Hindawi
Discrete Dynamics in Nature and Society
Volume 2021, Article ID 4533483, 14 pages
https://doi.org/10.1155/2021/4533483

Research Article

A Colored Traveling Salesman Problem with Varying City Colors

Xianghu Meng,[1,2] **Jun Li**,[2,3] **and MengChu Zhou**[4,5]

[1]*Anhui Provincial Key Laboratory of Power Electronics, and Motion Control, Anhui University of Techonlogy, Maanshan 243032, China*
[2]*Ministry of Education Key Laboratory of Measurement and Control of Complex Engineering Systems, Southeast University, Nanjing 210096, China*
[3]*Southeast University Shenzhen Research Institute, Shenzhen 518063, China*
[4]*Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark 07102, NJ, USA*
[5]*The Institute of Systems Engineering and Collaborative Laboratory for Intelligent Science and Systems, Macau University of Science and Technology, Macau 999078, China*

Correspondence should be addressed to Xianghu Meng; xdmengxianghu@163.com

Received 11 August 2021; Accepted 5 November 2021; Published 9 December 2021

Academic Editor: Shi Cheng

A colored traveling salesman problem (CTSP) is a path optimization problem in which colors are used to characterize diverse matching relationship between cities and salesmen. Namely, each salesman has a single color while every city has one to multiple salesmen's colors, thus allowing salesmen to visit exactly once the cities of their colors. It is noteworthy that cities' accessibilities to salesmen may change over time, which usually takes place in the multiwarehouse distribution of online retailers. This work presents a new CTSP with dynamically varying city colors for describing and modeling some scheduling problems with variable city accessibilities. The problem is more complicated than the previously proposed CTSP with varying edge weights. In particular, the solution feasibility changes as the cities change their colors, that is, a feasible original solution path may become no longer feasible after city colors change. A variable neighborhood search (VNS) algorithm is presented to solve the new problem. Specifically, a dynamic environment simulator with an adjustable frequency and amplitude is designed to mimic such color changes. Then, direct-route encoding, greedy initialization, and appropriate population immigrant are proposed to form an enhanced VNS, and then its performance is evaluated. The results of extensive experiments show that the proposed VNS can quickly track the environmental changes and effectively resolve the problem.

1. Introduction

A colored traveling salesman problem (CTSP) generalizes the well-known multiple traveling salesman problem [1]. Each salesman in it has a single color, and each city has one to multiple salesmen's colors. A city allows only one salesman of the same color to visit exactly once. Colors are used to characterize the cities' accessibility for salesmen and can be applied to some real-world scheduling problems in which the tasks to be performed own different accessibilities for movable processing devices [2]. Li and Meng apply the CTSP to schedule multibridge machining systems for avoiding collision [3, 4]. In addition, He and Hao [5] propose a two-phase local search for solving the CTSP. Recently, three CTSP variants have been proposed, that is, large-scale general CTSP [6], biobjective CTSP [7], and precedence-constrained CTSP [8]. They can be applied to many engineering systems, for example, multibridge machining systems [4], robot systems [9], and multitrip pickup and delivery systems [10].

However, these mentioned CTSPs are static in terms of connection weights and task accessibility. In practice, many optimization problems, such as job scheduling, assembly, transportation scheduling, and production planning, involve dynamically changing environments. Particularly, the complex networks where the optimization problems arise, for example, the communication networks [11–14] and intelligent transportation networks [15–18], have the changing connection weights among nodes and even the network topologies themselves over time.

The dynamics of a dynamic TSP (DTSP) [19] reflects the changes in the topology [20] and connection weights [21] of a network to be modeled. DTSP is widely used in practice. Tinós

[22] analyzes the effects of the edge weight changes on the fitness landscapes of TSP. Recently, Mavrovouniotis et al. [23] investigate a memetic ant colony optimization algorithm with local search to address symmetric and asymmetric DTSPs, and the experimental results show the efficiency of their proposed algorithm. Chen et al. [16] investigate the value of choosing the next stop to visit in a multistop trip based on current traffic conditions to minimize the expected total travel time of the tour, and they model this problem as a Markov decision process. In dynamic scenarios, datasets (city coordinate information) are generally incomplete and variable. Baykasoğlu and Durmuşoğlu [24] address a DTSP in which cities may be added to/ deleted from the city domain with a multiagent-based approach.

