Distributed Visitors Coordination System in Theme Park Problem

Takashi Kataoka[1], Hidenori Kawamura[1], Koichi Kurumatani[2], and Azuma Ohuchi[1]

[1] Graduate School of Information Science and Technology, Hokkaido University, North 14, West 9, Kita-ku, Sapporo 060-8628, Japan
{kataoka, kawamura, ohuchi}@complex.eng.hokudai.ac.jp
http://ses3.complex.eng.hokudai.ac.jp/index_e.html
[2] Information Technology Research Institute, National Institute of Advanced Industrial Science and Technology, Aomi 2-41-6, Koto-ku, Tokyo 135-0064, Japan
k.kurumatani@aist.go.jp

Abstract. A distributed visitors coordination system is proposed as an application of a massively multi-agent system. In the system, some agents register their next destination using an information device such as a cellular phone, and this information is used to reduce the effect of the time delay between decision-making and effect-emergence. This delay causes queue lengths to oscillate. However, it is troublesome for agents to continuously register their next destination. To compensate them, exclusive queues are made available to agents registering their next destination. Computer simulation of the theme park problem, showed that when all agents avoid the congestion by registering their next destination, the total waiting time is minimized and queue length oscillation is reduced.

1 Introduction

The ubiquitous environment is now being realized with the growing use of such devices as personal digital assistants (PDAs) and cellular phones and advances in the network environment using proximity communication represented by wireless LAN[1]. A ubiquitous environment enables users to share and transmit information easily, anytime and anywhere. For example, it enables realization of a navigation system, which provides information in real time. Moreover, it enables a user to make a reservation at a hotel or to operate household appliances while away form home[2, 3]. Thus, people's lives can be made more convenient by developing various types of user supports. A new type of support for the ubiquitous environment is called "mass user support"[1]. This support aims at not only optimizing individual utility but also supporting a social system consisting of many users. One important research area in mass user support is flow control in which there are many people and each person has individual utilities based on his or her preference and restrictions. An example application area is a congestion information service. Such services are gradually being realized though the

T. Ishida, L. Gasser, and H. Nakashima (Eds.): MMAS 2004, LNAI 3446, pp. 335–348, 2005.

use of information providing systems such as a vehicle information and communication system (VICS)[4]. Particularly important in such systems is a control for reducing congestion based on global information. Research on this has been done using simplified models of theme parks[7], event halls[5], traffic[2, 6], and so on. In these models, visitor agents select the next destination "congestion disregarding behavior" or "congestion avoiding behavior" by using congestion information. Simulations have shown that, the waiting time for all agents is not necessarily minimum if too many agents avoid congestion.

In a network, one way to solve the problem caused by time delay is to use a congestion avoidance algorithms, such as RED[10]. A packet routing algorithm may thus be applicable to theme parks, for example. In RED, some packets are discarded if crowding occurs, and they are transmitted again after a while. However, although it may be easy to discard packets, it is hard to make visitors leave a theme park. Therefore, applying congestion avoidance algorithms to a theme park is challenging.

We previously clarified the relationship between the waiting time for all agents and congestion avoiding behavior[7]. We showed by experiment that congestion avoiding agents continually went to the emptiest place if they received congestion information within a certain time frame, and they concentrated there. Since that place remains the emptiest until agents begin arriving there, other congestion avoiding agents continue to receive the same congestion information, despite the fact that some agents are already moving toward that place. Therefore, that place becomes overcrowded due to excessive agent concentration. Another place then becomes the emptiest place, and the cycle repeats. The queue length in each place thus oscillates. This is because it takes time from when agents select their next destination based on the latest congestion information to when they join the queue at the destination. Briefly, there is a delay existed between decision-making and effect-emergence. We can reduce total waiting time by reducing this delay.

A congestion information system that provides global information based on the current state of each service facility is thus not very effective. It is necessary to provide information that is calculated discretely as an application of a massively multi-agent system. To solve time delay problem, a method is needed that will enable the system server to estimate the future congestion situation at each service facility, discretely. The situation in which many agents share limited resources has been modeled as El Farol Bar problem[8] and the minority game[9]. In these models, many agents select one of two resources based on the past winners' selections. If the number of past selections is suitable, the resources are shared well. As the result of individual agent's selection based on the same information, optimal resource sharing is achieved. However, when the ratio of sharing the resources changes, these models are not effective.

Rump and Stidham tested a model in which each agent avoided congestion by estimating the future congestion situation and then visiting a service facility[11]. As a result, the queue lengths oscillated chaotically. Ikeda and Tokinaga achieved higher dimensional accuracy with genetic programming[12] and managed to control the chaotic oscillation . However, in these studies, there was only one service facility, and agents only selected whether to visit the facility or not. It may be more difficult to control the oscillation with genetic programming if there are multiple facilities. When an agent performs an action determined by genetic programming and the action is fed back to the environment, a more complicated phenomenon may appear that is impossible to control by genetic programming.

