UDC 519.854.2+519.714.71
DOI: 10.20535/SRIT.2308-8893.2023.2.08

A GENETIC ALGORITHM IMPROVEMENT BY TOUR CONSTRAINT VIOLATION PENALTY DISCOUNT FOR MARITIME CARGO DELIVERY

V.V. ROMANUKE, A.Y. ROMANOV, M.O. MALAKSIANO

Abstract. The problem of minimizing the cost of maritime cargo delivery is considered. The cost is equivalent to the sum of the tour lengths of feeders used for the delivery. The problem is formulated as a multiple traveling salesman problem. In order to find its solution as the shortest route of the tours of feeders, a genetic algorithm is used where we present two inequalities constraining the tour length of every feeder to lie between the shortest and longest lengths. Apart from the constant tour constraint violation penalty in the genetic algorithm, we suggest a changeable penalty as an exponential function of the algorithm iteration, where we maintain the possibility of the penalty rate to be either increasing or decreasing, whose steepness is controlled by a positive parameter. Our tests show that the changeable penalty algorithm may return shorter routes, although the constant penalty algorithms cannot be neglected. As the longest possible tour of the feeder is shortened, the changeable penalty becomes more useful owing to a penalty discount required either at the beginning or at the end of the algorithm run to improve the selectivity of the best feeder tours. In optimizing maritime cargo delivery, we propose to run the genetic algorithm by the low and constant penalties along with the increasing and decreasing penalties. The solution is the minimal value of the four route lengths. In addition, we recommend that four algorithm versions be initialized by four different pseudorandom number generator states. The expected gain is a few percent, by which the route length is shortened, but it substantially reduces expenses for maritime cargo delivery.

Keywords: maritime cargo delivery, tour length, genetic algorithm, tour constraint violation penalty, penalty discount.

INTRODUCTION

The up-to-date market of cargo delivery is divided into three branches of transportation: ground-surface, water, and air. Among them the water transportation has been the most used. In general, this is the maritime transportation which is the basis of the world trading and commerce. Roughly about 80% of all goods are transported by river, sea and ocean. The amount of maritime cargo has been dramatically growing since 1980. In 2020, there were about 1.85 billion metric tons shipped all over the world, whereas it was only 0.1 billion metric tons in 1980. Quite naturally, the world fleet of containers has expanded. The gross tonnage of container carriers since 1980 has increased from 11 up to 275 million metric tons [1, 2].

The main advantages of maritime transportation over competitors are the cost and reliability, and also the possibility to deliver any cargo. The main drawbacks are relatively low speed of delivery and dependence on weather. It is impossible to influence weather, when a delivery is scheduled, but it is possible to increase the delivery speed by routing the most efficient tours. The efficient tour implies its minimally possible length, expressed in units of either distance or time.

Tour length minimization is a transportation optimization problem [3]. In particular, this is a version of the assignment problem or the traveling salesman problem [4, 5].

In fact, the traveling salesman problem solves the task of routing efficient tours, using which optimizes the cost of the delivery. It is an NP-hard problem in combinatorial optimization, whose exact solution usually takes too long to be obtained because exact algorithms perform reasonably fast only for small-sized problems [6]. Heuristic algorithms perform far much faster producing approximated solutions and saving computational resources (which are equivalent to time and budget) [7, 8].

One of the best heuristics is the genetic algorithm allowing to find tours whose length is practically close to the minimal length of the delivery [9, 10]. Sometimes the length found heuristically coincides with the length in the exact solution. For maritime cargo delivery with using multiple tours, the genetic algorithm requires such input parameters as follows: a map of ports, a number of feeders (in maritime transportation, a cargo boat is called the feeder), a population size, and a series of additional inputs including mutation operators. In detail, the map of ports is the two-coordinate location of ports which should be visited en route. The number of feeders defines the maximal number of tours by which the cargo can be delivered. The population size is the number of randomly generated tours to be processed by the algorithm.

To obtain the best approximated solution, the adjustable inputs (like the population size, mutation operators, and others) should be optimally configured. The optimal configuration is a very tough task being itself an optimization problem (similar, e. g., to the optimization in AutoML [11, 12]). In this way, rules of thumb are widely accepted based on recent experience [13, 14]. Another way to optimize the algorithm performance is to use penalty when a tour length exceeds an upper length. The upper length is determined by the capability of the feeder which can cover only the upper length distance and after that it will require fuel refill. This is a tour constraint whose violation imposes a penalty that expunges too lengthy tour from the processing. However, the tour constraint penalty is taken by rules of thumb as well [15, 16]. Therefore, a proper rationalization of the penalty would improve the genetic algorithm performance.

PROBLEM STATEMENT

The goal is to algorithmize the tour constraint penalty in order to optimize the genetic algorithm itself. Moreover, the penalty algorithmization is expected to make possible further minimization of the tour cost. For achieving the goal, the following five tasks are to be fulfilled:

1. To formalize variables used in the genetic algorithm for a maritime cargo delivery model. The model is to be formulated based on [15].

2. To substantiate the inclusion of the tour constraint penalty into the algorithm. The penalty must be changeable depending on the state of the algorithm convergence.

3. To show the advantage of the algorithm using the changeable penalty compared to the algorithm using the constant penalty.

4. To discuss the significance and practical applicability of the suggested improvement in the genetic algorithm.

5. To make an unbiased conclusion on the contribution to the field of genetic algorithms used, in particular, to optimize maritime cargo delivery. An outlook of how the research should be extended and advanced is to be made as well.

MARITIME CARGO DELIVERY MODEL

We have N ports, from one of which every feeder starts its tour and ends up by returning to that port. By default, the port is assigned number 1 and is called the hub. Let p_{k1} and p_{k2} be coordinates of port k. Coordinates of all the ports are gathered in matrix

$$\mathbf{P} = [p_{kl}]_{N\times 2}. \tag{1}$$

The distance between port k and port j is

$$\rho(k, j) = \sqrt{(p_{k1} - p_{j1})^2 + (p_{k2} - p_{j2})^2} \text{ by } k = \overline{1, N} \text{ and } j = \overline{1, N}.$$

All the distances are gathered in matrix

$$\mathbf{D} = [\rho(k, j)]_{N\times N}. \tag{2}$$

Obviously,

$$\rho(k, j) = \rho(j, k) \quad \forall k = \overline{1, N} \text{ and } \forall j = \overline{1, N}$$

and

$$\rho(k, k) = 0 \quad \forall k = \overline{1, N}.$$

So, matrix (2) is symmetric:

$$\mathbf{D} = \mathbf{D}^{\mathrm{T}}.$$

Matrix of distances (2) is directly associated with durations of the maritime cargo delivery. The durations, in their turn, can be treated as the costs of the delivery.

The maximally possible number of feeders is denoted by $M_{\max}$, where $M_{\max} \in \mathrm{N} \setminus \{1\}$. We consider binary variable x_{kjm} associating ports k and j and feeder m, where $m = \overline{1, M}$ and M is a current total number of feeders:

$$M \leqslant M_{\max}. \tag{3}$$

Thus, $x_{kjm} = 1$ if ports k and j are included into the tour of feeder m, where the feeder visits either port j after port k or port k after port j: if $x_{kjm} = 1$ then $x_{jkm} = 0$ and if $x_{jkm} = 1$ then $x_{kjm} = 0$ for non-two-port tours. If a tour of feeder m is of just ports 1 and k, then $x_{1km} = x_{k1m} = 1$ because the feeder must return to the hub. Otherwise, if feeder m does not visit port j after port k nor port k after port j, $x_{kjm} = 0$ (although ports k and j still can be included into the tour of feeder m). So,

$$x_{kjm} \in \{0, 1\} \text{ by } k = \overline{1, N} \text{ and } j = \overline{1, N} \text{ and } m = \overline{1, M} \tag{4}$$

by

$$\sum_{j=2}^{N}\sum_{m=1}^{M} x_{1jm} = M \tag{5}$$

and

$$\sum_{k=2}^{N}\sum_{m=1}^{M} x_{k1m} = M \ , \tag{6}$$

where equality (5) means that each of M feeders only once departs from the hub, and equality (6) means that each of M feeders only once arrives to the hub.

Meanwhile, a feeder may not cover the distance greater than $d_{\max}$. Therefore, inequality

$$\sum_{k=1}^{N}\sum_{j=1}^{N} \rho(k, j) \cdot x_{kjm} \leqslant d_{\max} \quad \forall\, m = \overline{1, M} \tag{7}$$

constrains the tour of every feeder. Moreover, the feeder must not be charged for the delivery if its tour is too short. If $d_{\min}$ is the shortest possible tour of the feeder, inequality

$$\sum_{k=1}^{N}\sum_{j=1}^{N} \rho(k, j) \cdot x_{kjm} \geqslant d_{\min} \quad \forall\, m = \overline{1, M} \tag{8}$$

also constrains the tour of every feeder.

