SOLVING REAL-WORLD DELIVERY PROBLEM USING IMPROVED MAX-MIN ANT SYSTEM WITH LOCAL OPTIMAL SOLUTIONS IN WIDE AREA ROAD NETWORK

Junichi Ochiai[1] and Hitoshi Kanoh[2]

[1]Department of Computer Science, Graduate School of Systems and Information Engineering, University of Tsukuba, Tsukuba, Ibaraki, Japan
[2]Division of Information Engineering, Faculty of Engineering, Information and Systems, University of Tsukuba, Tsukuba, Ibaraki, Japan

ABSTRACT

This paper presents a solution to real-world delivery problems (RWDPs) for home delivery services where a large number of roads exist in cities and the traffic on the roads rapidly changes with time. The methodology for finding the shortest-travel-time tour includes a hybrid meta-heuristic that combines ant colony optimization (ACO) with Dijkstra's algorithm, a search technique that uses both real-time traffic and predicted traffic, and a way to use a real-world road map and measured traffic in Japan. We previously proposed a hybrid ACO for RWDPs that used a MAX-MIN Ant System (MMAS) and proposed a method to improve the search rate of MMAS. Since traffic on roads changes with time, the search rate is important in RWDPs. In the current work, we combine the hybrid ACO method with the improved MMAS. Experimental results using a map of central Tokyo and historical traffic data indicate that the proposed method can find a better solution than conventional methods.

KEYWORDS

Ant Colony Optimization, Dijkstra's Algorithm, Delivery Problem, Traffic, Real-World

1. INTRODUCTION

Ant colony optimization (ACO) is a stochastic search algorithm for problem solving that takes inspiration from the foraging behaviors of ants. The main idea of ACO rests on the indirect communication among individuals in an ant colony based on the pheromone trails that real ants use for communication. ACO has been formalized into a meta-heuristic for combinatorial optimization problems by Dorigo et al., and many applications are now available[1]-[3]. In particular, many studies on ACO have been performed using the traveling salesman problem (TSP[4]), and it has been shown that ACO is superior to other meta-heuristics [5],[6] for this type of problem.

In this paper, we deal with real-world delivery problems (RWDPs) for home delivery services as an extension of the TSP, where a large number of roads exist in cities and the traffic on the roads rapidly changes with time. This scenario reflects the typical traffic congestion in a wide area urban road network.

The problems that deal with finding optimal tours with time-dependent travel time have been studied as TSPs, delivery problems [7], and vehicle routing problems [8]. Conventional problem-solving methods using ACO repeat a search when the traffic flow changes during movement [9]-

DOI : 10.5121/ijaia.2014.5302

[11]. However, finding the global optimal solution by this method is difficult because when the traffic changes rapidly, the information obtained from an old search may not be helpful. Furthermore, research based on real road maps and traffic information services in the real world is seldom found.

In this paper, we propose a new method to solve RWDPs using ACO and Dijkstra's algorithm (DA). Search techniques based on only the predicted traffic have previously been presented for real-world time-dependent TSPs using ACO [12] and real-world time-dependent vehicle routing problems using ACO[13], an evolution strategy[14] , and a genetic algorithm[15]. When these methods make mistakes with the prediction values, solution accuracy may deteriorate. We previously proposed a hybrid ACO that combined real-time data with predicted traffic data for RWDPs[16]. In that method, the calculation time was not discussed, even though traffic on roads changes with time. The proposed method combines the hybrid ACO technique [16] with this previous method in order to improve the search rate of the MAX-MIN Ant System[12].

In the following section, we start by describing the problem. Then, we detail the algorithm of the proposed method. Finally, we present the results of experiments using a map of central Tokyo and real traffic data.

2. OVERVIEW

In this paper, we regard the RWDP as an extension of the TSP: a vehicle starts from a depot, visits all customers without any time constraint, and finally returns to the depot. This type of problem is also called a one-to-many-to-one delivery problem[7]. Here, we first describe the time-dependent TSP (TDTSP) [17] and the calculation of the tour travel time. Next, we explain the RWDP and a traffic information service in the real world. Then, we give a brief description of ACO and provide details about MAX-MIN Ant System (MMAS) as an example of typical ACO. Finally, we present our method to improve the search rate of MMAS.

2.1. Time-Dependent Traveling Salesman Problem (TDTSP)

The TSP [4] can be represented by a complete graph $G = (N, A)$, where N is a set of nodes, i.e. cities, $n = |N|$ is the number of nodes, and A is the set of arcs fully connecting the nodes. Each arc $(i, j) \in A$ is assigned a value d_{ij} $(= d_{ji})$, which represents the distance between nodes i and j. The TSP then is the problem of finding the shortest closed tour that visits each of the nodes of G exactly once. The TSP instances used in this paper are taken from the TSPLIB benchmark library: http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/.

The TDTSP [17] extends the original TSP so that traffic congestion can be included. Let $T_{ij}(t)$ be the travel time between nodes i and j at time t; $T_{ij}(0)$ means the original travel time of a given TSP, i.e. $T_{ij}(0) = d_{ij}$. Traffic congestion can be represented by a change in the travel time. Here, this change is defined by the following formulas:

$$T_{ij}(t+\Delta t) = [T_{ij}(t) \times (1 + R_{\text{jam}} \times u)]_{T_{\min}}^{T_{\max}}, \tag{1}$$

$$[x]_b^a = \begin{cases} a & \text{if } x > a, \\ b & \text{if } x < b, \\ x & \text{otherwise,} \end{cases} \tag{2}$$

$$T_{\max} = d_{ij} \times B_{\text{upper}}, \tag{3}$$

$$T_{\min} = d_{ij}, \tag{4}$$

where Δt is an updated interval of travel time, R_{jam} and B_{upper} are the parameters showing the rate and the upper bound of traffic congestion, respectively, and $u \in [-1, 1]$ is a uniform random number. The time when a salesman leaves city 1 is set to $t = 0$, and he always starts from and returns to city 1.

