Proactive Route Planning Based on Expected Rewards for Transport Systems

Naoto Mukai and Toyohide Watanabe
Department of Systems and Social Informatics,
Graduate School of Information Science, Nagoya University
Furo-cho, Chikusa-ku, Nagoya, 464-8603, Japan
{naoto,watanabe}@watanabe.ss.is.nagoya-u.ac.jp

Jun Feng
Hohai University, Nanjing, Jiangsu 210098, China
fengjun-cn@vip.sina.com

Abstract

Route planning is one of the important tasks for transport systems. Appropriate policies for route selections improve not only profitability of transport companies but also convenience of customers. Traditional ways for establishing the policies depend on manual efforts based on statistical data of transports. Moreover, traditional route planning techniques are reactive, i.e., an optimization based on information provided in advance. It is difficult for the manual policies and the reactive planning techniques to adjust dynamic changes of transport trends for customers such as amount and direction of transport demands, i.e., drivers of transport vehicles must follow the policies provided in advance. Therefore, in this paper we show how the proactive route planning based on expected rewards for transport systems can be modeled as a reinforcement learning problem. And, we show how agents as transport vehicles acquire their policies for route selection autonomously and dynamically. The learning ability of transport trends enables transport vehicles to foresee the next destination which provides high rewards. Finally, we report simulation results and make the effectiveness of our proposal strategy clear.

1 Introduction

In these years, many efforts have focused on developing novel intelligent transportation systems (ITS). Especially, real-time position information based on global positioning systems (GPS) contributes to various aspects of transport systems. One of the aspects is route planning for transport vehicles. Statistical data of transports (e.g., get-on/off positions) can help establishing policies for transport companies. For example, bus companies can determine where bus stops should be located in the city and which routes should be selected among the bus stops. Such transport policies have effects not only on profitability of transport companies but also on convenience of customers. However, traditional ways to acquire the policies strongly depend on manual efforts. Moreover, traditional route planning techniques are reactive, i.e., an optimization based on information provided in advance. Thus, it is difficult for the manual policies and the reactive planning techniques to adjust dynamic changes of transport trends for customers. Generally, it seems that transport trends such as amount and direction of transport demands are not fixed. Consider visitors in a theme park, we assume that there are some flows of them (e.g., to attractions or to restaurants). Trends of these flows would change, depending on its situations such as time and weather of day. Therefore, a learning architecture for adjusting such transport trends autonomously and dynamically is desired. We believe that driver-less cars equipped with the learning architecture will greatly contribute to the advancement of transport systems in the future.

In this paper, as the first step to the novel transport systems, we propose a strategy of proactive route planning based on expected rewards by Q-Learning which is a traditional adaptive reinforcement learning technique. The proactive routes means that transport vehicles can foresee the next destination which provides high rewards by the learning architecture. The route planning problem can be modeled as an agent paradigm with reinforcement learning. Agents as transport vehicles travel on a graph which consists of nodes and edges. In this model, the graph represents physical locations where customers get-on/off in a service area. While, the graph also represents states and actions of the Markov Decision Process (MDP) for Q-Learning. There are three key factors to be considered to acquire appropriate policies. The first key is to define states and actions. The states are given by a sequence of nodes, and the actions are given by an edge (i.e., a link-node). The second key is to define a reward function. The reward function influences transport behaviors of agents, i.e., they transport a large number of customers at once like buses or one by one like taxis. The third key is to define a penalty function. The penalty function influences curiosity behaviors of agents, i.e., they search over a large area or a small area. We developed a simulator to investigate an effectiveness of our strategy for route planning problem. Our experiment consists of two steps. First, we compare agents traveling on fixed routes with agents traveling on learned routes by using ten random patterns of customer flows. Second, we investigate influences related to the reward function and the penalty function.

The remainder of this paper is structured as follows: Section 2 overviews relevant studies. Section 3 formalizes the route planning problem for transport systems. Section 4 summarizes the concept of reinforcement learning. Section 5 presents our strategy based on Q-Learning for route planning. Section 6 reports simulation results and makes the effectiveness of our proposal strategy clear. Finally, Section 7 concludes and offers our future works.

