IEICE Proceeding Series

AD-Truck Routing Problem and Solution Methods

Takafumi MATSUURA, Yuri HARUNAGA, Kazumiti NUMATA

Vol. 1 pp. 535-538
Publication Date: 2014/03/17
Online ISSN: 2188-5079

2012 International Symposium on Nonlinear Theory and its Applications
NOLTA2012, Palma, Majorca, Spain, October 22-26, 2012

AD-Truck Routing Problem and Solution Methods

Takafumi MATSUURA, Yuri HARUNAGA, and Kazumiti NUMATA

Department of Management Science, Faculty of Engineering, Tokyo University of Science
1-3 Kagurazaka, Shinjuku-ku, Tokyo 162-8601, Japan
Email: matsuura@ms.kagu.tus.ac.jp, p7.yuyu8280@gmail.com, numata@ms.kagu.tus.ac.jp

Abstract—On town streets, we see a truck which displays advertisement of new products or local events on its side. The truck is moving media of advertisement to appeal to walkers on streets and called "AD-truck". The advertisement effects of the AD-track are generated at the intersections. Under the traffic rules: cars keep to the left and going straight is given priority over turning right, turning right takes a longer time than going straight at the intersections. Thus, the AD-trucks are recommended to turn right at the intersections as possible as many times for greater advertisement effect. In this paper, to determine an optimum tour of the AD-trucks, we define a new discrete optimization problem called "AD-Truck Routing Problem." We formulate it as integer programming, and propose a local search method and a chaotic search method which drives the local search method by using chaotic dynamics.

1. Introduction

Advertisement is one of inevitable activities in modern business. It needs some medium to send promoting messages to targeted people. An advertising truck (AD-truck) which runs around town streets with displaying a new product and local event is one of such media. Some type of AD-truck makes a round trip at the predetermined street, and the other type runs over the whole streets network. In this paper, we focus on the latter case.

The AD-truck starts from an base point (depot), runs around a downtown, and returns to the depot. The aim of the AD-truck is to promote the new products or local events displaying on its side for walkers on streets. When the AD-truck runs around the downtown, advertisement effects of the AD-truck are generated at the intersections, and influenced by the magnitude of crowds and elapsed times at the intersections. Under the traffic rules: cars keep to the left and going straight is given priority over turning right, it takes much longer time to turn right than to go straight at the intersection. By staying a long time in the intersections, the AD-trucks try to attract attention to the advertisements on its side. Hence, the AD-trucks are recommended to turn right at the crowded intersections as possible as many times for greater advertisement effects. In this paper, to determine an optimal tour of the AD-truck, we propose a new discrete optimization problem which is called "AD-Truck Routing Problem (ADT-RP)." In this paper, we first present the ADT-RP as a 0-1 linear programming problem, then, we propose two solution methods for it. The first one is a local search method for the ADT-RP. However, in general, the local search methods cannot obtain a global optimal solution due to the local minimum problem. To resolve the local minimum problem, we have already proposed effective heuristic search methods by using chaotic dynamics [1–9]. Thus, we propose a chaotic search method for solving the ADT-RP. In the method, the proposed local search method is driven by the chaotic dynamics. As the results, the method finds good feasible solutions for wide range of instances very quickly.

2. AD-Truck Routing Problem

The ADT-RP is assumed that an AD-truck runs over a grid streets network $\mathcal{N}$ consisted of $m \times n$ intersections (Fig.1(a)). A depot (base point of AD-truck) is located at (0th row, sth column). Each intersection, except for boundaries, has 4 entry points (from left, top, right, and bottom), 4 exit points (to left, top, right and bottom) and 12 edges from each entry point to 3 corresponding exit points (Fig.1(b)). The network also includes $2m(n-1)$ horizontal and $2n(m-1)$ vertical street segments which link adjacent intersections from an exit point to an entry point. Furthermore, 2 vertical edges from/to the depot to/from the intersection $(0, s)$ are added. An AD-truck starts from the depot, runs over the network and returns to the depot within given time limit T, where the required time for passing through each of the edges is positive and known. The "U-turn" is prohibited. We also assume that the AD-truck can enter the same crossing at most four times from different directions.