In the following, the time required to travel around a tour is called the tour travel time. The tour travel time for solution S can be calculated by the following formulas:

$$T(S) = \sum_{i=1}^{n} T_{i,i+1}(t_i), \tag{5}$$

$$t_i = \begin{cases} 0 & \text{if } i = 1, \\ t_{i-1} + T_{i-1,i}(t_{i-1}) & \text{otherwise}, \end{cases} \tag{6}$$

where t_i is the time when a salesman reaches or leaves node i.

2.2. Real-World Delivery Problem

The RWDP can be represented by a quadruple $G^{\text{RW}} = (N^{\text{RW}}, A^{\text{RW}}, M^{\text{RW}}, T^{\text{RW}})$, where $N^{\text{RW}} = \{C_i \mid i = 1,...\text{n}\}$, C_1 is a depot and $\{C_2,...C_n\}$ is a set of customers, $A^{\text{RW}} = \{P_{ij}(t_i) \mid i, j = 1,...\text{n}\ (i \neq j)\}$ is a set of optimal paths from C_i to C_j at time t_i when a vehicle reaches C_i, and M^{RW} and T^{RW} are a road map and a set of time-series traffic data in the real world, respectively. Each $P_{ij}(t_i)$, generally $\neq P_{ji}(t_i)$, can be calculated using T^{RW} on M^{RW}. It is necessary to calculate the optimal path between customers in RWDPs, while the distance between cities is given in TSPs. The RWDP then is the problem of finding a shortest-travel-time tour when a vehicle starts from C_1 and returns to C_1 visiting each of the customers $\{C_2,...C_n\}$ exactly once.

The road map M^{RW} used in this paper is the standard map database that is used in actual car navigation systems. This map includes all drivable roads in Japan and its format was developed and established by the Navigation System Researchers' Association.

Historical time-series traffic data T^{RW} are also used to calculate the travel time of a vehicle. In Japan, traffic meters are installed at more than 20,000 locations along principal roads throughout the country. These meters measure the average travel time of cars passing through specific road links at intervals of one or five minutes. The data so obtained is collected at a traffic information center and provided to subscribers in real time. Figure 1 shows an example of time-series traffic data. The vertical axis represents the average speed of cars on a link calculated directly from the traffic data. In this figure, data in the range of 0:00 to 5:00, 5:00 to 8:00, 8:00 to 18:00, and 18:00 to 20:00 correspond to no congestion, outbreak of congestion, heavy congestion, and dissolution of congestion, respectively. Thus, time-series traffic data is highly nonlinear, which makes it difficult to perform accurate predictions. Application to real-world scenarios must take the prediction error rate into consideration.

2.3. Ant Colony Optimization (ACO)

The generic ACO meta-heuristic [1] is shown in Fig. 2. After initialization, the meta-heuristic iterates over two phases. First, a number of solutions are constructed by the ants, and second,

before the start of the next iteration, the pheromone trails are adapted to reflect the search experience of the ants.

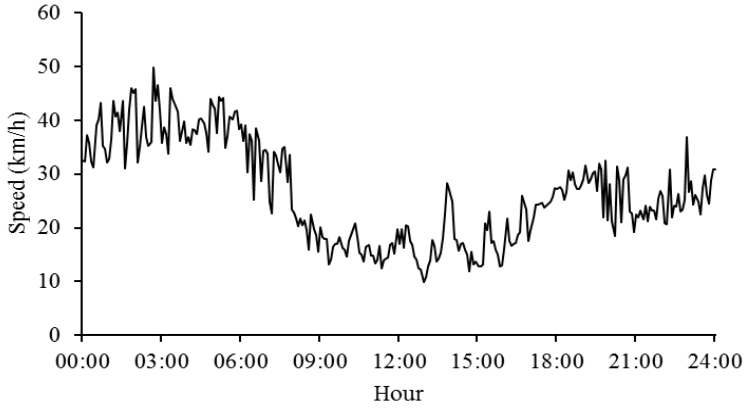

Figure1. Example of time-series traffic data.

```
procedure ACO ( )
   Set parameters;
   Initialize pheromone trails;
   while ( terminal condition not met ) do
      Construct ant solutions;
      Update pheromone trails;
   end-while
end-procedure
```

Figure 2. ACO meta-heuristic.

There have been many attempts to improve the performance of ACO. MAX-MIN Ant System (MMAS) [18] has demonstrated an especially impressive performance[1], so we used the MMAS, except for constructing the ant solution described in section 3.4, as the ACO in the proposed method.

2.4. MAX-MIN Ant System (MMAS)

Here, we describe the MMAS for original TSPs. The MMAS constructs ant solutions and updates pheromone by following (7) and (12), respectively. The MMAS has two main characteristics [2] : only the best ant updates the pheromone trails, and the value of the pheromone is bound. Here, the best means that the length of the tour is shortest. The pheromone τ_{ij}, associated with the arc joining cities i and j, is updated using (7):

$$\tau_{ij} \leftarrow [(1-\rho)\times\tau_{ij} + \Delta\tau_{ij}^{\text{best}}]_{\tau_{\min}}^{\tau_{\max}}, \tag{7}$$

$$\Delta\tau_{ij}^{\text{best}} = \begin{cases} 1/L_{\text{best}} & \text{if } (i, j) \text{ belongs to the best tour,} \\ 0 & \text{otherwise,} \end{cases} \tag{8}$$

where ρ is the evaporation rate, $\tau_{\max}$ and $\tau_{\min}$ are respectively the upper and lower bounds imposed on the pheromone, and L_{best} is the length of the tour of the best ant.

