Swarm Reinforcement Learning for traffic signal control based on cooperative multi-agent framework

Mohammed TAHIFA, Jaouad BOUMHIDI, Ali YAHYAOUY
Computer Science Department, LIIAN Lab.
Sidi Mohamed Ben Abdellah University, Faculty of Science Dhar-Mahraz
Fez, 3000, Morocco.
Email: mohammed.tahifa@usmba.ac.ma, jaouad.boumhidi@usmba.ac.ma, ayahyaouy@yahoo.fr

***Abstract*—Congestion, accidents, pollution, and many other problems resulting from urban traffic are present every day in most cities around the world. The growing number of traffic lights in intersections needs efficient control, and hence, automatic systems are essential nowadays for optimally tackling this task. Agent based technologies and reinforcements learning are largely used for modelling and controlling intelligent transportation systems, where agents represent a traffic signal controller. Each agent learns to achieve its goal through many episodes. With a complicated learning problem, it may take much computation time to acquire the optimal policy. In this paper, we use a population based methods such as particle swarm optimization to be able to find rapidly the global optimal solution for multi-modal functions with wide solution space. Agents learn through not only on their respective experiences, but also by exchanging information among them, simulation results show that the swarm Q-learning surpass the simple Q-learning causing less average delay time and higher flow rate.**

Keywords*—multi-agent systems, reinforcement learning, traffic signal control, Q-learning, Particle swarm optimization.

I. Introduction

Nowadays, most cities in the world suffer from an extreme vehicular traffic that provokes severe problems like pollution, congestion, security and many others. the construction of a new infrastructure is expensive, thus the solution is to move towards the utilization of the existing resources like intelligent transportation systems (ITS) for traffic management and control. Traffic control contains a set of technics used to improve the traffic network performance by, for example, controlling the traffic flow to minimize congestion, waiting times, fuel consumption and avoid accidents, allowing vehicles to travel more quickly through the system. Traffic control generally uses different actors, such as, controlling the traffic signals in urban areas, enforcing variable speed limits, ramp-metering in highways, and using driver-assistance systems (e.g., adaptive cruise control). In this paper, we particularly focus on controlling traffic signals in urban areas. The use of multi-agents systems and artificial Intelligence (AI) are becoming more and more reliable in ITS, it allows to decompose the system into multiple agents that interact with each others to achieve a desired goal, improving the ability of a traffic signal to efficiently serve vehicles at an intersection, to reduce the delay experienced by vehicular, thus increasing average network speed.
We compare our proposed method to a multi-agent architecture with Q-learning algorithm controlling a network with junctions [1]. In this work, we try to enhance the performance of the Q-learning algorithm by introducing the concept of population-based method to find optimal policies rapidly. The simulation results show that swarm Q-learning outperforms the Simple Q-learning. In summary, the following main contributions are made in this paper Modelling a traffic network using multi-agent system where each agent represents a junction to decompose the traffic into a decentralized system. The development of a strategy control, based on swarm reinforcement learning which increase reliability and performance of the system. The rest of the paper is organized as follows: section 2 reviews the related works and preliminaries about multi-agent and learning model. The proposed learning method is presented in section 3, this section also includes review of Q-learning and swarm reinforcement learning. Experimental results are presented in section 4 and finally the paper is concluded in section 5.

II. Related Works

In transportation systems, many works [2] [3] used Multi-agents systems, as a sub-domain of artificial intelligence, in this section, we reference some related works that use control of traffic signals based on multi-agent systems in order to find an optimized control policy. In [4] [5], the system is composed of three types of agents trying to achieve a global optimal performance by means of coordination. It adapts itself to the environment based on internal rules. A centralized control, is presented in [6] composed of four types of agents, the efficiency of the system highly depends on the centralized control and therefore the disadvantages of centralized methods can be seen in this system. An individual traffic control is presented in [7], the system is modelled by coordinated intelligent agents. The agents coordinate among themselves based on the information received from each others to control the network.
In [1] and [8], they model a relatively large traffic network as a multi-agent system and use techniques from multi-agent reinforcement learning. [9] And [10] presented a hierarchical multi-agents system containing three layers of agent, and use techniques to be adapted dynamically to the change of the environment. The implementation of agents is based on feed-forward neural network and fuzzy logic theories. The system is hard to extend due to the computational cost of the implementation.
The learning techniques developed for multi-agent systems, can actually give contributions to the control and management