2 Relevant Studies

In this section, we overview relevant studies to our paper. Finding the shortest cycle visiting all nodes of a graph is well known as Traveling Salesman Problem (TSP). TSP can be regarded as the simplest transport problem. In other words, a single vehicle visits all nodes twice (i.e., get-on/off) by the shortest route. But, TSP does not consider the convenience for customers at all. The problem in which a fleet of vehicles is routed in order to visit distributed customers on a graph is called Vehicle Routing Problem (VRP) [2, 9]. Depending on the types of demands (i.e., pre-reservation type or on-demand type), the following two categories exist: Static Vehicle Routing Problem (SVRP) and Dynamic Vehicle Routing Problem (DVRP). These problems are known to be NP-complete. Thus, most approaches for these problems are based on heuristic algorithms such as a genetic algorithm (GA) [11, 7, 4] or an ant colony system (ACS) [3, 6]. Our problem belongs to DVRP. We can say that such traditional approaches for route planning are reactive, i.e., an optimization based on information provided in advance. On the other hand, our approach to the problem is proactive, i.e., transport vehicles can foresee the next destination which provides high rewards by the reinforcement technique.

In recent years, Hugo Santana et al. addressed a multi-agent patrolling problem [1]. The purpose of the problem is to minimize the time lag between two visits to the same node by multi-agent. The multi-agent patrolling task can be regarded as an extended problem of TSP. At first, they mainly focused on several multi-agent architectures such as agent communications, coordination scheme, agent perception, etc [5]. Then, they proposed a strategy based on a reinforcement learning technique for multi-agent patrolling [8]. Although our problem is basically different from their problem in the purposes, our paper has a strong relevance to this strategy.

3 Formalization

In this section, we formalize our route planning problem for transport systems and make the differences between our problem and other problems clear.

3.1 Graph

A space for this route planning problem is given by a graph G as Equation (1). The graph consists of nodes N and edges E. In this model, the graph represents physical locations where customers get-on/off in a service area. While, the graph also represents states and actions of Markov Decision Process (MDP) for Q-Learning.

$$\left\{ \begin{array}{lcl} G & = & (N, E) \\ N & = & \{n_1, n_2, \cdots, n_l\} \\ E & = & \{e(n, n') | n, n' \in N\} \end{array} \right. \tag{1}$$

3.2 Agents

Let V be a set of k agents as Equation (2). The agents travel on the graph at a fixed speed. Each agent acts as a transport vehicle. However, in this paper we assume that the riding capacities of all agents are infinite. Thus, we ignore types of vehicles such as buses and taxis (i.e., agents can be buses or taxis depending on their situations).

$$V = \{v_1, v_2, \cdots, v_k\} \tag{2}$$

3.3 Flows

Let C be a set of customers as Equation (3). The customers are not given in advance of the start time of transport service. Thus, we cannot schedule traveling routes of agents in advance. The patterns of customer occurrence follow a set of m flows F as Equation (4). A flow f is defined by a tuple: n_r and n_d are the nodes where customers get on/off, and η is occurrence rate of customers at a unit time T_u. Generally, the customer occurrences are not uniform distributions (i.e., transport trends such as amount and direction are different). Therefore, the flows F represent such transport trends.

$$C = \{c_1, c_2, \cdots\} \tag{3}$$

$$\begin{cases} F = \{f_1, f_2, \cdots, f_m\} \\ f = (n_r, n_d, \eta) \end{cases} \tag{4}$$

3.4 Objective

The objective function is defined by Equation (5), where $T_w(c)$ and $T_r(c)$ are the waiting time and the traveling time of customer c, respectively. Generally, there is a trade-off relation between T_w and T_r. For example, if agents transport large customers at once like buses, T_w decreases and T_r increases. While, if agents transport one by one like taxis, T_w increases and T_r decreases. Therefore, it is desirable to adjust behaviors of agents to their situations. We introduce weighting factors which decide behaviors of agents into the reward function.

$$\min \left(\sum_{c \in C} \frac{T_w(c) + T_r(c)}{|C|} \right) \tag{5}$$

4 Reinforcement Learning

In this section, we summarize the concept of reinforcement learning. The framework is defined by the theory of Markov Decision Process (MDP).