Only one feeder can arrive at port j, being not the hub, from only one port (which can be the hub). This is expressed by equality

$$\sum_{k=1}^{N}\sum_{m=1}^{M} x_{kjm} = 1 \quad \forall\, j = \overline{2, N} \, . \tag{9}$$

Symmetrically, only one feeder can depart from port k, being not the hub, towards only one following port (which can be the hub). This is expressed by equality

$$\sum_{j=1}^{N}\sum_{m=1}^{M} x_{kjm} = 1 \quad \forall\, k = \overline{2, N} \, . \tag{10}$$

Every feeder must depart from the hub and arrive at it, so its tour is a closed loop. This is ensured by the following requirement:

$$\sum_{k \in Q_m} \sum_{j \in Q_m \setminus \{k\}} x_{kjm} \leqslant |Q_m| - 1$$

$$\forall\, Q_m \subset T_m = \{1, \{q_l^{(m)}\}_{l=2}^{A_m}\} \subset \{\overline{1, N}\} \text{ by } 2 \leqslant |Q_m| < A_m \text{ and } \forall\, m = \overline{1, M} \tag{11}$$

with tour

$$T_m = \{1, \{q_l^{(m)}\}_{l=2}^{A_m}\} \subset \{\overline{1, N}\} \tag{12}$$

of feeder m. Constraint (11) eliminates any subtours of every feeder. This ensures that a feasible route of delivering maritime cargo is of closed loops only, where every loop is a feeder tour starting off the hub and ending up by returning to the hub.

To optimize the maritime cargo delivery, we minimize the sum of all the tours of the feeders: objective function

$$\rho_{\Sigma}\left(N, M, \left\{\{\{x_{kjm}\}_{k=1}^{N}\}_{j=1}^{N}\right\}_{m=1}^{M}, d_{\min}, d_{\max}\right) = \sum_{k=1}^{N}\sum_{j=1}^{N}\sum_{m=1}^{M} x_{kjm}\cdot\rho(k, j)$$

is to be minimized subject to constraints (3)–(12). The minimization is implied to be done over binary variables (4) along with trying to minimize the total number of feeders used in the tours. That is, the minimization goal is to find such

$$M^{*} \in \{\overline{1, M_{\max}}\}$$

and

$$x_{kjm}^{*} \in \{0, 1\} \text{ for } k = \overline{1, N} \text{ and } j = \overline{1, N} \text{ by } m = \overline{1, M^{*}}$$

at which

$$\sum_{k=1}^{N}\sum_{j=1}^{N}\sum_{m=1}^{M^{*}} x_{kjm}^{*}\cdot\rho(k, j) =$$

$$= \rho_{\Sigma}\left(N, M^{*}, \left\{\{\{x_{kjm}^{*}\}_{k=1}^{N}\}_{j=1}^{N}\right\}_{m=1}^{M^{*}}, d_{\min}, d_{\max}\right) =$$

$$= \min_{\substack{\{\{x_{kjm}\}_{k=1}^{N}\}_{j=1}^{N},\\ m=\overline{1, M},\, M=\overline{1, M_{\max}}}} \sum_{k=1}^{N}\sum_{j=1}^{N}\sum_{m=1}^{M} x_{kjm}\cdot\rho(k, j). \qquad (13)$$

The solution given formally as

$$\left\{\left\{\{x_{kjm}^{*}\}_{k=1}^{N}\right\}_{j=1}^{N}\right\}_{m=1}^{M^{*}} \qquad (14)$$

allows to build a set of M^{*} the most rational tours of M^{*} feeders. Sum (13) of these tours is the shortest route to deliver maritime cargo and return to the hub.

THE TOUR CONSTRAINT PENALTY

Even for a few tens of ports, it is an intractably time-consuming computational task to find an exact solution of problem (13) subject to constraints (3)–(12). A solution whose route length is quite close to the shortest route length is obtained by genetic algorithms. One of the best genetic algorithms designed for solving problem (13) subject to constraints (3)–(7) and (9), (10) was presented in [15]. Herein, we add constraint (8) cutting off too short feeder tours, and add constraint (11) with tour (12) of feeder m eliminating subtours.

The genetic algorithm uses four forms of chromosome mutations: flip, swap, slide, and crossover. The crossover operation takes two chromosomes, cuts each chromosome in two parts in random places, and interchanges those parts. For generating a random place of the chromosome cut, the minimal number of ports every feeder should visit without counting the hub after starting off port 1 (hub) is used. This number is

$$H_{\min} = \psi\left(\frac{N-1}{M_{\max}}\right), \qquad (15)$$

where function $\psi(x)$ returns the integer part of number x [17]. In addition, within the crossover operation, two chromosomes as tours of two different feeders may be merged into a single tour allowing to decrease the number of feeders used to deliver maritime cargo. This is done with using a merging probability β given at the input of the genetic algorithm.

Let H_m be the number of ports which feeder m should visit after starting off port 1 (hub). Thus, we denote the vector of the tour of feeder m (vector of ports which feeder m should visit in the order of the sequence of the vector elements) by

$$\mathbf{F}_m = [f_h^{(m)}]_{1\times H_m}. \tag{16}$$

So,

$$\bigcup_{m=1}^{M} \{f_h^{(m)}\}_{h=1}^{H_m} = \{\overline{2, N}\}. \tag{17}$$

Initially, tours $\{\mathbf{F}_m\}_{m=1}^{M}$ of feeders are randomly generated by breaking the set of non-hub ports $\{\overline{2, N}\}$ with using integers (15) and M. Each feeder has a series of such tours called population.

For every element of the population, the following routine is executed during an iteration of the algorithm. First,

$$d_m = 0 \text{ for } m = \overline{1, M}.$$

The distance to the port following the hub is calculated as

$$d_m = \rho(1, f_1^{(m)}).$$

Then, the remaining distances except the last one are accumulated into d_m:

$$d_m^{(\text{obs})} = d_m,\ d_m = d_m^{(\text{obs})} + \rho(f_k^{(m)}, f_{k+1}^{(m)}) \text{ for } k = \overline{1, H_m - 1}.$$

Finally, the distance of returning to the hub is:

$$d_m^{(\text{obs})} = d_m,\ d_m = d_m^{(\text{obs})} + \rho(f_{H_m}^{(m)}, 1). \tag{18}$$

To improve selectivity of the best feeder tours to solution (14), and to expunge tours which violate conditions (7), (8), the tour constraint violation penalty is applied. Thus, as it was shown in [15], if $d_m > d_{\max}$ then a current accumulated distance d_m after (18) can be increased with a factor $\lambda > 0$:

$$d_m^{(\text{obs})} = d_m,\ d_m = d_m^{(\text{obs})} + (d_m^{(\text{obs})} - d_{\max}) \cdot \lambda.$$

The increment of d_m in the case of $d_m < d_{\min}$ can be done in the same way.

However, we introduce a more flexible tour constraint violation penalty. In our version, the penalty rate depends on the iteration (denoted by i) of the genetic algorithm. It is either increasing or decreasing that is controlled by a positive parameter α:

$$r(i) = 1 + \frac{1 + \operatorname{sign}(\alpha - 1)}{2} + \operatorname{sign}(1 - \alpha) e^{-i \cdot |1-\alpha|},$$

where the case $\alpha = 1$ is excluded. The penalty rate is increasing if $\alpha > 1$, and it is decreasing if $\alpha < 1$. For instance, if $\alpha = 1.01$ then the penalty rate exponentially increases from a number close to α (because $r(1) \approx \alpha$ in this case) up to 2 (Fig. 1). If, say, $\alpha = 0.995$ then the penalty rate exponentially decreases from a number close to $1+\alpha$ (in this case $r(1)$ is slightly greater than $1+\alpha$) down to 1 (Fig. 2).

