current knowledge of the environment to act. We deduce that the greedy approach is 100% exploitation. However, the problem based on such an approach will never explore new actions.

- Random approach

This approach is the opposite of the greedy approach. It is simply taking a random action, which means that we have an equal probability between all actions. As a result, the random approach is 100% exploration.

- Є-greedy approach

This approach gathers the two previous approaches: greedy and random. It is one of the most used methods to balance between exploration and exploitation [29]. In this method, the behavior is greedy. In most cases, the agent chooses the action that gives the maximum action-state value, except in some cases it chooses a random action. The ε in ε-greedy is an adjustable parameter that varies between 0 and 1. It determines the probability of taking a random action.

The probability of this random behavior is ε. The major disadvantage of the ε-greedy method is that it gives equal priority to all actions during the exploration. It is possible to choose the worst action instead of choosing the next best action [6].

- Softmax approach

This approach varies the probabilities of action according to the estimated value. During exploration, ε -greedy gives an equal priority to all actions, while Softmax involves choosing an action with weighted probabilities. In other words, the agent ignores the actions that he considers suboptimal, and pay more attention to potentially promising actions, i.e., the action that the agent thinks is optimal is the most likely to choose. In general, the Boltzman distribution is used to define this probability. The main role of the Boltzman distribution is to choose an action with weighted probabilities instead of always taking the optimal action, or taking a random action. The probability of choosing the action in state s is:

$$P(a \mid \text{state}=s)=\frac{\exp(\frac{Q(s,a)}{\tau})}{\sum_{b=1}^{\text{all actions}}\exp(\frac{Q(s,b)}{\tau})} \quad (7)$$

Where τ a positive parameter named temperature. This parameter controls the propagation of the Boltzman distribution, so that all actions are considered equal at the beginning of the experiment, and over time, they are dispersed. More the temperature increases, more likely to choose one of the actions is almost equal (i.e. more exploration). On the other hand, a small value of the temperature, the actions will be selected proportionally to their estimated gain (i.e. more exploitation).

- Є-Softmax approach

The Є-Softmax approach consists of combining the two approaches Є-greedy and Softmax, proposed by Wahba [38]. This approach consists of a mixture of greedy and softmax models, with 1-ε probability of exploitation and ε probability of exploration

When exploiting,

$$P_s(a)=\begin{cases}1 & Q(s,a)=max_{a'\in A}\,Q(s,a') \\ 0 & otherwise\end{cases} \quad (8)$$

When exploring,

$$P_s(a)=\frac{e^{Q(s,a)/\tau}}{\sum_{b\in A}e^{Q(s,b)/\tau}} \quad (9)$$

In this method, the greedy action always has the highest probability of selection, and all other actions are weighted according to their estimated values [39].

Reward definition

The reward function indicates what is good in an immediate sense. Three definitions are considered in the literature.

- Reward Definition 1: Minimizing the delay

The major objective of the reward is to maximize the cumulative future reward. In this definition, we have to minimize the negative value of the delay. Indeed, the reward function considers the vehicle delay negative value of two successive decision points [5], [34], [40].The disadvantage of this definition is that it doesn't consider how long the vehicles have been delayed before the last two decision points.

- Reward Definition 2: Maximizing the Reduction in the Total Cumulative Delay

This definition considers the reward as the reduction of the cumulative total delay. In other words, the difference between the total cumulative delays of two consecutive decision points [20]. If the reward has a positive value, this implies that the delay of the trip is reduced by executing the previous action. Similarly, a negative reward value indicates that the chosen action results in a relative increase in the total delay of the trip [37]. However, this definition does not guarantee the prevention of total or partial blocking in ntersections. Which will lead to jeopardize the upstream intersections.

- Reward Definition 3: Minimizing and balancing the queue

This reward is defined as the reduction of the sum of the squares of the maximum queue length. The aim is to minimize and balance the queue length through the different phases[23], [35]. This definition is represented by the following equation:

$$r^t=\sum_{i\in N}(max_{i\in L_i}q_i^t)^2-\sum_{i\in N}(max_{i\in L_i}q_i^{t-1})^2 \quad (10)$$

3.4 RL algorithms

Numerous RL algorithms have been studied in the literature [29], [41], [42].