An agent senses his current state s_t and takes any action a_t, and transits to the next state s_{t+1}. The probability of transition is given by conditional probability P as Equation (6).

$$P = P\{s_{t+1} | s_t, a_t\} \tag{6}$$

When the agent transits to some state s_t, the agent may receive reward R_t depending on the state. The probability of an agent takes an action a at the state s is denoted by policy $\pi(s, a)$. The purpose of agents is to establish optimal policy $\pi^*(s, a)$ which maximizes expected value of the rewards V_t^π as Equation (7), where γ ($0 < \gamma < 1$) is a discount rate. Thus, if γ is close to 0, the agents give greater importance to immediate reward. While, if γ is close to 1, the agents give greater importance to the sum of rewards in the future. The estimated value for the expected rewards is denoted by $Q(s, a)$. The agents have to update the value of $Q(s, a)$ through their experiences in an appropriate strategy.

$$V_t^\pi = R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \cdots \tag{7}$$

Q-Learning [10], which we applied to this route planning problem, is one of the strategies to update the value of $Q(s, a)$. The update formula of Q-Learning is defined by Equation (8), where α is a learning rate, and γ is a discount rate. This strategy is guaranteed to converge to an optimal Q value in the limit when the learning rate α is set to appropriate value.

$$\begin{aligned} Q(s_t, a_t) = & (1-\alpha) Q(s_t, a_t) \\ & + \alpha \left[R_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) \right] \end{aligned} \tag{8}$$

5 Route Planning with Reinforcement Learning

In this section, we present our strategy based on Q-Learning for transport systems. There are three key definitions to be considered to acquire appropriate policies: state and action, reward, and penalty.

5.1 State and Action

An agent can transit to an adjacent node from the current node at one transition time T_e which is defined by Equation (9), where $length(e)$ is a length of edge e, and $speed(v)$ is speed of agent v. For simplicity, in this paper lengths of all edges and speeds of all agents are set to the same value, respectively. Thus, the transition time T_e is a fixed value and can be used as a unit time in the Markov Decision Process (MDP).

$$T_e = \frac{length(e)}{speed(v)} \tag{9}$$

A state is defined by a sequence of nodes which includes the current node n_t and a history of δ visited nodes in the recent past $n_{t-\delta}, \cdots, n_t$ as Equation (10).

$$s_t = (n_{t-\delta}, \cdots, n_{t-1}, n_t) \tag{10}$$

Let $L(n_t)$ be a set of link-nodes of node n_t. An action a_t is defined by a link-node as Equation (11).

$$a_t = n_{t+1} \in L(n_t) \tag{11}$$

Here, setting the length of histories δ to 1, a pair of a state and an action is given by a sequence of three nodes as Equation (12).

$$(s_t, a_t) = (n_{t-1}, n_t, n_{t+1}) \tag{12}$$

This definition means that expected rewards depend on not only the current node but also nodes visited in the past. For example, in Figure 1, two agents v_1 and v_2 are located in the same node n_c. While, nodes visited in the recent past are different between v_1 and v_2, i.e., the state of v_1 is (n_a, n_c), and the state of v_2 is (n_b, n_c). In this case, expected rewards, which the agents receive when they arrive at n_d in the next time, are different between v_1 and v_2. In this problem model, there are trends of routes (i.e. from where to where) although each customer may have different get-on/off nodes so that the history of visited nodes is useful to estimate expected rewards. However, it is obvious that the long length of history causes the explosion of the number of states. The maximum number of states is calculated by Equation (13) where $|N|$ is the number of nodes, and $|L|$ is the maximum size of link-nodes.