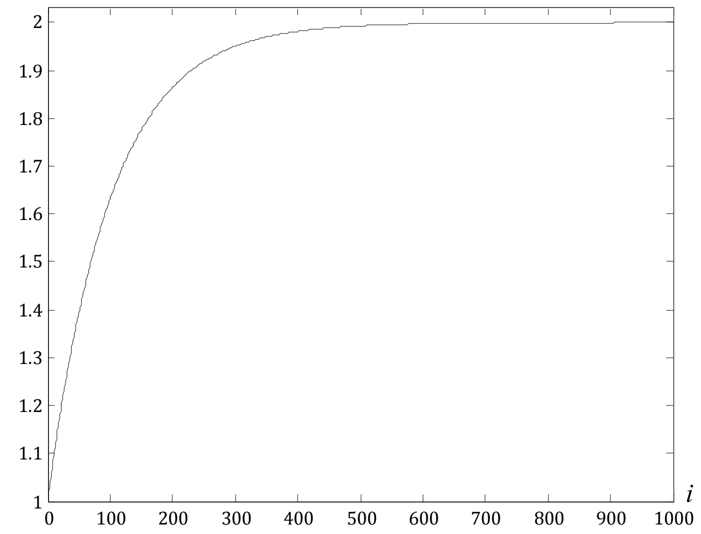

Fig. 1. The increasing tour constraint violation penalty by $\alpha = 1.01$

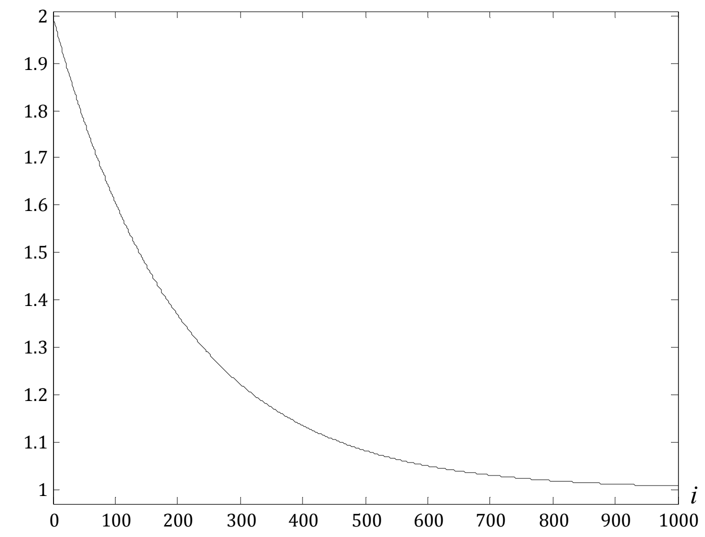

Fig. 2. The decreasing tour constraint violation penalty by $\alpha = 0.995$

Therefore, upon obtaining accumulated distance d_m by (18), if $d_m > d_{\max}$ then

$$d_m^{(\text{obs})} = d_m,\ d_m = d_m^{(\text{obs})} + (d_m^{(\text{obs})} - d_{\max}) \cdot r(i).$$

If $d_m < d_{\min}$ then

$$d_m^{(\text{obs})} = d_m,\ d_m = d_m^{(\text{obs})} + (d_{\min} - d_m^{(\text{obs})}) \cdot r(i).$$

Finally, sum

$$\tilde{\rho}_{\Sigma}(N, M, \{\mathbf{F}_m\}_{m=1}^{M}, d_{\min}, d_{\max}; \alpha) = \sum_{m=1}^{M} d_m$$

is calculated and minimized over the population. Obviously,

$$\tilde{\rho}_{\Sigma}(N, M, \{\mathbf{F}_m\}_{m=1}^{M}, d_{\min}, d_{\max}; \alpha) \geqslant \rho_{\Sigma}\left(N, M^*, \left\{ \left\{ \{x^*_{kjm}\}_{k=1}^{N} \right\}_{j=1}^{N} \right\}_{m=1}^{M^*}, d_{\min}, d_{\max} \right).$$

Herein, the question is which α is to be selected. The matter is that at different values of α the output of the genetic algorithm varies. This is so due to the state of the algorithm convergence varies depending on how tours violating requirements (7), (8) are expunged. Therefore, it is better to run through a set of the values and to select such a value at which the route is the shortest. The first run of the algorithm is done at $\alpha = 1.01$, whereupon the value is decreased by a factor slightly less than 1:

$$\alpha^{(\text{obs})} = \alpha, \quad \alpha = 0.999\alpha^{(\text{obs})}. \qquad (19)$$

After 10 runs, the penalty rate is still an exponentially increasing curve because $\alpha \approx 1.000946$. Since the 11-th run, the penalty rate decreases because then $\alpha \approx 0.999945$. In fact, $\alpha > 0.95$ for the first 62 runs, whereas $\alpha < 0.95$ after the 63-rd run. So, we re-run the algorithm until $\alpha > 0.95$ starting with $\alpha = 1.01$ and proceeding by (19). Besides, we use an early stop condition imposed on the re-running. Denote by

$$\tilde{\rho}^*_{\Sigma} = \tilde{\rho}_{\Sigma}\left(N, M^*, \{\mathbf{F}_m\}_{m=1}^{M^*}, d_{\min}, d_{\max}; \alpha^* \right)$$

the shortest route length found so far. Denote by s_{fail} the counter of fails to improve the route (i. e., to shorten its length), and denote the maximal number of such fails by $s_{\text{fail}}^{(\max)}$. If

$$\tilde{\rho}_{\Sigma}\left(N, M^*, \{\mathbf{F}_m\}_{m=1}^{M^*}, d_{\min}, d_{\max}; \tilde{\alpha} \right) < \tilde{\rho}^*_{\Sigma} \qquad (20)$$

for a next value of $\alpha = \tilde{\alpha}$, then

$$\tilde{\rho}^*_{\Sigma} = \tilde{\rho}_{\Sigma}\left(N, M^*, \{\mathbf{F}_m\}_{m=1}^{M^*}, d_{\min}, d_{\max}; \tilde{\alpha} \right)$$

and

$$\alpha^* = \tilde{\alpha},$$

whereupon the counter of fails is set at 0:

$$s_{\text{fail}} = 0.$$

Otherwise, if (20) is false,

$$s_{\text{fail}}^{(\text{obs})} = s_{\text{fail}}, \; s_{\text{fail}} = s_{\text{fail}}^{(\text{obs})} + 1.$$

So, the algorithm is re-run while $\alpha > 0.95$ and $s_{\text{fail}} < s_{\text{fail}}^{(\max)}$.

To see whether the algorithm performs better with a decreasing tour constraint violation penalty, we need $s_{\text{fail}}^{(\max)} \geqslant 10$. In any way, the last value of α in ascertaining the performance must be less than 1. Therefore, in short, the described flexible penalty may be called the tour constraint penalty discount, although an increasing tour constraint violation penalty can give the shortest route as well (for simplicity, the route returned by the algorithm we will further call the shortest, although a shorter route may exist). In this case, it can be said that a penalty discount is given at the start of the algorithm run; as the run advances (the number of passed iterations increases), the discount decays.

TESTING

First, we test the algorithm for 10 to 50 ports randomly scattered. In this case, all ports coordinates (1) are in matrix

$$\mathbf{P} = 50 \cdot \Theta(N, 2)$$

by

$$N = 5 + 5n\,,\ n = \overline{1, 9}$$

and an operator $\Theta(N, 2)$ returning a pseudorandom $N \times 2$ matrix whose entries are drawn from the standard uniform distribution on the open interval $(0; 1)$. The remaining parameters are:

$$M_{\max} = 2\,,\ s_{\text{fail}}^{(\max)} = 15\,,\ \beta = 0.05\,,$$

$$d_{\max} = \psi\left(1.25 \cdot \psi\left(0.5 \cdot \max_{k=\overline{1, N}} \left\{ \sum_{j=1}^{N} \rho(k, j) \right\}\right)\right),$$

$$d_{\min} = \zeta(0.1 \cdot d_{\max})\,, \tag{21}$$

where function $\zeta(x)$ rounds number x to the nearest integer towards infinity. The maximal number of iterations is 3600, whereas the algorithm early stop condition is used, by which (a run of) the algorithm is stopped if the shortest route length does not change for 720 iterations (a one fifth of the maximal number of iterations). The test is repeated for 100 times for the algorithm used in three versions: with the tour constraint penalty discount, with the constant penalty by

$$r(i) = 1 \quad \forall\, i = \overline{1, 3600}\,, \tag{22}$$

and with the augmented constant penalty by

$$r(i) = 100 \quad \forall\, i = \overline{1, 3600}\,. \tag{23}$$

Overall, there are 900 route lengths (for instances of randomly generated ports) returned by each of the three versions, where $M^* = 1$ (the shortest route is of a single tour in every solution). We will refer to them as v_α, v_1, v_{100}, respectively. Whereas the pseudorandom number generator outputs the same instance for each of the three algorithm versions, re-runs for v_α are not initialized with the same pseudorandom number generator seed. The advantage of the algorithm using the changeable penalty compared to the algorithm using constant penalties by (22), (23) can be seen in Table 1, where "better than" implies producing a shorter route. Table 2 shows that v_{α} has been worse than v_{100} in less than 1.5 %. Compared v_{α} to v_1, this percentage is even far smaller – just 0.1111 % (only one of those 900 route lengths produced by v_{α} has appeared to be longer than the respective route length produced by v_1; this is an instance of 50 ports whose v_{α}-route length is 274.8312 and v_1-route length is 274.6006). Table 3 shows that v_1 and v_{100} are more likely to produce the same result (strictly speaking, the equal lengths of the routes, whereas the routes themselves may differ in particular regions). The percentage of instances where v_{α} performs identically to v_1 or v_{100} is not that small. At least, this is 20% on average.