It is difficult to estimate future states based only on for the current congestion situation. To solve this problem, we developed the distributed visitors coordination system (DVCS). Visitor agents select a destination based on either congestion disregarding behavior or congestion avoiding behavior. The congestion avoiding agents register their next destination into a system server using some kind of information device, such as a cellular phone. This enables the server to estimate future queue lengths with more accuracy and provide congestion information that moderates the oscillation in queue lengths. However, it is troublesome for an agent to continually register its next destination. To compensate them for their efforts, we added exclusive queues for them. We evaluated the effectiveness of this system for the theme park problem with two types of agents, ones not using the congestion information provided and ones using it.

In Section 2 we introduce the general definition of the theme park problem. Section 3 describes the distributed visitors coordination system, and Section 4 describes the simulation. Section 5 presents some experimental results, and the results are discussed in game-theoretic terms in Section 6. We conclude in Section 7 with a summary and a look at future work.

2 Theme Park Problem

The objective in the theme park problem is to maximize the satisfaction of all agents by controlling their visiting schedule. Many agents visit a theme park consisting of several attraction, roads, and so on, as shown in Fig. 1. We model the theme park using a directed graph in which each segment is a node.

Segment i has three static attributes and three dynamic ones. The static ones are capacity (c_i), service time (st_i) and visited probability, p_i. Capacity is the maximum number of visitor agents who can be served at one time by a segment. Service time in the number of steps a visitor agent wait at each segment. Visited probability is a parameter of the attractions. The higher the probability, the more agents who visit. The dynamic ones are $Ls_i(t)$, the list of agents being served by a the segment at time t, $Lq_i(t)$, the list of agents waiting for service by a segment at time t, and the remaining service time, $st_{rest,i}(t)$, which is a parameter of the attractions. Initially, this value is equal to st_i, is decremented in steps while Ls_i has at least one agent. When it reaches 0, it is set to st_i again.

Agent j has p_{ji}, v_{ji}, des_j, wt_j, mt_j, and pt_j as parameters, v_{ji} represents the state whether agent j has visited segment i or not. This value is initially 0, it is set to 1 if the agent visits segment i. Preference value p_{ji} represents the agent's preference degree for attraction i, and the value is 0 or 1. If it is 1, the agent wants to visit attraction i, if it is 0, the agent does not want to visit there.

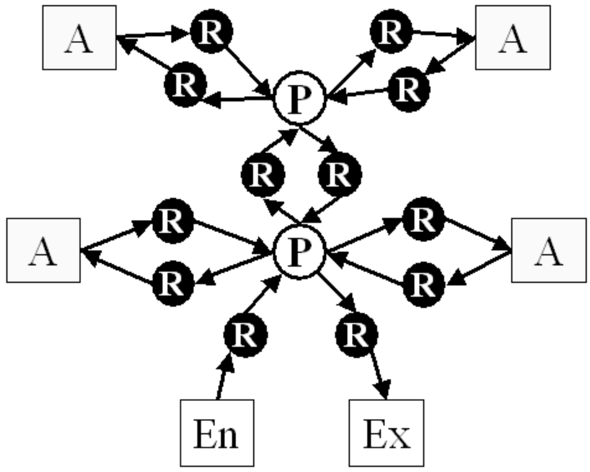

Fig. 1. Example theme park modeled by directed graph. Notations, A, R, P, En, and Ex represent attraction, road, plaza, entrance, and exit, respectively

The preference values are decided randomly for all attractions based on p_i when an agent enters the park. The current destination segment is des_j, wt_j and mt_j represents total waiting time and total moving time, respectively. Past time pt_j is the time that agent j was on the list of agents being served by the present segment.

An agent coming to segment i at time t is added to the end of queuing list $Lq_i(t)$. If the transition condition is satisfied, the agent at the head of $Lq_i(t)$ can transit to $Ls_i(t)$, the list of agents being served. An agent who transits to $Ls_i(t)$ is deleted from Lq_i. Thus, an agent's priority is FIFO (first-in first-out) queuing. The transition condition is as follows.

if segment i is an attraction,	$st_{rest,i} = 0$ in time t
otherwise,	$\lvert Ls_i(t)\rvert + 1 \leq c_i$

The $|Ls_i(t)|$ represents the number of agents on the list of agents being served. We set that an attraction cannot service visiting agents until it finishes serving the previous visiting agents. For the other segments, an agent can transit to list $Ls_i(t)$ from list $Lq_i(t)$ even if the segment is serving another agents in time t.