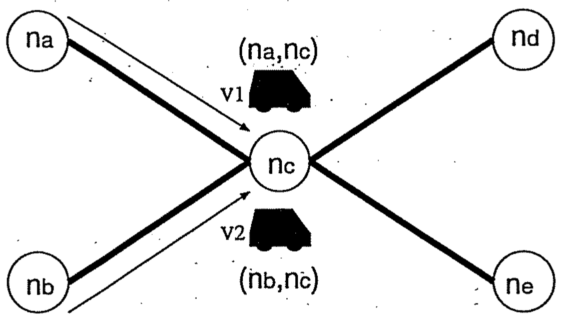

Figure 1. History of visited nodes

$$|N| \times |L|^{\delta} \tag{13}$$

5.2 Reward

A reward function for agent v which arrives at a node n is calculated by Equation (14). A reward value for riding customers is given by the left term where $|C_r^v(n)|$ is the number of customers getting on, and ω_r is its weighting factor. While, a reward value for dropping customers is given by the right term where $|C_d^v(n)|$ is the number of customers getting off, and ω_d is its weighting factor. Note that the rewards which agents received may be different even if the agents arrive at the same node. The balance between ω_r and ω_d represents preferences of transport behaviors for agents (i.e., large customers at once or one by one).

$$R^v(n) = \omega_r \cdot |C_r^v(n)| + \omega_d \cdot |C_d^v(n)| \tag{14}$$

5.3 Penalty

If once an agent v arrives at a node n, the reward of node n for agent v is absolutely 0, independently of its expected reward although the reward increases with time after that. However, agents always lean toward nodes with high expected rewards. Agents should avoid such nodes temporarily. Therefore, we introduce a penalty function as Equations (15) and (16), where $idle(n)$ is the idle time (i.e., the time lag between two visits), and ζ is the number of penalties. The value of penalty increases from 0 to 1 with increasing idle time. If the idle time is more than μ (the time period under penalty), the value of penalty is always 1. The penalty $P^v(a_t)$ is used as a weighting factor of $Q(s_t, a_t)$.

$$\mu = T_e(\zeta + 1) \tag{15}$$

$$P^v(a_t) = \begin{cases} 1 & (if \;\; \mu < idle(n_{t+1})) \\ \log\left(\frac{(idle(n_{t+1}) - T_e)(e-1)}{\zeta \times T_e} + 1\right) & (else) \end{cases} \tag{16}$$

5.4 Policy

The selection of an action a_t is followed by ϵ-roulette method. In other words, an action is selected randomly in probability $\epsilon (0 \leq \epsilon \leq 1)$. Otherwise (i.e., with probability $1 - \epsilon$), an action is selected by Equation (17). The equation represents the probability that action a_t is selected from $L(n_t)$. Not that the expected value $Q(s_t, a_t)$ is penalized by weighting factor $P^v(a_t)$.

$$Pr^v(s_t, a_t) = \frac{P^v(a_t) \cdot Q(s_t, a_t)}{\sum\limits_{a'_t \in L(n_t)} P^v(a'_t) \cdot Q(s_t, a'_t)} \tag{17}$$

5.5 Cooperation

In the previous section, we presented the strategy for independent learners. In multi-agent case, a cooperation among agents is required. However, we have some problems to be solved for extending our strategy to cooperation learners. The complexity mainly results from the fact that the rewards, which agents received, may be different depending on their assigned customers even if the agents arrive at the same node. We have an idea to overcome the complexity. It is to divide a service area into sub-areas and each agent is assigned to one of the sub-areas. Each agent has only to learn trends of flows in its responsible sub-area. The definition of boundaries among the sub-areas is a key problem to be solved. However, this work is our future work. Thus, we only consider a single-agent case in the next section.

6 Experiments

In this section, we report our experimental results and make the effectiveness of our proposal strategy clear.

6.1 Environment

We developed a simulator to investigate an effectiveness of our strategy. A graph in the simulator consists of 10 nodes and 15 edges. We used ten sets of flow patterns ($F1, \cdots, F10$). Each set includes five flows which is randomly generated. However, the sum of occurrence rates η is set to 0.3 in all the sets (i.e., the total number of customers is almost same). We simulated 10000 cycles for training and 20000 cycles after the training for evaluation. In the evaluation cycles, the probability of random selection ϵ is set to 0. The default parameter setting is summarized in Table 1.