Some guidelines have been provided for defining τ_{max}, τ_{min}, and L_{best} on the basis of analytical and empirical considerations. This paper defines these parameters by the following three formulas [18] :

$$\tau_{\max} = \frac{1}{\rho \times L_{\mathrm{gb}}}, \tag{9}$$

$$\tau_{\min} = \frac{1-\sqrt[n]{p_{\mathrm{best}}}}{(n/2-1)\times\sqrt[n]{p_{\mathrm{best}}}} \times \tau_{\max}, \tag{10}$$

$$L_{\mathrm{best}} = L_{\mathrm{ib}}, \tag{11}$$

where L_{gb} is the length of the best tour from the beginning of the algorithm (global-best), L_{ib} is the length of the best tour found in the current iteration (iteration-best), and p_{best} is a parameter.

In the construction of a solution, ants select the next city to be visited through a stochastic mechanism. When ant k is in city i and has so far constructed the partial solution, the probability of going to city j is given by

$$p_{ij}^{k} = \begin{cases} \dfrac{[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{l\in N^{k}}[\tau_{il}]^{\alpha}[\eta_{il}]^{\beta}} & \text{if } j \in N^{k}, \\ 0 & \text{otherwise,} \end{cases} \tag{12}$$

where N^k is the set of cities not yet visited by the ant k. The parameters α and β control the relative importance of the pheromone trail and the heuristic information η_{ij}, which is given by

$$\eta_{ij} = \frac{1}{d_{ij}}. \tag{13}$$

2.5. Effective Initialization of Pheromone Trail

An effective solution of TDTSP requires the early convergence for the reasons below.

- It is necessary to calculate the optimal path between customers in practical problems, while the distance between cities is given in TSPs. Consequently, the order of visiting customers is calculated by ACO and the path between customers is calculated by Dijkstra's algorithm, etc.

- When traffic congestion changes during traveling, the path should be re-evaluated. Though the traffic congestion information on main roads is provided every one or five minutes in Japan, this calculation must be performed within one minute.

We previously proposed a method to improve the search rate of MMAS for TDTSPs[12]. The strategies of this method are listed below.

- In addition to the usual initial pheromone trails, additional pheromones are deposited on the solution built before the first iteration. The method helps to guide the search and so might reduce the number of points visited in the search space by giving deviations from the initial pheromone trails.

- This reduction is conducted using solutions built by a greedy algorithm. These solutions, i.e., initial solutions, can include candidate partial solutions of the global optimal solution. Moreover, the computational time of the greedy algorithm is very short; it can be considered to have no influence on the computational time of the method.
- In TDTSPs, a salesman starts from city 1 and the travel time between cities might change. In order to generate many tours using the greedy algorithm, we change the first visited city from city 1 in the greedy algorithm.

The pheromone trail initialization is shown in Fig. 3 and below:

$$\tau_{ij} \leftarrow \frac{1}{\rho \times T(S_0^{\text{best}})}, \tag{14}$$

$$\tau_{ij} \leftarrow (1-r)\times\tau_{ij} + \frac{r}{n-1}\sum_{k=1}^{n-1}\delta\tau_{ij}^{k}, \tag{15}$$

$$\delta\tau_{ij}^{k} = \begin{cases} 1/T(S_{0k}) & \text{if } (i,j)\in S_{0k}, \\ 0 & \text{otherwise,} \end{cases} \tag{16}$$

where S_0^{best} is the best solution of the initial solutions, S_{0k} (k = 1 to n –1) is the kth initial solution, and r ($0 \le r \le 1$) is a parameter.

Here, the first term in the right side of (15) means the uniform distribution of pheromones, and the second term means centralizing of the pheromones around the initial solutions. The parameter r indicates the ratio of the decentralization of the initial pheromone trails and the centralization of the initial pheromone trails. When r = 0, the method corresponds to the MMAS. In contrast, when r = 1, the initial pheromone is distributed only on the arcs contained in the initial solutions.

3. PROPOSED METHOD

3.1. General Procedure

The general procedure of the proposed method is shown in Fig. 4. A target road map including the depot and all customers is prepared along with the historical traffic data for principal roads on the map. Index i in the outermost loop corresponds to the turn at which a vehicle visits customers. When the vehicle is moving, the search is repeated. Real-time traffic data for the principal roads are input at Δt intervals, so updating the travel time, re-calculating the predicted traffic, and planning the tour of the vehicle are performed at this interval. The prediction system, as we have already reported [19] , [20] has an interpolation function as well as a prediction function. This system can estimate traffic on roads not installed with detectors from the traffic on roads that are installed with detectors. Constructing a tour by hybrid ACO is described in the next section.

procedure Initialize-Pheromone ( )
  Construct (n −1) solutions using greedy algorithm with (n −1) different second cities;
  Initialize pheromone trails as well as original MMAS; // see **Error! Reference source not found.**
  Update τ_{ij} individually using **Error! Reference source not found.**;
end-procedure

Figure.3. General procedure of pheromone trail initialization.

```
procedure main ( )
   Input RWDP;
   C_v = 1;    // 1 is depot number
   N_v = { 1 };
   for ( i = 1 to n ) do
      if ( Δt passed or i = 1) then
         Input T^RW ( real-time );
         Predict travel time for all roads on M^RW;
         Execute Hybrid-ACO ( C_v, N_v );    // see Fig. 5
      end-if
      Move a vehicle to a next customer according to best tour;
      C_v ← the number of current customer of the vehicle;
      N_v ← N_v ∪ { C_v };
   end-for
   Move the vehicle to the depot according to best tour;
end-procedure
```

Figure4. General procedure of proposed method.

