The implementation of reinforcement learning algorithms on the elevator control system

Hang Li
Institute of Automation and Information System
Technische Universität München
Garching near Munich, Germany
STUD_Li@ais.mw.tum.de

***Abstract*— the machine learning technology has been implemented in control systems successfully especially in the stochastic process. Among the machine learning algorithms, the reinforcement learning algorithm is fairly significant. On the other hand, the elevator, as a facility in daily life, is implemented with several heuristic control methods in the past practice. Simultaneously, the elevator working procedure can be viewed as a stochastic process, so this paper tries to discuss the implementation of several reinforcement learning algorithms on the elevator control system. Through the application of the reinforcement learning algorithms, the elevator system can perform better.**

Keywords—reinforcement algorithms; elevator control system; Q-learning.

I. INTRODUCTION

The technology of machine learning, which aims to help the computer learn by itself and simulate the behavior of learning like a real person, is one vital domain of artificial intelligence [1]. Since it is proposed, the dramatic growth in practical applications has been accompanied by many significant developments in the underlying algorithms and techniques over the last decades [2]. And it exhibits kinds of advantages e.g. the higher learning speed, the higher portability, and regardless of the restrictions upon life cycle.

Among all the algorithms in machine learning, the reinforcement learning (RL) algorithm is a bright spot. RL algorithms are a series of learning policies that makes the programs improve their performance by receiving rewards or punishments from the environment. Most RL policies optimize the total discounted reward received by an agent [3]. Therefore, the aim of the controller is to find one control policy that can maximize the cumulative reward during its exploring procedure. Therefore, three RL algorithms are introduced and implemented in this paper, i.e. Q-learning algorithm, Q-value algorithm and multi-step Q-learning algorithm.

In temporary life, the elevators play an essential role in industrial countries. In skyscrapers, even several elevators are installed. Correspondingly, several classical control approaches are implemented in the elevator system such as collective control strategy, the zone approaching control strategy, search based approach and rule-based approach etc. But these classical heuristic approaches have obvious disadvantages that they will response to every hall call regardless of the current state of the car, and that increases the average waiting time and the burden of the elevator inevitably. Besides, they can also become inappropriate if new traffic patterns arise that were not anticipated during the research and design phases [4].

Besides, there are also several adaptive control approaches. Imasaki et al. described a system that used a fuzzy neural network for various sets of parameters [5]. Tobita et al. adopted a floor-attribute-based control algorithm combined floor-attribute-based evaluation and car-attribute-based evaluation [6]. Markon et al. introduced a neural network system to react to immediate call [7]. These control policies reduce the average waiting time of the passengers up to 10%. But in the following part of this paper, we find that the average waiting time shrinks substantially with the reinforcement learning algorithms.

The rest of this paper is organized as follows. The three RL algorithms are introduced in the second section. The third section describes the simulating system. The experiment results are displayed and analyzed in the fourth section. The conclusion and future work are discussed in the fifth section.

II. AN INTRODUCTION TO THE ALGORITHMS

In a stochastic process, the controller has multiple choices to proceed. At a discrete time step t, the system state is described as a variable $s_t \in S$ of the state space, and an action $a_t \in A$ is employed according to the control policy $a_t = \pi(s_t)$. Then, the system transits to a new state s_{t+1}. Synchronously, the controller will be assigned reward $r_{t+1} = \rho(s_t, a_t, s_{t+1})$, which is used to evaluate the quality of the learning stages. The aim of the controller is to maximize the accumulated value of the reward function [8].

$$R_t = \sum_{i=0}^{\infty} \gamma^i r_{t+i+1} \tag{1}$$

Here the parameter γ represents the discount rate with the domain of [0, 1]. This can be realized with the best control policy in each step, donated by $Q^{\pi}(s,a)$ under the state-action

pair(SAP), that means the value $Q^{\pi}(s,a)$ is determined by the current state value s_t and action value a_t under control policy π.

$$Q^{\pi}(s,a) = E^{\pi}\left\{R_t \middle| s_t = s, a_t = a\right\} \tag{2}$$

The optimal value $Q^{*}(s,a)$ is defined as the maximum of $Q^{\pi}(s,a)$. Similarly, in case the optimal value $Q^{*}(s,a)$ is known, the optimal control strategy is also determined. Here, $Q^{*}(s,a)$ is the unique solution to the Bellman optimality equation:

$$Q^{*}(s,a) = \sum_{s' \in S} f(s,a,s')[R(s,a,s') + \gamma \max_{a' \in A} Q^{*}(s',a')] \tag{3}$$

To learn this Q-function, RL algorithms repeatedly do:

1) Select action at a given state s_t.
2) Collect reward r_t and observe successor state s_{t+1}.
3) Update the Q-function using the latest experience (s_t, a_t, r_t, s_{t+1}).