Table 1. Parameter setting

learning rate α	0.1
discount rate γ	0.5
history size δ	1
transition time T_e	$5T_u$
weighting factor of riding ω_r	1
weighting factor of dropping ω_d	1
number of penalties ζ	9
probability of random selection ϵ	0.2

6.2 Fixed route vs. Learned route

First, we compare two types of agent (fixed route vs. learned route). The fixed route is the shortest possible route visiting all nodes of the graph. The fixed route can be regarded as an optimal route under condition of the agent without knowledge about flows. While, the learned route corresponds to the policy of agent which acquired in the training.

Figures 3 and 4 show the average of waiting and traveling times of customers. Figure 5 shows the average of total time (i.e., the sum of waiting and traveling times). The flows ($F_1, \cdots, F_{10}$) are sorted in increasing order of the total time. From the results, it can be seen that learned route agents are superior or equivalent to fixed route agents. The number of superior results related to total time is eight flows. Even if not so (i.e., F_6 and F_{10}), the differences of them are not significant. Paying attention to the fixed route, we can see that the variance of waiting time is small, while the variance of traveling time is large. The reason is that the visit order of nodes strongly influence the traveling time. For example, consider a moniliform graph, it is important to decide whether the agent visit nodes in clockwise direction or not. In $F4$, the average number of nodes between get-on/off on the fixed route is about 7.0 so that the value of traveling time is high. If the agent visits in the reverse order of the fixed route, the average number is about 2.9. However, the traveling time of other flows (i.e. all flows except for $F4$) may increase. While, paying attention to the learned route, we can see that the both variances are large. It means that the agent changes its transport behavior (i.e., large customers at once or one by one) depending on the flow. For example, in $F1$ there are two flows: $f_1 = (n_1 \rightarrow n_3)$ and $f_2 = (n_2 \rightarrow n_3)$. And, the number of nodes between n_1 and n_2 is 1 on the topology of graph. In such situation, the agent visits nodes in the order n_1, n_2, and n_3 (or n_2, n_1, and n_3) in Figure 2. This behavior of the agent can significantly decrease the waiting time of customers although the traveling time may slightly increase. Consequently, we can say that this adaptability of the agent enables proactive route selections for improving the profitability of transport systems and the convenience of customers.

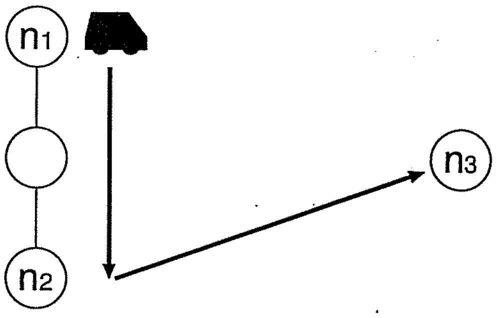

Figure 2. Transport behavior

6.3 Reward and Penalty

Second, we investigate influences related to the reward function and the penalty function.

Figure 6 shows the results related to the rate of rewards. The rate of rewards $\omega_r : \omega_d$ is changed from 0.4 : 1.6 to 1.6 : 0.4. From the result, it can be seen that the rate of rewards can emphasize the transport behaviors of the agent (i.e., large customers at once or one by one). Here, let's focus on a relativity between the rate of rewards and the total time. In this case, the high rate of ω_r causes the increasing of total time although the waiting time decreases. It means