At time t, the agents act in turn based on their numbers. All agents necessarily belong to one of two lists, $Lq_i(t)$ and $Ls_i(t)$, in one of the segments. If agent j is on queuing list $Lq_i(t)$ and the transition condition described above is satisfied, it transits to $Ls_i(t)$, otherwise,

$$wt_j \leftarrow wt_j + 1, \tag{1}$$

and the agent remains on the list Lq_i. If agent j is on the list $Ls_i(t)$, segment i is an attraction, and $st_{rest,i} = 0$, agent j selects an attraction as his next destination based on a strategy described later. Then, des_j is changed to that attraction, and agent j transits to the queuing list of the next segment. If segment

i is an attraction and $st_{rest,i} \neq 0$, agent j does nothing special. If segment i is not an attraction,

$$pt_j \leftarrow pt_j + 1, \tag{2}$$

and if it is a road,

$$mt_j \leftarrow mt_j + 1. \tag{3}$$

Then, if $pt_j \geq st_i$, $pt_j = 0$ and agent j transits to the queuing list of the next segment.

3 Distributed Visitors Coordination System

As mentioned above, the waiting time for all agents is not necessarily minimum if too many agents try to avoid congestion by using current congestion information. This is because there is a delay between decision-making and effect-emergence. Therefore, the attractions in turn become overcrowded due to excessive agent concentration, and their queue lengths oscillate.

To prevent this, the system server must provide congestion information that reduces the effect of the delay. One approach is to have the system server estimate the future state of each attraction's queue and provide information to the agents that stimulates them to visit attraction that will become empty in the future . To achieve high accuracy, the system server needs to know agent's next action. Therefore, we considered making agents register their next destination into the server. If the server knows the number of agents moving toward each attraction at a certain time beforehand, it can prevent agent concentrate on at those attractions. In our proposed distributed visitors coordination system, agents register their next destination into the system server using some kind of information devices, such as a cellular phone, immediately after they select next destination. In particular, the system server provides agents the estimated wait time at each attraction as congestion information. At time t, the estimated wait time at attraction i is

$$EstimatedTime_i(t) = \alpha(|Lq_i(t)| + |Lq_{ex,i}(t)|) + \beta num_{i,reg}(t), \tag{4}$$

where $Lq_{ex,i}$ denotes the exclusive queue, described later, and $num_{i,reg}(t)$ denotes the number of agents who have registered attraction i as their next destination. It is decremented by 1 when an agent who has registered it as their next destination arrives there and is added to either of the two queuing lists, $Lq_i(t)$ or $Lq_{ex,i}(t)$.

Provision of the estimated wait time calculated using (4) makes the attraction with the minimum value the emptiest one when a congestion avoiding agent using this information arrives there. The result is no overcrowding. However, it is troublesome for agents to always register their next destination. To compensate them for their trouble, we added another queue, $Lq_{ex,i}(t)$, at each attraction. It is an exclusive queue for agents who registered attraction i as their next destination. The other agents stand in either $Lq_i(t)$ and $Lq_{ex,i}(t)$, depending on

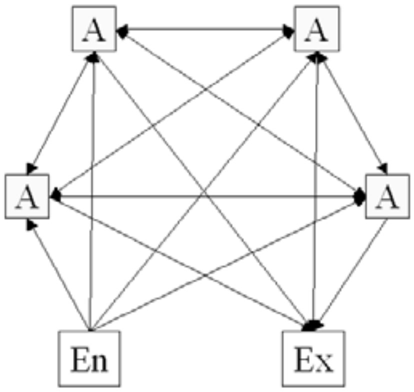

Fig. 2. Example situation for simulation. Each arrow represents a road segment

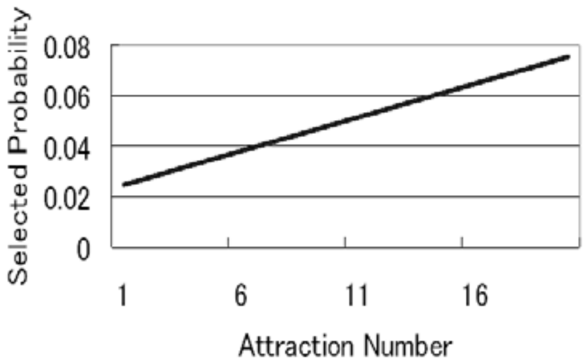

Fig. 3. Selected probability of each attraction

which one. When agents transit from $Lq_i(t)$ or $Lq_{ex,i}(t)$ to $Ls_i(t)$, they transit alternately from the two lists. Therefore,

$$\forall t, |Lq_i(t)| \geq |Lq_{ex,i}(t)|. \tag{5}$$

The agents who registered the attraction as their next destination have a shorter wait, which promotes agent registering.