Table 1. The percentage of instances where one algorithm version produces a shorter route than the other version

		v_{α} **better than** v_1	v_{α} **better than** v_{100}	v_1 **better than** v_{100}
Overall average		75.7778	75.8889	19.3333
N	10	10	17	11
	15	55	51	8
	20	78	71	9
	25	83	84	14
	30	88	88	19
	35	88	89	22
	40	95	95	29
	45	90	90	28
	50	95	98	34

Table 2. The percentage of instances where one algorithm version produces a longer route than the other version

		v_{α} **worse than** v_1	v_{α} **worse than** v_{100}	v_1 **worse than** v_{100}
Overall average		0.1111	1.4444	21.5556
N	10	0	0	2
	15	0	0	15
	20	0	0	17
	25	0	0	17
	30	0	0	19
	35	0	3	28
	40	0	3	24
	45	0	6	40
	50	1	1	32

Table 3. The percentage of instances where two algorithm versions produce the same route lengths

		v_α **and** v_1	v_α **and** v_{100}	v_1 **and** v_{100}
Overall average		24.1111	22.6667	59.1111
N	10	90	83	87
	15	45	49	77
	20	22	29	74
	25	17	16	69
	30	12	12	62
	35	12	8	50
	40	5	2	47
	45	10	4	32
	50	4	1	34

In the test, v_α has produced 653 route lengths (72.5556 %) by $\alpha > 1$ (i. e., by an increasing tour constraint violation penalty). All the 100 instances of 10 ports are solved by $\alpha > 1$. Then, however, the percentage of instances where v_α has produced the shortest route by an increasing tour constraint penalty starts decreasing (Fig. 3). The way how the best values of α are distributed shown in Fig. 4 delusively hints at that the increasing penalty is better than the decreasing one. Meanwhile, it is worth noting that 224 of 900 instances has been v_α-solved by starting with $\alpha = 1.01$. The distributions of the best values of α for the number of ports in Fig. 5 confirm that the decreasing penalty becomes more influential as the number increases.

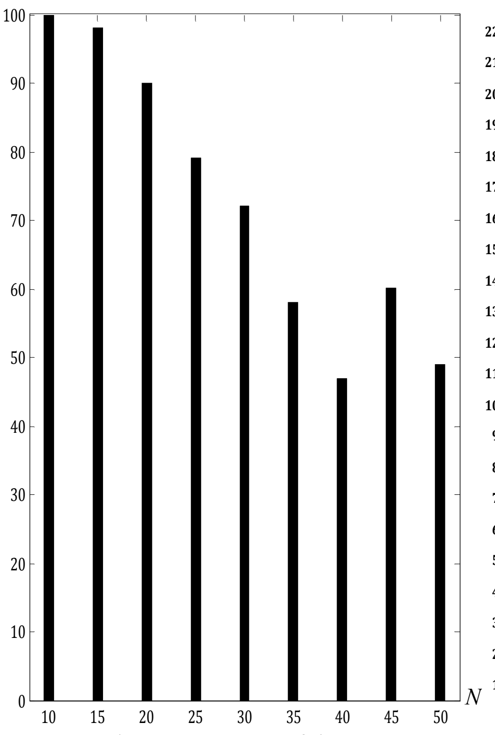

Fig. 3. The percentage of instances per number of ports where v_α has produced the shortest route by $\alpha > 1$

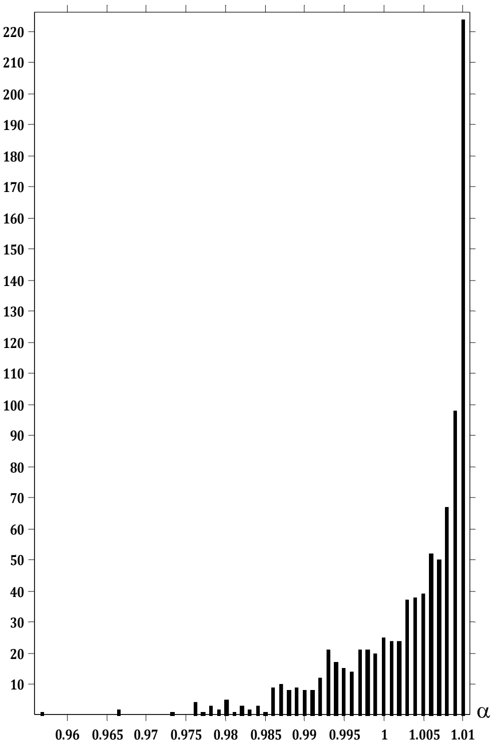

Fig. 4. The distribution of instances per α at which the shortest route is obtained

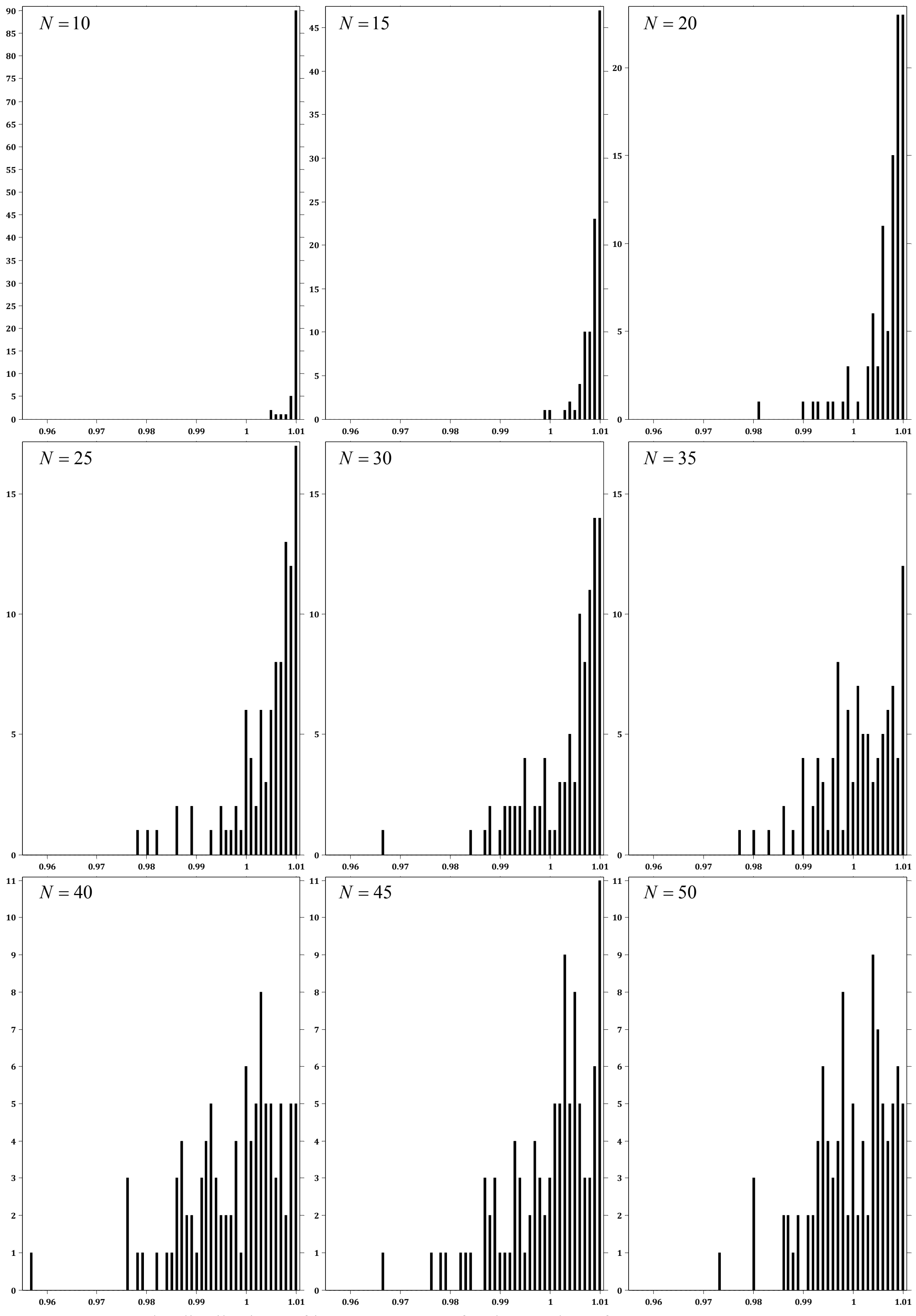

Fig. 5. The distributions of instances per α for the number of ports

A simple example of how the changeable penalty algorithm outperforms the two constant penalty algorithm versions is presented in Fig. 6. Compared to Fig. 7, where both the constant penalty algorithm versions produce the same route, the changeable penalty algorithm shortens the route by 3.7837 %, which is quite considerable and significant improvement. A more intricate example is presented in Fig. 8, where the shortest route through 50 ports is found at an increasing tour constraint violation penalty as well. Compared to Fig. 9, showing the shortest route found by v_1, the changeable penalty algorithm shortens the route by 11.1689 %. Moreover, algorithm version v_{100} seeming to be a slightly more robust than v_1 produces a longer route (Fig. 10). Its length is 15.5299 % greater than that in Fig. 8.