The most relevant algorithms are the temporal difference learning algorithms TD. They allow you to learn directly from experience without the need for a dynamic model of the environment. Two types of TD algorithms are cited in the literature[29], namely TD (0) and TD eligibility traces (λ). The TD (λ) is a learning algorithm invented by Richard S. Sutton. The agent looks forward to the end of the learning horizon and updates the value functions based on all future rewards. We will be interested in TD (0) for the rest of the article. TD (0) looks like dynamic programming, where the agent looks ahead a step in time and updates the value functions based on immediate reward. We find:

- SARSA algorithm

SARSA means State-Action-Reward-State-Action. It is an on-policy algorithm because it starts with a simple policy that improves after determining the sample of the state space. In the SARSA algorithm, the estimates of the values of the pair (state-action) are carried out starting from the experiment. The state-action value functions are updated after the execution of the actions, which are selected by the current policy [15]. The disadvantage of this algorithm is that the update formula follows the choice of the action (which is not always optimal) [43]. In addition, the SARSA algorithm does not perform better than fixed signal plans when tested in different levels of congestion [6].

- Q-learning algorithm

It is an " off-policy " algorithm since it collects the information in a random way, then evaluates the states by reducing slowly the randomness. Q-learning is one of the most widely used reinforcement learning methods because of its simplicity. It deals only with the Q value function (s, a) iteration independently of the policy, and does not require a complete model of the environment [28]. Q-learning is used for episodic tasks. The advantage of Q-Learning is that it takes into account that the policy changes regularly, which makes it more effective. Q-learning is more efficient in all levels of congestion. In addition, it works better than signal plans in all levels of congestion [6].

The difference between SARSA and Q-learning is: in SARSA, the update formula exactly follows the choice of action (which is not always optimal). In the case of Q-learning, the update formula uses the optimal value of the possible actions after the next state.

In order to illustrate and highlight the different RL techniques mentioned above, we will apply them in the case of an isolated intersection. The objective of this article is to make a rapprochement between the different action selection strategies namely Є-greedy, Softmax, using the Q-learning algorithm. In order to deduce which of these methods we will adopt for our overall architecture.

4 Case study

This section presents a comparison study of the key parameters of a simple RL-based signal control problem for a single intersection. We will then analyze the effect of the design parameters of RL namely the exploration methods by applying Q-learning. Note that this case study is based on data from one of Samah el-Tantawy's examples [39]. Indeed, the traffic light consists of a simulated signal in two phases controlling the two roads intersection. One is considered a major street and the other a minor street. Vehicles arrivals at the minor street at a given time t is not known a priori. Therefore, the minor street queue length is random. At different times, observations on the queue length of this street are considered a stochastic process.

4.1 State definition

Typically, for the case of an isolated intersection, the available information includes the queue lengths for each roadway [5]. For that, we opted for the first state definition, which is the queue length. Therefore, the random variable represents the queue length on the minor street. Thus, the state space of the random variable will have two elements: S = {s1, s2}, where s1 is the state in which the number of cars in the queue is less than or equal to 5, and s2 is the state in the case where the queue length is superior than 5. At t = 0, s0 = s1 it means that the number of cars in the queue is low at the beginning. After observing the process over time, the set of transition probabilities is calculated. T1 and T2 are the transition probabilities associated with a1 and a2 respectively:

$$T^1 = \begin{array}{c c} & \begin{array}{cc} s_1 & s_2 \end{array} \\ \begin{array}{c} s_1 \\ s_2 \end{array} & \begin{bmatrix} 0{,}5 & 0{,}5 \\ 0{,}1 & 0{,}9 \end{bmatrix} \end{array} \qquad T^2 = \begin{array}{c c} & \begin{array}{cc} s_1 & s_2 \end{array} \\ \begin{array}{c} s_1 \\ s_2 \end{array} & \begin{bmatrix} 0{,}9 & 0{,}1 \\ 0{,}7 & 0{,}3 \end{bmatrix} \end{array}$$

4.2 Action definition

At each state, we have two actions: A = {a1, a2}. Action a1 denotes the extension of the green phase for the major street (for example 30 sec). Action a2 represents the allocation of the green light to the minor street (for example 15 seconds), then return the green to the major street (for example, 15 seconds).