While this system may seem like a "time-specific reservations" system, which is being used in some theme parks, the waiting time for all agents may not be reduced as much with a "time-specific reservations" system. This is because the waiting time of an agent who has been able to make a reservation decreases while that of an agent who has not been able to make a reservation increases inversely. The advantage of that system is that the agent who has been able to make a reservation can receive service with a shorter waiting time at certain attractions. On the other hand, in our proposed system, the objective is to reduce the effect of time delay by increase the accuracy of estimating future queue lengths by learning agents' actions beforehand.

4 Simulation

We simulated our system using the theme park model shown in Fig. 2. Each attraction is connected to every other attraction by a road segment. Service time st_i for each road segment was 30, meaning it takes 30 steps to traverse each road segment. These 30 steps are the time delay because it takes 30 steps for agents to arrive at their destination from the time they select their next destination immediately after visiting the current attraction. There were 20 attractions in the park.

Each agent used one of two strategies for destination selection. With the congestion disregarding (CD) strategy, the agent does not consider congestion and

Table 1. Parameter settings

Capacity of attractions	10
Service time of attractions	15
Capacity of roads	∞
Service time of roads	30
Average arrival rate	0.8

selects attraction i randomly as his next destination from among the attractions for which p_{ij} is maximum, from the non-visited attractions. With the congestion avoiding (CA) strategy, the agent selects the attraction with the shortest expected wait time, which was received as congestion information, from the non-visited attractions.

An agent enters the theme park at the average arrival rate, λ, of 0.8. When an agent enters, he is defined as either a CD agent or a CA agent with probability P_{CA}, which is the ratio of CA agents in the park. Once an agent is defined as one or the other, he cannot change. Each agent selects 15 attractions to visit based on each selected probability at the entrance. The selected probability of each attraction is shown in Fig. 3. Agent j is set to $p_{ji} = 1$ with a higher probability for an attraction that has a large i. That is, more agents visit attractions with a large i. The selected probability of each attraction means the degree of its popularity. We set the attraction visited by many agents and not one purposely. Therefore, it is necessary to evenly distribute the agents by providing them congestion information. Since the average arrival rate was 0.8, there were empty attractions and overcrowded ones at a just suitable rate.

We simulated three cases to determine the effect of using our system.

1. **onlyCA**
   In the first case, CA agents avoid congestion based on the estimated wait time, which is calculated simply from the queue length at each attraction at that time. That is, CA agents are provided the estimated wait time at each attraction as congestion information, which is the value given by Eq. (4) when $\alpha = 1$ and $\beta = 0$. CA agents do not register their next destination, and there is no exclusive queue at each attraction.
2. **CA_noDelay**
   In the second case, the service time of the road segments between attractions is set to 0, i.e., no time delay. This means that agents can join the queue of the emptiest attraction instantly. Although this situation is unachievable, we used it for comparison. The other settings were the same as in case 1.
3. **CA_DVCS**
   In third case, we applied our proposed distributed visitors coordination system (DVCS). CA agents register their next destination and can stand in an exclusive queue. They are provided the estimated wait time at each attraction as congestion information, which is the value given by Eq. (4) when $\alpha = 1$ and $\beta = 1$. These parameters were selected to equally treat

the queue length and number of agents registering as factors of congestion information. However, they must be investigated further to determine which value is optimal. On the other hand, CD agents select their next destination randomly and cannot stand in the exclusive queues.

The parameter values for the simulations are shown in Table. 1. Each simulation ended once 5000 agents had left the park. We used the waiting time as the measure of satisfaction, averaged over 100 simulations.

5 Results

The average waiting time for all agents is shown in Fig. (4). A ratio of CA (congestion avoiding) agents equals 0.0 means there are only CD (congestion disregarding) agents, and $P_{CA} = 1$ means there are only CA agents. In the onlyCA case, the average waiting time for all agents was not minimum at $P_{CA} = 1$, as in previous studies[5, 6, 7], which is the situation in which all agents simply avoid congestion. In the range of $P_{CA} \leq 0.3$, the average waiting time increased as P_{CA} decreased because of the time delay. In the CA_noDelay case, which is the ideal situation, the average waiting time decreased as P_{CA} increased and was minimum at $P_{CA} = 1$. Without a time delay, CA agents do not crowd excessively, and they fill the attraction queues uniformly. In the CA_DVCS case the tendency was the same as in the CA_noDelay case. This means that CA agents are directed toward the attractions uniformly if the system server knows the number of agents moving, even if an agent cannot immediately actually stand in a queue. As shown in Fig. 4, the average waiting time for all agents in the CA_DVCS case was almost equal to that in the CA_noDelay case. However, in