978-1-4799-7511-2/15/$31.00 ©2015 IEEE

of traffic systems in dynamic environments. Reinforcement Learning (RL) offers some advantages in dealing with dynamic environments. RL is a good solution to control single intersections and also network of intersections especially when there is no previous knowledge about the environment.
In [11], weiring describes the use of reinforcement learning by traffic signal agents in order to minimize the overall waiting time of vehicles in a small grid. The agent learns the value function on different setting of the traffic signals. The value functions are learned by traffic signal and vehicles.
Another model-based RL method is presented in [12] to minimize the total time of vehicles in the network. The traffic signals are controlled by the agents located in each intersection. The state representation is calculated from the waiting time of the individual vehicles. The network in the simulation includes fifteen intersections.
Particle Swarm optimizations have been used in [13]. They propose a Swarm intelligence approach to find successful cycle programs of traffic lights. The solution is evaluated in the context of two large and heterogeneous metropolitan areas located. In [14] they focus on computing a consistent traffic signal configuration at each junction that optimizes multiple performance indices, it includes minimizing trip waiting time, total trip time, they formulate the system as a multi-agent system and a multi-objective RL.

A. Q-learning

In an environment in which an agent has no previous knowledge, Q-learning is the best suited method for this case. It perceives the state of the environment, and receives data from it, and learns an optimal policy, by mapping states to actions. Q-learning algorithm was proposed by Watkins [15]. As known, is independent of existing knowledge of the environment.
Q-Learning is modelled by using Markov decision process formalism. The method quantifies the effectiveness of selecting a given action a perceived in a state s.

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha[r_{t+1} + \gamma Max_a Q(s_{t+1}, a_{t+1})] - Q(s_t, a_t) \tag{1}$$

The agent perceives the state s from the environment, selects an action a randomly and receives a reward r from the environment. This leads us to another state s_{t+1} due to a probability transition. The agents purpose is to link each state s to an action a which maximizes cumulative rewards over the time. The discount factor $\gamma \in [0, 1]$, awards greater weight to future reinforcements if set closer to 1. We represent the state action pair $Q(s, a)$ as the value of cumulative rewards from taking an action a in a state s and therefore representing the matrix Q of optimal policies. As the agent explore each state action pair an infinite time, Q-learning converges to an optimal decision policy for a finite Markov decision process.

B. Particle swarm optimization

Motivated from the social behaviour of birds within a flock, particle swarm optimization (PSO) is a population-based originally designed for continuous optimization problems. In

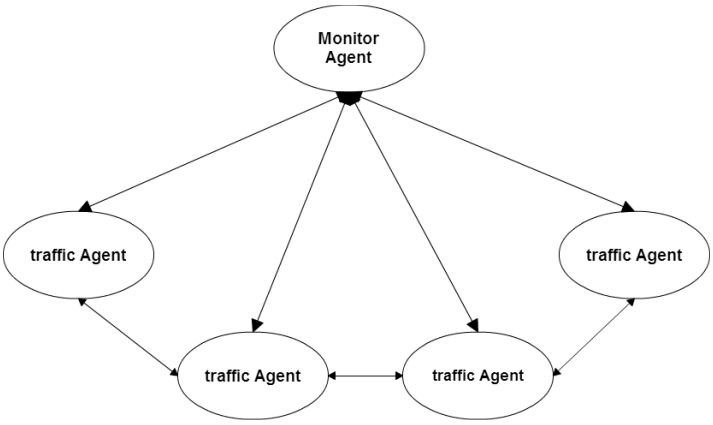

Fig. 1. multi-agent architecture

PSO, each possible solution to the problem is called particle position and the population of particles is called the swarm. In this algorithm, each particle position mentioned by x, is updated for each iteration g by means of Where term v^i_{g+1} is the velocity of the particle, given by the following equation:

$$\begin{aligned} v^i_{g+1} &= w.v^i_g + c_1 r_1.(p^i_g - x^i_g) + c_2 r_2.(b_g - x^i_g) \\ x^i_{g+1} &= x^i_g + v^i_{g+1} \end{aligned} \tag{2}$$