4.3 Reward definition

In our case, the reward is considered as a penalty, because it is the total delay sustained by the vehicles between the successive decision points. Thus, the reward function is defined as the squared of the negative queue lengths sum. It aims to minimize and balance the queue lengths. The goal is to find the control policy that maximizes the expected cumulative reward. R1 and R2 are the reward matrices associated with a1 and a2 respectively:

$$R^1 = \begin{array}{c c} & \begin{array}{cc} s_1 & s_2 \end{array} \\ \begin{array}{c} s_1 \\ s_2 \end{array} & \begin{bmatrix} -1 & -16 \\ -10 & -50 \end{bmatrix} \end{array} \qquad R^2 = \begin{array}{c c} & \begin{array}{cc} s_1 & s_2 \end{array} \\ \begin{array}{c} s_1 \\ s_2 \end{array} & \begin{bmatrix} -10 & -50 \\ -16 & -1 \end{bmatrix} \end{array}$$

4.4 Exploration methods

In a first moment, and in order to visualize the results of the different actions selection in RL, we performed simulations using the VB-EXCEL tool. Our concern is to position ourselves more on the side of the exploration techniques mentioned before. We eventually tested each of the following approaches:

- Є-Greedy

Let the exploration rate ε equal to C / t, where C is a constant = 0.9 and t is the iteration number. In the first iterations, we usually perform explorations since our experience in the environment is zero. Over time, the value of ε will converge to 0 in a fast way, which will lead more to exploitation. That's why we proposed a second value of ε defined by C / log (10.t) to slow down this convergence.

Pseudo Code

```
Rnd = random number [0,1]
Є = exploration rate
If Rnd <Є then
  Action = an action chosen randomly from all actions
If not
  Action = an action chosen in random among the actions that maximize Q (st)
End if
```

- Softmax

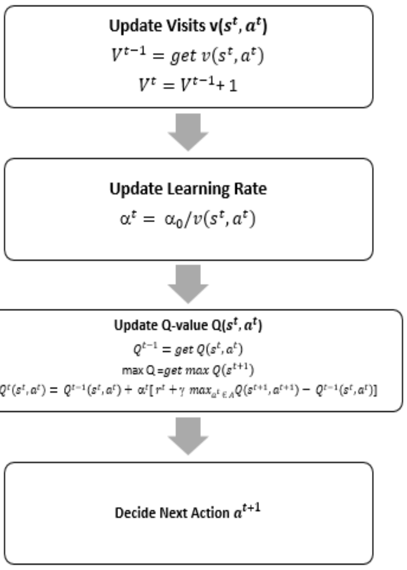
As explained in the previous section, this approach depends on a parameter τ (temperature), in this case study, we tested 3 values, τ = 1 to do more exploitation, τ= 10 and τ= 50 to make more exploration.

Pseudo Code

```
Vp = Boltzmann probabilities vector generated from the values of Q (st, a)
Vp_c = Cumulative probabilities vector of Vp (distribution function)
Rnd = random number [0,1]
i = 1
while Rnd> Vp_c (i)
i = i + 1
action = action [i]
```

