Department of Industrial Engineering and Management

Technical Report

No. 2013-8

MISOCP formulation and route generation algorithm for ship navigation problem

Mirai Tanaka and Kazuhiro Kobayashi

June, 2013

Tokyo Institute of Technology

2-12-1 Ookayama, Meguro-ku, Tokyo 152-8552, JAPAN
http://www.me.titech.ac.jp/index-e.html

MISOCP FORMULATION AND ROUTE GENERATION ALGORITHM FOR SHIP NAVIGATION PROBLEM

MIRAI TANAKA AND KAZUHIRO KOBAYASHI

Abstract. We consider a problem to find a shipping route between two ports and shipping speed on each leg which minimize the total fuel consumption. Since this problem can be formulated as a mixed-integer second-order cone optimization problem (MISOCP), we can apply general-purpose solvers to this problem. However, it is difficult to solve large-scale problems with them. We propose an efficient algorithm for solving this problem named the route generation algorithm. The basic idea of our algorithm is to generate shipping routes and to optimize shipping speed for each shipping route. Since we employ a relaxation technique, we need not to enumerate all shipping routes. The numerical results provide the effectiveness of our algorithm.

1. Introduction

We consider a problem to find a shipping route from the origin port to the destination port and shipping speed on each leg in the route which minimize the total fuel consumption and let the ship reach at the destination by the designated time. We call this problem *ship navigation problem*. In our model, we incorporate the effect of the weather conditions in a deterministic way. That is, we assume that the available information correctly depicts the actual weather conditions. If the difference between the information and the actual condition is small, this would provide a reasonable solution.

Recently, the reduction of the fuel emission has grown in importance since the price of fuel has increased (see the latest survey [5]). The fuel emissions depend on a choice of shipping route and shipping speed. Thus, it is important to optimize them to reduce the fuel emissions in the navigation of ships.

Some researchers have considered problems to minimize the cost of operating a fleet of ships. Perakis and Papadakis [8, 9] introduced a model to select shipping speeds of ships in a fleet which carry cargoes between two ports to minimize the annual fleet operating cost.

Some researchers have considered problems to optimize shipping speed of a single ship in visiting multiple ports. Fagerholt *et al.* [6] introduced a model to minimize fuel consumption of a single ship which visits multiple ports in a voyage. The decision variable is shipping speed between each pair of ports in the voyage. In this model, the sequence of ports to visit and the shipping route between two ports are fixed. Ronen [10] considered a problem of a single ship to maximize the revenue, in which the decision variable is shipping speed on a leg. They considered a trade-off between fuel savings through the reduction of the shipping speed and the loss of revenues due to the resulting voyage extension. In the model, the shipping route is fixed.

Problems to optimize the shipping speed between two ports also have been investigated. Lo and McCord [7] considered the effect of the current and treated the problem as an adaptive optimization problem under uncertainty. In their model, the decision variables are the heading of the ship and its log speed (the speed through the water) and the objective is to minimize the expected value of the fuel consumption. Moreover, the ship is required to depart from a origin and arrive at a destination at a prescribed time. Azaron and Kianfar [2] treated the problem as a dynamic shortest path problem in stochastic dynamic networks. In the model, the decision variable is shipping route. They assume that we know the environmental states upon arriving at each node.

In this article, a problem to optimize both of a shipping route and log speed on each leg in a route to move from a port to another port is studied. Such a problem has not been considered by the previous researchers. Since we take into account of a short time span, we can treat the weather condition as deterministic.

We model the ship navigation problem as an optimization problem on a network. We identify a set of geographical locations on the ocean on which the ship may pass through. For each of the locations, we define a node. Let N be the set of such nodes. Especially, node $s \in N$ and $t \in N$ represent the origin and the destination,

Key words and phrases. shipping routes, shipping speed, fuel emissions, mixed-integer second-order cone optimization (MISOCP), constrained shortest path problem.

respectively. When the ship could move directly from node i to node j, we define arc (i, j). Let A be the set of such arcs. Thus, a shipping route is represented as an s–t path in network (N, A).

Each arc $(i, j) \in A$ is associated with three values d_{ij}, v_{ij}, and r_{ij}, where d_{ij} is the distance from node i to node j, v_{ij} is the log speed on arc (i, j), and r_{ij} is the speed reduction on arc (i, j) (the details are described later). The log speed must lie between the minimum value $v_{\min}$ and the maximum value $v_{\max}$.

In our model, we assume that the fuel consumption per unit distance is given as the cubic function $\alpha v^3 + \beta v^2 + \gamma v$ of log speed v for the interval $[v_{\min}, v_{\max}]$. This model has been given by Fagerholt *et al.* [6]. They gave the estimated values of $\alpha = 0.0036$, $\beta = -0.1015$, and $\gamma = 0.8848$ with $v_{\min} = 14$, and $v_{\max} = 20$.

We take into account the effect of the weather condition. More specifically, the transition time from each node to an adjacent node is assumed to be dependent on the weather conditions of the locations of the nodes. We assume that the weather condition on arc (i, j) is represented as the speed reduction of the ship, which is denoted by constant r_{ij}. In other word, when the ship passes through arc (i, j) with log speed v_{ij}, the ground speed (the speed over the ground) is given by $v_{ij} - r_{ij}$. In what follows, we assume that $r_{ij} < v_{\min}$. Under this assumption, the transition time on arc (i, j) is given by travelling distance d_{ij} divided by ground speed $v_{ij} - r_{ij}$. Thus, the fuel consumption on arc (i, j) is given by $(\alpha v_{ij}^3 + \beta v_{ij}^2 + \gamma v_{ij}) d_{ij} / (v_{ij} - r_{ij})$. Then, the ship navigation problem can be formulated as the following mixed-integer nonlinear optimization problem (MINLP):

$$\text{(1)} \qquad \text{minimize} \quad \sum_{(i,j)\in A} \frac{\alpha v_{ij}^3 + \beta v_{ij}^2 + \gamma v_{ij}}{v_{ij} - r_{ij}} d_{ij}$$

$$\text{(2)} \qquad \text{subject to} \quad \sum_{j:(s,j)\in A} x_{sj} = 1,$$

$$\text{(3)} \qquad \sum_{j:(i,j)\in A} x_{ij} - \sum_{j:(j,i)\in A} x_{ji} = 0 \quad (i \in N \setminus \{s, t\}),$$

$$\text{(4)} \qquad \sum_{i:(i,t)\in A} x_{it} = 1,$$

$$\text{(5)} \qquad x_{ij} \in \{0, 1\} \quad ((i, j) \in A),$$

$$\text{(6)} \qquad \sum_{(i,j)\in A} \frac{d_{ij} x_{ij}}{v_{ij} - r_{ij} x_{ij}} \leq T,$$

$$\text{(7)} \qquad v_{\min} x_{ij} \leq v_{ij} \leq v_{\max} x_{ij} \quad ((i, j) \in A),$$

where $x_{ij} = 1$ if the ship passes through arc (i, j) and $x_{ij} = 0$ otherwise, and v_{ij} represents the log speed if arc (i, j) is used and $v_{ij} = 0$ otherwise. Note that if $x_{ij} = 0$ then the corresponding summands in objective function (1) and those in constraint (6) vanish. Thus, objective function (1) and the left-hand side of constraint (6) represent the total fuel consumption and the total transition time in the route, respectively.