v^i_{g+1} : Velocity vector of particle g,
w : Weight parameter called inertia weight,
c_1, c_2 : Weight parameters called acceleration coefficients,
r_1, r_2 : Uniform random numbers in the range from 0 to 1,
p^i_g : Best solution found by particle g so far, which is called personal best,
b_g : Best solution found by all particles so far, which is called global best.
equation (2) describes the pseudo-code of PSO. The algorithm starts by initializing the swarm, which includes both the positions and velocities of the particles. The corresponding p^i of each particle is randomly initialized, and the leader b is computed as the best particle of the swarm. Then, during a maximum number of iterations, each particle moves through the search space updating its velocity and position. The particles are then evaluated, and their personal best positions p^i are also recalculated. At the end of each iteration, the leader b is also updated.

III. THE PROPOSED ARCHITECTURE AND METHOD

A. Proposed architecture

In this paper, we use a hierarchical structure to model the architecture of the system. The agents communicate with two types of interaction:
Intra-level: agents interacts with each others in the same level, it's referred as a horizontal interaction,
Inter-level: agents in different levels of hierarchy communicate with each others.
This is illustrated in Fig 1 by the interaction between traffic agents on the same level, by sharing information through swarm reinforcement learning. Furthermore, the communication between the monitor-agent and the traffic agents are inter-level.

TABLE I. STATE SPACE REPRESENTATION

state number	Edge Order	state number	Edge Order
1	$e_1 \geq e_2 \geq e_3 \geq e_4$	13	$e_2 \geq e_3 \geq e_1 \geq e_4$
2	$e_1 \geq e_2 \geq e_4 \geq e_3$	14	$e_2 \geq e_4 \geq e_1 \geq e_3$
3	$e_1 \geq e_3 \geq e_2 \geq e_4$	15	$e_3 \geq e_2 \geq e_1 \geq e_4$
4	$e_1 \geq e_4 \geq e_2 \geq e_3$	16	$e_4 \geq e_2 \geq e_1 \geq e_3$
5	$e_1 \geq e_3 \geq e_4 \geq e_2$	17	$e_3 \geq e_4 \geq e_1 \geq e_2$
6	$e_1 \geq e_4 \geq e_3 \geq e_2$	18	$e_4 \geq e_3 \geq e_1 \geq e_2$
7	$e_2 \geq e_1 \geq e_3 \geq e_4$	19	$e_2 \geq e_3 \geq e_4 \geq e_1$
8	$e_2 \geq e_1 \geq e_4 \geq e_3$	20	$e_2 \geq e_4 \geq e_3 \geq e_1$
9	$e_3 \geq e_1 \geq e_2 \geq e_4$	21	$e_3 \geq e_2 \geq e_4 \geq e_1$
10	$e_4 \geq e_1 \geq e_2 \geq e_3$	22	$e_4 \geq e_2 \geq e_3 \geq e_1$
11	$e_3 \geq e_1 \geq e_4 \geq e_2$	23	$e_3 \geq e_4 \geq e_2 \geq e_1$
12	$e_4 \geq e_1 \geq e_3 \geq e_2$	24	$e_4 \geq e_3 \geq e_2 \geq e_1$

TABLE II. ACTION SPACE REPRESENTATION

state number	Edge Order	state number	Edge Order
1	$33, 33, 13, 13$	11	$23, 33, 13, 23$
2	$33, 13, 33, 13$	12	$13, 33, 23, 23$
3	$33, 13, 13, 33$	13	$23, 23, 33, 13$
4	$13, 33, 33, 13$	14	$23, 13, 33, 23$
5	$13, 33, 13, 33$	15	$13, 23, 13, 33$
6	$13, 13, 33, 33$	16	$23, 23, 13, 33$
7	$33, 23, 23, 13$	17	$23, 13, 23, 33$
8	$33, 23, 13, 23$	18	$13, 23, 23, 33$
9	$33, 13, 23, 23$	19	$23, 23, 23, 23$
10	$23, 33, 23, 13$		

B. Proposed Method

The traffic network is considered as a system composed of 4 agents that each controls an intersection. We use Q-learning to learn these agents. In each learning iteration, the agents perceive the environment and receive statistical data in a fixed cycle. The sate estimation of the Q-learning is based on the average queue length of the edges in a junction on each cycle. Twenty-for is the number of states. We calculate them from the average queue length of the approaches links and rank them from max to min. We have four roads in each intersection, so the number of permutation is $4! = 24$, the state space is figured in Table I. We assume e_i is the i^th road of a junction. $e_1 > e_2 > e_3 > e_4$ Represents a state in which the value e_1 is the longest and e_4 is the shortest. If we have equality, we rank from the north following the direction of the clock.