3.2. Constructing a Tour by Hybrid ACO

In the proposed method, the paths between customers are planned by Dijkstra's algorithm (DA) and the route of visiting customers is constructed by ACO. The DA is widely used as a path planning method, and ACO is superior to other meta-heuristics including genetic algorithms [5] and simulated annealing [6] in terms of constructing a tour.

Figure 5 shows the general procedure of hybrid ACO, where C_v is a current customer of a vehicle and N_v is a set of customers visited by the vehicle. Ants start from the current location of the vehicle and construct a route to the depot. When the ant k is at the customer C_i in constructing a route, selection probabilities for all $C_j \in N^k$ are calculated by the following formulas:

$$p_{ij}^{k}(t_i) = \begin{cases} \dfrac{[\tau_{ij}]^{\alpha}[\eta_{ij}(t_i)]^{\beta}}{\sum_{l \in N^k}[\tau_{il}]^{\alpha}[\eta_{il}(t_i)]^{\beta}} & \text{if } j \in N^k, \\ 0 & \text{otherwise,} \end{cases} \tag{17}$$

$$\eta_{ij}(t_i) = \frac{1}{T_{ij}(t_i)}. \tag{18}$$

In conventional methods [13]-[15], paths connecting each pair of customers are calculated before the algorithm, and the calculation time of the algorithm is not considered. In the proposed method, we calculate these paths individually when they are required for the first time. The DA procedure is shown in Fig. 6, where s is the starting node, g is the destination node, t_s is the time at which a vehicle leaves node s, and n_{cl} is a parameter of the candidate lists[1]. These candidate lists constitute a small set of promising neighbors of the current partial solution and their use enables the ACO algorithms to focus on the more relevant components, thus significantly reducing the dimensions of the search space. A sort algorithm is normally required in order to use candidate lists, but since the DA already includes a sort function, the proposed method does not need any additional sort algorithm for making the candidate lists.

```
procedure Hybrid-ACO ( C_v, N_v )
   Input RWDP;
   Execute Initialize-Pheromone ( );    // see Fig. 3
   S_gb ← S_0^best;
   while ( terminal condition not met ) do
      for ( k = 1 to m ) do    // m is the number of ants
         N^k ← N^RW − N_v;
         for ( j = |N_v| to n ) do    // ants start from C_v
            Move ant k to a next customer;    // see Error! Reference source not found.
         end-for
         Move ant k to depot;
      end-for
      S_ib ← the best tour in current iteration;
      if ( T ( S_ib ) < T ( S_gb ) ) then
         S_gb ← S_ib;
      end-if
   end-while
end-procedure
```

Figure 5. General procedure of hybrid ACO.

```
procedure DA ( s, g, t_s, n_cl )
   Initialization for Dijkstra's algorithm;
   c = 0;
   while ( c ≤ n_cl ) do
      Execute Dijkstra's algorithm until an optimal path between customers is defined;
      c += 1;
   end-while
   Create candidate list of s at time t_s;
   while ( an optimal path between s and g is not defined ) do
      Execute Dijkstra's algorithm until the optimal path between s and g is defined;
   end-while
   Store the optimal paths from s to other nodes in the memory;
   Store the travel time from s to other nodes in the memory;
end-procedure
```

Figure 6. General procedure of path planning by DA.

4. EXPERIMENTS

4.1. Experiments with TSP Instances (TDTSP)

4.1.1 Experimental Methods

To evaluate how well the proposed method performs, we first conducted experiments using the TSP instances eil51, eil76, kroA100, u159, and d198 from the TSPLIB. The number of cities seems small in TSPLIB. However, considering correspondence with real-world problems, since the number of cities (or customers) that a salesman (or a vehicle) can visit in one day is at most 200, this number should be suitable as a scale for benchmark problems.

TDTSPs were generated by the method described in section 2.1. Parameters R_{jam} and B_{upper} were respectively set to 0.5 and 5 by reference to real-world traffic. Minutes and seconds are assumed as units of travel time for instances {eil51, eil76} and {kroA100, u159, d198}, respectively. So, the update interval of travel time Δt is 5 and 300 for instances {eil51, eil76} and {kroA100, u159, d198}, respectively.

We compared the minimal tour travel time obtained by the proposed method with those by the conventional methods below.

- Plain method: Search is conducted once before a vehicle starts using static traffic T_{ij} (0).
- Repeat method [9]-[11]: Search is repeated while a vehicle is moving using real-time traffic T_{ij} (t).
- Prediction method [13] , [21] : Search is conducted once before a vehicle starts using predicted traffic T'_{ij} (t, t_i).
- Prediction and repeat method: The proposed method.

We assumed two kinds of prediction errors, i.e., 20% and 50% in Fig. 7. The predicted traffic with error rate can be calculated as follows,

$$T'_{ij}(t,t_i) = T_{ij}(t_i) \times (1 + E(t_i - t) \times u), \tag{19}$$

where t is the current time, t_i is the departure time from C_i, E ($t_i - t$) is the error rate given by Fig. 7, and $u \in [-1, 1]$ is a uniform random number. This is based on experience in which short-term prediction has a smaller error rate than long-term prediction. In the prediction method, since the search is conducted only once before a vehicle starts, T'_{ij} (t, t_i) is calculated only once (t = 0). In contrast, since the proposed method repeatedly searches for the best tour while a vehicle is moving, T'_{ij} (t, t_i) is calculated every time real-time traffic is updated.

In this section, we assume that computational time is not limited and so we used original MMAS (r = 0) for the tour planning and chose 3000 iterations as the terminal condition. The values of parameters of the MMAS were as follows: α = 1.0, β = 5.0, ρ = 0.02, p_{best} = 0.05, and m is the number of customers not yet visited by a vehicle.