In this paper, we propose an algorithm to solve the ship navigation problem quickly. The remainder of this paper is organized as follows: In Section 2, we reformulate the ship navigation problem as a mixed-integer second-order cone optimization problem by approximating objective function (1) by a convex quadratic function. In Section 3, we propose an efficient algorithm called the route generation algorithm. The basic idea of our algorithm is to generate a shipping route and to optimize log speed on each leg in the route iteratively. In Section 4, we report the result of the numerical experiments and verify the effectiveness of our algorithm. Section 5 is devoted to the concluding remark.

2. Formulation as a mixed-integer second-order cone optimization problem

The ship navigation problem can be described as MINLP (1)–(7). By approximating objective function (1) by a quadratic function, we can reformulate MINLP (1)–(7) as a mixed-integer second-order cone optimization problem (MISOCP).

Each term in objective function (1) can be approximated by a quadratic function as the followings: It is represented as

$$\text{(8)} \qquad \frac{\alpha v_{ij}^3 + \beta v_{ij}^2 + \gamma v_{ij}}{v_{ij} - r_{ij}} = \alpha v_{ij}^2 + \beta'_{ij} v_{ij} + \gamma'_{ij} + \frac{\delta_{ij}}{v_{ij} - r_{ij}},$$

where

$$\beta'_{ij} = \alpha r_{ij} + \beta, \qquad \gamma'_{ij} = \alpha r_{ij}^2 + \beta r_{ij} + \gamma, \qquad \delta_{ij} = \alpha r_{ij}^3 + \beta r_{ij}^2 + \gamma r_{ij}.$$

We approximate the right-hand side of (8) by a quadratic function using the Taylor approximation. The Taylor approximation at $\bar{v} := (v_{\min} + v_{\max})/2$ of the forth term of the right-hand side of (8) is given by

$$\frac{\delta_{ij}}{v_{ij} - r_{ij}} \simeq \frac{\delta_{ij}}{\bar{v} - r_{ij}} - \frac{\delta_{ij}}{(\bar{v} - r_{ij})^2}(v_{ij} - \bar{v}) + \frac{\delta_{ij}}{(\bar{v} - r_{ij})^3}(v_{ij} - \bar{v})^2. \tag{9}$$

Then, each term for (i, j) in (1) is given as a quadratic function of log speed v_{ij}:

$$\tilde{\alpha}_{ij} v_{ij}^2 + \tilde{\beta}_{ij} v_{ij} + \tilde{\gamma}_{ij}, \tag{10}$$

where

$$\begin{aligned}
\tilde{\alpha}_{ij} &= \alpha + \frac{\delta_{ij}}{(\bar{v} - r_{ij})^3}, \\
\tilde{\beta}_{ij} &= \beta'_{ij} - \frac{\delta_{ij}}{(\bar{v} - r_{ij})^2} - 2\frac{\delta_{ij}}{(\bar{v} - r_{ij})^3}\bar{v}, \\
\tilde{\gamma}_{ij} &= \gamma'_{ij} + \frac{\delta_{ij}}{\bar{v} - r_{ij}} + \frac{\delta_{ij}}{(\bar{v} - r_{ij})^2}\bar{v} + \frac{\delta_{ij}}{(\bar{v} - r)^3}\bar{v}^2.
\end{aligned}$$

Under a mild assumption, $\tilde{\alpha}_{ij} > 0$ holds. Therefore, approximate objective function (10) is convex quadratic. In addition, the approximation error in (9) is small for small r_{ij}. More specifically, when $v_{ij} \geq \bar{v}$, the approximation error, *i.e.*, the remainder term of the Taylor approximation in (9), is $-\delta_{ij}(v - \bar{v})^3/(v^\dagger - r_{ij})^4$ for some $v^\dagger$ with $\bar{v} \leq v^\dagger \leq v_{ij}$. For example, the error for $v_{\min} = 14$, $v_{\max} = 20$, and $r_{ij} = 1$ is evaluated as the followings:

$$\left| -\frac{\delta_{ij}}{(v^\dagger - r_{ij})^4}(v - \bar{v})^3 \right| \leq \frac{\delta_{ij}}{(\bar{v} - r_{ij})^4}(v_{\max} - \bar{v})^3 < 3.3 \times 10^{-4}.$$

Each term of (10) should vanish when arc (i, j) is not used, *i.e.*, $x_{ij} = 0$. Thus, we replace the constant term $\tilde{\gamma}_{ij}$ by the linear term $\tilde{\gamma}_{ij} x_{ij}$. As a result of this, we obtain the following optimization problem:

$$\begin{array}{ll}
\text{(SNP)} & \\
\text{minimize} & \displaystyle\sum_{(i,j)\in A} (\tilde{\alpha}_{ij} v_{ij}^2 + \tilde{\beta}_{ij} v_{ij} + \tilde{\gamma}_{ij} x_{ij}) \\
\text{subject to} & (2), (3), (4), (5), (6), (7).
\end{array}$$

As we describe below, (SNP) is reformulated as an MISOCP. In order to define an MISOCP, we introduce the second-order cone. For positive integer n, the n-dimensional second-order cone $\mathcal{Q}^n$ is defined as

$$\mathcal{Q}^n := \left\{ \boldsymbol{x} = (x_1, x_2, \ldots, x_n)^\top \in \mathbb{R}^n : x_1 \geq \sqrt{\sum_{i=2}^n x_i^2} \right\}.$$

A second-order cone optimization problem (SOCP) is described as follows:

$$\begin{array}{lll}
\text{minimize} & \boldsymbol{c}^\top \boldsymbol{x} & \\
\text{subject to} & \boldsymbol{a}_i^\top \boldsymbol{x} = b_i & (i = 1, \ldots, m), \\
& \boldsymbol{x} \in \mathcal{Q}^{n_1} \times \cdots \times \mathcal{Q}^{n_k}, &
\end{array} \tag{11}$$

where $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m, \boldsymbol{c} \in \mathbb{R}^n$ $(n = n_1 + \cdots + n_k)$ and $b_1, \ldots, b_m \in \mathbb{R}$ are constants and $\boldsymbol{x} \in \mathbb{R}^n$ is a decision variable. The n-dimensional nonnegative orthant can be represented as $\mathbb{R}^n = \mathcal{Q}^1 \times \cdots \times \mathcal{Q}^1$. Therefore, an LP is a special case of SOCPs. In addition, we can express a number of convex constraints by using second-order cone constraint (11). For example, a system of inequalities

$$\boldsymbol{z}^\top \boldsymbol{z} \leq uv, u \geq 0, v \geq 0 \tag{12}$$

of $u, v \in \mathbb{R}$ and $\boldsymbol{z} \in \mathbb{R}^p$ is equivalent to the second-order cone constraint below:

$$(u + v, u - v, 2\boldsymbol{z})^\top \in \mathcal{Q}^{p+2}.$$

Constraint (12) is called the *restricted hyperbolic constraint.* We use this relationship to reformulate (SNP) as an MISOCP. In addition, SOCPs can be solved in polynomial time with interior-point methods. For more information about SOCPs, see [1, 3, 4].