The action space determines a plan for agents in such a way that each road has its own portion of green time. For each cycle, all available phases should at least appear once. So that waiting vehicles have enough time to pass the intersection. The calculation of the effective cycle length is given as:

$$\delta = n_{ex} * h_{ex} + n_{ph} * t_{min} \tag{3}$$

n_ph: Phases
t_min: Minimum of green time
n_ex: Number of Extensions
h_ex: fixed time length of green time
The action set determines how many extensions to add for each phases. To illustrate that, imagine a cycle length of 50 seconds, the cycle contains 3 phases, and 10 seconds as the minimum green time. $3 * 10$ Seconds is assigned to the phases, and let the action selection calculate the number of extensions to associate to each three phase. The action space representation are presented in Table II We mention by x_i the time split to add to each phase, this number is controlled by the action selection by:

$$d_i = t_{min} * x_i * h_{ex} \tag{4}$$

d_i Is the green time assigned to phase i.
To perceive if an action is suitable for a certain state, the existence of an accurate and efficient reward function is elemental. The reward function in this work is based on the number of vehicles waiting in an edge nbw_i, and the number of vehicles crossed the junction coming from an edge nbc_i, the number total of vehicles present in a junction is $nbtotal$
The reward function is given by :

$$r = \sum_{i=1}^{n} \frac{nbw_i * nbc_i}{nbtotal} \quad with \quad n = n_{ex} \tag{5}$$

A penalty is counted if the number of vehicles crossing the roads is superior to the number of vehicles that are waiting for it. To calculate the data of vehicles crossing the network, we place sensors and detectors at the start and the end of each road in the simulator.
For the exploration vs exploitation selection, we use an ϵ-greedy selection to select an action in Q-learning, we use a special ϵ-greedy based iteration, so the ϵ-greedy is favoured to explore for the first iterations, and so on, the ϵ start to decrease with the increase of iterations until $\epsilon = 0$.
To simulate this work, we use a network with four junctions, each one considered as a intelligent agent. The four-way junction is the most common and suitable to be considered in our approach. The figure 3 Illustrates the network used in this network. During each time step, new vehicles are generated by a uniform process; they are placed at the end of the queue of their respective destination lanes. Time intervals between two consecutive vehicles arrivals at input sections are sampled from a uniform distribution. We managed to operate this network with four phases; each phase had a minimum green time equal to 13 seconds. For the extension intervals, we have four 10-seconds extensions. i.e. $n_ex = 4$ and $h_ex = 10$, if we refer to (3) , the effective cycle length is equal to 92.To avoid conflict between two approaching links, and collisions in the junction, we add 2 seconds at the end of each phase 2. We have four phases, so 8 seconds are added to the effective cycle length = 100.

C. Proposed swarm reinforcement learning

The swarm reinforcement learning method is motivated by population-based methods in optimization problems as described before, the principal of the framework is as follows: In an environment where we have multiple agents learning with two learning strategies: each agent learn individually and learning through exchanging information. In this strategy, each one uses a reinforcement learning algorithm to extract its optimal policy. On the other hand, the agents exchange

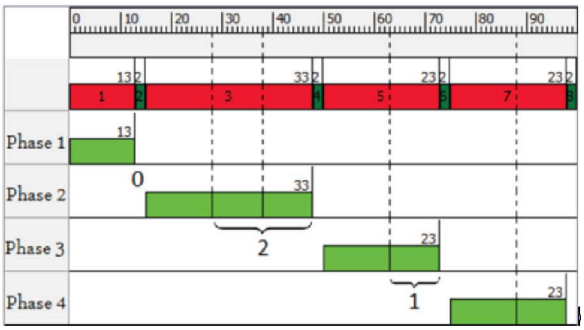

Fig. 2. Example of action space for 13, 33, 23, 23

information among them regularly during the individual learning and they learn based on the exchanged information. When learning with an ordinary reinforcement learning algorithm, it may take a useless action bringing a small reward, which makes learning time longer. Another issue with single learning came up when learning a non-stationary environment; the agent may not visit a State often time to explore all the possible actions, so the final optimal policy becomes a semifinal optimal policy. In the swarm reinforcement learning method, since the multiple agents are prepared, some of these agents could take useful actions bringing a larger reward. Because each agent learns based on information exchanged with the agents that takes the useful actions, it is expected that agents can acquire the optimal policy in a shorter learning time.