The AD-truck obtains the advertisement effects only at the intersections. The values of effects are decided by average magnitude of crowds and average waiting time in the intersections. The values of effects are known and given

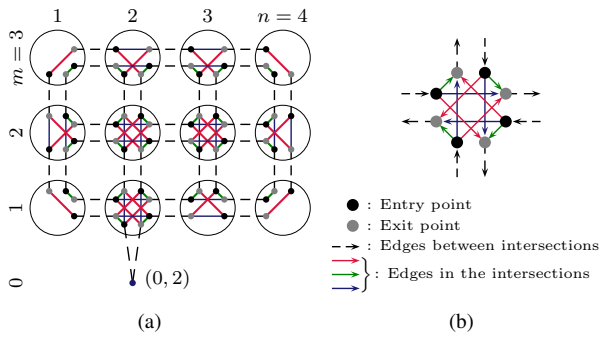


Figure 1: (a) An example of the network $\mathcal{N}$ whose size is $(m, n) = (3, 4)$. (b) An intersection in the network $\mathcal{N}$.

for all edges in the intersections. The values of edges corresponding to turning right are the largest in all edges. Second is turning left and the least effects are going straight. The values of effects are set to 0 for all edges between intersections. The aim of the ADT-RP is to determine the tour of AD-truck so that it generates the maximum expected advertisement effect under the condition that time length of the tour does not exceed the time limit T.

3. Formulation of the ADT-RP

The grid network $\mathcal{N}$ is given by (V, E, g, c). $V = \{0\} \cup V_1 \cup V_2$ is the set of all points, where "0" denotes the depot, V_1 the set of all entry points, and V_2 the set of all exit points in the intersections. $E = E_0 \cup E_1 \cup E_2$ is the set of all edges, where E_0 is the set of 2 edges between "0" (depot) and the intersection $(1, s)$, E_1 the set of all edges in the intersections, and E_2 the set of all edges between intersections. g_e represents the expected effect of advertisement on edge e ($e \in E_1$) and c_e is the required time to pass through the edge e ($e \in E$). Let y_v denote whether the AD-truck visits point v ($y_v = 1$) or not ($y_v = 0$), and x_e denote whether it moves on edge e ($x_e = 1$) or not ($x_e = 0$). Decision variables y_v and x_e determine a tour. Let f_e denote the counting value which AD-truck holds when it moves on edge e. If it does not move on e, f_e is set to 0. The value of f_e is counted up by 1 whenever the AD-truck visits a point. Decision variables f_e are used to eliminate sub-tours. Using these notations, the ADT-RP is formulated as follows:

(ADT-RP)

$$
\begin{cases}
\text{max.} & \displaystyle\sum_{e \in E_1} g_e x_e & & (1) \\
\text{s.t.} & \displaystyle\sum_{e \in E} c_e x_e \le T & & (2) \\
& x_{\epsilon^-(v)} = y_v, & (\forall v \in V_1) & (3) \\
& \displaystyle\sum_{e \in \delta^+(v)} x_e = y_v, & (\forall v \in V_1) & (4) \\
& x_{\epsilon^+(v)} = y_v, & (\forall v \in V_2) & (5) \\
& \displaystyle\sum_{e \in \delta^-(v)} x_e = y_v, & (\forall v \in V_2) & (6) \\
& f_e \le Q x_e, & (\forall e \in E) & (7) \\
& \displaystyle\sum_{e \in \delta^+(v)} f_e - f_{\epsilon^-(v)} = y_v, & (\forall v \in V_1) & (8) \\
& \displaystyle\sum_{j \in V, j \ne 1} f_{1j} = 0 & & (9) \\
& f_{\epsilon^+(v)} - \displaystyle\sum_{e \in \delta^-(v)} f_e = y_v, & (\forall v \in V_2) & (10) \\
& y_0 = 1 & & (11) \\
& x_{\epsilon^+(0)} = 1 & & (12) \\
& x_{\epsilon^-(0)} = 1 & & (13) \\
& y_v \in \{0, 1\}, & (\forall v \in V) & (14) \\
& x_e \in \{0, 1\}, & (\forall e \in \{0\}E) & (15) \\
& f_e \ge 0 : \text{integer}, & (\forall e \in E) & (16)
\end{cases}
$$

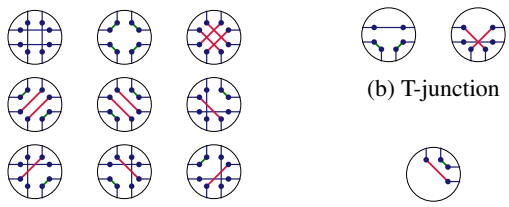
where, $\epsilon^-(v)$ and $\epsilon^+(v)$ represent the unique edge entering to v ($v \in \{0\} \cup V_1$) and the unique edge exiting from v ($v \in \{0\} \cup V_2$), respectively. $\delta^+(v)$ represents the set of all edges going from an entry point v ($\in V_1$) to 3 exit points ($\in V_2$) in the same intersection. $\delta^-(v)$ represents the set of all edges going to an exit point v ($\in V_2$) from 3 entry points ($\in V_1$) in the same intersection.