An MISOCP is an SOCP in which some variables are integral. It can be solved by a combination of techniques in integer optimization and interior-point methods. For example, Gurobi Optimizer, which is one of the most popular solvers for MISOCPs, employs the branch-and-cut method with an interior-point method.

For each quadratic term of (10), we introduce auxiliary variable u_{ij}. Then, (SNP) can be described as the followings:

$$
\begin{aligned}
&\text{minimize} && \sum_{(i,j)\in A} (\tilde{\alpha}_{ij}u_{ij} + \tilde{\beta}_{ij}v_{ij} + \tilde{\gamma}_{ij}x_{ij})d_{ij} \\
&\text{subject to} && v_{ij}^2 \le u_{ij} \qquad ((i,j)\in A), \\
& && (2), (3), (4), (5), (6), (7).
\end{aligned} \tag{13}
$$

Since inequality (13) can be described as the restricted hyperbolic constraint, it can be represented as a second-order cone constraint below:

$$(u_{ij}+1, u_{ij}-1, 2v_{ij})^\top \in \mathcal{Q}^3 \quad ((i,j)\in A).$$

Also, inequality (6) can be described as a combination of a linear constraint and second-order cone constraints. For each $(i,j)\in A$, we introduce auxiliary variable y_{ij}. Then, (6) can be represented as

$$
\begin{aligned}
&\sum_{(i,j)\in A} d_{ij}y_{ij} \le T, \\
&\frac{x_{ij}}{v_{ij} - r_{ij}x_{ij}} \le y_{ij} \quad ((i,j)\in A).
\end{aligned} \tag{14}
$$

Since $v_{ij} \ge v_{\min} > r_{ij}$ and $x_{ij} \in \{0,1\}$, the relations $v_{ij} - r_{ij}x_{ij} \ge 0$ and $x_{ij}^2 = x_{ij}$ hold. From these relations, we see that inequality (14) is represented as

$$x_{ij}^2 \le y_{ij}(v_{ij} - r_{ij}x_{ij}) \quad ((i,j)\in A).$$

Since this inequality can be described as the restricted hyperbolic constraint, this inequality can be represented as a second-order cone constraint as the followings:

$$(y_{ij} + v_{ij} - r_{ij}x_{ij}, y_{ij} - v_{ij} + r_{ij}x_{ij}, 2x_{ij})^\top \in \mathcal{Q}^3 \quad ((i,j)\in A).$$

Consequently, (SNP) can be described as the following MISOCP:

$$
\begin{aligned}
&\text{minimize} && \sum_{(i,j)\in A} (\tilde{\alpha}_{ij}u_{ij} + \tilde{\beta}_{ij}v_{ij} + \tilde{\gamma}_{ij}x_{ij})d_{ij} \\
&\text{subject to} && \sum_{j:(s,j)\in A} x_{sj} = 1, \\
& && \sum_{j:(i,j)\in A} x_{ij} - \sum_{j:(j,i)\in A} x_{ji} = 0 && (i \in N\setminus\{s,t\}), \\
& && \sum_{i:(i,t)\in A} x_{it} = 1, \\
& && x_{ij} \in \{0,1\} && ((i,j)\in A), \\
& && \sum_{(i,j)\in A} d_{ij}y_{ij} \le T, \\
& && v_{\min}x_{ij} \le v_{ij} \le v_{\max}x_{ij} && ((i,j)\in A), \\
& && (u_{ij}+1, u_{ij}-1, 2v_{ij})^\top \in \mathcal{Q}^3 && ((i,j)\in A), \\
& && (y_{ij} + v_{ij} - r_{ij}x_{ij}, y_{ij} - v_{ij} + r_{ij}x_{ij}, 2x_{ij})^\top \in \mathcal{Q}^3 && ((i,j)\in A).
\end{aligned}
$$

3. Route generation algorithm

We have already given an MISOCP formulation of (SNP) in Section 2. When we solve it with general-purpose solvers, it may take long computation time. In order to obtain an optimal solution quickly, we propose a new algorithm named the *route generation algorithm*. In our algorithm, we do not optimize a shipping route and log speed simultaneously. Instead, we firstly generate candidates of an optimal shipping route and subsequently optimize the shipping speed on each leg in each candidate route.

Clearly, we can find an optimal solution of (SNP) by enumerating all s–t paths and optimizing shipping speed for each s–t path. For a fixed s–t path, we can optimize the shipping speed on each arc in the path by solving a small SOCP as we see below. Let $P \subset (N, A)$ be the fixed s–t path and $A(P)$ denote the arc set of P. Then, we obtain the optimal shipping speed on each arc $(i, j) \in A(P)$ by solving the following SOCP:

$$
\begin{array}{lll}
\text{(SNP-S)} & & \\
\text{minimize} & \displaystyle\sum_{(i,j)\in A(P)} (\tilde{\alpha}_{ij}u_{ij} + \tilde{\beta}_{ij}v_{ij} + \tilde{\gamma}_{ij})d_{ij} & \\
\text{subject to} & \displaystyle\sum_{(i,j)\in A(P)} d_{ij}y_{ij} \le T, & \\
& v_{\min} \le v_{ij} \le v_{\max} & ((i,j) \in A(P)), \\
& (u_{ij} + 1, u_{ij} - 1, 2v_{ij})^\top \in \mathcal{Q}^3 & ((i,j) \in A(P)), \\
& (y_{ij} + v_{ij} - r_{ij}, y_{ij} - v_{ij} + r_{ij}, 2)^\top \in \mathcal{Q}^3 & ((i,j) \in A(P)).
\end{array}
$$

Note that we can obtain the problem by fixing variable x_{ij} as $x_{ij} = 1$ for $(i, j) \in A(P)$ and $x_{ij} = 0$ for $(i, j) \notin A(P)$ in (SNP). We can solve (SNP-S) quickly with interior-point methods since the size of (SNP-S) is much smaller than the continuous relaxation problem of (SNP). Suppose that we have an optimal s–t path P^* but do not know an optimal speed v^*_{ij} on each arc $(i, j) \in A(P^*)$. In such a case, v^*_{ij} can easily be obtained by solving (SNP-S) corresponding to P^*.