In the proposed update scheme, the personal best of each agent and the global best are determined by comparing evaluated values and are stored. Let the personal best P_i be the best Q-values found by the agents i so far, and let the global best G be the best Q-values found by all the agents so far. Each agent updates its Q-values by using these two kinds of best Q-values. We give the update equation as:

$$\begin{aligned} v_g^i &= w.v_g^i + c_1 r_1.(p_g^i - Q_g^i(s,a)) + c_2 r_2.(b_g - Q_g^i(s,a)) \\ Q_g^i(s,a) &= Q_g^i(s,a) + v_g^i \end{aligned} \tag{6}$$

To implement the scheme of updating Q-values of each agent, we have to consider which Q-values in the state-action table are updated by performing the update scheme. We propose a procedure in preforming the update scheme; and develop swarm reinforcement learning algorithms using this procedure. Since the evaluation of Q-values of each agent are determines at the end of episode and the personal best Q-values and the global best Q-values are calculated, it is appropriate to perform the update scheme just after this evaluation, we simply apply the update scheme to the Q-values presenting the actual state-action visited earlier, determine the personal best values, which will be the last value of this state-action. The global best value will be determined by comparing values of the same state-action of all agents and choosing the highest value.

To optimize the calculation, this procedure will be skipped if the global and personal best value are still equal to zero, another criteria is taken in consideration, when the global best will be equal to the personal best this will conduct to a velocity equal to zero, so there is no need to calculate it.

The algorithm of this procedure is in Algorithm (1). where

TABLE III. Q-LEARNING PARAMETERS

Parameter	description	value
β	Learning rate	0.9
γ	discount factor	0.9
ϵ	exploration rate	0.1

TABLE IV. NETWORK CONFIGURATION

Properties	Value
Number of junctions	4
Number of Edges	16
Average length of edges	600
Number of lanes per edge	2
Maximum speed	60Km/h
Centroids	8
Simulation duration	6h
Average Traffic flow	600 veh/hour

X^i is evaluated as the value of the Q-value of agent i. P^i represents the evaluated value for the personal best Q-values of agent i. G presents the value of the global best Q-value. $V(s,a)$ as the velocity. i the number of agents. T Total number of agents and t is the number of iterations.

Algorithm 1 Proposed swarm Reinforcement Learning algorithm

```
while t > T do
    calculate X^i with equation (1)
    calculate P^i and G
    if P^i != G^i and P^i > 0 and G^i > 0 then
        calculate V(s,a)
    end if
    if velocity > 0 then
        Q(s,a) ← X^i + V(s,a)
    else if then
        Q(s,a) ← X^i
    end if
end while
return Q(s,a)
```

IV. SIMULATION RESULTS

To implement a multi-agent system controlling a traffic environment, we need a multi-agent platform and a traffic simulator. There are a number of open-source agent platforms available in the literature, Which aid developers to build a complex agent system in a simplified fashion. In this work, we choose Jade multi-agent platform, fully implemented in the Java language. It simplifies the implementation of multi-agent systems through a middle-ware, which complies to the FIPA specifications.

For the traffic simulator, the open source SUMO simulation environment has been chosen for a number of reasons, including portability, presence of an active development community and availability of a graphical user interface. Furthermore, SUMO has been used by other intelligent systems researchers, such as [13] and [16]

The test of the method has been done on the network cited in Fig 3. It contains four junctions in a grid presentation with