4. Proposed Methods

4.1. Local Search Method

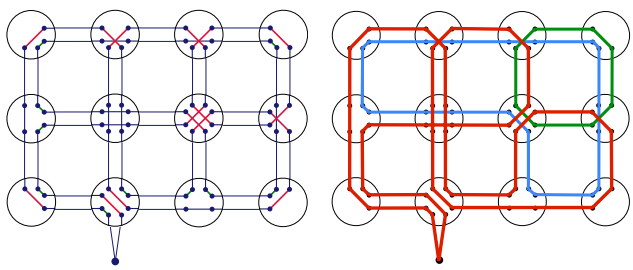
The proposed method solves the ADT-RP by using a new network $\mathcal{N}'$. In the network, traffic patterns of the intersections are restricted. The traffic patterns are constructed to satisfy the following conditions: each entry point is connected only one exit point and each exit point is connected only one enter point. Then, nine and two traffic patterns are constructed for the cross road and the T-junction, respectively. Figure 2 shows the traffic patterns of the cross road and the T-junctions.

By assigning a restricted traffic pattern to each intersection, a network $\mathcal{N}'$ is constructed (Fig.3(a)). The network gives one or more disjoint closed paths (sub-tours) (Fig.3(b)), then, a depot is included in only one sub-tour. Namely, one feasible tour and the advertisement effect of the tour can be obtained from the network, if the tour satisfies time limit T. If a traffic pattern at an intersection in the current network is changed, a new network is constructed. As the result, different tour and the advertisement effect are obtained form the new network. In the proposed local search method, one traffic pattern at a intersection is changed to improve the current advertisement effect, until no furthermore improvement can be obtained.



(a) Cross road (b) T-junction (c) Corner

Figure 2: Traffic patterns of intersections in the network $\mathcal{N}'$.



(a) (b)

Figure 3: An example of (a) the network $\mathcal{N}'$ and (b) three sub-tours in the network $\mathcal{N}'$. The red sub-tour shows the feasible solution obtained from the network.

4.2. Chaotic Search Method for the ADT-RP

In general, the local search methods cannot find an optimal solution due to the local minimum problem. To resolve the problem, we have already proposed effective heuristic methods by using chaotic dynamics [1–9]. The methods show good results for different type of combinatorial optimization problems such as traveling salesman problems [1–4], quadratic assignment problems [5], and motif extraction problems [6–9]. Thus, we propose a new chaotic search method for solving the ADT-RP.

In the chaotic search methods [1–9], a chaotic dynamics is used to avoid local minima. To realize the chaotic dynamics, a chaotic neural network model constructed by chaotic neurons [10] is used. To realize a chaotic search method for the ADT-RP, the chaotic neurons are assigned to each traffic pattern in all intersections. For the cross roads, nine chaotic neurons are assigned because the number of the traffic patterns at the cross road is nine (Fig.3).

If the ijkth chaotic neuron fires, the kth traffic pattern becomes a new traffic pattern at the intersection (i, j). The firing of the ijkth neuron $x_{ijk}(t)$ is defined by

$$x_{ijk}(t) = f(y_{ijk}(t)), \tag{17}$$

where $f(y) = 1/(1 + \exp(-y/\epsilon))$. $y_{ijk}(t)$ is the internal state of the ijkth chaotic neuron at time t. If $x_{ijk}(t) > \frac{1}{2}$, the ijkth neuron fires at the time t, otherwise, the neuron is resting. $y_{ijk}(t)$ is decomposed into two parts, $\zeta_{ijk}(t)$ and $\xi_{ijk}(t)$. Each component represents different factor to the dynamics of chaotic neurons, a gain effect and a refractory effect, respectively.