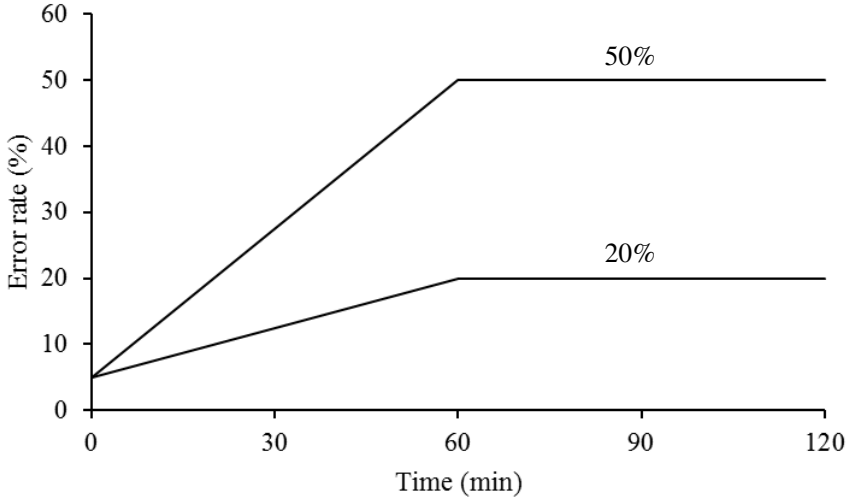

Figure 7. Relationship between prediction error and time.

4.1.2 Experimental Results

First, since the optimal solutions of TDTSPs are not known, we obtain them by the prediction method without error, which knows the exact travel time on all the links at all times. We assumed the best solution of 30 trials to be the optimal solution. Table 1 lists the number of cities, the optimal solution known in a static environment, the optimal solution in a dynamic environment, the update interval of travel time in a dynamic environment, and the number of updates for each instance. The number of updates ranged from 68 to 207—in other words, the traffic flow changed very frequently.

Figure 8 shows the computational results of the prediction method without error. The plots on the left show the development of the ratio of the tour travel time to the optimal tour travel time and the plots on the right show the development of the average λ-branching factor, which measures the distribution of the pheromone trail values[1]. Each plot in Fig. 8 is the average of 30 trails using different random number sequences. These results demonstrate that the terminal condition, 3000 iterations, is sufficient to achieve convergence on every instance.

Table 2 lists the ratio of the tour travel time of each method to the optimal tour travel time. Each value in Table 2 is the average of 30 trials using different random number sequences; the standard deviation was about 5 to 10%. Table 2 reveals three main findings.

Table 1. TSP instances used in experiments, optimal tour travel time, and updated travel time.

Instance	Optimal solution in TSP	Optimal solution in TDTSP	Update interval	Number of updates
eil51	426	574	5	114
eil76	538	730	5	146
kroA100	21282	29942	300	99
u159	42080	62355	300	207
d198	15780	20700	300	68

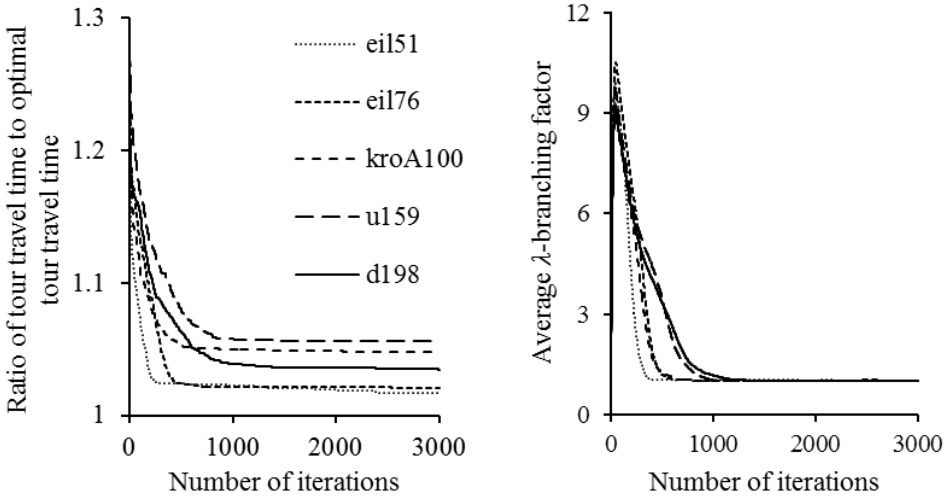

Figure 8. Computational results of prediction method without error: the ratio of tour travel time to optimal tour travel time (left) and the average λ-branching factor, $\lambda = 0.05$ (right).

Table.2. Experimental results for TDTSP. Each value indicates the ratio of the tour travel time of each method to the optimal tour travel time.

Instance	Plain	Repeat	Error rate 20%		Error rate 50%	
			Prediction	Prediction + Repeat	Prediction	Prediction + Repeat
eil51	1.50	1.31	1.05	1.02	1.41	1.12
eil76	1.39	1.36	1.29	1.06	1.44	1.10
kroA100	1.27	1.18	1.16	1.03	1.17	1.05
u159	1.26	1.27	1.29	1.03	1.39	1.08
d198	1.39	1.27	1.17	1.03	1.26	1.08
(Mean)	1.39	1.27	1.17	1.03	1.33	1.08

- Compared with the plain method, the performances of the other methods are improved.
- When the prediction error rate becomes large (50%), the prediction method is inferior to the repeat method.
- The proposed method is superior to the other methods, even when the error rate is large (50%).