Note that an optimal s–t path P^* is expected to be a short s–t path. Based on this observation, we generate short s–t paths and optimize speeds on arcs in each generated s–t path. In order to generate short s–t paths, we introduce the following constrained shortest path problem:

$$
\begin{array}{ll}
\text{(CSP)} & \\
\text{minimize} & \displaystyle\sum_{(i,j)\in A} (\tilde{\alpha}_{ij}\tilde{v}_{ij}^2 + \tilde{\beta}_{ij}\tilde{v}_{ij} + \tilde{\gamma}_{ij})d_{ij}x_{ij} \\
\text{subject to} & (2), (3), (4), (5), \\
& \displaystyle\sum_{(i,j)\in A} \frac{d_{ij}}{\tilde{v}_{ij} - r_{ij}} x_{ij} \le T,
\end{array}
$$

where $\tilde{v}_{ij}$ is a fixed shipping speed. Note that we can obtain this problem by fixing variable v_{ij} as $\tilde{v}_{ij}$ in (SNP). By solving this problem, we can find a short s–t path P. Subsequently, we solve (SNP-S) corresponding to P and obtain an approximate optimal solution of (SNP). After solving (SNP-S), we search for another short s–t path by solving (CSP). One of the simplest way to find another s–t path is to add the constraint below to (CSP):

$$
\sum_{(i,j)\in A(P)} x_{ij} \le |A(P)| - 1. \tag{15}
$$

Although we can optimize shipping speed v_{ij} by solving (SNP-S) in polynomial time, it is difficult to solve (SNP-S) for all s–t paths since there are an enormous number of s–t paths. To overcome this difficulty, we make it possible to stop the enumeration of s–t paths. For this purpose, we introduce the following optimization problem (SNP-R):

$$
\begin{array}{ll}
\text{(SNP-R)} & \\
\text{minimize} & \displaystyle\sum_{(i,j)\in A} (\tilde{\alpha}_{ij}v_{ij}^{*2} + \tilde{\beta}_{ij}v^*_{ij} + \tilde{\gamma}_{ij})d_{ij}x_{ij}
\end{array}
$$

$$\text{subject to} \quad (2), (3), (4), (5),$$
$$\sum_{(i,j)\in A} \frac{d_{ij}}{\nu_{max} - r_{ij}} x_{ij} \leq T.$$

where ν^*_{ij} is the minimizer of $\tilde{\alpha}_{ij}\nu^2_{ij} + \tilde{\beta}_{ij}\nu_{ij} + \tilde{\gamma}_{ij}$ in $\nu_{min} \leq \nu_{ij} \leq \nu_{max}$. It is obtained by replacing $\tilde{\nu}_{ij}$ in the objective function of (CSP) with ν^*_{ij} and $\tilde{\nu}_{ij}$ in the last constraint of (CSP) with ν_{max}. Instead of (CSP), we use (SNP-R) to generate short s–t paths. By solving (SNP-R), we obtain a lower bound for the optimal value θ^* to (SNP) as well as a short s–t path. The reason is as follows: For any feasible solution (x_{ij}, ν_{ij}) for $(i, j) \in A$ to (SNP), x_{ij} for $(i, j) \in A$ is a feasible solution to (SNP-R) since the following relations hold:

$$\sum_{(i,j)\in A} \frac{d_{ij}}{\nu_{max} - r_{ij}} x_{ij} \leq \sum_{(i,j)\in A} \frac{d_{ij}}{\nu_{ij} - r_{ij}} x_{ij} = \sum_{(i,j)\in A} \frac{d_{ij}x_{ij}}{\nu_{ij} - r_{ij}x_{ij}} \leq T.$$

In addition, the corresponding objective value in (SNP-R) is not larger than that in (SNP) since ν^*_{ij} is the minimizer of $\tilde{\alpha}_{ij}\nu^2_{ij} + \tilde{\beta}_{ij}\nu_{ij} + \tilde{\gamma}_{ij}$.

Employing (SNP-S) and (SNP-R), we can construct an efficient algorithm for solving (SNP) as described in Algorithm 1. There exist several ways to implement "delete P from the network." In our implementation, we add constraint (15).

Algorithm 1 Route generation algorithm

set $\theta^\dagger$ to $+\infty$
while (SNP-R) is feasible **do**
  solve (SNP-R) and let P and θ_R be the optimal s–t path corresponding to an optimal solution and the optimal value, respectively
  if $\theta^\dagger \leq \theta_R$ **then**
    break
  end if
  solve (SNP-S) corresponding to P and let θ_S denote its optimal value
  if $\theta_S < \theta^\dagger$ **then**
    update $\theta^\dagger$ by θ_S
  end if
  delete P from the network
end while
output $\theta^\dagger$ as the optimal value

An advantage of our route generation algorithm is that we can stop the enumeration of s–t paths in this algorithm. The reason is as follows: In each iteration, the optimal value θ_R of (SNP-R), θ_S of (SNP-S), θ^* of (SNP), and the best known objective value $\theta^\dagger$ satisfies $\theta_R \leq \theta_S$ and $\theta^* \leq \theta^\dagger$. Note that θ_R is monotonically increasing since (SNP-R) finds the shortest s–t path which is different from the ones found in the previous iterations. Suppose that $\theta^\dagger \leq \theta_R$ holds in a certain iteration. In such a case, $\theta^* \leq \theta^\dagger \leq \theta_R \leq \theta_S$ holds. Since θ_R is monotonically increasing, this relation always holds in the following iterations. This means that any s–t path which will be found in the following iterations cannot be better than the best known solution. This guarantees the optimality of $\theta^\dagger$ so that we can stop the enumeration of s–t paths. Another advantage of this algorithm is that it generates a feasible solution at each iteration. Owing to these advantages, we can easily obtain an approximate optimal solution for the feasible problem and the certificate of the infeasibility for the infeasible one.