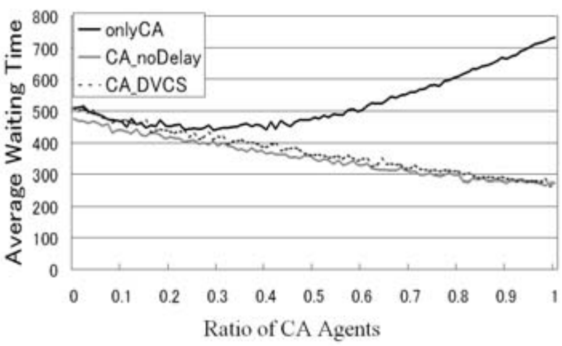

Fig. 4. Average waiting time for all agents in each case. X-axis indicates P_{CA}, ratio of CA agents, and Y-axis indicates average waiting time of whole agents

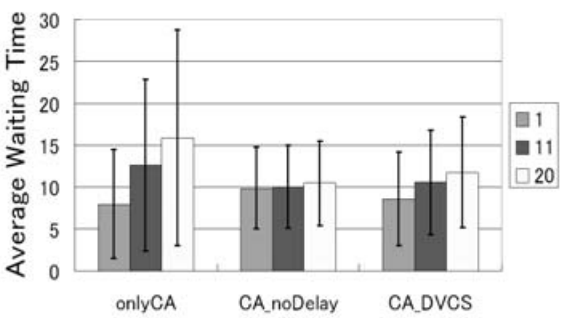

Fig. 5. Average waiting time and standard deviation for agents who visited attractions 1, 11, and 20. X-axis indicates each case, and Y-axis indicates average waiting time. Vertical lines show standard deviations. Each number indicator corresponds to attraction number

Table 2. Average waiting time for attractions 1, 11, 20

onlyCA	CA_noDelay	CA_DVCS
36.54120751	30.3851502	30.93780554

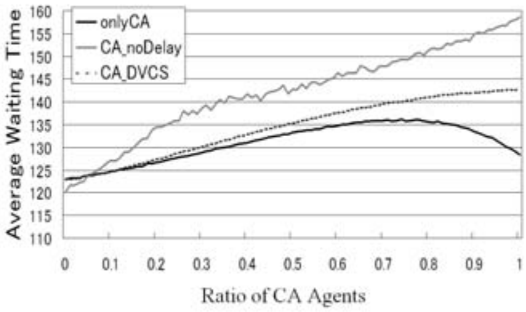

Fig. 6. Average waiting time for all agents in each case with average arrival rate of 0.3. X-axis indicates P_{CA}, ratio of CA agents, and Y-axis indicates average waiting time for all agents

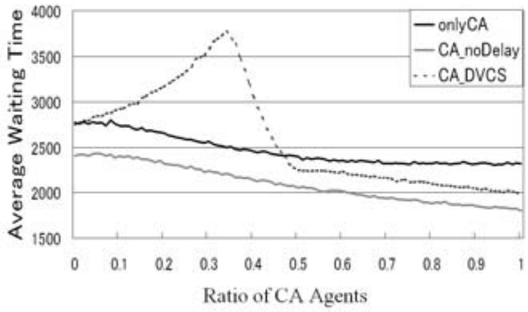

Fig. 7. Average waiting time for all agents in each case with average arrival rate of 1.1. X-axis indicates P_{CA}, ratio of CA agents, and Y-axis indicates average waiting time for all agents

the real world, the distances between attractions are not equal, so users do not take the same number of steps to get to the next attraction. These factors are not serious although they do have somewhat of an effect. Therefore, the average waiting time for all agents in the CA_DVCS case is no longer than that in the onlyCA case. However, it may be longer than that in the CA_noDelay case.

Figure 5 shows the average waiting times and standard deviations for agents who visited attractions 1, 11, 20, for $P_{CA} = 1$. Table 2 shows the average waiting times. Since more agents visited the attractions with a higher number, i, the average waiting time of agents who visited attraction 20 was maximum in all cases. However, the average waiting times in the CA_noDelay case did not differ much between attractions. This is because providing congestion information that did not include the time delay prevented the CA agents from crowding certain attractions, so they stood in the attraction queues uniformly, as mentioned above. In the CA_DVCS case, CA agents went to empty attractions although not as effectively as in the CA_noDelay case. In the onlyCA case, the queue lengths oscillated greatly, as shown by the large standard deviations. On the other hand, in the CA_noDelay and CA_DVCS cases, the oscillations were moderated. That is the distributed visitors coordination system reduced the effect of the time delay and moderated the oscillations in the queue lengths.