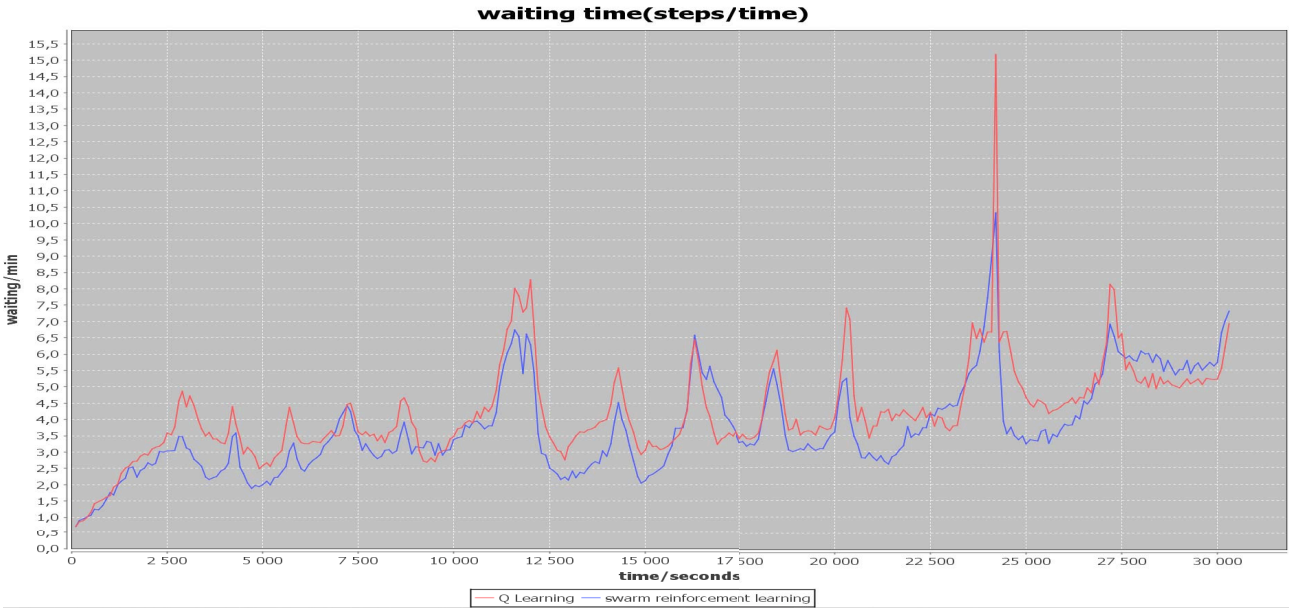

Fig. 4. Comparison between Swarm reinforcement learning and Q-learning

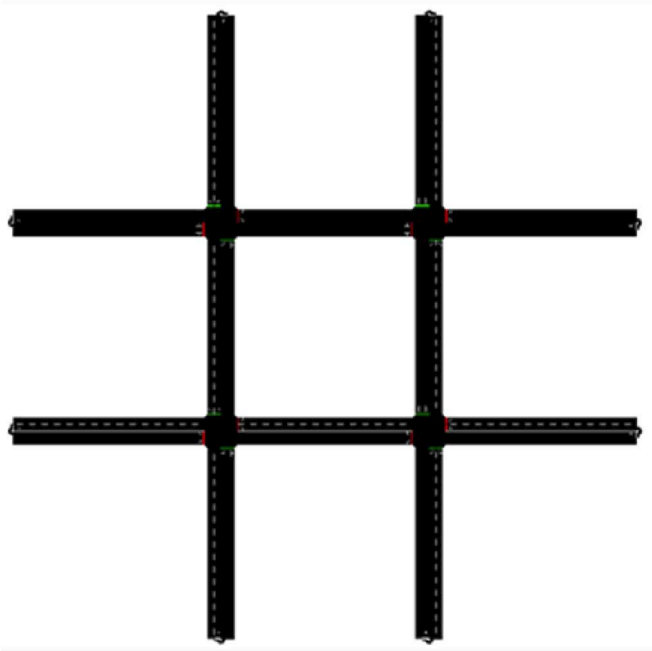

Fig. 3. Network used in this simulation

double lane edges. Table III shows the Q-learning parameters. We set the Learning rate to 0.9, to force delayed rewards have an impact on the learning. Exploration rate equals to 0.1, we use an e-greedy policy to favourite the system to select random actions in the beginning, the exploration rate starts to decrease as we move forward in the learning. Table IV present the network configurations, we choose a normal four way junctions with edges length equal to 600 m. The number of lanes in each edge are two and maximum speed of vehicles is equal to 60km/h. The duration of simulation is 6h and we let 600 vehicles/hour circulate inside the network to test the effectiveness of the learning system.