4.2. Experiments with Real-World Problems (RWDP)

4.2.1 Experimental Methods

Next, to evaluate the proposed method in a real-world environment, we applied it to the RWDP described in section 2.2. Figure 9 shows a map of central Tokyo, which was the target area of this experiment. This map is 9 km long by 11 km wide and includes 10056 links and 3542 nodes and represents the most congested area in Japan. The positions of the depot and the customers

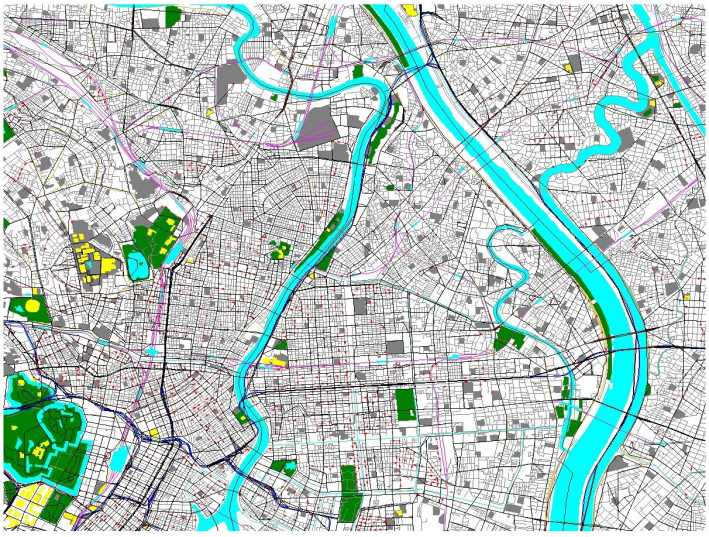

Figure 9. Road map of central Tokyo. Area is 9 km long by 11 km wide and there are 10,056 links and 3,542 nodes.

are randomly selected from the nodes. The travel time of a vehicle was calculated from historical traffic data on June 17, 2003. An example of the data is shown in Fig. 1. We also assumed the

prediction error rate in Fig. 7. We performed nine experiments. The numbers of customers are 50, 100 and 200. The departure times from the depot are 6:00 (morning), 12:00 (noon), and 18:00 (night). The instances are described as "the number of customers–the departure time from the depot," i.e., 50–6, 50–12 ... 200–18.

In a real-world scenario, the calculation time of the tour planning is limited. In Japan, traffic meters measure the average travel time of cars passing through specific road links at 1-minute intervals. When a state-of-the-art prediction system [22] is used in the proposed method, the calculation time of the prediction system is about 20 seconds. Therefore, we chose 10 seconds, 20 seconds, 40 seconds, and 3000 iterations (unlimited calculation time) as the terminal condition of MMAS. The other parameter values of MMAS were set the same as in section 4.1. The code was written in C++ and the experiments were executed on an Intel Core i7-870 (2.93 GHz).

4.2.2 Experimental Results

Figure 10 shows the best tours obtained by all experiments. The red and blue circles indicate a depot and customers, respectively. As shown, the tour depends on the time period. Crosses and returns appear in some parts of the tour in 2D, but in the 3D real-world, the tour is a complete circuit.

Table 3 lists the best tour travel time (minutes) and the tour travel time (minutes) of each method. The terminal condition of MMAS is 3000 iterations. Each value in Table 3 is the

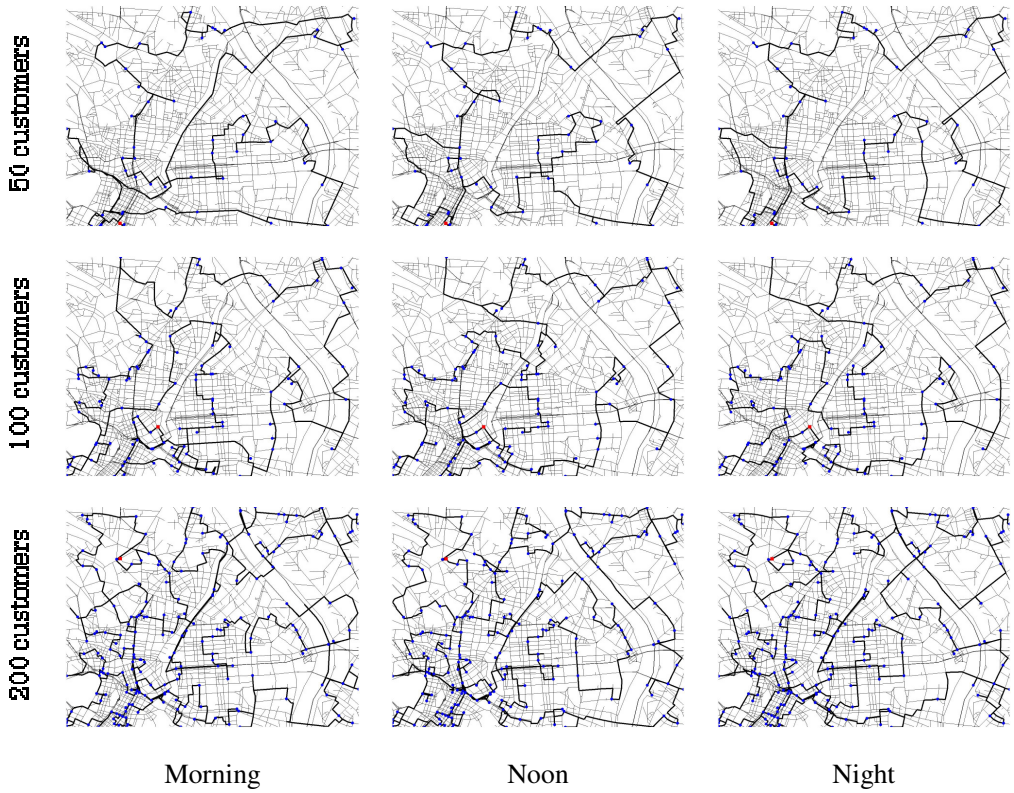

Figure10. Best tours in RWDP.