Figure 6 shows the average waiting time for all agents with average arrival rate of 0.3, meaning there were few agents in the park. The average waiting times in each case increased with P_{CA}, so providing congestion information was not effective. The magnitude relation of the average waiting times in two cases was inverse to that in Fig. 4. Service at the attractions begins when agents enter the queue, even if there is only one agent. In the CA_noDelay case, agents still had to wait until a few agents, which was much less than capacity c_i, finished being served even if they arrived there immediately. Since this was repeated, their waiting time increased. On the other hand, in the onlyCA case, some CA agents visited the same attraction at the same time. Since the attraction could service many agents at once, their waiting times were shorter than in the CA_noDelay case. The average waiting time in the CA_DVCS case was between those in the onlyCA and CA_noDelay cases.

Figure 7 shows the average waiting time for all agents with average arrival rate of 1.1, meaning there were too many agents in the park. In the range of $P_{CA} \leq 0.5$, the average waiting time in the CA_DVCS case was much longer. Since attractions had an exclusive queue, many CD agents waited for a longer time while CA agents waited for a shorter time by standing in the exclusive queue. The advantage of this queue decreased gradually as P_{CA} increased, and it was mostly gone at about $P_{CA} = 0.5$. However, as the number of CA agents increased, the effect of registering their next destination grew, and the average waiting time for all agents decreased. Finally, the average waiting time in the CA_DVCS case is between those in the onlyCA and CA_noDelay cases.

Thus, the distributed visitors coordination system is not necessarily effective. When there are few agents or too many agents, it is less effective. Otherwise, it is effective, such as when the average arrival rate is 0.8. We can thus say that the distributed visitors coordination system is effective in most situations.

6 Discussion

Our testing showed that having agents register their next destination reduces the time delay effect and moderates the oscillations in queue lengths. Then, what was the effect of the excusive queues for agents registering their next destination? To investigate this, we defined another case, CA_noEQ, in which there are no exclusive queues although CA (congestion avoiding) agents register their next destination. They stand in the same queues as the CD (congestion disregarding) agents. The other settings are the same.

Figure 8 shows the average waiting times of CD and CA agents in the CA_noEQ case. Figure 9 shows them in the CA_DVCS case. Figure 10 shows the average waiting time for all agents in the CA_noEQ and CA_DVCS cases. In the CA_noEQ and CA_DVCS cases, the average waiting times of CD and CA agents differed greatly although it did not differ for all agents. In the CA_noEQ case, the average waiting times of CD and CA agents were almost the same. However, the average waiting time of CA agents was slightly longer than that of CD agents, consistently. This is because the CA agents postponed visiting the crowded attractions until they became less crowded. On the other hand, in the CA_DVCS case, the average waiting times of CD and CA agents differed greatly. In the range of $P_{CA} \leq 0.8$, the average waiting time of CA agents was shorter

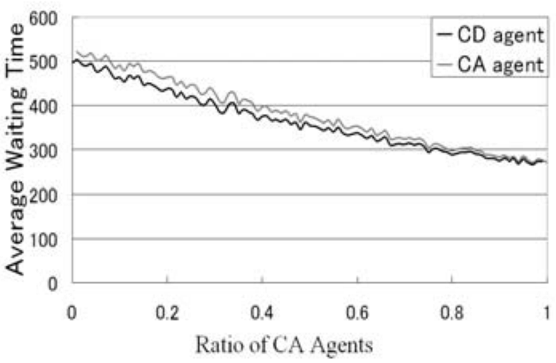

Fig. 8. Average waiting times of two kinds of agents in the CA_noEQ case. X-axis indicates P_{CA}, ratio of CA agents, and Y-axis indicates average waiting time of each agent

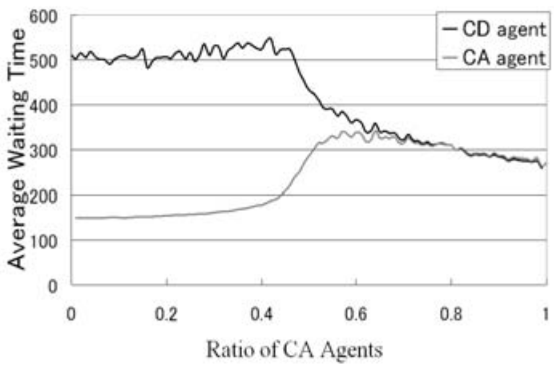

Fig. 9. Average waiting times of two kinds of agents in the CA_DVCS case. X-axis indicates P_{CA}, ratio of CA agents, and Y-axis indicates average waiting time of each agent

than that of CD agents, although it was longer in other range. The average waiting time changed at around $P_{CA} = 0.5$ because the advantage of standing in the exclusive queues drops when the number of agents eligible to stand in them exceeds 50%.

