Routing an Autonomous Taxi with Reinforcement Learning

Miyoung Han, Pierre Senellart, Stéphane Bressan, Huayu Wu

Routing an Autonomous Taxi with Reinforcement Learning

Miyoung Han
Télécom ParisTech, Paris
IPAL, I2R, A*STAR, Singapore

Pierre Senellart
Télécom ParisTech, Paris
IPAL, NUS, Singapore

Stéphane Bressan
IPAL, NUS, Singapore

Huayu Wu
I2R, A*STAR, Singapore

ABSTRACT

Singapore's vision of a Smart Nation encompasses the development of effective and efficient means of transportation. The government's target is to leverage new technologies to create services for a demand-driven intelligent transportation model including personal vehicles, public transport, and taxis. Singapore's government is strongly encouraging and supporting research and development of technologies for autonomous vehicles in general and autonomous taxis in particular. The design and implementation of intelligent routing algorithms is one of the keys to the deployment of autonomous taxis. In this paper we demonstrate that a reinforcement learning algorithm of the Q-learning family, based on a customized exploration and exploitation strategy, is able to learn optimal actions for the routing autonomous taxis in a real scenario at the scale of the city of Singapore with pick-up and drop-off events for a fleet of one thousand taxis.

1. INTRODUCTION

Singapore's Prime Minister Office outlines its vision of a Smart Nation as the harnessing of information and communication technologies "to support better living, create more opportunities, and support stronger communities"[1]. The vision encompasses the development of effective and efficient means of transportation and aims at the development of a demand-driven intelligent transportation model including personal vehicles, public transports, and taxis. In particular, Singapore's government is strongly encouraging and supporting research and development of technologies for autonomous vehicles (AV). Its agencies, together with academia and industry, are involved in learning and understanding opportunities and challenges for autonomous vehicle technologies and autonomous taxis. For example, the Land Transport Authority of Singapore [9] has been working with A*STAR's Institute for Infocomm Research to evaluate their AV Proof-of-Concept on-road trials[2] since last year. "Unlike autonomous vehicle trials elsewhere, Singapore is focusing on applying the technology to public buses, freight carriers, autonomous taxis and utility operations such as road sweepers." [8]. Self-driving riders have already been experienced by the public at Gardens by the Bay last December[3].

The design and implementation of intelligent routing algorithms is one of the keys to the deployment of autonomous taxis. While prior studies have explored reinforcement learning approaches to taxi routing, they have mostly done so with synthetic models and data in small and static state spaces. In this paper we demonstrate that a reinforcement learning algorithm is able to progressively, adaptively, efficiently, and effectively learn optimal actions for the routing to passenger pick-up points of an autonomous taxi in a real scenario at the scale of the city of Singapore.

We present a solution based on a customized exploration and exploitation strategy for an algorithm of the *Q-learning* family [10]. Q-learning is an approach to solve *reinforcement learning* [7] problems. Reinforcement learning is used for learning an optimal policy in a dynamic environment: an agent takes an action in a state, receives a reward, moves to some next state, and repeats this procedure. Initially, the agent has no knowledge of which action has to be taken in a given state. The algorithm has the choice between *exploiting* its knowledge by choosing the action with highest estimated value and *exploring* its environment by taking any other action. The trade-off between exploration and exploitation is crucial. We devise an exploration strategy that follows an almost-greedy policy. The strategy first selects a set of candidate actions with long-term reward above a given threshold and then selects, among candidates, the action with the highest probability of finding a passenger.

We evaluate how Q-learning progressively learns and how to balance exploration and exploitation. We quantify the influence of the parameters. We propose an original selection strategy. The evaluation uses a dataset of taxi pickups and drop-offs for a fleet of 1000 taxis for one month in Singapore.

2. BACKGROUND AND RELATED WORK

Most studies addressing the taxi routing problem focus on providing the fastest route and a sequence of pick-up points [6] by mining historical data [6,11–13]. Yuan et al. [11]

[1]http://www.pmo.gov.sg/smartnation

CIKM'16 , October 24 - 28, 2016, Indianapolis, IN, USA

ISBN 978-1-4503-4073-1/16/10. . . $15.00

DOI: http://dx.doi.org/10.1145/2983323.2983379

[2]http://www.lta.gov.sg/apps/news/page.aspx?c=2&id=7dac9bd2-e6d1-448f-ba0e-f31fd13a8c3e

[3]http://www.opengovasia.com/articles/6792-asias-first-fully-operational-autonomous-vehicle-running-at-gardens-by-the-bay-singapore

cluster road segments and travel time to build a landmark graph of traffic patterns and time-dependent fastest routes. They then present in [12] a recommendation system for taxi drivers and passengers by clustering road segments extracted from GPS trajectories. The system recommends a parking place and road segments to taxi drivers and passengers. Qu et al. [6] propose a method to recommend a route. They develop a graph representation of a road network by mining the historical taxi GPS traces and generate an optimal route for finding passengers. Those models rely on the availability of accurate historical data and trajectories. They might not be relevant in dynamic environments such as an autonomous taxi looking for optimal passenger pick-up points.