Table 3. Experimental results for RWDP. The terminal condition of MMAS is 3000 iterations. Each value is shown in minutes.

Instance	Best tour	Plain	Repeat	Error rate 20%		Error rate 50%	
				Prediction	Prediction + Repeat	Prediction	Prediction + Repeat
50–6	193	270	226	210	210	233	210
50–12	205	282	232	221	214	244	221
50–18	182	200	190	190	188	199	192
100–6	276	351	323	287	283	316	291
100–12	288	410	342	323	306	337	313
100–18	250	314	282	259	272	271	275
200–6	420	546	495	447	439	484	448
200–12	427	590	488	461	447	475	455
200–18	375	445	411	398	385	404	397

average of 30 trials using different random number sequences, and the standard deviation was about 2 to 5%. Table 3 indicates the same results as the experiments for TDTSPs can be obtained in a real-world environment.

When the calculation time of the tour planning is limited, the search rate of MMAS is quite important. Here, we compare the performance of the proposed method using original MMAS (r = 0) with the performance of the proposed method using improved MMAS (r = 0.9). Table 4 lists the tour travel time (minutes) of each method. Each value of each method in Table 4 is the average and the standard deviation of 30 trials using different random number sequences. Table 4reveals four main findings.

- The improved MMAS did not deteriorate compared to the original MMAS.
- When the problem size was small (50 customers), 10 seconds was enough to obtain the best tour.
- When the problem size was large (200 customers), 40 seconds was not sufficient to achieve convergence on the original MMAS.
- The larger the problem size was, the more superior the improved MMAS was compared to the original MMAS.

5. CONCLUSIONS

In this paper, we presented three techniques: a hybrid meta-heuristic that combines ACO with Dijkstra's algorithm, a search method that combines repeat and prediction, and a way to use a real-world road map and measured traffic data in Japan. The experimental results suggest that the proposed method is effective in a wide area road network. The results presented in this paper are based on five benchmark problems and a real-world problem. Further investigation using other maps and traffic from additional days is necessary. Although the proposed method is for delivery problems, the basic idea can be used for other combinatorial optimization problems in networks. In future work, we will combine the proposed method with traffic prediction systems.

Table 4. Experimental results for RWDP using the proposed method. The terminal condition of MMAS is 10 seconds, 20 seconds, and 40 seconds. Each value of each method is shown in minutes.

Instance	Terminal condition (sec)	Error rate 20%				Error rate 50%			
		$r=0$		$r=0.9$		$r=0$		$r=0.9$	
		Avg.	SD	Avg.	SD	Avg.	SD	Avg.	SD
50–6	10	210	8	210	8	210	9	210	8
	20	210	8	210	8	210	9	210	8
	40	210	8	210	8	210	9	210	8
50–12	10	214	7	214	6	221	7	221	6
	20	214	7	214	6	221	7	221	6
	40	214	7	214	6	221	7	221	6
50–18	10	188	4	188	5	192	5	192	5
	20	188	4	188	5	192	5	192	5
	40	188	4	188	5	192	5	192	5
100–6	10	296	8	292	11	321	13	300	17
	20	288	12	285	11	304	24	299	12
	40	283	13	283	11	291	14	291	11
100–12	10	316	13	316	11	317	12	317	12
	20	311	10	310	7	314	11	313	9
	40	306	7	306	7	313	9	313	9
100–18	10	295	14	274	11	299	12	277	8
	20	275	12	272	9	279	11	276	11
	40	272	10	272	9	275	11	275	10
200–6	10	463	14	451	14	479	30	462	20
	20	445	14	442	11	452	14	449	14
	40	441	15	439	10	449	14	448	12
200–12	10	471	14	469	14	477	14	477	14
	20	459	11	448	12	465	11	460	12
	40	454	16	447	12	456	12	455	11
200–18	10	402	10	393	10	415	10	407	14
	20	396	10	389	8	410	11	399	8
	40	388	11	385	7	401	10	397	8

ACKNOWLEDGEMENTS

The authors wish to thank Mr. Yosuke Kameda for his considerable advice in carrying out this research. This research was partly supported by a Grant-in Aid for Scientific Research (C) of the Japan Society for the Promotion of Science (23500169).

REFERENCES

[1] M. Dorigo and T. Stützle, (2004) Ant Colony Optimization, MIT Press.

[2] M. Dorigo, M. Birattari, and T. Stützle, (2006) "Ant Colony Optimization—Artificial Ants as a Computational Intelligence Technique," IEEE Computational Intelligence Magazine, vol. 1, no. 4, pp. 28–39.

[3] M. Dorigo and T. Stützle, (2010) "Ant Colony Optimization: Overview and Recent Advances," Handbook of Metaheuristics, International Series in Operations Research & Management Science, Springer-Verlag, vol. 146, pp. 227–263.

[4] G. Reinelt, (1994) The Traveling Salesman Problem: Computational Solution for TSP Applications, LNCS, Springer-Verlag, vol. 840.

[5] M. Albayrak and N. Allahverdi, (2011) "Development a new mutation operator to solve the Traveling Salesman Problem by aid of Genetic Algorithms," Expert Systems with Applications, vol. 38, no. 3, pp. 1313–1320.

[6] X. Genga, Z. Chen, W. Yang, D. Shi, and K. Zhao, (2011) "Solving the traveling salesman problem based on an adaptive simulated annealing algorithm with greedy search," Applied Soft Computing, vol. 11, no. 4, pp. 3680–3689.