In the CA_noEQ and CA_DVCS cases, when each agent rationally selects one of the two strategies, what ratio of strategies does it become? In the CA_noEQ case, although the average waiting time for all agents was minimum at $P_{CA} = 1$, the average waiting time of CA agents was consistently longer than that of CD agents. When there are n CA agents out of $(N-1)$ agents, and Nth agent's profit, which is less waiting time, is $CA(n+1)$ or $CD(n)$ according to his selection of strategy,

$$CA(n+1) \leq CD(n), \qquad n = 0, 1, 2, \cdots, N-1 \tag{6}$$

$$CD(0) \leq CA(N). \tag{7}$$

That is, a CD agent's profit is always larger than a CA agent's one, but the profit when all agents select the CD strategy is smaller than when all agents select the CA strategy. In game theory, this is called the dilemma game of N agents. Figure 11 shows the profit function of the dilemma game with N agents, and the situation in Fig. 8 shows the same tendency because an agent's profit is less waiting time in a theme park. In Fig. 11, the Pareto optimal point is at $n/N = 1$, and the equilibrium point is at $n/N = 0$. Similarly, in Fig. 8, the Pareto optimal point is at $P_{CA} = 1$, and the equilibrium point is at $P_{CA} = 0$. That is, in the noEQ case, if each agent selects a strategy rationally, his profit will be the waiting time for $P_{CA} = 0$ because all agents select the CD strategy. This is an undesirable state.

In contrast, the two waiting times in the CA_DVCS case seem like the profit function of the minority game, which is a game in which selecting different

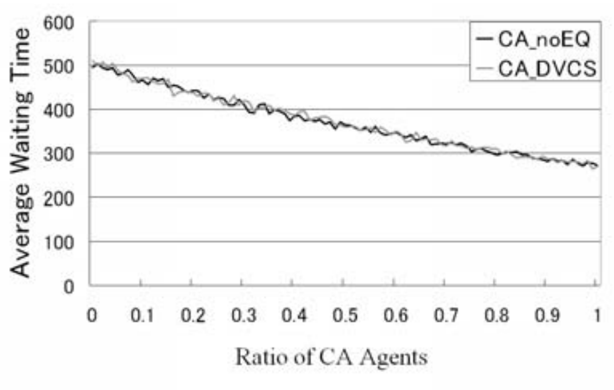

Fig. 10. Average waiting times of whole agents in the CA_noEQ and CA_DVCS cases. The X-axis indicates P_{CA}, ratio of CA agents, and Y-axis indicates average waiting time for all agents

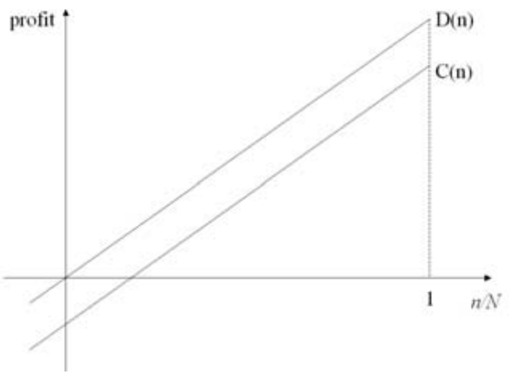

Fig. 11. Profit function of dilemma game with N agents. X-axis indicates ratio of agents who select strategy C, and Y-axis indicates profit of each strategy

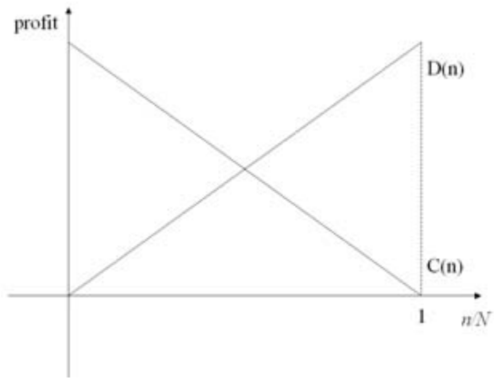

Fig. 12. Profit function of minority game with N agents. X-axis indicates ratio of agents who select strategy C, and Y-axis indicates profit of each strategy

strategies results in the optimal state. (Fig. 12 shows the profit function of the minority game.) However, in the range of $P_{CA} \geq 0.8$, both waiting times are downward-sloping although the magnitude relation of the average waiting times in two cases was inverse to that in other range. This is different them in the minority game. In the CA_DVCS case, the Pareto optimal state is also the state in which all agents select the CA strategy. This means that the situation in the CA_DVCS case is also dilemma game in the range of $P_{CA} \geq 0.8$. Therefore, the equilibrium point is at $P_{CA} = 0.8$.