Reinforcement learning [7] has the potential to continuously and adaptively learn from interaction with the environment. The algorithm discovers which actions produce the greatest reward by experience and estimates how good it is to take a certain action in a given state. Yet reinforcement learning aims to maximize cumulated reward. The notion of *Markov Decision Process* (*MDP*) underlies much of the work on reinforcement learning, and is at the basis of our work.

Q-learning [10] is widely used because of its computational simplicity. In Q-learning, one does not require a model of transition functions and reward functions but learns directly from observed experience. This is an advantage as learning a model often requires exhaustive exploration that is not suitable for a large state space. Thus, Q-learning seems suitable for taxi routing in a city of the scale of Singapore.

While taxi routing has often been used as the example application for reinforcement learning algorithms, it often remained relegated to toy or small scale examples, as it is the case of the seminal 5×5 grid introduced by the authors of [1] and used for experimental purposes by the authors of [2–5]. Learning pick-up points is a somewhat new application of reinforcement learning.

3. Q-LEARNING FOR TAXI ROUTING

We consider an autonomous taxi completely relying on reinforcement learning. The algorithm decides where the autonomous taxi should go in order to pick up passengers by learning the existence probability of passengers from its gathered experience. Reinforcement learning models an agent taking an action a in state s, receiving a reward r, and moving to the next state s'. With Q-learning, the estimated value of taking action a in state s, denoted $Q(s,a)$, is updated as: $Q(s,a) \coloneqq Q(s,a) + \alpha\left[r + \gamma \max_{a'} Q(s',a') - Q(s,a)\right]$.

Here, α is a positive fraction such that $0 < \alpha \leqslant 1$, the step-size parameter that influences the rate of learning. When $\alpha = 1$, the agent considers only the most recent information for learning. If α is properly reduced over time, the function converges [7]. The discount rate γ ($0 \leqslant \gamma < 1$) determines the present value of future rewards. If $\gamma = 0$, the agent is only concerned with the immediate reward. The agent's action influences only the current reward. If γ approaches 1, the agent considers future rewards more strongly.

To maximize total reward, the agent must select the action with highest value (exploitation), but to discover such action it has to try actions not selected before (exploration). The ε-greedy method works as follows: most of the time, select an action with the highest estimated value, but with small probability ε select an action uniformly at random. This exploration enables to experience other actions not taken before and it may increase the greater total reward in the long run because we would discover better actions.

Algorithm 1 Taxi Routing for Learning Pick-up Points

```
1: Initialize Q(s,a), existence probability of passengers p
2: repeat
3:   repeat
4:     if greedy then
5:       V ≔ { a ∈ A | Q(s,a) ⩾ max_a' Q(s,a') − η }
6:       if |V| > 1 then
7:         Select action a with highest probability p
8:     else /* not greedy */
9:       Select action a uniformly at random
10:    Take action a, obtain reward r, observe next state s'
11:    Q(s,a) ≔ Q(s,a) + α [r + γ max_a' Q(s',a') − Q(s,a)]
12:    Increment visit count on s'
13:    Update existence passenger probability p(s')
14:    if passenger found in s' then
15:      Increment found count on s'
16:      s becomes the end of the passenger route from s'
17:    else
18:      s ≔ s'
19:  until a passenger is found
20: until algorithm converges
```

We call an episode the movement of a taxi to an actual pick-up point. For the first episode, the taxi located at a random position moves according to the initial oblivious learning policy. The episode ends when the taxi finds a passenger. Then, it moves to the passenger's destination and starts a new episode. As the taxi moves it receives rewards and updates its action-value and the existence probability. The road network is discretized and the movements correspond to steps in the discretized network. At each step, the taxi learns where passengers are likely to be located.

The Taxi Routing algorithm for learning pick-up points is outlined in Algorithm 1. According to the ε-greedy policy, an action a is selected in a given state s.