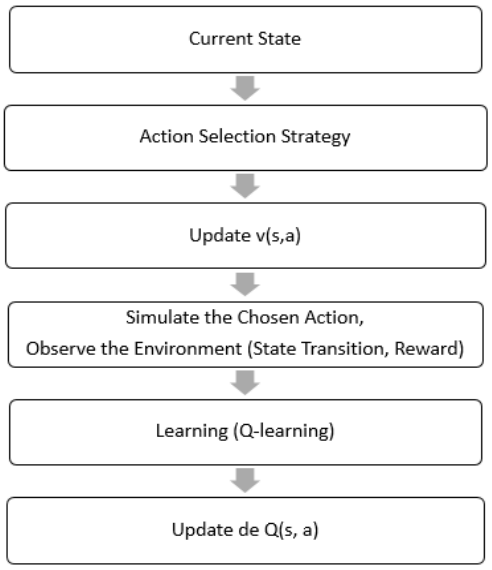
4.5 Learning Method

As discussed in the above, Q-learning offers a significant advantages over dynamic programming and SARSA, i.e., Q-learning does not require a predefined model of the environment, on which action selection is based [5]. As a result, we tested the exploration methods using the Q-learning algorithm. In this case study, the discount rate equals 0.8. The learning rate α is defined as follows: B / v (s, a), where B is a constant = 0.2 and v is the number of visits of the pair (state, action). The Q-learning process is as follows:


Update Visits $v(s^t, a^t)$
$V^{t-1} = get\ v(s^t, a^t)$
$V^t = V^{t-1} + 1$
Update Learning Rate
$\alpha^t = \alpha_0 / v(s^t, a^t)$
Update Q-value $Q(s^t, a^t)$
$Q^{t-1} = get\ Q(s^t, a^t)$
max Q = $get\ max\ Q(s^{t+1})$
$Q^t(s^t, a^t) = Q^{t-1}(s^t, a^t) + \alpha^t[r^t + \gamma\ max_{a^t \in A} Q(s^{t+1}, a^{t+1}) - Q^{t-1}(s^t, a^t)]$
Decide Next Action a^{t+1}

4.6 Steps to follow

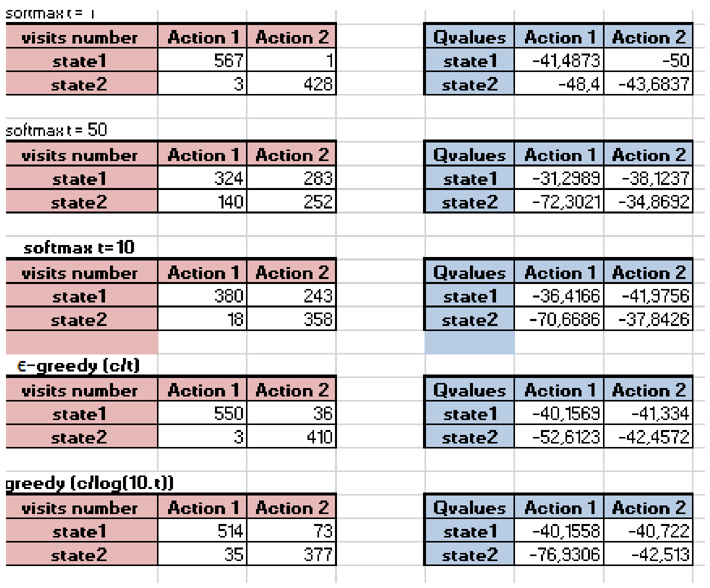
At each iteration, we will adopt the following approach


Current State
Action Selection Strategy
Update v(s,a)
Simulate the Chosen Action,
Observe the Environment (State Transition, Reward)
Learning (Q-learning)
Update de Q(s, a)

4.7 Result and analysis of the simulation

We opted for 1000 iterations in order to have results that are more significant.

After the simulations made on the different approaches, the first observation is that all these methods led to the same result. Indeed, if we are in state s1, we must take action a1 and if we are in state s2 we must take action a2. However, the balance between exploration and exploitation is not always assured:



softmax t = 1

visits number	Action 1	Action 2
state1	567	1
state2	3	428

Qvalues	Action 1	Action 2
state1	-41,4873	-50
state2	-48,4	-43,6837

softmax t = 50

visits number	Action 1	Action 2
state1	324	283
state2	140	252

Qvalues	Action 1	Action 2
state1	-31,2989	-38,1237
state2	-72,3021	-34,8692

softmax t=10

visits number	Action 1	Action 2
state1	380	243
state2	18	358

Qvalues	Action 1	Action 2
state1	-36,4166	-41,9756
state2	-70,6686	-37,8426

Є-greedy (c/t)