If each agent learns repeatedly and selects a strategy rationally in such situation, the ratio of CA agents will converge at $P_{CA} = 0.8$, which is the equilibrium point, although this depends on the learning method. The average waiting time for all agents at $P_{CA} = 0.8$ is less than at $P_{CA} = 0$. This means that the average waiting time for all agents balances out at a more desired state than in the CA_noEQ case.

In addition, the onlyCA and CA_noDelay cases did not have exclusive, as in the CA_noEQ case, and they resulted in the dilemma game, which has a Pareto optimal point at $P_{CA} = 1$ and an equilibrium point at $P_{CA} = 0$.

Thus, adding exclusive queues shifts the equilibrium point in the increasing P_{CA} direction, although there was no change in these situations being dilemma game. In short, adding exclusive queues for agents who have the same strategy as many other agents is effective way to make agents select the same strategy.

7 Conclusion

We have described own distributed visitors coordination system that reduces the effect of the time delay between decision-making and effect-emergence. In this system, CA (congestion avoiding) agents register their next destination and can stand in exclusive queues. Simulation showed that an increase in the number of CA agents registering their next destination reduced the waiting time for all agents. Providing congestion information, the estimated future state of each attraction's queue based on knowing agents' next actions, effectively reduced the effect of the delay.

Whether the exclusive queues are added or not, the profit function for whole agents did not differ. However, doing so shifted the equilibrium point in the state where more CA agents avoided congestion, although there was no change in these situations being dilemma game. Therefore, the distributed visitors coordination system encouraged the rational agents to register their next destination, which reduced the waiting time for all agents.

Thus, our system effectively stimulates agent coordination and reduces global congestion, even if the number of agents is massive.

Some settings were simplified for the simulation. For example, the number of steps between attractions was uniform. In the real world, the number differs, so the effect of using our system may differ from the experimental one. Investigation of this is future works.

References

1. Koichi Kurumatani: User Intention Market for Multi-Agent Navigation - An Artificial Intelligent Problem in Engineering and Economic Context. In Working Note of the AAAI-02 Workshop on Multi-Agent Modeling and Simulation of Economic Systems, MAMSES-02, Technical Report WS-02-10, AAAI Press (2002) 1–4
2. Tomohisa Yamashita, Kiyoshi Izumi, Koichi Kurumatani: Effect of Using Route Information Sharing to Reduce Traffic Congestion. Multi-Agent for Mass User Support, International Workshop, MAMUS Acapulco, Mexico, August 10, 2003 Revised and Invited Papers (2003) 86–104

3. Hidenori Kawamura, Koichi Kurumatani, Azuma Ohuchi: Modeling of Theme Park Problem with Multiagent for Mass User Support. Working Note of The IJCAI-03 Workshop on Multiagent for Mass User Support, Acapulco, Mexico (2003) 1–7
4. VICS (Vehicle Information and Communication System) is the information service in ITS (Intelligent Transport Systems): http://www.its.go.jp/ITS/
5. R. Suzuki, T. Arita: Effects of Information Sharing on Collective Behaviors in Competitive Populations. Proceedings of the Eight International Symposium on Artificial Life and Robotics (2003) 36–39
6. T. Shiose, T. Onitsuka, T. Taura: Effective Information Provision for Relieving Traffic Congestion. the Proceedings of ICCIMA'01, 4th International Conference on Intelligence and Multimedia Applications (2001) 138–142
7. Hidenori Kawamura, Takashi Kataoka, Koichi Kurumatani, Azuma Ohuchi: Investigation of Global Performance Effected by Congestion Avoiding Behavior in Theme Park Problem. IEEJ Transaction EIS, Vol. 124, No. 10 (2004) 1922–1929
8. W.B. Arthour: Inductive Reasoning and Bounded Rationality. American Economic Association Papers Proceedings, Vol. 84, No. 2 (1994) 406–411
9. D. Challet, Y.C.Zhang: Emergence of Cooperation and Organization in an Evolutionary Game. Physica A, Vol. 246 (1997) 407–418
10. S. Floyd, V. Jacobson: Random Early Detection Gateways for Congestion Avoidance. IEEE/ACM Transactions on Networking, Vol. 1 (1993) 397–413
11. C.M. Rump, S. Stidham,Jr.: Stability and Chaos in Input Pricing for a Service Facility with Adaptive Customer Response to Congestion. Management Science, Vol. 44 No. 2 (1998) 246–261
12. Y. Ikeda, S. Tokinaga: Controlling the Chaotic Dynamics by Using Approximated System Equations Obtained by the Genetic Programming. Trans. IEICE, Vol. E84-A, No. 9 (2001) 2118–2127

Author Index