[7] G. Berbeglia, J. Cordeau, and G. Laporte, (2010) "Dynamic pickup and delivery problems," European Journal of Operational Research, vol. 202, no. 1, pp. 8–15.

[8] V. Pillac, M. Gendreau, C. Gueret, and A. L. Medaglia, (2013) "A review of dynamic vehicle routing problems," European Journal of Operational Research, vol. 225, no. 1, pp. 1–11.
[9] M. Guntsch and M. Middendorf, (2001) "Pheromone modification strategies for ant algorithms applied to dynamic TSP," in Proc. EvoWorkshops 2001, LNCS, Springer-Verlag, vol. 2037, pp. 213–222.
[10] C. J. Eyckelhof and M. Snoek, (2002) "Ant system for a dynamic TSP: ants caught in a traffic jam," in Proc. 3rd International Workshop on Ant Algorithms (ANTS 2002), LNCS, Springer-Verlag, vol. 2463, pp. 88–99.
[11] M. Mavrovouniotis and S. Yang, (2013) "Ant colony optimization with immigrants schemes for the dynamic traveling salesman problem with traffic factors," Applied Soft Computing, vol. 13, no. 10, pp. 4023–4037.
[12] J. Ochiai and H. Kanoh, (2014) "Solving Time-dependent Traveling Salesman Problems Using Ant Colony Optimization Based on Predicted Traffic and Its Application to Wide Area Road Network," IPSJ Transactions on Mathematical Modeling and Its Applications (in Japanese), vol. 7, no. 1, pp. 94–105.
[13] A. V. Donati, R. Montemanni, N. Casagrande, E. A. Rizzoli, and L. M. Gambardella, (2008) "Time dependent vehicle routing problem with a multi ant colony system," European Journal of Operational Research, vol. 185, no. 3, pp. 1174–1191.
[14] H. Kanoh and S. Tsukahara, (2010) "Solving Real-World Vehicle Routing Problems with Time Windows using Virus Evolution Strategy," International Journal of Knowledge-Based and Intelligent Engineering Systems, vol. 14, no. 3, pp. 115–126.
[15] A. Haghani and S. Jung, (2005) "A dynamic vehicle routing problem with time-dependent travel times," Computers & Operations Research, vol. 32, no. 11, pp. 2959–2986.
[16] J. Ochiai and H. Kanoh, (2014) "Hybrid Ant Colony Optimization for Real-World Delivery Problems Based on Real Time and Predicted Traffic in Wide Area Road Network," in Proc. 2nd International Conference on Artificial Intelligence, Soft Computing (AISC 2014), pp. 379–389.
[17] L. Gouveia and S. Vob, (1995) "A classification of formulations for the (time-dependent) traveling salesman problem," European Journal of Operational Research, vol. 83, no. 1, pp. 69–82.
[18] T. Stützle and H. H. Hoos, (2000) "MAX-MIN Ant System," Future Generation Computer System, vol. 16, no. 8, pp. 889–914.
[19] H. Kanoh, T. Furukawa, S. Tsukahara, K. Hara, H. Nishi, and H. Kurokawa, (2005) "Short-Term Traffic Prediction Using Fuzzy C-Means and Cellular Automata in a Wide-Area Road Network," in Proc. IEEE International Conference on Intelligent Transportation Systems (ITSC 2005), pp. 984–988.
[20] D. Ichiba, K. Hara, and H. Kanoh, (2006) "Spatial Interpolation of Traffic Data by Genetic Fuzzy System," in Proc. IEEE 2nd International Symposium on Evolving Fuzzy Systems (EFS 2006), pp. 280–285.
[21] H. Kanoh and J. Ochiai, (2012) "Solving Time-Dependent Traveling Salesman Problems using Ant Colony Optimization Based on Predicted Traffic," in Proc. International Symposium on Distributed Computing and Artificial Intelligence (DCAI 2012), Advances in Intelligent and Soft Computing, vol. 151, pp. 25–32.
[22] W. Min and L. Wynter, (2011) "Real-time road traffic prediction with spatio-temporal correlations," Transportation Research Part C, vol. 19, no. 4, pp. 305–320.

Authors

Junich Ochiai received M.E. degree and Ph.D. degree in computer science from University of Tsukuba, Japan, in 2011 and 2014, respectively. He currently is a postdoctoral researcher with the Center for Service Research, National Institute of Advanced Industrial Science and Technology. His research interests include ant colony optimization and evolutionary computation.

Dr. Ochiai is a member of the Information Processing Society of Japan. He received the best presentation award from the Mathematical Modeling and Problem Solving Group, Information Processing Society of Japan in 2012.

Hitoshi Kanoh received M.S. degree in physics and Ph.D. degree in computer science from University of Tsukuba, Japan, in 1980 and 1992, respectively. In 1980, he joined Hitachi Cable, Ltd. where he has been engaged in Research of expert systems, fuzzy systems and neural networks as a section chief. In 1993 he moved to the University of Tsukuba. He currently is a professor with the Division of Information Engineering, Faculty of Engineering, Information and Systems, University of Tsukuba. His main research interests include evolutionary computation, swarm intelligence, and soft computing with special application to intelligent transportation systems.

Dr. Kanoh is a member of the IEEE, the Japanese Society for Artificial Intelligence, and Information Processing Society of Japan. He received the best paper award from the Institute of electrical Engineers of Japan in 1992, the best paper and presentation award from the World Conference on Soft Computing in Industrial Applications in 1999, and the best paper and presentation award from the Intelligent Transport Systems Group, Information Processing Society of Japan in 2003 and 2006.