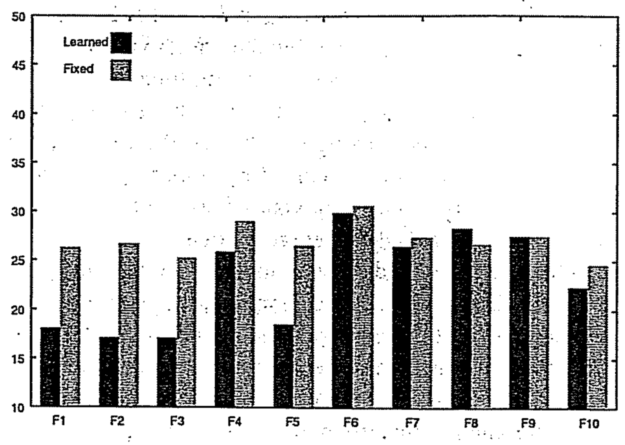

Figure 3. Waiting time

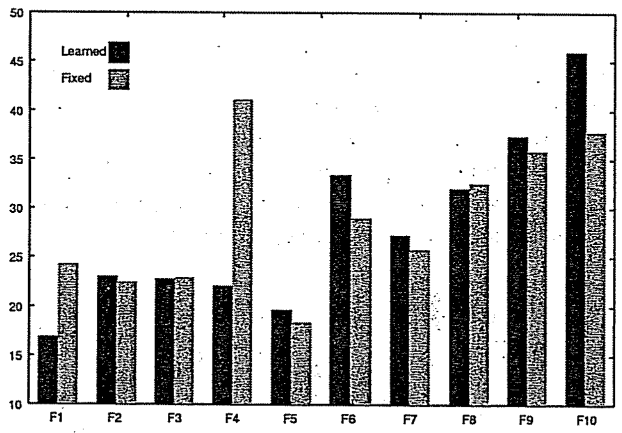

Figure 4. Traveling time

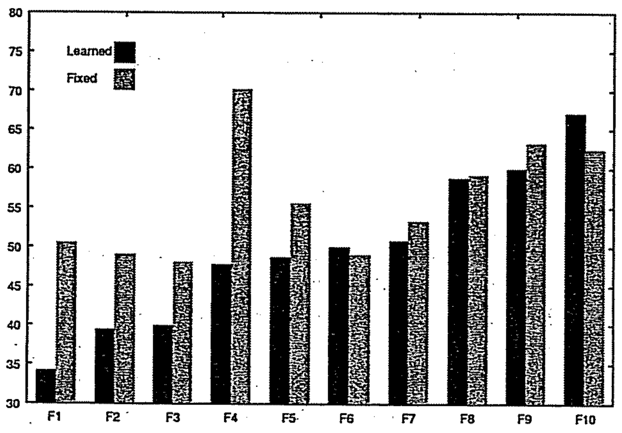

Figure 5. Total time

that the rate of rewards should change depending on not only the types of vehicles but also the patterns of flows.

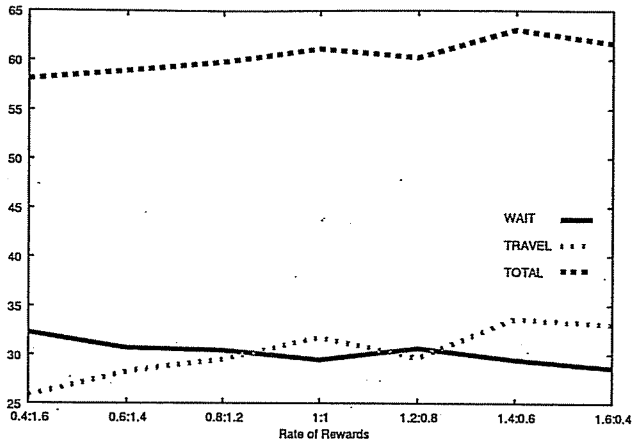

Figure 6. Results related to rate of rewards

Figure 7 shows the results related to the number of penalties. The number of penalties ζ is changed from 0 to 9. From the result, it can be seen that the large number of penalties decreases the amount of times for customers. However, too many number of penalties cause the diminishing of search effects followed by expected rewards (i.e., the agent selects next action randomly independently of expected rewards). Thus, an appropriate number of penalties must be set on the basis of the environment (e.g. the number of nodes).