4. Numerical results

We test our route generation algorithm of (SNP). Test instances are generated as the following: We consider the grid network illustrated in Figure 1. The network has m nodes in the vertical direction and n nodes in the horizontal direction. For each node in the network except the rightmost nodes and bottommost nodes, three arcs are defined: One with the direction to its right node, one with the direction to its lower node, and the other

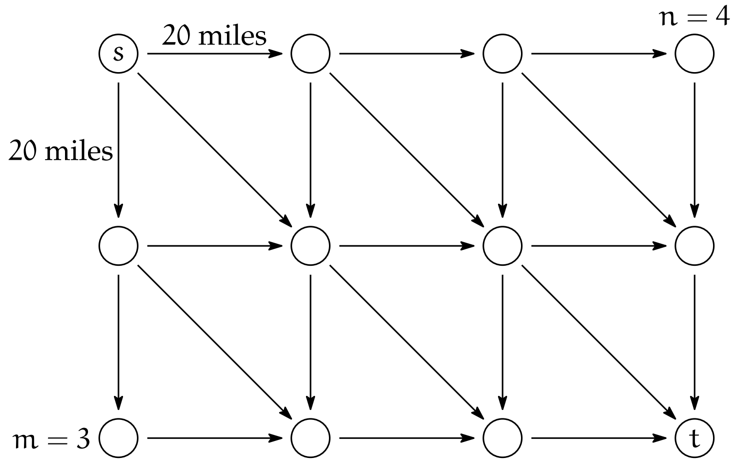

FIGURE 1. Network structure

with the direction to its lower-right node. The lengths of the first two arcs are 20 miles. The length of the last arc is $20\sqrt{2}$ miles. For each of the rightmost nodes in the network, an arc with the direction to its lower node is defined, whose length is 20 miles. For each of the bottommost nodes in the network, an arc with the direction to its right node is defined, whose length is 20 miles. The value of speed reduction r_{ij} is a randomly chosen integer from the set $\{1, \dots, 4\}$. In our numerical experiments, we used the same values of $\nu_{\min}$, $\nu_{\max}$, α, β, and γ in (SNP) as those in Fagerholt *et al.* [6].

We solved the instances by the route generation algorithm and compared the results with those of Gurobi Optimizer 5.1.0 with its default setting. The route generation algorithm is implemented in Python 2.7.3, and (SNP-R) and (SNP-S) are solved as an IP and an SOCP by Gurobi Optimizer, respectively. For evaluating the effectiveness of the route generation algorithm, we also solved (SNP) as an MISOCP by Gurobi Optimizer. In our experiments, we stopped the algorithm when the elapsed time exceeds ten minutes. The numerical experiments were executed on a computer having Intel Xeon 2.80 GHz 6-Core CPUs and 12 GB of RAM with Scientific Linux release 5.9.

In the tables in this section, we use the following symbols and notation. Figures in "m" and "n" columns denote the number of nodes in the vertical and horizontal direction, respectively. A figure in "T" column denotes the value in the right-hand side of total transition time constraint (6). A figure in "time" column denotes the total computational time in seconds of the route generation algorithm or Gurobi Optimizer. A figure in "obj." column denotes the best known objective value by the route generation algorithm or Gurobi Optimizer. For the route generation algorithm, a figure in "mip gap" column denotes the relative difference between the lower bound θ_R and the best known objective value $\theta^\dagger$ at the last iteration defined by $\max\{\theta^\dagger - \theta_R, 0\}/\theta^\dagger$. For Gurobi Optimizer, we used the value defined similarly. We use the notation "infeasible" to denote that the algorithm detected the infeasibility of the instance, the notation "optimal" to denote that it guaranteed the optimality of the best known solution, and the notation "no info" to denote that it could not detect the infeasibility nor find any feasible solution. A figure in "diff." column denotes the relative difference between the best known objective value by the route generation algorithm and that by Gurobi Optimizer at the last iteration defined by $(\theta^\dagger_{GO} - \theta^\dagger_{RG})/\theta^\dagger_{RG}$, where $\theta^\dagger_{RG}$ denote the best known objective value by the route generation algorithm and $\theta^\dagger_{GO}$ the best known objective value by Gurobi Optimizer. Note that a positive value of the relative difference indicates that the route generation algorithm returned the better solution.

Table 1 shows the numerical results on the route generation algorithm and Gurobi Optimizer applied to the instances of $m = 5$. For these instances, we changed n to be 50, 100, 150, and 200. We also changed T to make the instance feasible except the last one with each pair of (m, n). We observe that the route generation algorithm found a feasible solution of each instance or detected its infeasibility. In contrast to this, Gurobi Optimizer could neither find a feasible solution nor detect the infeasibility of four instances with $n = 150$ and four instances with

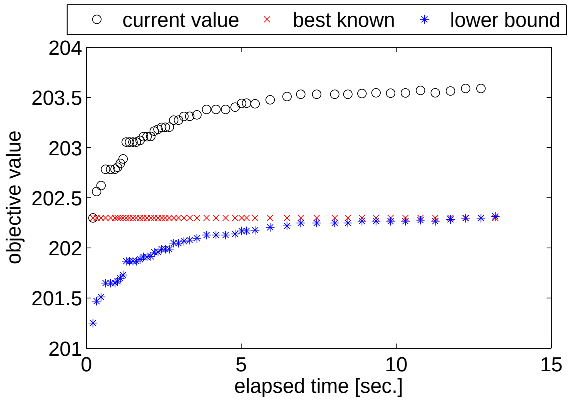

FIGURE 2. The behavior of the computed values by the route generation algorithm for the instance of $m = 5$, $n = 50$, and $T = 80.0$

$n = 200$. We note that the "obj." values by the route generation algorithm are equal to or smaller than those by Gurobi Optimizer for all instances. We also see that the route generation algorithm found a feasible solution of each instance with smaller "mip gap" value than Gurobi Optimizer except two instances. In particular, we observe some critical differences in the "mip gap" values between the route generation algorithm and Gurobi Optimizer for the instances with large T. Concerning the computational time, the route generation algorithm found an optimal solution in short computational time for each instance with small T or large T.

Table 2 shows the results on the route generation algorithm and Gurobi Optimizer applied to the instances of $m = 10$. As in the case of Table 1, the "obj." values by the route generation algorithm are equal to or smaller than those by Gurobi Optimizer for all instances. Moreover, as T becomes small, the computational time of the route generation algorithm becomes smaller. We also observe that the "mip gap" and "diff." values of Gurobi Optimizer are of the larger magnitude than those in Table 1.

Figure 2 shows the behavior of the route generation algorithm for the instances of $m = 5$, $n = 50$ and $T = 80$. Note that the value of $T = 80$ is relatively large (the time constraint is loose) among the test instances. Three values shown in the figure are optimal value θ_S of (SNP-S) (denoted by "current value"), best known objective value $\theta^\dagger$ (denoted by "best known"), and optimal value θ_R of (SNP-R) (denoted by "lower bound") at each iteration. Figure 3 and 4 shows the behavior of Gurobi Optimizer for the same instance. Figure 4 shows the details of the "best known" values in Figure 3. From Figure 2, we see that the route generation algorithm found a good feasible solution and good lower bounds in quite early iterations. Moreover, it guaranteed the optimality within fifteen seconds, *i.e.*, the value of θ_R ("lower bound") became larger than the value of θ^* ("best known"). In contrast to this, from Figure 3, we see that Gurobi Optimizer obtained loose lower bounds in ten minutes.