The proposed method is compared with the standard Q-learning. The aim of the experience is to calculate the waiting time of vehicles in the network, it determines how much time does a vehicle, to cross the network from an entry to an exit. We test the method for A medium traffic flow equal to 600veh/hour.

The performance of the proposed method is shown in Fig 4. The performances are equal for the first two hours. After that, the swarm reinforcement learning start to make difference and decrease the waiting time of vehicles in the network. the power of swarm-Q-learning comes from its ability of cooperation between agents, when each agent have some knowledge about his junctions, it start to share this knowledge to its neighbours and collecting informations from them. This can help each agent to complete its back-knowledge about his junctions without visiting each state-action. This contribution affect the time of learning of the system and start to give better result in a shortest time. The performances oscillate because of the increase and the drop of the vehicles flow to model a non-deterministic environment, the proposed method shows a reasonable waiting time of vehicles in the network.

V. Conclusion

In this paper, we demonstrated the effectiveness of the Swarm reinforcement learning in a non-deterministic traffic network. The proposed method gathers information from junctions in the network, share informations between agents and calculates an optimal policy to select the best action in different situations. The performance of the method is based on the waiting time of vehicles to cross the network. We use the average halting vehicles in approaching links to define the states space. The paper presents a method for cooperation between agents through exchanging informations about Q-values.

The experimental result shows that the proposed method

surpasses the standard Q-learning method.
Future works include the implementations of techniques to use the Q-values in a continuous space for a large state-space action.

REFERENCES

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

[2] B. Chen, H. H. Cheng, and J. Palen, "Integrating mobile agent technology with multi-agent systems for distributed traffic detection and management systems," *Transportation Research Part C: Emerging Technologies*, vol. 17, pp. 1–10, 2009.

[3] M. Tlig and N. Bhouri, "A multi-agent system for urban traffic and buses regularity control," in *Procedia - Social and Behavioral Sciences*, vol. 20, 2011, pp. 896–905.

[4] D. A. Roozemond, "Using intelligent agents for pro-active, real-time urban intersection control," *European Journal of Operational Research*, vol. 131, pp. 293–301, 2001.

[5] A. L. C. Bazzan, "A Distributed Approach for Coordination of Traffic Signal Agents," pp. 131–164, 2005.

[6] C. Q. Cai and Z. S. Yang, "Study on Urban Traffic Management Based on Multi-Agent System," in *Machine Learning and Cybernetics 2007 International Conference on*, vol. 1, 2007, pp. 25–29.

[7] R. T. Van Katwijk, B. De Schutter, and J. Hellendoorn, "Multi-agent coordination of traffic control instruments," in *2008 1st International Conference on Infrastructure Systems and Services: Building Networks for a Brighter Future, INFRA 2008*, 2008.

[8] I. Arel, C. Liu, T. Urbanik, and A. Kohls, "Reinforcement learning-based multi-agent system for network traffic signal control," p. 128, 2010.

[9] "Cooperative, hybrid agent architecture for real-time traffic signal control," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 33, pp. 597–607, 2003.

[10] D. Srinivasan, M. C. Choy, and R. L. Cheu, "Neural networks for real-time traffic signal control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, pp. 261–272, 2006.

[11] M. Wiering, "Multi-Agent Reinforcement Learning for Traffic Light Control," in *Proc Intl Conf Machine Learning*, 2000, pp. 1151–1158.

[12] S. El-Tantawy, B. Abdulhai, and H. Abdelgawad, "Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (marlin-atsc): Methodology and large-scale application on downtown toronto," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, pp. 1140–1150, 2013.

[13] J. Garcia-Nieto, A. C. Olivera, and E. Alba, "Optimal cycle program of traffic lights with particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 17, pp. 823–839, 2013.

[14] M. A. Khamis and W. Gomaa, "Adaptive multi-objective reinforcement learning with hybrid exploration for traffic signal control based on cooperative multi-agent framework," *Engineering Applications of Artificial Intelligence*, vol. 29, pp. 134–151, 2014.

[15] C. Watkins and P. Dayan, "Q-Learning," *Machine Learning*, vol. 8, pp. 279–292, 1992.

[16] L. S. Passos and R. Rossetti, "Traffic light control using reactive agents," *Information Systems and Technologies (CISTI), 2010 5th Iberian Conference on*, 2010.