B. Q-learning algorithm

Q-learning is a model-free RL algorithm [9]. It makes the controller learn how to act optically without building the environment model. Also, it can be viewed as a method of asynchronous dynamic programming. Most importantly, it provides agents with the capability of learning to act optimally in Markovian domains by experiencing the consequences of actions, without requiring them to build maps of the domains. The procedures algorithm is displayed in Fig. 1.

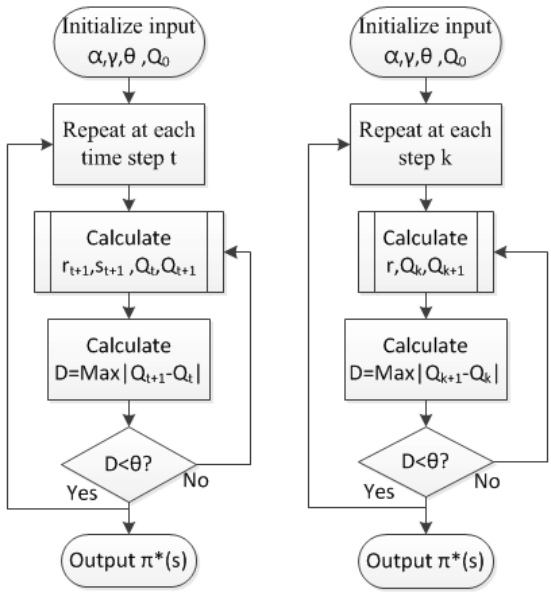

Fig. 1. The flowchart of Q-learning and Q-value algorithms

At the beginning, a discount rate γ, a learning rate α_t and the convergence threshold θ are set. In the Q-learning algorithm, the initial values of Q function are set arbitrarily. Here set $Q_0(s,a) = 0$, for all $s \in S, a \in A$. Then a discrete distribution function is chosen to evaluate the action optionally, e.g. Boltzmann distribution, Uniform discrete distribution (UD), Poisson distribution. Here, we use the UD as the transition function. Next, the values of Q-function are calculated under all conditions of SAP until the deviation between two steps is less than the given threshold. Finally, the optimal value of state variable x in SAP is chosen to continue.

$$Q_{t+1}(s_t,a_t) = Q_t(s_t,a_t) + \alpha_t\left[r_{t+1} + \gamma \max_{a \in A} Q_t(s_{t+1},a) - Q_t(s_t,a_t)\right] \tag{4}$$

$$\pi^{*}(s) = \arg\max_{a \in A} Q_t(s,a), \forall s \in S \tag{5}$$

C. Q value algorithm

The Q-value algorithm is an iteration algorithm that computes the Bellman equation iteratively and it is model-based [10]. The Q-value algorithm works only offline and ensures monotonous convergence to the optimal policy [11].

The flowchart of Q-value algorithm is also displayed in Fig.1. Where, f represents the transition function. It is same with the Q-learning algorithm. Also, different distribution functions are suitable for this stochastic process. Here, Bellman optimality equation is employed.

D. Multi-step Q-learning algorithm

The multi-step Q-learning algorithm combines QL algorithm and TD(λ) method. TD, short for temporal difference, is a discrete method of extracting information from observations of sequential stochastic processes so as to improve predictions of future outcomes [12]. The multi-step Q-learning algorithm uses TD(λ) method to accelerate the Q-learning algorithm [13]. An eligibility trace

$$l_t(s,a) = \sum_{i=1}^{t-1} (\gamma\lambda)^{t-i} \eta^{i}(s,a) \tag{6}$$

is employed for each SAP (s,a). Then, the error functions are computed.

$$e_t' = r_t + \gamma Q^{*}(s_{t+1}) - Q(s_t,a_t) \tag{7}$$

$$e_t = r_t + \gamma Q^{*}(s_{t+1}) - Q^{*}(s_t) \tag{8}$$

Next, for each $SAP \in H$, the new value of $l(s,a)$ and $Q(s,a)$ will be computed.

$$l(s,a) \leftarrow \gamma\lambda l(s,a) \tag{9}$$

$$Q_{t+1}(s,a) = Q_t(s,a) + \alpha_k(s_t,a_t)e_t l_t(s,a) \tag{10}$$

At the same time, a Boolean variable $visited(s,a)$ is used to make sure that every SAP in H is unique. If $l(s,a) < \theta$, then $visited(s,a) = 0$. Then set $l_t(s,a) \leftarrow l_t(s,a) + 1$, $Q(s,a) \leftarrow Q(s,a) + \alpha_k(s_t,a_t)e'_t$. If $visited(s,a) = 0$, then set $visited(s,a) = 1, H \leftarrow H \cup (s_t,a_t)$. The last step, output $\pi^*(s) = \arg\max_{a \in A} Q_k(s,a), \forall s \in S$.