visits number	Action 1	Action 2
state1	550	36
state2	3	410

Qvalues	Action 1	Action 2
state1	-40,1569	-41,334
state2	-52,6123	-42,4572

greedy (c/log(10.t))

visits number	Action 1	Action 2
state1	514	73
state2	35	377

Qvalues	Action 1	Action 2
state1	-40,1558	-40,722
state2	-76,9306	-42,513

In Є-greedy where Є = C / t. The pair (s2, a1) constitutes only 3/1000 of the cases visited. We observe that this possibility has not been explored sufficiently to confirm our final conclusion, in addition, the selected exploration rate C/ t converges rapidly to 0, for example from the 20th iteration, the probability of performing explorations is less than 4.5%. Thus the model goes into operation mode very quickly. To remedy this problem, we thought to slow down this convergence by opting for a new function Є = C / log (10 * t). With this value, the probability of exploring in the 20th iteration and 39% against 4.5% of the first value. We therefore ensure balance; we can say that our system has explored enough before confirming the choice of appropriate actions.

In Softmax τ= 1, the model exploits too much without exploring, unlike τ = 50 we find too much exploration, for τ = 10, the equilibrium is ensured if we are in state s2. However, if one is in the state s1, the system performs too much exploitation since the 2 values of Q are close.

5 Conclusion and perspectives

Reinforcement learning method offers significant results in real-time road traffic management. In this paper, we have examined the existing work in the literature concerning RL-based signal control systems; we have also simplified these concepts. The Q-learning learning algorithm has been implemented, and tested with the different action selection policies. In this paper, we presented a case study of a two-phase isolated traffic signal to test several values for each of the parameters of Є-greedy and Softmax. The results of the simulation showed that the balance is ensured between exploitation and exploration for the Є-greedy method with Є = C / log (10 * t). Softmax guarantees this equilibrium with the value of the temperature τ = 10. One of the limitations of this study is that demand levels are constant during the evaluation process. Therefore, the next step will be dedicated to performing a simulation of a more complex network using the open source simulator SUMO. We will test thereafter all the action selection approaches namely: Є-greedy, Softmax and Є-Softmax. In addition, future research will consider more representative state and actions definitions. The multi criteria reward functions are also intended to be considered.

REFERENCES

[1] M. Rezzai, W. Dachry, F. Mouataouakkil, and H. Medromi, 'Ville intelligente: un nouveau paradigme.', presented at the Communication aux journées doctorales JDTIC, Morocco, 2014.

[2] N. Shah, S. Kumar, F. Bastani, and I.-L. Yen, 'Optimization models for assessing the peak capacity utilization of intelligent transportation systems', Eur. J. Oper. Res., vol. 216, no. 1, pp. 239–251, Jan. 2012.

[3] M. Rezzai, W. Dachry, F. Mouataouakkil, and hicham medromi, 'Designing an Intelligent System for Traffic Management', Journal of Communication and Computer, 2015.

[4] B. Abdulhai and L. Kattan, 'Reinforcement learning: Introduction to theory and potential for transport applications', Can. J. Civ. Eng., vol. 30, no. 6, pp. 981–991, Dec. 2003.

[5] B. Abdulhai, R. Pringle, and G. J. Karakoulas, 'Reinforcement learning for true adaptive traffic signal control', J. Transp. Eng., vol. 129, no. 3, pp. 278–285, 2003.

[6] H. M. Abdul Aziz, F. Zhu, and S. V. Ukkusuri, 'Reinforcement learning based signal control using R - Markov Average Reward Technique (RMART) accounting for neighborhood congestion info rmation sharing', presented at the Transportation Research Board Conference and publication in Transportation Research Record, 2012.

[7] J. C. Medina and R. F. Benekohal, 'Reinforcement Learning Agents for Traffic Signal Control in Oversaturated Networks', 2011, pp. 132–141.

[8] M. J. Moghaddam, M. Hosseini, and R. Safabakhsh, 'Traffic light control based on fuzzy Q-leaming', 2015, pp. 124–128.