Figure 5 and 6 shows the behavior on the route generation algorithm for the instance of $m = 5$, $n = 50$, and $T = 57.1$. From Figure 5, we see that the route generation algorithm did not find tight "lower bounds." "best known" values in quite early iterations (see Figure 6). Figures 7 and 8 shows the results on Gurobi Optimizer applied to the same instance. In contrast to the route generation algorithm, Gurobi Optimizer did not find good "best known" values in early iterations. Moreover, "lower bounds" are worse than those by the route generation algorithm. However, it found good "lower bound" values in late iterations and it eventually guaranteed the optimality of the solution. An interesting observation is that the route generation algorithm found an optimal solution in early iterations, although it cannot guaranteed its optimality. We also see the similar result for the instance of $m = 5$, $n = 100$, and $T = 114.1$. In the route generation algorithm, we exploit the structure of the problem. More specifically, we generate short s–t paths, and try to find an optimal solution among these paths. This strategy enables us to obtain a good solution in early iterations. In contrast to this, Gurobi Optimizer does not exploit the problem structure since it is a general-purpose solver. These characteristics lead to the difference between the behaviors of the route generation algorithm and Gurobi Optimizer.

Table 1. Numerical results for the instances of $m = 5$

			Route Generation			Gurobi Optimizer			
m	n	T	time	obj.	mip gap	time	obj.	mip gap	diff.
5	50	90.0	0.3	201.3	optimal	600.0	201.7	82.4%	0.2%
		80.0	13.2	202.3	optimal	600.0	206.2	137.2%	1.9%
		70.0	600.8	223.1	8.9%	600.0	225.9	109.9%	1.3%
		60.0	601.5	292.4	30.5%	600.0	298.2	75.7%	2.0%
		57.1	601.3	328.5	38.1%	181.0	328.5	optimal	0.0%
		57.0	221.3	330.0	optimal	151.0	330.0	optimal	0.0%
		56.9	35.3	331.5	optimal	20.9	331.5	optimal	0.0%
		56.8	3.9	333.0	optimal	67.7	333.0	optimal	0.0%
		56.7	0.3	334.5	optimal	72.0	334.5	optimal	0.0%
		56.6	0.2	—	infeasible	14.3	—	infeasible	—
5	100	170.0	0.6	405.7	optimal	600.0	406.3	170.0%	0.1%
		160.0	602.2	408.4	0.2%	600.0	411.0	193.0%	0.6%
		150.0	601.9	421.9	3.4%	600.0	429.7	202.5%	1.8%
		140.0	602.1	451.5	9.7%	600.1	458.3	185.1%	1.5%
		130.0	601.5	504.2	19.2%	600.1	524.2	28.0%	4.0%
		120.0	601.4	591.1	31.1%	600.2	601.4	137.5%	1.7%
		114.1	602.5	665.5	38.8%	449.2	665.5	optimal	0.0%
		114.0	259.1	667.0	optimal	399.8	667.0	optimal	0.0%
		113.9	34.2	668.5	optimal	544.5	668.5	optimal	0.0%
		113.8	3.1	670.1	optimal	377.8	670.1	optimal	0.0%
		113.7	0.7	671.7	optimal	315.1	671.7	optimal	0.0%
		113.6	0.2	—	infeasible	344.4	—	infeasible	—
5	150	250.0	1.1	609.1	optimal	600.0	616.9	274.3%	1.3%
		240.0	602.5	613.1	0.4%	600.0	621.4	278.1%	1.4%
		230.0	602.2	624.3	2.2%	600.1	633.3	248.4%	1.4%
		220.0	600.4	644.9	5.3%	600.0	654.5	225.9%	1.5%
		210.0	601.0	677.5	9.9%	600.0	690.3	223.0%	1.9%
		200.0	602.8	725.3	15.8%	600.1	747.8	224.8%	3.1%
		190.0	601.6	792.8	23.0%	600.0	810.9	207.0%	2.3%
		180.0	601.1	886.0	31.1%	600.0	908.1	153.4%	2.5%
		170.8	604.2	1002.5	39.1%	600.0	—	no info	—
		170.7	87.6	1004.1	optimal	600.0	—	no info	—
		170.6	6.5	1005.6	optimal	600.0	—	no info	—
		170.5	1.1	1007.2	optimal	600.0	—	no info	—
		170.4	0.6	—	infeasible	554.5	—	infeasible	—
5	200	340.0	1.2	810.1	optimal	600.0	821.6	344.2%	1.4%
		330.0	4.6	810.6	optimal	600.0	824.3	394.7%	1.7%
		320.0	602.2	815.1	0.4%	600.1	837.1	173.1%	2.7%
		310.0	600.9	825.0	1.6%	600.1	851.5	211.0%	3.2%
		300.0	602.4	841.4	3.6%	600.1	854.7	166.2%	1.6%
		290.0	600.9	865.7	6.3%	600.1	880.1	186.1%	1.7%
		280.0	600.1	899.4	9.8%	600.1	921.1	237.0%	2.4%
		270.0	602.3	944.5	14.1%	600.0	976.6	248.4%	3.4%
		260.0	601.8	1003.4	19.1%	600.0	1022.0	396.7%	1.9%
		250.0	602.1	1078.9	24.8%	600.1	1115.6	160.4%	3.4%
		240.0	600.5	1174.9	30.9%	600.1	1203.6	144.5%	2.4%
		230.0	603.0	1296.0	37.4%	600.0	1329.1	252.9%	2.6%
		227.2	600.8	1336.5	39.3%	600.7	—	no info	—
		227.1	97.2	1338.1	optimal	603.4	—	no info	—
		227.0	7.1	1339.7	optimal	600.1	—	no info	—
		226.9	0.9	—	infeasible	600.0	—	no info	—

Table 2. Numerical results for the instances of $m = 10$.