In the algorithm of multi-step QL algorithm, two error function values e'_t, e_t are used to decline the uncertainty caused by the stochastic process. The flowchart of the multi-step Q-learning algorithm is showed in Fig. 2.

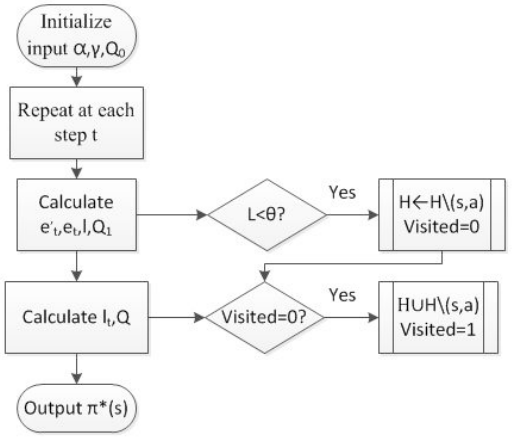

Fig. 2. The flowchart of the multi-step QL algorithm

III. THE DESIGN OF THE SYSTEM

A. elevator control system overview

In this paper, the model of elevator is designed as follows.

- To simplify the problem, one-tunnel-one-car elevator is simulated in the system.
- The number of the floors is set to 5.
- The travelling time of the elevator to go through 2 adjacent floors is 2 second.
- If someone calls the car in the hall, the elevator will go to his waiting floor and stay at the floor for 2 seconds for him.
- The elevator capacity is set to fairly high. So it will not happen that some people cannot enter the car because of full-load.

In a manufactory building, the heavy production facilities are usually put in the ground floor. Usually, the majority of workers such as engineers and accountants are working in the first floor, and the managers are in higher stage with a bigger office. So it means the higher stage, the fewer clerks, and the smaller probability to use the elevator. The proportions of people working from the ground floor to fourth floor are set {1/16, 11/16, 2/16, 1/16, and 1/16}.

In the simulation, a down-peak traffic pattern is discussed. That pattern occurs at the closing time. In the down peak traffic, there are several departure floors but only one destination floor, namely the ground floor. So the probability of people emerging in the first floor is same as the proportion of people working in that floor. And it is assumed that there will be a person waiting in the hall every 6 seconds so the elevator will have enough time to fetch the person. The initial position for the elevator is the ground floor. That also means after quick long time no persons waiting in the hall, and then the elevator will go back to the ground floor.

B. State-action space

The discrete state space of the elevator control system is assigned 5 parameters. Namely,

$$x = [s_1, s_2, s_3, s_4, p]^T \tag{11}$$

Here,

- $s_i, i = 1,2,3,4$. It represents the hall call request in each floor. Obviously, there will be no hall call signal in the ground floor. They are binary value with the domain {0, 1}.
- p represents the position of elevator. It is integer with the domain {0, 1, 2, 3, and 4}.

The elevator controller has three kinds of actions: {-1, 0, 1}. The action value "-1" states that the elevator goes downstairs, while the value "1" indicates that the elevator goes upwards. The value "0" means that the elevator doesn´t move. There is also one restriction on the action value, i.e. the value "1" cannot transit to "-1" directly, it must firstly transit to "0", then "-1", and vice versa.

The reward function is set as a negative integer. If and only if when there are no hall calls on any floor, the value of reward function is 0.

$$r(x) = -\sum_{i=1}^{4} s_i \tag{12}$$

IV. THE RESULT AND ANALYSIS OF THE EXPERIMENT

As introduced in the former section, the average waiting time is used to evaluate whether an algorithm is better. There are several kinds of methods to represent the average waiting time e.g.

- Simple average of every person

- Weighted average
- Median value, middle floor value(t_{2_j}, t_{3_j})
- Maximum weight-stage t_i (in this paper, t_1)
- Close-to-mean-stage value.

The simple way and the weighted means are intuitive. The median value method is easy and intuitive, but it extends one bubble sort calculation which increases the amount of calculation. Owing to the specialty of the model in this paper, the average waiting time of the people working in the first floor is the best way to evaluate the quality of the algorithms.

The heuristic control policy is used as a reference. The elevator system randomly chooses a waiting passenger and takes him to the ground floor. Then it chooses another passenger randomly. This reference control policy pushes waiting time up to 46 seconds [5].

The parameters applied in the three algorithms are set as follows.

TABLE I. ALGORITHMS PARAMETER SETTING

parameters	value
Number of tests	50
Discount rate γ	0.98
Learning rate α	0.38
Convergence threshold θ	0.01

The reinforcement algorithms have perfect results of the passenger average waiting time about 5.3 seconds. The results of the three algorithms are displayed in Fig. 3.