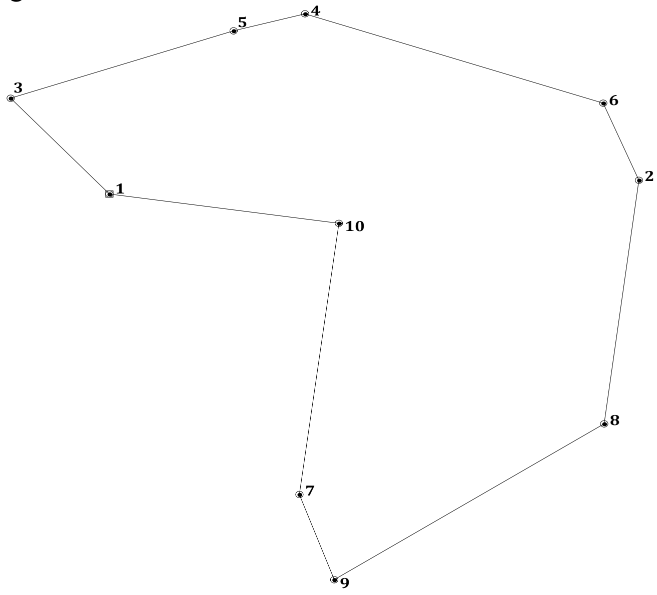

Fig. 6. The v_α-solution of an instance with 10 ports by $\alpha = 1.008$, where $\tilde{\rho}^*_\Sigma = 139.5179$

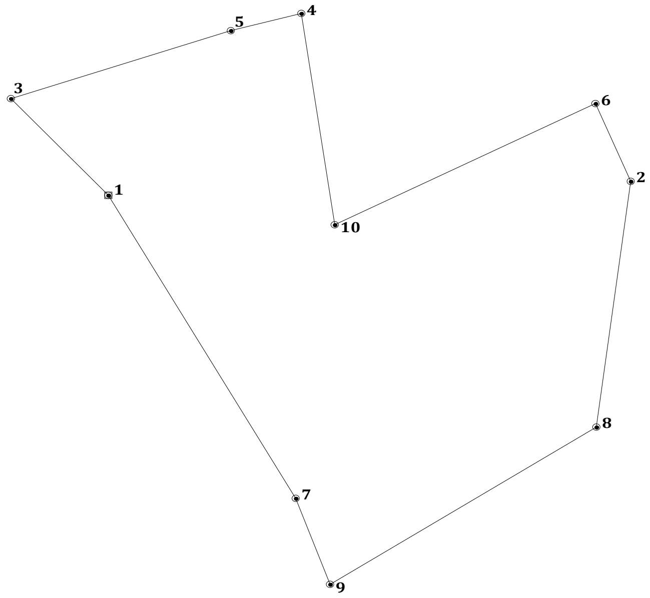

Fig. 7. The worse solution of the instance with 10 ports in Fig. 6 by v_1 and v_{100}, where $\tilde{\rho}^*_\Sigma = 145.0045$

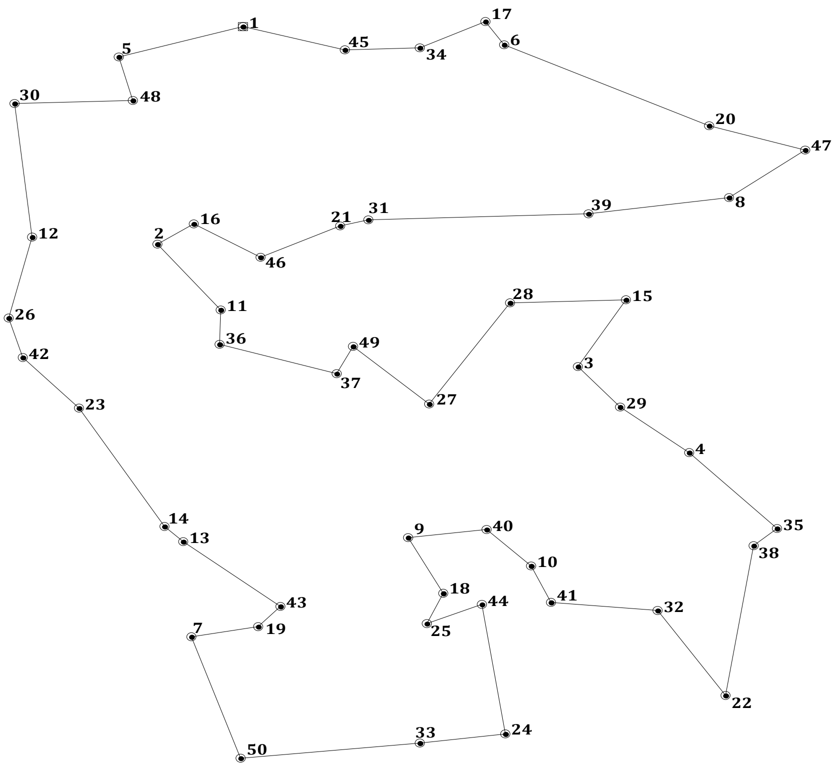

Fig. 8. The v_α-solution of an instance with 50 ports by $\alpha = 1.003$, where $\tilde{\rho}^*_\Sigma = 284.7905$

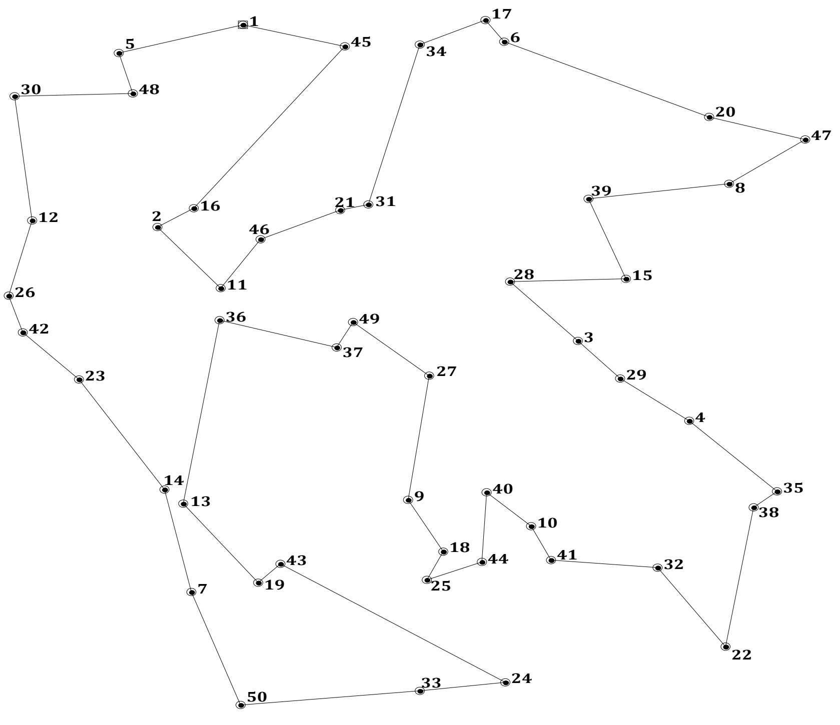

Fig. 9. The v_1-solution of the instance with 50 ports in Fig. 8, where $\tilde{\rho}^*_\Sigma = 320.5976$

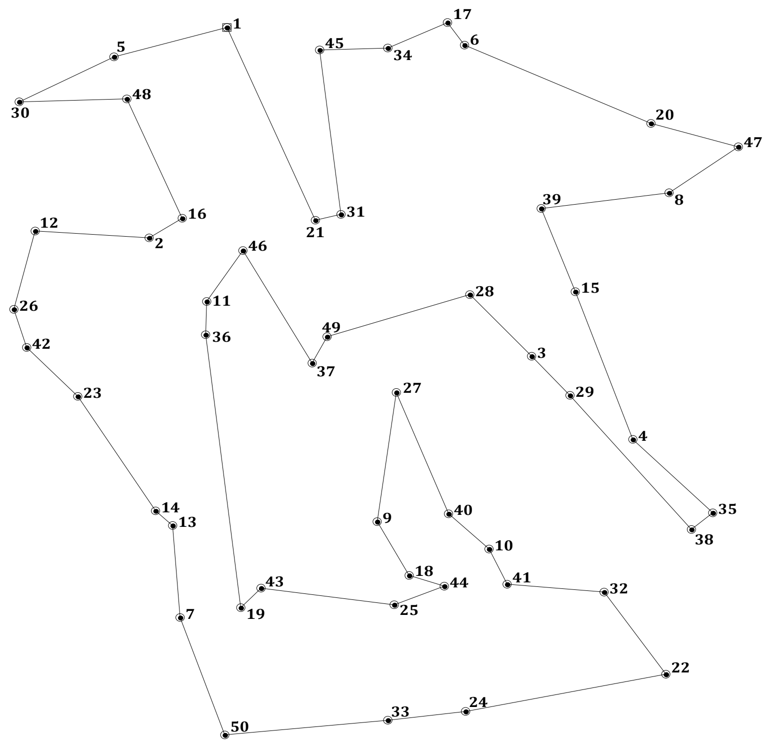

Fig. 10. The v_{100}-solution of the instance with 50 ports in Fig. 8, where $\tilde{\rho}^*_{\Sigma} = 337.1495$

It is noteworthy that the results reported in Tables 1–3 and Figs. 3–5 are statistically reliable, i. e. they are approximately repeatable for other pseudorandom number generator seeds. Thus, in another series of 900 instances, the overall average percentages from Table 1 first row are now 73.6667 %, 75.7778 %, 21 % (there have been obtained 189 routes by v_1 whose lengths are shorter than lengths by v_{100}), respectively. So, v_α is indeed "better than" v_1 and v_{100} in about 75 % of the modeled instances. The overall average percentages from Table 2 first row are now 0.1111 % (once again only one of those 900 route lengths produced by v_α has appeared to be longer than the respective route length produced by v_1), 0.8889 % (the difference is so big due to a few instances where v_α is "worse than" v_{100}), 16.8889 %, respectively. Just like in the first series, the only instance whose v_α-route is longer than v_1-route has appeared to be of 50 ports, where the v_α-route length is 272.0435 and the v_1-route length is 271.7023 (the difference in the first series is even smaller). Eventually, the overall average percentages from Table 3 first row are now 26.2222 %, 23.3333 %, and 62.1111 %, respectively, being really close to those ones in Table 3.