			Route Generation			Gurobi Optimizer			
m	n	T	time	obj.	mip gap	time	obj.	mip gap	diff.
10	50	90.0	0.6	206.2	optimal	600.0	218.3	574.0%	5.8%
		80.0	600.8	209.2	0.9%	600.0	232.6	524.1%	11.2%
		70.0	601.7	237.3	12.6%	600.0	264.0	496.1%	11.2%
		60.0	600.7	321.0	35.4%	600.0	345.6	383.8%	7.6%
		58.6	601.8	340.4	39.1%	600.0	346.1	176.3%	1.7%
		58.5	175.9	342.0	optimal	549.9	342.0	optimal	0.0%
		58.4	3.8	343.5	optimal	389.1	343.5	optimal	0.0%
		58.3	0.3	—	infeasible	443.0	—	infeasible	—
10	100	170.0	1.2	407.0	optimal	600.0	429.4	609.4%	5.5%
		160.0	601.4	410.4	0.6%	600.1	440.0	612.0%	7.2%
		150.0	601.8	425.5	4.1%	600.1	465.3	589.2%	9.3%
		140.0	600.3	457.5	10.8%	600.0	515.5	541.6%	12.7%
		130.0	601.9	514.0	20.6%	600.0	581.1	490.9%	13.1%
		120.0	600.3	606.6	32.7%	600.1	695.4	428.2%	14.6%
		114.9	602.1	673.8	39.5%	600.0	—	no info	—
		114.8	90.2	675.3	optimal	600.0	—	no info	—
		114.7	4.0	676.9	optimal	600.0	—	no info	—
		114.6	1.1	—	infeasible	600.0	—	no info	—
10	150	250.0	4.6	604.7	optimal	600.1	635.8	642.6%	5.1%
		240.0	600.4	608.0	0.4%	600.1	642.4	635.8%	5.6%
		230.0	600.6	618.9	2.2%	600.1	668.0	620.9%	7.9%
		220.0	601.4	639.2	5.3%	600.0	682.6	602.4%	6.8%
		210.0	601.8	671.7	9.9%	600.1	729.9	570.7%	8.7%
		200.0	601.6	719.9	15.9%	600.1	784.0	536.0%	8.9%
		190.0	600.7	788.1	23.2%	600.1	893.4	478.0%	13.3%
		180.0	601.3	882.7	31.4%	600.1	982.2	456.7%	11.3%
		170.6	600.5	1003.8	39.7%	600.0	—	no info	—
		170.5	139.8	1005.3	optimal	600.0	—	no info	—
		170.4	1.7	—	infeasible	600.0	—	no info	—
10	200	340.0	2.7	811.2	optimal	600.1	860.0	629.2%	6.0%
		330.0	351.4	811.9	optimal	600.1	860.2	609.8%	5.9%
		320.0	601.0	816.9	0.6%	600.1	871.2	608.4%	6.7%
		310.0	600.2	827.4	1.9%	600.1	888.3	571.4%	7.4%
		300.0	600.2	844.6	3.9%	600.1	909.7	581.1%	7.7%
		290.0	600.2	870.0	6.7%	600.1	947.9	571.7%	9.0%
		280.0	601.2	905.0	10.3%	600.1	986.8	551.9%	9.0%
		270.0	601.4	951.7	14.7%	600.1	1033.0	525.0%	8.5%
		260.0	600.0	1012.5	19.8%	600.1	1136.3	489.6%	12.2%
		250.0	600.6	1090.6	25.5%	600.2	1189.9	490.1%	9.1%
		240.0	601.4	1189.6	31.7%	600.1	1285.8	449.1%	8.1%
		230.0	602.0	1314.6	38.2%	600.1	—	no info	—
		228.0	601.4	1344.4	39.6%	600.1	—	no info	—
		227.9	10.3	1346.0	optimal	600.0	—	no info	—
		227.8	2.3	—	infeasible	600.0	—	no info	—

When T is small (the time constraint is tight), our algorithm also found an optimal solution quickly or detect the infeasibility in only one iteration. When our algorithm stopped, (SNP-R) became infeasible since all feasible s–t paths were enumerated.

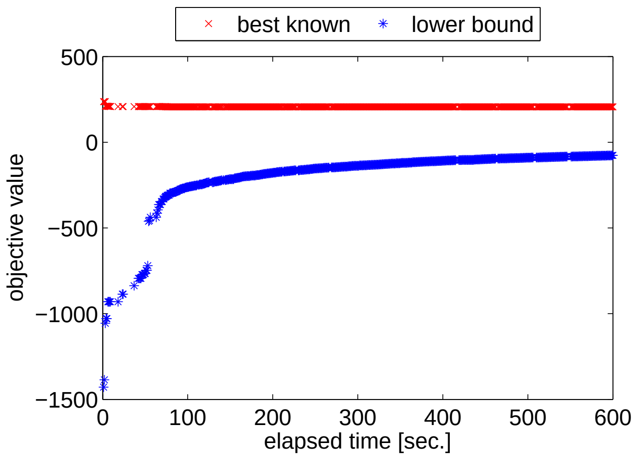

Figure 3. The behavior of "best known" and "lower bound" by Gurobi Optimizer for the instance of $m = 5$, $n = 50$, and $T = 80.0$

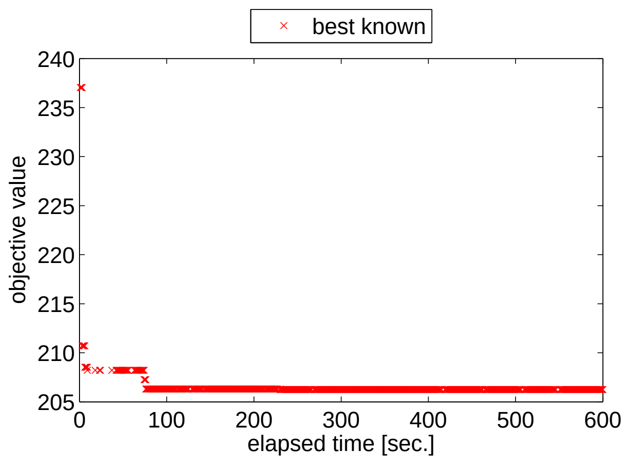

Figure 4. The behavior of "best known" by Gurobi Optimizer for the instance of $m = 5$, $n = 50$, and $T = 80.0$ in detail

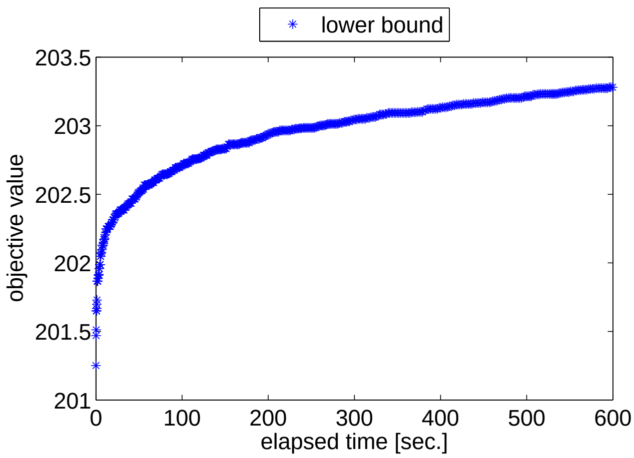

Figure 5. The behavior of "lower bound" by the route generation algorithm for the instance of $m = 5$, $n = 50$, and $T = 57.1$