[9] F. Zhu, H. M. A. Aziz, X. Qian, and S. V. Ukkusuri, 'A junction-tree based learning algorithm to optimize network wide traffic control: A coordinated multi-agent framework', Transp. Res. Part C Emerg. Technol., vol. 58, pp. 487–501, Sep. 2015.

[10] A. Stevanovic, J. Stevanovic, J. So, and M. Ostojic, 'Multi-criteria optimization of traffic signals: Mobility, safety,

and environment', Transp. Res. Part C Emerg. Technol., vol. 55, pp. 46–68, Jun. 2015.

[11] T. Le, P. Kovács, N. Walton, H. L. Vu, L. L. H. Andrew, and S. S. P. Hoogendoorn, 'Decentralized signal control for urban road networks', Transp. Res. Part C Emerg. Technol., vol. 58, pp. 431–450, Sep. 2015.

[12] Z. Cong, B. De Schutter, and R. Babuška, 'Co-design of traffic network topology and control measures', Transp. Res. Part C Emerg. Technol., vol. 54, pp. 56–73, May 2015.

[13] B. Cesme and P. G. Furth, 'Self-organizing traffic signals using secondary extension and dynamic coordination', Transp. Res. Part C Emerg. Technol., vol. 48, pp. 1–15, Nov. 2014.

[14] D. Pescaru and D.-I. Curiac, 'Ensemble based traffic light control for city zones using a reduced number of sensors', Transp. Res. Part C Emerg. Technol., vol. 46, pp. 261–273, Sep. 2014.

[15] S. El-Tantawy and B. Abdulhai, 'Multiagent Reinforcement Learning for Integrated Network of Adaptive Traffic Signal Controllers (MARLIN-ATSC): Methodology and Large-Scale Application on Downtown Toronto', 2013.

[16] Madhavi Arora and D. V. . Banga, 'Intelligent Traffic Light Control System using Morphological Edge Detection and Fuzzy Logic', presented at the nternational Conference on Intelligent Computational Systems (ICICS'2012), Dubai, 2012.

[17] M. Keyarsalan and G. Ali Montazer, 'Designing an intelligent ontological system for traffic light control in isolated intersections', Eng. Appl. Artif. Intell., vol. 24, no. 8, pp. 1328–1339, Dec. 2011.

[18] M. Abdoos, N. Mozayani, and A. L. Bazzan, 'Traffic light control in non-stationary environments based on multi agent Q-learning', in Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on, 2011, pp. 1580–1585.

[19] Y. Dujardin, F. Boillot, D. Vanderpooten, and P. Vinant, 'Multiobjective and multimodal adaptive traffic light control on single junctions', 2011, pp. 1361–1368.

[20] I. Arel, C. Liu, T. Urbanik, and A. G. Kohls, 'Reinforcement learning-based multi-agent system for network traffic signal control', IET Intell. Transp. Syst., vol. 4, no. 2, p. 128, 2010.

[21] M. Shakeri, H. Deldari, A. Rezvanian, and H. Foroughi, 'A novel fuzzy method to traffic light control based on unidirectional selective cellular automata for urban traffic', 2008, pp. 300–305.

[22] T. Tari, L. T. Koczy, C. Gaspar, and J. Hontvari, 'Control of Traffic Lights in High Complexity Intersections Using Hierarchical Interpolative Fuzzy Methods', 2006, pp. 1045–1048.

[23] D. De Oliveira, A. L. Bazzan, and V. Lesser, 'Using cooperative mediation to coordinate traffic lights: a case study', in Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems, 2005, pp. 463–470.

[24] E. D. Ferreira, E. Subrahmanian, and D. Manstetten, 'Intelligent agents in decentralized traffic control', 2001, pp. 705–709.

[25] E. Bingham, 'Reinforcement learning in neurofuzzy traffic signal control.', 2001.

[26] T. L. Thorpe and C. W. Anderson, 'Traffic light control using sarsa with three state representations', Citeseer, 1996.

[27] S. Mikami and Y. Kakazu, 'Genetic reinforcement learning for cooperative traffic signal control', 1994, pp. 223–228.