The action selection rule selects the action with the maximum estimated action value (greedy action). However, with this rule, the algorithm ignores other actions that, although they have slightly lesser value, may lead to a state having a higher chance to pick a passenger up. Hence, instead of selecting one greedy action, we loosen the selection condition by setting a lower bound below the maximum value in order to choose from more potentially valuable candidate actions (Line 5). The candidate actions are compared with existence probabilities of passengers in their corresponding states (Line 7). We later refer to the algorithm with this selection strategy as *Q-learning using LB/Proba*.

After taking an action, we update the Q-value in the current state s with reward r and next state s'. As we visit a new state s', the visit and found counts are incremented and the existence probability of passengers is also recalculated. We repeat this procedure until we find a passenger.

4. PERFORMANCE EVALUATION

For the sake of simplicity, in this paper, we present the results for a map discretized into cells of 0.01 degree longitude × 0.01 degree latitude (about 1.1km × 1.1km) forming a 38 × 20 grid. At each cell of the grid, eight actions are possible: up, down, right, left, and diagonally. A step is the movement from one cell to an adjacent one. Although such

a representation does not capture several natural constraints on the traffic, it is sufficient, with limited loss of generality, to evaluate the effectiveness of the algorithm.

Since popular pick-up points generally depend on the time of the day, we run the experiments for selected time intervals. Here, we present the results for two off-peak hours (12h to 13h and 14h to 15h), but we obtain comparable results for other time slots. At each episode we select 300 passengers according to actual geographical distribution in the given time interval in a dataset of taxi pickups and drop-offs for a fleet of one thousand taxis for one month in Singapore.

We first look into the impact of the step-size parameter α, the discount rate γ, and the probability of exploration ε. We evaluate how these parameters influence the learning performance with ordinary Q-learning. We compare the average number of steps. The average steps are calculated at every 100 episode by dividing the total steps from the first episode to the last by the total number of episodes.

Figures 1a–1b show the average number of steps with different step-size parameter α values for different time intervals. We compare four different α with a fixed γ (= 0.5) and ε (= 0.3). For all the time intervals, as the α is smaller, the average number of steps also decreases. Lower step-size values perform better. This indicates that accumulated experience affects value estimation more significant than recent experience, i.e., that the problem is indeed stochastic.

For the discount rate γ experiment, we fixed α (= 0.5) and ε (= 0.3) and changed the γ. The average number of steps with different γ values for different time intervals are shown in Figures 1c–1d. In Figure 1d, the lowest γ (= 0.25) perform better. This means that immediate rewards are more important than future rewards. In Figure 1c, as episodes continue, a higher γ (= 0.75) is slightly better. In 1c, relatively longer step counts than those of the other time intervals are needed to achieve a goal. In this case, future rewards are more significant than current rewards.

Figures 1e–1f show the average number of steps with different ε values for different time intervals, given $\alpha = 0.5$ and $\gamma = 0.5$. The average number of steps first decreases dramatically and then converges gradually. For all time intervals, when ε is 0.1, the average number of steps is higher than the other cases in early episodes but it dominates after about 30,000 episodes. At the beginning, exploration is more effective and relatively inexpensive. Eventually, sufficient knowledge is accumulated and exploitation is worthy.

We now compare Q-learning using LB/Proba (our algorithm) with ordinary Q-learning. For experiments, we select three parameter values shown in the previous section. The step-size parameter α is set to 0.25 because low learning rate is appropriate to our problem. The probability of exploration ε is set to 0.1. Since loosened selection for maximum action has the effect of exploration, high ε is not needed. We take $\eta = 0.01$ to set the lower bound on the maximum action value per state. Two algorithms are compared by varying the discount rate γ value.

In Figures 1g–1j, when $\gamma = 0.75$ or 0.5, Q-learning using LB/Proba converges faster than Q-learning. On the other hand, when γ is 0.25 (Figures 1k–1l), Q-learning performs similary well or slightly better than Q-learning using LB/Proba. These experiments show that when the learning rate α is low and the discount rate γ approaches 1, Q-learning using LB/Proba outperforms Q-learning. In other words, it has to accumulate much experience for value prediction and it considers future rewards more strongly. The reason is that Q-learning using LB/Proba depends on existence probability of passengers that requires enough experience and that is more related to long-term high rewards.