In logistics and distribution, traffic network topology and customer demands are changing over time. If some practical constraints in logistics distribution are introduced into DTSP, it becomes a dynamic vehicle routing problem (DVRP) [25]. DVRP requires solving not only a dynamic path planning problem with varying edge weights but also the problem with some uncertain constraints in real delivery. It is more complex than the DTSP.

DVRP can be divided into two categories according to the customer's location information. One is those with known customer locations, but a traffic network topology varies and customer demands are random. Kim et al. [26] propose a DVRP model with nonstationary stochastic travel time under traffic congestion, and a Markov decision process model is adopted to solve this problem. A vehicle routing problem with stochastic demands (VRPSDs) is the one in which customer locations are known, but the demands of customers are random. For example, a flexible solution method is adopted to transform VRPSD into a small set of capacitated VRP, where Monte Carlo simulation with statistical distributions is used to estimate customer demands [27]. Mavrovouniotis et al. [23] present a DVRP with stochastic demands where customers arrive dynamically and randomly, and thereby their requests are not completely known. To solve it, an ant algorithm with an immigrant scheme is developed. Customer service time may be uncertain in a DVRP. Vonolfen and Affenzeller [28] resolve a dynamic pickup and delivery problem with time windows by using a novel waiting for heuristic strategy based on historical request information. Juan et al. [27] propose a method to estimate truck arrival time at each customer location for truck route planning in nonstationary stochastic networks.

Another DVRP category is the real-time or online VRP, that is, new customer orders are randomly generated, and both customer locations and demands are unknown. In this situation, it is necessary to reschedule vehicles based on previous schedules, that is, redistribute these vehicles for all not-served customers. Many scholars have studied the rule of accepting or rejecting new service requirements in a DVRP [29–31]. Bopardikar et al. [32] investigate a DVRP with the Poisson distribution orders and design routing policies for a service vehicle to maximize the number of demands at the steady state. Pavone et al. [33] propose a distributed and adaptive segmentation algorithm to solve a DVRP and validate its effectiveness.

The above studies focus on path optimization over traffic networks with varying edge weights and all the cities with the same accessibility. So far, there is no research on the optimization problem with dynamic city accessibilities, which can well describe a supply and distribution problem faced by online retailers. Therefore, this work focuses on delivery route optimization for online retail with multiwarehouse and dynamic task accessibility.

For example, aiming at delivering goods faster and better, online retailers tend to build their own logistics system consisting of widely located warehouses in different countries or regions containing goods of different types and stocks. Amazon owns a total of 90 warehouses in the world, JD Company has built 143 warehouses in 43 cities of China, and Suning Logistics Group has 12 large-scale warehouses in China.

Once random customer orders are generated, an online retailer has to provide the customers with their ordered goods. The stocks of the goods differ at distinct warehouses and are constantly changing with sales and supplies. Therefore, decision-making is needed to decide which warehouse should provide the goods to a specific customer, and optimize the delivery routes of the different goods to different customers to meet customer requirements and reduce the supply cost. So far, there is no satisfying decision-making mechanism for this kind of online retail.

Suppose that each warehouse has one unique color. It can be considered as a salesman with the color to deliver goods for customers by its own transport system. Those online ordering customers, located in different areas, can be provided with the ordered goods stocked at one of the different warehouses. In other words, such a customer carries the warehouses' colors, and thereby the customers can be regarded as the cities with one to multiple warehouses' colors. Thus, the warehouses' supply problem can be described as a CTSP. With the customers' online purchases, each warehouse's stock changes over time. When a commodity ordered by a customer is out of stock in a warehouse, the customer loses his/her color. This means that a customer whose warehouses' colors will change in the sales process.

Therefore, we propose a new CTSP with varying city colors (CTSP-VCC). It can be applied to supply decision and delivery route optimization for online retail with multiwarehouse and varying task accessibility.