[28] Onivola Henintsoa Minoarivelo, 'Application of Markov Decision Processes to the Control of a Traffic Intersection', University of Barcelona, Spain, 2009.

[29] R. S. Sutton and A. G. Barto, Introduction to reinforcement learning. London, England: Cambridge, Massachusetts, 1998.

[30] T. M. Mitchell, 'Does machine learning really work?', AI Mag., vol. 18, no. 3, p. 11, 1997.

[31] X. Xu, L. Zuo, and Z. Huang, 'Reinforcement learning algorithms with function approximation: Recent advances and applications', Inf. Sci., vol. 261, pp. 1–31, Mar. 2014.

[32] M. MACIEJEWSKI, 'A comparison of microscopic traffic flow simulation systems for an urban area', Transp. Probl., vol. 5, no. 4, pp. 27–38, 2010.

[33] S. Richter, D. Aberdeen, and J. Yu, 'Natural actor-critic for road traffic optimisation', in Advances in neural information processing systems, 2007, pp. 1169–1176.

[34] M. Wiering, J. Van Veenen, J. Vreeken, and A. Koopman, 'Intelligent traffic light control', Inst. Inf. Comput. Sci. Utrecht Univ., 2004.

[35] E. Camponogara and W. Kraus, 'Distributed Learning Agents in Urban Traffic Control', in Progress in Artificial Intelligence, vol. 2902, F. M. Pires and S. Abreu, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 324–335.

[36] A. Salkham, R. Cunningham, A. Garg, and V. Cahill, 'A Collaborative Reinforcement Learning Approach to Urban Traffic Control Optimization', 2008, pp. 560–566.

[37] J. Jin and X. Ma, 'Adaptive Group-based Signal Control by Reinforcement Learning', Transp. Res. Procedia, vol. 10, pp. 207–216, 2015.

[38] M. M. A. A.-L. Wahba, 'MILATRAS: microsimulation learning-based approach to transit assignment', 2008.

[39] S. El-Tantawy, 'Multi-Agent Reinforcement Learning for Integrated Network of Adaptive Traffic Signal Controllers (MARLIN-ATSC)', University of Toronto, Canada, 2012.

[40] Shoufeng Lu, Ximin Liu, and Shiqiang Dai, 'Incremental multistep Q-learning for adaptive traffic signal control based on delay minimization strategy', 2008, pp. 2854–2858.

[41] C. Watkins and P. Dayan, 'Q-learning. Machinelearning', University of Toronto, 1992.

[42] A. Gosavi, Simulation-Based Optimization, vol. 55. Boston, MA: Springer US, 2015.

[43] B. Bouzy, 'Apprentissage par renforcement (3)', Cours D'apprentissage Autom., 2005.

BIOGRAPHIES

Author Maha REZZAI

Received a master degree in Engineering and Optimization of Transport and Logistics System in 2012 from Faculty of science FSAC. In 2013 she joined The system architecture team of the ENSEM. Maha's actual main research is Traffic Control System based on Multi-Agent System.

Author Wafaa DACHRY

Received his PhD. in information system engineering from the Hassan II University, Casablanca, Morocco, 2013. She is an Assistant Professor at Faculty of Science and Techniques, Hassan 1st University, Settat city, Morocco. Her main research focuses on IT Governance, reverse logistics and traffic control system based on multi-agent system.

Author Fouad MOUTAOUAKKIL

Received the PhD in computer science from the ENSEM, Hassan II University in November 2010, Casablanca, Morocco in 2005 he obtained the Engineer Degree in industrial automation from the ENSEM, Hassan II University, Casablanca, Morocco. In 2006 he joined the system architecture team of the ENSEM. His actual main research interests concern Remote Control over Internet Based on Multi agents Systems.

Author Hicham MEDROMI

Received his PhD in engineering science from the Sophia Antipolis University in 1996, Nice, France. He is responsible of the system architecture team of the ENSEM Hassan II University, Casablanca, Morocco. His current main research interests Concern Control Architecture of Mobile SystemsBased on Multi Agents Systems. Since 2003 he is a full professor for automatic productic and computer sciences at the ENSEM School, Hassan II University, Casablanca.