The gain effect is expressed as:

$$\xi_{ijk}(t+1) = \begin{cases} G_{ijk}(t) - \hat{G} & (T_{ijk} \leq T), \\ -G_{ijk}(t) & (\text{otherwise}) \end{cases} \tag{18}$$

where $\hat{G}$ is the expected advertisement effect of a current tour; $G_{ijk}(t)$ is the expected advertisement effect of a new tour, when the kth traffic pattern is assigned to the intersection (i, j); $T_{ijk}(t)$ is the tour time of the new tour; and $\beta(t + 1)$ is a scaling parameter. The scaling parameter increase in proportion to time t: $\beta(t + 1) = \beta(t) + \lambda$. If we use these functions, the searching space is gradually limited as the simulated annealing [11]. If the $\beta(t)$ takes a small value, the proposed method can explore a large solution space. On the other hand, if $\beta(t)$ takes a large value, the proposed method works like a greedy algorithm.

The second factor is a refractory effect which works to avoid the local minima. In the chaotic search, past firings are memorized as previous states to decide strength of the refractory effect. The strength of the refractory effect increases just after corresponding neuron firings and recovers exponentially with time. The chaotic search might permit to select the same solutions if a corresponding neuron fires due to a larger gain than the refractory effect or an exponential decay of the refractory effect. The refractory effect is expressed as:

$$\begin{aligned} \zeta_{ijk}(t+1) &= -\alpha \sum_{d=0}^{t} k_r^d x_{ijk}(t-d) + \theta & (19) \\ &= k_r \zeta_{ijk}(t) - \alpha x_{ijk}(t) + \theta(1 - k_r), & (20) \end{aligned}$$

where α controls the strength of the refractory effect after the firing ($\alpha > 0$); k_r is a decay parameter of the refractory effect ($0 < k_r < 1$); θ is a threshold value.

The algorithm of the proposed method can be described as follows:

1. Given a data set (V, E, g, t): V is the set of all points; E is the set of all edges; g_e represents the expected advertisement effect on edge e ($e \in E_1$), and c_e the required time for passing through the edge e ($e \in E$). The time limit is T.
2. To make a network $\mathcal{N}'$, traffic patterns are randomly assigned to each intersection, then, an initial solution (tour) and its advertisement effect are obtained from the network.
3. To change the traffic pattern at intersections, a intersection (i, j) is selected randomly.
   (a) The kth neuron is randomly selected from the intersection (i, j). If the ijkth neuron fires ($x_{ijk} \geq 0.5$), the corresponding traffic pattern becomes a new pattern at the intersection (i, j), and the value of $\hat{G}$ is updated.
   (b) Repeat step 3(a) for all neurons of the intersection (i, j).
4. Repeat step 3 for all intersections.
5. The procedure of a single iteration in the proposed method is finished. Repeat steps 3-4 until the number of iterations is satisfied.

5. Simulations and Results

To investigate solving performances of the proposed methods, first, we obtained an optimization solution of many instances by using a general purpose mixed integer program solver. In the simulation, we used a Gurobi 5.0.0 solver [12] on iMac (2.8 GHz Intel Core i7) with 6GB memory running Mac OS X 10.7.4. All simulations are carried out a single thread. The size of instances is set to $m = 5$ and $n = 5, 6, 7, 8, 9$. The effects of advertisement and the required time for passing through the edges are set to uniform random numbers. The maximum and minimum values of the uniform random numbers are shown in Table 1. The time limit T is set to ∞.

Table 1: The minimum and maximum values of the transit time c_e and the advertisement effects g_e.

		segments	left	straight	right
c_e	min	5	1	5	10
	max	20	10	15	20
g_e	min	0	25	5	50
	max	0	150	10	500

Figure 4 shows the calculation time of the Gurobi until optimal solutions are found. From the results in Fig. 4, the instances whose size ($m \times n$) is less than 45 can be solved by general purpose MIP solver through (ADT-RP) formulation in section 3. However, optimal solutions cannot be obtained in a reasonable time frame for large size of instances ($n \geq 10$). These results indicate that it is difficult to solve by general purpose MIP solver for large size of instances because the calculation time increases exponentially with the size of network.

To evaluate the performances of the chaotic search method, we solved same instances. In this simulation, we set same values of parameters in Eqs.(18) and (19) for all instance: $\beta(0) = 0.0$, $\lambda = 0.000001$, and $\epsilon = 0.01$. The α and k_r are set to various values. In this experiment, we have applied $5,000$ iteration for each calculation.