A very specific property of the genetic algorithm is that its result as the shortest route length depends on the pseudorandom number generator state seeded at the beginning of a test. Hence, we try re-running v_α initialized with the same pseudorandom number generator seed. By this set-up of the test, the shortest route length does depend on whether $\alpha > 1$ or $\alpha < 1$. Nevertheless, this dependence is weak: 769 instances have been v_α-solved by $\alpha = 1.01$ (the starting value for the increasing penalty rate), whereas just 99 instances have been v_α-solved by

$\alpha = 0.999945$ (the starting value for the decreasing penalty rate). Similarly to Tables 1–3, the comparison to v_1 and v_{100} is presented in Tables 4–6. These tables clearly show that the advantage of v_α is not that big, if any.

Table 4. The "better than" percentages for the same pseudorandom number generator seed test

		v_α **better than** v_1	v_α **better than** v_{100}	v_1 **better than** v_{100}
Overall average		42.5556	45.6667	21
N	10	11	15	10
	15	32	32	6
	20	37	37	8
	25	48	49	17
	30	46	48	25
	35	50	50	24
	40	52	63	28
	45	50	62	35
	50	57	55	36

Table 5. The "worse than" percentages for the same pseudorandom number generator seed test

		v_α **worse than** v_1	v_α **worse than** v_{100}	v_1 **worse than** v_{100}
Overall average		40.5556	37.3333	16.8889
N	10	6	4	4
	15	27	28	9
	20	47	45	9
	25	46	44	12
	30	51	47	15
	35	47	48	17
	40	48	37	26
	45	50	38	25
	50	43	45	35

Table 6. The "equal route lengths" percentages for the same pseudorandom number generator seed test

		v_α **and** v_1	v_α **and** v_{100}	v_1 **and** v_{100}
Overall average		16.8889	17	62.1111
N	10	83	81	86
	15	41	40	85
	20	16	18	83
	25	6	7	71
	30	3	5	60
	35	3	2	59
	40	0	0	46
	45	0	0	40
	50	0	0	29

Table 7 shows what the pure advantage is (when the algorithm version is simultaneously compared to the two other versions). Moreover, the average route length by v_α is 233.5604 (averaged over 900 route lengths), whereas the average route lengths by v_1 and v_{100} are 234.0213 and 234.41296, respectively. Therefore, at least a tiny advantage of v_α does exist, i. e. using the tour constraint penalty discount may indeed shorten the route.

T a b l e 7 . The percentages of performance comparison for the same pseudorandom number generator seed test

	better than the two other versions	**worse than the two other versions**	**not worse than the two other versions**
v_α	36	31.5556	53.6667
v_1	14.1111	12.5556	53.1111
v_{100}	9.6667	15.2222	48.5556

The final test is done for each of algorithm versions v_1, v_{100}, v_α by $s_{\text{fail}}^{(\max)} = 19$ and resetting every re-run with a new pseudorandom number generator seed being the same for v_1, v_{100}, v_α. This is the purest experiment, where every instance is solved at least 20 times (and there are 19 attempts to shorten the very first route length). It is v_α-solved for 10 times by the increasing penalty and 10 times by the decreasing penalty, unless a shorter route is found by a decreased α. Now, the performance comparison similar to Table 7 is presented in Table 8. It is clearly seen that the purest advantage of v_α does exist (because there have been v_α-found 33 routes, which is 3.6667 %, each shorter than any of 20 routes by v_1 and any of 20 routes by v_{100} in the respective 33 instances). However, the purest advantage of v_{100} seems to be stronger. Amazingly enough, there have been v_α-found 134 routes (14.8889 %) each shorter than any of 20 routes by v_{100} in the respective 134 instances. Contrariwise, there have been v_{100}-found 143 routes (15.8889 %) each shorter than any route by v_α in the respective 143 instances. In other words, the algorithm version using the tour constraint penalty discount has an efficiency comparable (roughly speaking, almost the same) to the efficiency of the algorithm version using the high constant penalty. The respective percentages for v_α and v_1 are 8.3333 % (v_α outperforms v_1) and 11.2222 % (v_1 outperforms v_α).

T a b l e 8 . The percentages of performance comparison for the purest experiment

	better than the two other versions	**worse than the two other versions**	**not worse than the two other versions**
v_α	3.6667	6.1111	79
v_1	7.4444	5.8889	81.8889
v_{100}	12.2222	12.5556	80.2222

Consequently, the algorithm using the changeable penalty sometimes outperforms the algorithm using the constant penalty. Although this occasion is not very likely, there are expectedly about one problem of 25, when the changeable penalty algorithm will produce a shorter route than routes by the constant penalty algorithm versions. The latter can outperform as well, with slightly higher likelihoods.

The final test has revealed another interesting peculiarity. Among those 900 instances, 215 shortest routes by v_{α} have been found by $\alpha = 1.01$, 88 shortest v_{α}-routes have been found by $\alpha = 1.009$, and 68 shortest v_{α}-routes have been found by $\alpha = 1.008$ (the two nearest values to 1.01). The distribution resembles that one in Fig. 4. The ratio of the number of instances solved by $\alpha > 1$ to the number of instances solved by $\alpha < 1$ is about 2. Consequently, the increasing penalty has its one advantage over the decreasing penalty, but still the latter is "needed" roughly in every third problem solved by the changeable penalty algorithm.

DISCUSSION OF THE CONTRIBUTION

The experiment with controllable seed for generating maritime cargo delivery problem instances and for randomly generating tours (16) for (17) has shown that the changeable penalty, along with the constant penalty, is an important parameter of the genetic algorithm. It has been also revealed that the algorithm output depends on the seed. It is unclear how the best value of α could be selected. Moreover, it is impossible to foresee that the constant penalty algorithm version will not be outperformed by the changeable penalty algorithm. Therefore, the best decision is to use both the constant and changeable penalty versions (say, by running them on parallel processor cores), whereupon the shortest route length is trivially selected. In our case of study, we propose to run simultaneously four versions: v_1, v_{100}, v_{α} by $\alpha = 1.01$, and v_{α} by $\alpha = 0.999945$ (here the penalty has the slowest descent; during the starting few thousand iterations, it decreases almost linearly).

An example of how the suggested changeable penalty improves the genetic algorithm is presented in Figs. 11–13. A maritime cargo delivery problem with 45 ports is solved, starting with the same pseudorandom number generator seed, by using the low constant penalty by (22), the high constant penalty by (23), and the increasing penalty with $\alpha = 1.01$. As we can see, the high constant penalty has improved the low constant penalty route length by just 0.08 % ($\tilde{\rho}_{\Sigma}^{*} = 300.90902$ in Fig. 12 against $\tilde{\rho}_{\Sigma}^{*} = 301.1532$ in Fig. 11). The improvement by the increasing penalty is far more significant: it is 3.8194 % ($\tilde{\rho}_{\Sigma}^{*} = 289.4161$ in Fig. 13) compared to $\tilde{\rho}_{\Sigma}^{*} = 300.90902$ in Fig. 12, and it is 3.8974 % compared to $\tilde{\rho}_{\Sigma}^{*} = 301.1532$ in Fig. 11.