5. CONCLUSIONS AND FUTURE WORK

A*STAR's Institute for Infocomm Research, with its academic partners, is contributing to Singapore's Land Transport Authority's efforts to evaluate the viability of autonomous vehicles. We explored the use of reinforcement learning for autonomous taxi routing. The evaluation results provide empirical support of the effectiveness of reinforcement learning for the task at hand. We are planning more extensive studies as soon as extensive on-road trials are carried out and more data is available to us. We are also currently extending this work to consider a more dynamic environment. We are considering a multi-agent approach to scale to larger and more complex state spaces. We also investigate hierarchies, factored representations, and other extended methods that can improve adaptiveness, effectiveness and efficiency of learning.

Acknowledgment. This research was supported in part by the Télécom ParisTech Chair on Big Data & Market Insights, and in part by the National Research Foundation, Prime Minister's Office, Singapore under the CREATE program, NUS ref. R-702-005-101-281.

6. REFERENCES

[1] T. G. Dietterich. The MAXQ method for hierarchical reinforcement learning. In *ICML*, 1998.

[2] C. Diuk, A. L. Strehl, and M. L. Littman. A hierarchical approach to efficient reinforcement learning in deterministic domains. In *AAMAS*, 2006.

[3] M. Ghavamzadeh and S. Mahadevan. A multiagent reinforcement learning algorithm by dynamically merging Markov decision processes. In *AAMAS*, 2002.

[4] M. Ghavamzadeh and S. Mahadevan. Learning to communicate and act using hierarchical reinforcement learning. In *AAMAS*, 2004.

[5] T. Hester and P. Stone. Generalized model learning for reinforcement learning in factored domains. In *AAMAS*, 2009.

[6] M. Qu, H. Zhu, J. Liu, G. Liu, and H. Xiong. A cost-effective recommender system for taxi drivers. In *KDD*, 2014.

[7] R. S. Sutton and A. G. Barto. *Introduction to Reinforcement Learning*. MIT Press, 1998.

[8] C. Tan. Driverless vehicles hit the road in trials around singapore. *Straits Times*, 2015. http://www.straitstimes.com/singapore/transport/driverless-vehicles-hit-the-road-in-trials-around-singapore.

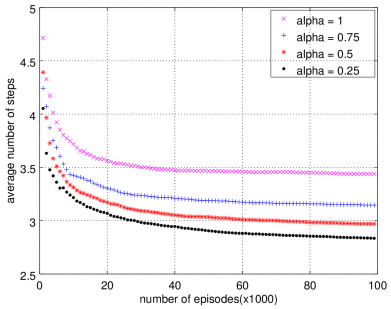
[9] T. K. S. TAN Cheon Kheong. Autonomous vehicles, next stop: Singapore. *Journeys*, 2014.

[10] C. J. C. H. Watkins and P. Dayan. Technical note: Q-learning. *Mach. Learn.*, 8(3-4), 1992.

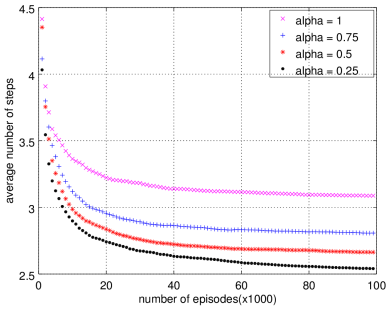
[11] J. Yuan, Y. Zheng, X. Xie, and G. Sun. T-drive: Enhancing driving directions with taxi drivers' intelligence. *TKDE*, 25(1), 2013.

[12] N. J. Yuan, Y. Zheng, L. Zhang, and X. Xie. T-finder: A recommender system for finding passengers and vacant taxis. *TKDE*, 25(10), 2013.

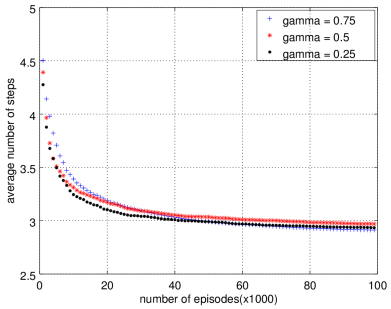
[13] Y. Zheng. Trajectory data mining: An overview. *ACM Trans. Intell. Syst. Technol.*, 6(3), 2015.