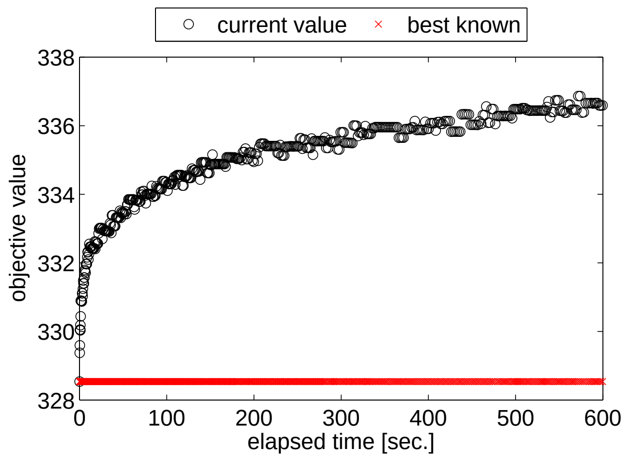

Figure 6. The behavior of "current value" and "best known" by the route generation algorithm for the instance of $m = 5$, $n = 50$, and $T = 57.1$

5. Concluding remarks

We have given a mixed-integer second-order cone optimization formulation of the ship navigation problem. In addition, we proposed the route generation algorithm for this problem. Our route generation algorithm generates short shipping routes and optimizes the shipping speed on each leg for each shipping route. By using the lower bound for the optimal value, we have made it possible to stop the enumeration of shipping routes and guarantee the optimality of the obtained solution or detect the infeasibility of the problem.

From the results of numerical experiments, we see the following features of the algorithm: If the total transition time constraint is either loose or tight, our algorithm obtain an optimal solution in short computational time. Otherwise, our algorithm returns a good feasible solution although our algorithm might not guarantee its optimality.

We have future work described below: In this paper, we approximate the objective function by the quadratic function. Therefore, the solution to the MISOCP model is an approximate solution to the exact MINLP formulation. Thus, our future work includes the comparison of the solution by the route generation algorithm

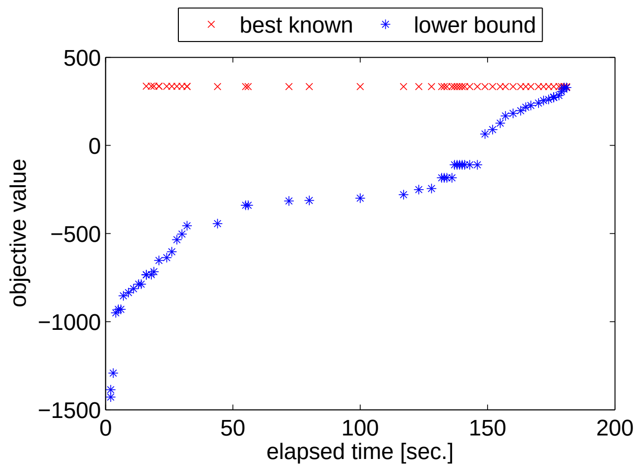

FIGURE 7. The behavior of "best known" and "lower bound" by Gurobi Optimizer to the instance of $m = 5, n = 50$, and $T = 57.1$

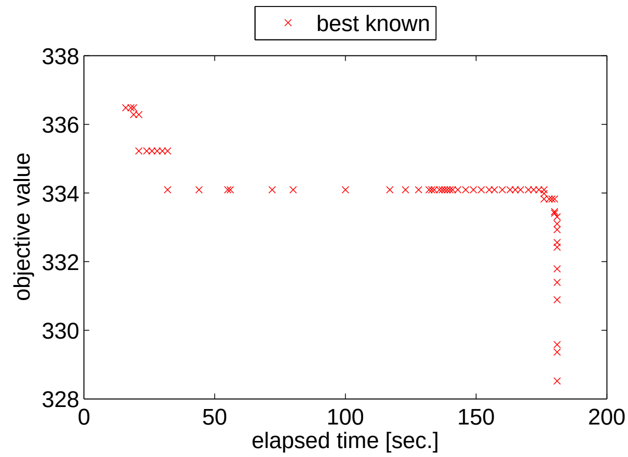

FIGURE 8. The behavior of "best known" by Gurobi Optimizer to the instance of $m = 5, n = 50$, and $T = 57.1$

with an exact optimal solution to the MINLP formulation. In our numerical experiments, our route generation algorithm cannot guarantee the optimal of the obtained solution for some instances. Our future work also include the development of a method to compute tighter lower bounds.

ACKNOWLEDGEMENTS

We thank Yuichiro Miyamoto and Kazuhide Nakata for providing useful advice on the implementation and the experiments.

REFERENCES

[1] ALIZADEH, F. AND GOLDFARB, D.: Second-order cone programming, *Mathematical Programming* **95** (2003) 3–51.
[2] AZARON, A. AND KIANFAR, F.: Dynamic shortest path in stochastic dynamic networks: Ship routing problem, *European Journal of Operational Research* **144** (2003) 138–156.
[3] BEN-TAL, A. AND NEMIROVSKI, A.: *Lectures on Modern Convex Optimization*, Society for Industrial and Applied Mathematics, 2001.
[4] BOYD, S. AND VANDENBERGHE, L.: *Convex Optimization*, Cambridge University Press, 2004.
[5] CHISTIANSEN, M., FAGERHOLT, K., NYGREEN, B., AND RONEN, D.: Ship routing and schedulling in the millennium, *European Journal of Operational Research* **228** (2013) 467–483.
[6] FAGERHOLT, K., LAPORTE, G., AND NORSTAD, I.: Reducing fuel emissions by optimizing speed on shipping routes, *Journal of the Operational Research Society* **61** (2010) 523–529.
[7] LO, H. K. AND MCCORD, M. R.: Adaptive ship routing through stochastic ocean current: general formulations and empirical results, *Transportation Research Part A: Policy and Practice* **32** (2003) 138–156.
[8] PERAKIS, A. N. AND PAPADAKIS, N. A.: Fleet deployment optimization models. Part 1, *Maritime Policy & Management* **14** (1987) 127–144.
[9] PERAKIS, A. N. AND PAPADAKIS, N. A.: Fleet deployment optimization models. Part 2, *Maritime Policy & Management* **14** (1987) 145–155.
[10] RONEN, D.: The effect of oil price on the optimal speed of ships, *Journal of the Operational Research Society* **33** (1982) 1035–1040.

(Tanaka, M.) GRADUATE SCHOOL OF DECISION SCIENCE AND TECHNOLOGY, TOKYO INSTITUTE OF TECHNOLOGY, 2-12-1-W9-60, OOKAYAMA, MEGURO-KU, TOKYO, 152-8552, JAPAN
E-mail address: `tanaka.m.aa@m.titech.ac.jp`

(Kobayashi, K.) NAGIVATION & LOGISTICS ENGINEERING DEPARTMENT, NATIONAL MARITIME RESEARCH INSTITUTE, 6-38-1 SHINKAWA, MITAKA-SHI, TOKYO, 181-0004, JAPAN
E-mail address: `kobayashi@nmri.go.jp`