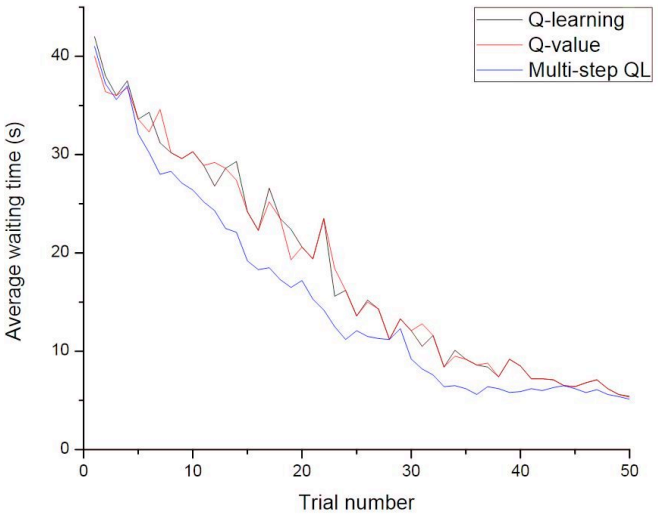

Fig. 3. evolution of the trial waiting time with reinforcement algorithms

Each experiment includes 50 tests. From the very beginning, the average waiting time is around 40 seconds. And it will weaken very quickly in the following trials. Although the average waiting time does not decrease monotonically, it converges to a low value with stark vibration. This is partly caused by the random choice between the waiting people of the system. The system maybe chooses a person in the 2nd floor in the former trial, but may choose a person in the 3rd floor in the following trial.

Compared with the Q-learning algorithm, the Q-value algorithm performs slightly better. The multi-step Q-learning algorithm has the best result, the best convergence rate and the least vibration. But the results of all Q-learning algorithms are much better than the result of heuristic control policy.

V. CONCLUSION AND FUTURE WORK

In this paper, a model of elevator system is designed, and then three reinforcement algorithms are implemented in the simulated elevator control system. The average waiting time is used to evaluate the performance of three algorithms. The results prove that all the reinforcement algorithms are suit for elevator control system and perform much better than the classical methods.

The future work should be put at which extend that the parameters influence the results and which parameter makes the greatest influence.

REFERENCES

[1] J. R. Anderson. "*Machine learning: An artificial intelligence approach"*. Eds. Ryszard Stanisław Michalski, Jaime Guillermo Carbonell, and Tom Michael Mitchell. Vol. 2. Morgan Kaufmann, 1986.

[2] C. M. Bishop. "*Pattern recognition and machine learning* "(Vol. 4, No. 4, p. 12). New York: springer, 2006.

[3] P. Tadepalli, D. Ok. "Model-based average reward reinforcement learning". Artificial Intelligence, 100(1), 177-224, 1998.

[4] T.Walczak, & P. Cichosz, P. "A distributed learning control system for elevator groups". In *Artificial Intelligence and Soft Computing–ICAISC 2006* (pp. 1223-1232). Springer Berlin Heidelberg, 2006.

[5] S. Sekine, N. Imasaki, & T. Endo. "Application of fuzzy neural network control to automatic train operation and tuning of its control rules". *International Joint Conference of the Fourth IEEE International Conference on Fuzzy Systems and The Second International Fuzzy Engineering Symposium., Proceedings of IEEE Int* (Vol. 4, pp. 1741-1746). March 1995.

[6] A. Fujino, T. Tobita, K. Segawa, K. Yoneda, & A. Togawa. "An elevator group control system with floor-attribute control method and system optimization using genetic algorithms". *Industrial Electronics, IEEE Transactions on*, *44*(4), 546-552, 1997.

[7] R. H. Crites, & A. G. Barto. "Elevator group control using multiple reinforcement learning agents". *Machine Learning*, *33*(2-3), 235-262, 1998.

[8] C. J. Watkins, & P. Dayan. "Q-learning". *Machine learning*, *8*(3-4), 279-292,1992.

[9] A. Al-Tamimi, F. L. Lewis, & M. Abu-Khalaf. "Model-free Q-learning designs for linear discrete-time zero-sum games with application to H-infinity control". Automatica, 43(3), 473-481, 2007.

[10] X. Yuan, L. Busoniu, & R. Babuška. "Reinforcement learning for elevator control". In Proceedings 17th IFAC World Congress (IFAC-08) (pp. 2212-2217), 2008.

[11] D. P. Bertsekas. "Dynamic Programming and Optimal Control 3rd Edition, Vol. I, vol. 2 of Athena Scientific optimization and computation series." Athena Scientific , 2007.

[12] P. Dayan, and J. S. Terrence "TD (λ) converges with probability 1." *Machine Learning* 14.3: 295-301, 1994.

[13] M. Wiering, and J. Schmidhuber. "Fast online Q (λ)." *Machine Learning* 33.1: 105-115, 1998.