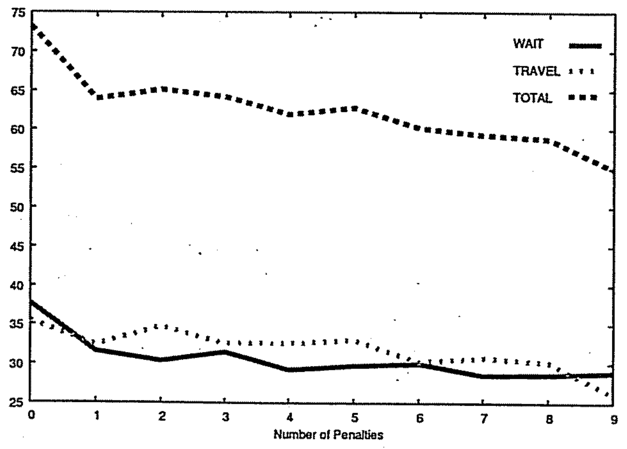

Figure 7. Results related to number of penalties

7 Conclusions

In this paper, we proposed a learning architecture for adjusting transport trends autonomously and dynamically. The route planning problem for transport systems is modeled as a reinforcement learning problem. In the model, agents acquire an optimal policy based on Q-Learning. The policies which agents acquired enable proactive route planning, i.e., transport vehicles can foresee the next destination which provides high rewards. There are three key definitions to acquire the policy: state and action, reward, and penalty. Finally, we reported our simulation results. The result indicates that our strategy outperforms the strategy of fixed route when there are characteristic transport trends in a service area.

In our future work, we have to consider the reinforcement learning in multi-agent cases. As discussed previously, the complexity of the multi-agent cases mainly results from the fact that the rewards which agents received may be different even if the agents arrive at same node. Our final target of this work is to achieve a future city in where there are driver-less cars equipped with a learning architecture.

8 Acknowledgments

We would like to thank the 21st Century COE Program for 2002. And, we acknowledge to Prof. Naohiro Ishii of Aichi Institute of Technology.

References

[1] A. Almeida, G. Ramalho, H. Santana, P. A. Tedesco, T. Menezes, V. Corruble, and Y. Chevaleyre. Recent advances on multi-agent patrolling. In *SBIA*, pages 474–483, 2004.

[2] M. Desrochers, J. Lenstra, M. Savelsbergh, and F.Soumis. Vehicle routing with time windows: Optimizatin and approximation. *Vehicle Routing: Methods and Studies*, pages 65–84, 1988.

[3] L. M. Gambardella, Éric Taillard, and G. Agazzi. MACS-VRPTW: A Multiple Ant Colony System for Vehicle Routing Problems with Time Windows. In D. Corne, M. Dorigo, and F. Glover, editors, *New Ideas in Optimization*, pages 63–76. McGraw-Hill, 1999.

[4] S. J. Louis, X. Yin, and Z. Y. Yuan. Multiple vehicle routing with time windows using genetic algorithms. In P. J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, and A. Zalzala, editors, *Proceedings of the Congress on Evolutionary Computation*, volume 3, pages 1804–1808, Mayflower Hotel, Washington D.C., USA, 6-9 1999. IEEE Press.

[5] A. Machado, G. Ramalho, J.-D. Zucker, and A. Drogoul. Multi-agent patrolling: an emprical analysis of alternative architectures. In *3rd. International Workshop on Multiagent Based Simulation*, 2002.

[6] R. Montemanni, L. Gambardella, A. Rizzoli, and A. Donati. A new algorithm for a dynamic vehicle routing problem based on ant colony system. In *Proceedings of ODYSSEUS 2003: Second International Workshop on Freight Transportation and Logistics*, pages 27–30, Palermo, Italy, 2003.

[7] J.-Y. Potvin and S. Bengio. The vehicle routing problem with time windows — part II: Genetic search. *INFORMS Journal on Computing*, 8:165–172, 1996.

[8] H. Santana, G. Ramalho, V. Corruble, and B. Ratitch. Multi-agent patrolling with reinforcement learning. In *Proceedings of International Conférence on Autonomous Agents and Multi-Agents Systems*, pages 1120–1127, 2004.

[9] M. Solomon and J. Desrosiers. Time window constrained routing and scheduling problems. *Transportations Science*, 22:1–13, 1988.

[10] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction.* MIT Press, Cambridge, MA, 1998. A Bradford Book.

[11] S. Thangiah. Vehicle routing with time windows using genetic algorithms. *Application Handbook of Genetic Algorithms: New Frontiers, Volume II. Lance Chambers (Ed.), CRC Press*, pages 253–277, 1995.