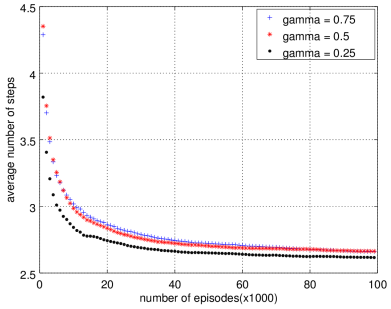
(a) Varying α, 12h to 13h



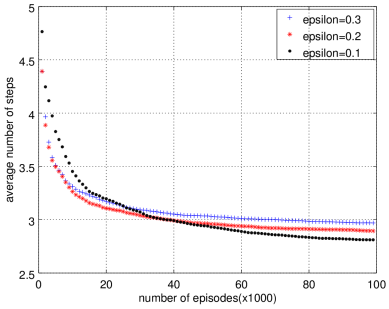
(b) Varying α, 14h to 15h



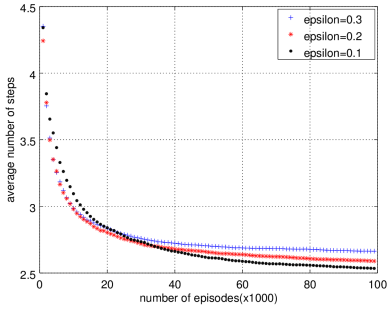
(c) Varying γ, 12h to 13h



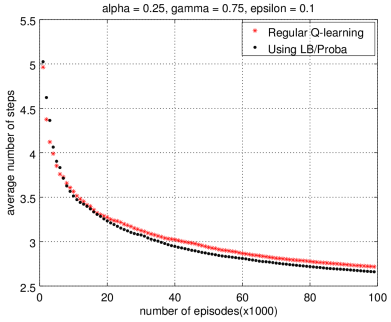
(d) Varying γ, 14h to 15h



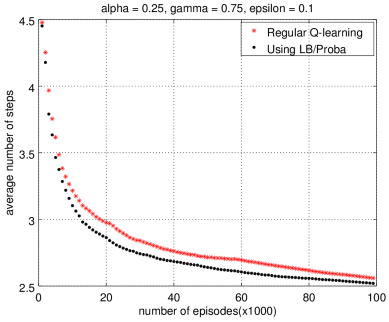
(e) Varying ε, 12h to 13h



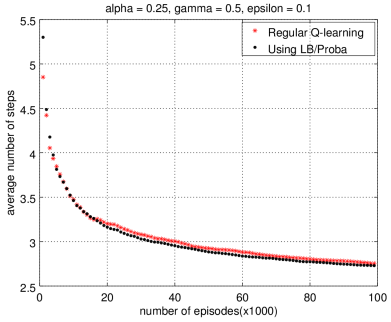
(f) Varying ε, 14h to 15h



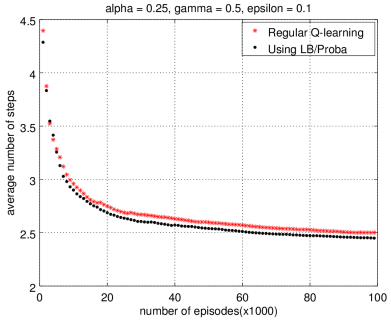
(g) Regular Q-learning vs LB/Proba, $\gamma = 0.75$, 12h to 13h



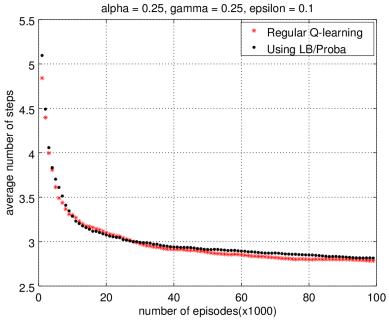
(h) Regular Q-learning vs LB/Proba, $\gamma = 0.75$, 14h to 15h



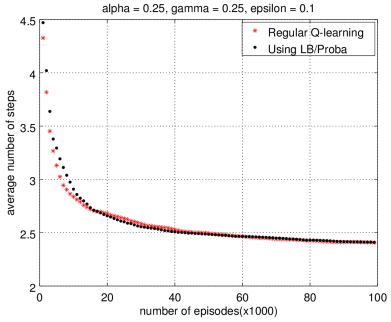
(i) Regular Q-learning vs LB/Proba, $\gamma = 0.5$, 12h to 13h



(j) Regular Q-learning vs LB/Proba, $\gamma = 0.5$,14h to 15h



(k) Regular Q-learning vs LB/Proba, $\gamma = 0.25$, 12h to 13h



(l) Regular Q-learning vs LB/Proba, $\gamma = 0.25$, 14h to 15h

Figure 1: Average number of steps as the number of episodes increases.