The results in Table 2 are shown by percentage of gaps between obtained solutions by the proposed local search method and the optimal solutions. From Table 2, the average gaps of the local search method are about 10.0% for all instances. In addition, in 30 trials, the local search method cannot find optimal solutions.

Figure 5 shows the average gaps between the obtained best solutions of the chaotic search method and the optimal solutions. Table 3 shows the CPU-time of the chaotic search method. From these results, the chaotic search method finds good solutions in less CPU-time, when we appropriately set to the values of parameters. These results indicate that to search solution space effectively, it is necessary to control the strength of the refractory effect. However, the good parameters sets are relatively large (Fig.5).

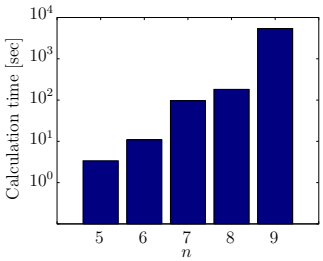


Figure 4: Calculation time of the Gurobi [12].

Table 2: Performances of the local search method. The average gaps (%) between obtained solutions and the optimal solutions (%) in 30 trials are shown.

n	5	6	7	8	9
Average gaps	10.02	9.37	10.09	10.38	10.15

Table 3: CPU-time of the chaotic search method [sec].

n	5	6	7	8	9
CPU-time	26.72	47.55	81.11	118.92	168.48

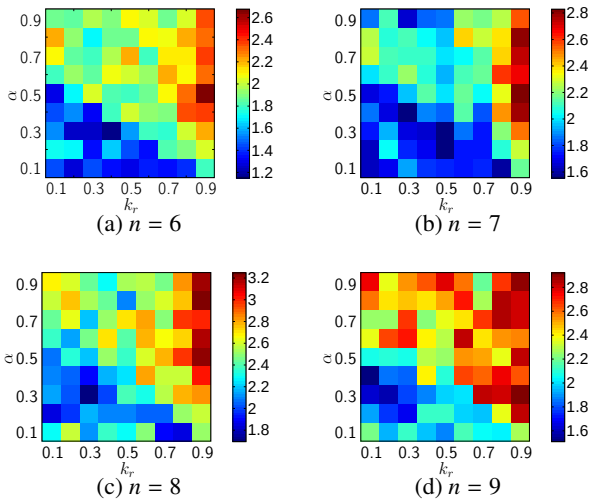


Figure 5: Performances of the chaotic search method. The average gaps (%) between obtained solutions and the optimal solutions in 30 trials are indicated by shading.

6. Conclusions

In this paper, we defined a new discrete optimization problem called "Ad-Truck Routing Problem (ADT-RP)" to decide effective tours of the AD-truck. Then, we proposed a local search method and a chaotic search method for solving the ADT-RP. From the computational results, we confirmed that although the proposed local search method is simple, the chaotic search method which drives the proposed local search method by the chaotic dynamics obtains good solutions in less CPU-time. Although we have to set optimal parameters to find the good solutions by the chaotic search method, the good parameters sets are relatively large.

References

[1] Hasegawa, M. et al. (1997): *PRL*, **79** (12), 2344–2347.
[2] Hasegawa, M. et al. (2002): *Neural Networks*, **15** (2), 271–283.
[3] Matsuura, T. and Ikeguchi, T. (2008): *Proc. of NOLTA 2008*, 77–80.
[4] Matsuura, T. and Ikeguchi, T. (2008): *LNCS*, **5164** 587–596.
[5] Hasegawa, M. et al. (2002): *EJOR*, **39** (3), 543–556.
[6] Matsuura, T. et al. (2005): *Proc. of MIC 2005*, 677–682.
[7] Matsuura, T. and Ikeguchi, T. (2006): *LNCS*, **4224**, 1103–1110.
[8] Matsuura, T. and Ikeguchi, T. (2009): *Proc. of NOLTA 2009*, 368–371.
[9] Matsuura, T. and Ikeguchi, T. (2010): *NOLTA*, **1**(1), 207–220.
[10] Aihara, K. et al. (1990): *Phys. Lett. A*, **144**, 333–340.
[11] Kirkpatrick, S. et al. (1983): *Science*, **220**, 671–680.
[12] Gurobi Optimization, http://www.gurobi.com/