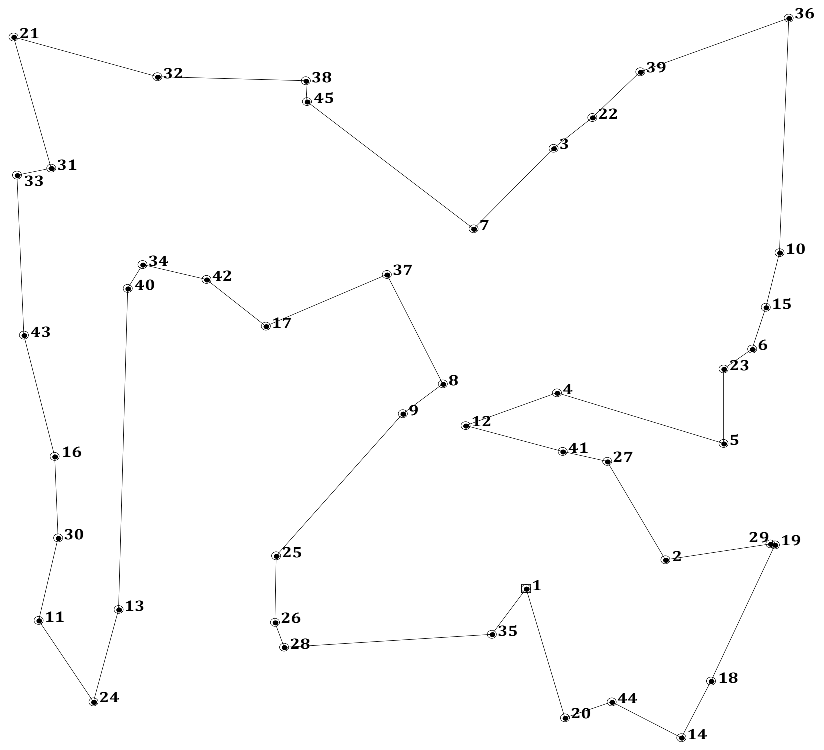

Fig. 11. The v_1-solution of an instance with 45 ports, where $\tilde{\rho}_{\Sigma}^{*} = 301.1532$

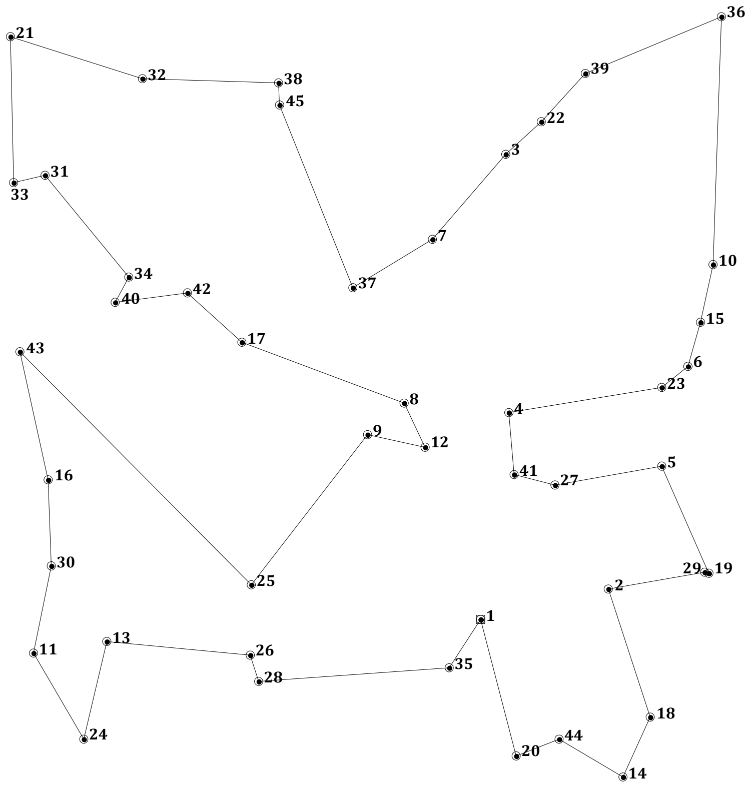

Fig. 12. The v_{100}-solution of the instance with 45 ports in *Fig. 11*, where $\tilde{\rho}_{\Sigma}^{*} = 300.90902$

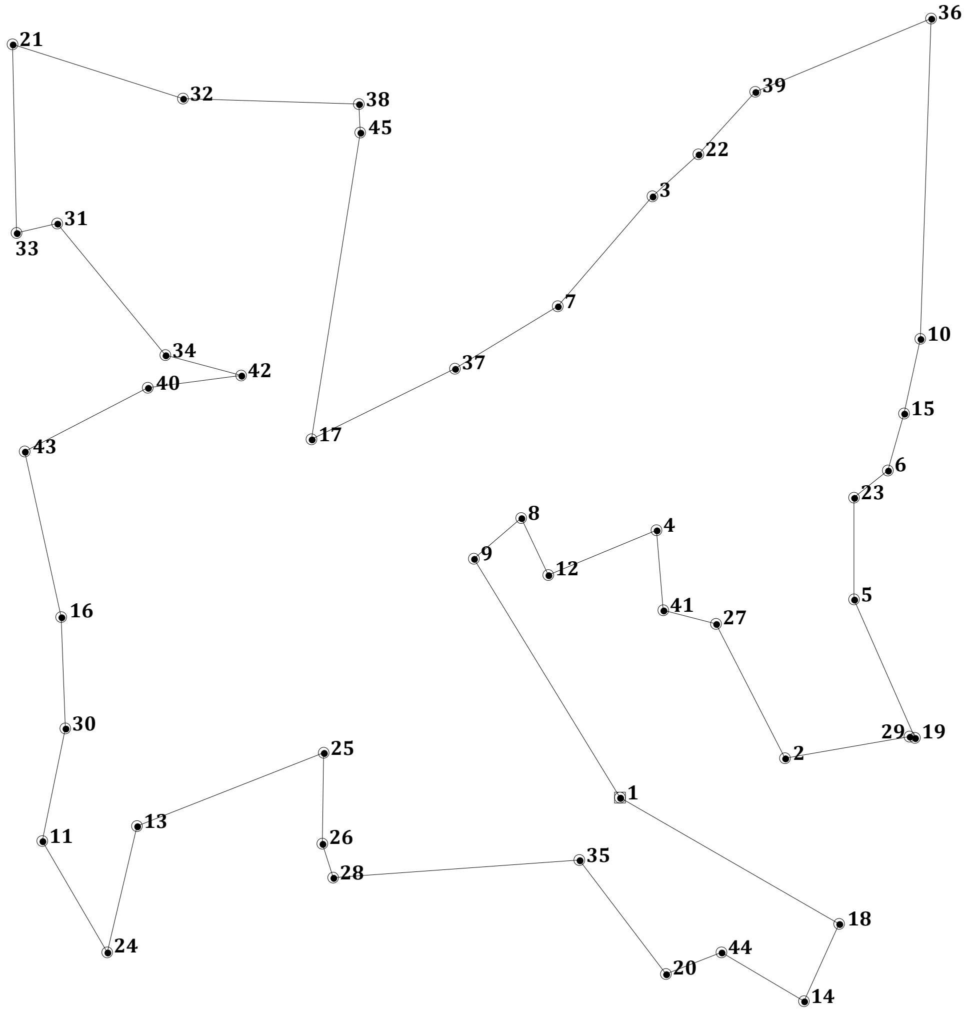

Fig. 13. The v_{α}-solution ($\alpha = 1.01$) of the instance with 45 ports in Fig. 11, where $\tilde{\rho}^{*}_{\Sigma} = 289.4161$

Despite the example in Figs. 11–13 is a good demonstration of that the changeable penalty is a real improvement of the genetic algorithm, a counterexample is easily generated just on the same maritime cargo delivery problem by re-running the constant penalty algorithm versions and v_{α} (a re-run implies that the pseudorandom number generator state is changed). Thus, in a series of 302 re-runs, the shortest route length by v_1 varies between 272.8407 and 323.0998, and the shortest route length by v_{α} varies between the same lower and upper boundaries. The shortest route length by v_{100} varies between 272.8407 (the same lower boundary) and 317.6751, so its upper boundary is less than that for v_1 and v_{α}.

Nevertheless, as the maximally possible number of feeders is increased, the suggested changeable penalty further improves the genetic algorithm. In this case, we have

$$d_{\max} = \psi\left(1.25 \cdot \psi\left(\frac{1}{M_{\max}} \cdot \max_{k=\overline{1,\,N}} \left\{ \sum_{j=1}^{N} \rho(k,\,j) \right\}\right)\right)$$

and (21), i. e. the longest and shortest possible tours of the feeder are shortened. For the same instance with $M_{\max} = 6$, the performance of v_{α} in a series of 738 re-runs is significantly better. The lower and upper boundaries in this case are 274.1295 and 321.0341 for v_1, 274.70303 and 317.0557 for v_{100}, 272.8407 and 323.7964 for v_{α}, where the shortest route length $\tilde{\rho}^*_{\Sigma} = 272.8407$ is found in a re-run with $\tilde{\rho}^*_{\Sigma} = 288.4775$ (it is even shorter than that in Fig. 13) by v_1 and $\tilde{\rho}^*_{\Sigma} = 297.49702$ by v_{100}. Furthermore, in this "local" test series v_{α} is better than the two other versions in 27.9133 % and is worse than the two other versions in 23.9837 % of all re-runs. These rates are 3.9735 % and 1.9868 % for the series with $M_{\max} = 2$. By the way, the performance of the high constant penalty algorithm with $M_{\max} = 6$ significantly drops: the average route length is 290.4455 within the 302 re-runs with $M_{\max} = 2$, and it is 295.3313 with $M_{\max} = 6$.

At last, it is important to note that all the generated instances in our testings have been such that, in the solution returned by the algorithm, only one feeder was required to cover the shortest route ($M^* = 1$). This is practically possible and applicable due to we set the longest possible tour of the feeder at a relatively high value. Even though the shortened route length by the suggested improvement in the genetic algorithm is small, it is a significant decrement of the maritime cargo delivery cost.

CONCLUSION

We have presented a tour constraint violation penalty to the genetic algorithm for solving a maritime cargo delivery problem formulated as a multiple traveling salesman problem. The penalty is an exponential function of the iteration, where we maintain the possibility of the penalty rate to be either increasing or decreasing whose steepness is controlled by a positive parameter α. Our tests have shown that the changeable penalty algorithm may return shorter routes, although the constant penalty algorithms cannot be neglected. Therefore, our contribution to the field of genetic algorithms is the monotonous flexibility of the tour constraint violation penalty. The usefulness of this flexibility grows as the longest possible tour of the feeder is shortened. It is so due to a penalty discount is required either at the beginning or at the end of the algorithm run to improve selectivity of the best feeder tours. In optimizing maritime cargo delivery, we propose to run the genetic algorithm by the low and constant penalties along with the increasing and decreasing penalties, whereupon the solution is the minimal value of the four route lengths. In addition, we recommend the four algorithm versions to be initialized by four different pseudorandom number generator states. Although the gain is just a few percent (by which the route length is shortened) or less, it is a substantial reduction of expenses for maritime cargo delivery.

The research should be extended and advanced in the way of studying the pseudorandom number generator state influence. As we have revealed that the state influences the algorithm output, it is to ascertain lower and upper boundaries of the route length. Another open question is how many re-runs should be made (by changing the state) to obtain the shortest possible route in a maritime cargo delivery problem.

REFERENCES

1. T.R. Walker et al., "Chapter 27 — Environmental Effects of Marine Transportation," in *World Seas: an Environmental Evaluation*. Cambridge, Massachusetts, USA: Academic Press, 2019, pp. 505–530. Available: https://doi.org/10.1016/B978-0-12-805052-1.00030-9
2. W. Li, R. Pundt, and E. Miller-Hooks, "An updatable and comprehensive global cargo maritime network and strategic seaborne cargo routing model for global containerized and bulk vessel flow estimation," *Maritime Transport Research*, vol. 2, 100038, 2021. Available: https://doi.org/10.1016/j.martra.2021.100038
3. C. Archetti, L. Peirano, and M. G. Speranza, "Optimization in multimodal freight transportation problems: A Survey," *European Journal of Operational Research*, vol. 299, iss. 1, pp. 1–20, 2022. Available: https://doi.org/10.1016/j.ejor.2021.07.031
4. X. Wu, J. Lu, S. Wu, and X. Zhou, "Synchronizing time-dependent transportation services: Reformulation and solution algorithm using quadratic assignment problem," *Transportation Research Part B: Methodological*, vol. 152, pp. 140–179, 2021. Available: https://doi.org/10.1016/j.trb.2021.08.008
5. P.A. Miranda, C.A. Blazquez, C. Obreque, J. Maturana-Ross, and G. Gutierrez-Jarpa, "The bi-objective insular traveling salesman problem with maritime and ground transportation costs," *European Journal of Operational Research*, vol. 271, iss. 3, pp. 1014–1036, 2018. Available: https://doi.org/10.1016/j.ejor.2018.05.009
6. D.-Z. Du and P.M. Pardalos, *Handbook of Combinatorial Optimization*. New York, NY, USA: Springer, 1998, 2406 p. Available: https://doi.org/10.1007/978-1-4613-0303-9
7. A. Hertz and M. Widmer, "Guidelines for the use of meta-heuristics in combinatorial optimization," *European Journal of Operational Research*, vol. 151, iss. 2, pp. 247–252, 2003. Available: https://doi.org/10.1016/S0377-2217(02)00823-8
8. A. Colorni, M. Dorigo, F. Maffioli, V. Maniezzo, G. Righini, and M. Trubian, "Heuristics from nature for hard combinatorial optimization problems," *International Transactions in Operational Research*, vol. 3, iss. 1, pp. 1–21, 1996. Available: https://doi.org/10.1016/0969-6016(96)00004-4
9. L.D. Chambers, *The Practical Handbook of Genetic Algorithms*. Chapman and Hall/CRC, 2000, 544 p.
10. R. L. Haupt and S.E. Haupt, *Practical Genetic Algorithms*. John Wiley & Sons, 2003, 253 p. Available: https://doi.org/10.1002/0471671746
11. C. Thornton, F. Hutter, H.H. Hoos, and K. Leyton-Brown, "Auto-WEKA: Combined selection and hyperparameter optimization of classification algorithms," in *KDD'13: Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 847–855, 2013.
12. V.V. Romanuke, "Optimal training parameters and hidden layer neurons number of two-layer perceptron for generalized scaled objects classification problem," *Information Technology and Management Science*, vol. 18, pp. 42–48, 2015. doi: 10.1515/itms-2015-0007
13. E. Merhej, S. Schockaert, and M. De Cock, "Repairing inconsistent answer set programs using rules of thumb: A gene regulatory networks case study," *International Journal of Approximate Reasoning*, vol. 83, pp. 243–264, 2017. Available: https://doi.org/10.1016/j.ijar.2017.01.012
14. A. Santini, A. Viana, X. Klimentova, and J.P. Pedroso, "The probabilistic travelling salesman problem with crowdsourcing," *Computers & Operations Research*, vol. 142, 105722, 2022. Available: https://doi.org/10.1016/j.cor.2022.105722
15. A. Király and J. Abonyi, "Redesign of the supply of mobile mechanics based on a novel genetic optimization algorithm using Google Maps API," *Engineering Applica-*

tions Of Artificial Intelligence, vol. 38, pp. 122–130, 2015. Available: https://doi.org/10.1016/j.engappai.2014.10.015

16. L. Kota and K. Jarmai, "Mathematical modeling of multiple tour multiple traveling salesman problem using evolutionary programming," *Applied Mathematical Modelling*, vol. 39, iss. 12, pp. 3410–3433, 2015. Available: https://doi.org/10.1016/j.apm.2014.11.043
17. V. V. Romanuke, "Job order input for efficient exact minimization of total tardiness in tight-tardy progressive single machine scheduling with idling-free preemptions," *Scientific Papers of O. S. Popov Odessa National Academy of Telecommunications*, no. 1, pp. 19–36, 2020. Available: https://doi.org/10.33243/2518-7139-2020-1-1-19-36

Received 04.07.2022

INFORMATION ON THE ARTICLE

Vadim V. Romanuke, ORCID: 0000-0001-9638-9572, Vinnytsia Institute of Trade and Economics of State University of Trade and Economics, Ukraine, e-mail: romanukevadimv@gmail.com

Andriy Y. Romanov, ORCID: 0000-0002-1714-3310, Odessa National Maritime University, Ukraine, e-mail: andreygorogogo@ gmail.com

Mykola O. Malaksiano, ORCID: 0000-0002-4075-5112, Odessa National Maritime University, Ukraine, e-mail: malaksiano@gmail.com

ПОКРАЩЕННЯ ГЕНЕТИЧНОГО АЛГОРИТМУ НА ОСНОВІ ДИСКОНТУ ШТРАФУ ЗА ПОРУШЕННЯ ОБМЕЖЕНЬ РЕЙСУ ДЛЯ МОРСЬКОЇ ДОСТАВКИ ВАНТАЖІВ / В.В. Романюк, А.Ю. Романов, М.О. Малаксіано

Анотація. Розглянуто задачу мінімізації вартості морської доставки вантажів. Ця вартість еквівалентна сумі довжин рейсів фідерів, що використовуються для доставки. Задача формулюється у формі задачі декількох комівояжерів. Для знаходження розв'язку у формі найкоротшого маршруту, що складається з рейсів фідерів, використовується генетичний алгоритм, у якому дві нерівності, котрі обмежують довжину рейсу кожного фідера до інтервалу між найкоротшою та найбільшою довжинами. Окрім сталого штрафу за порушення обмежень рейсу у генетичному алгоритмі запропоновано змінюваний штраф у формі експоненціальної функції ітерації алгоритму, де залишається можливість як зростаючого, так і спадного штрафу, чия крутизна контролюється деяким додатним параметром. Тести показують, що алгоритм зі змінюваним штрафом може повертати коротші маршрути, хоча алгоритми зі сталими штрафами не можуть бути відкинуті. Зі скороченням найдовшого рейсу фідера змінюваний штраф стає більш корисним завдяки тому, що деякий дисконт штрафу потрібний на початку або наприкінці прогону алгоритму задля покращення селективності найкращих рейсів фідерів. Для оптимізації морської доставки вантажів запропоновано запускати даний генетичний алгоритм за низького та високого штрафів разом зі зростаючим та спадаючим штрафами, після чого розв'язком є мінімальне значення з чотирьох відповідних довжин маршрутів. Рекомендовано ініціалізувати ці чотири версії алгоритму чотирма різними станами генератора псевдовипадкових чисел. Очікуваний виграш складає декілька відсотків скорочення довжини маршруту, але для морської доставки вантажів це є значним скороченням витрат.

Ключові слова: морська доставка вантажів, довжина рейсу, генетичний алгоритм, штраф за порушення обмежень рейсу, дисконт штрафу.