Considering the outstanding performance of Variable Neighborhood Search (VNS) [2] in solving static CTSP, we adopt it to solve CTSP-VCC. To overcome its instability under a dynamic circumstance, greedy initialization and best population immigrant are incorporated into it to maintain its fast convergence and solution diversity.

This work intends to make the following contributions:

(1) It formulates a CTSP-VCC in which city colors are changing over time. CTSP-VCC can be applied to supply decision and delivery route optimization for a online retail with multiwarehouse.

(2) It constructs a rigorous mathematical optimization problem for CTSP-VCC and designs a dynamic environment simulator to mimic the change of city colors.

(3) It develops an improved VNS to solve CTSP-VCC in which a greedy operation and a novel population immigrant scheme are adopted to achieve better convergence and solution quality.

The remainder of this paper is organized as follows: Section 2 introduces CTSP-VCC. Section 3 presents a greedy-initialization VNS with a best population immigrant scheme. Section 4 shows simulation results. Section 5 concludes the work.

2. Problem Formulation

CTSP uses the colors to characterize city accessibility to salesmen [1, 2]. It owns m salesmen and n cities, where $m < n$, $m, n \in Z = \{1, 2, \ldots\}$. It can be formulated over a digraph $G = (V, E)$ with vertex set $V = \{1, 2, \ldots, n\}$ representing n cities; each edge $(i, j) \in E$ and $i \neq j$ represents a visit cost between cities i and j, associated with weight ω_{ij}. Salesman $k \forall k \in Z_m = \{1, 2, \ldots, m\}$ is allocated with color c_k. A city color matrix (CCM) M is defined to indicate the accessibility of cities to salesmen.

$$M = \left[a_{ij}\right]_{n\times m} = \begin{array}{c} \\ v_1 \\ v_2 \\ \vdots \\ v_n \end{array} \begin{array}{c} \begin{array}{cccc} c_1 & c_2 & \cdots & c_m \end{array} \\ \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \ldots & a_{nm} \end{bmatrix}_{n\times m} \end{array}, \quad (1)$$

where a row vector represents the color set allocated to city i, $c(i) = \{c_j | a_{ij} = 1, j \in Z_m\}$, $i \in V$. Clearly, $1 \leq |c(i)| \leq m$, namely, city v_i allows at least one salesman and at most m salesmen to visit. The j-th column vector in M expresses which cities in the city set can be visited by salesman j. The accessible city set with respect to salesman j is denoted by $I_j = \{i | a_{ij} = 1, i \in V\}$, $j \in Z_m$. In addition, we define the set of single-colored cities $T_j = \{i | a_{ij} = 1, \sum_j a_{ij} = 1, i \in V\}$ to collect the cities that carry the same color c_j, $j \in Z_m$.

Here, we define a dynamic CTSP whose city colors are changing over time. CCM in CTSP-VCC is not a constant matrix, that is, a_{ij}, $i \in V$, $j \in Z_m$, becomes a function $a_{ij}(t)$ that varies over time. Let $a_{ij}(t) = 1$, if city i has color c_j at time t; otherwise $a_{ij}(t) = 0$.

Next, a 0-1 integer programming model of CTSP-VCC is given as follows: the binary access variables $x_{ijk} = 1, i \neq j$, $\forall i, j \in V$, if salesman $k \in Z_m$ passes through edge (i, j); otherwise, $x_{ijk} = 0$.

$$\text{Minimize } f = \sum_k \sum_i \sum_j \omega_{ij} x_{ijk}, \quad \forall k \in Z_m, \forall i, j \in V. \quad (2)$$

Subject to:

Each salesman k is required to start from and return to depot $d_k \in V$. The equation is as follows:

$$\begin{aligned} &\sum_i x_{d_k ik} = 1, \\ &\sum_i x_{id_k k} = 1, \quad \forall i \in I_k \backslash \{d_k\}, \forall k \in Z_m. \end{aligned} \quad (3)$$

Salesman k cannot visit a city whose color set does not includec_k:

$$\begin{aligned} &\sum_i \sum_j x_{ijk} = 0, \\ &\sum_i \sum_j x_{jik} = 0, \quad \forall i \in V, \forall j \in V \backslash I_k, \forall k \in Z_m. \end{aligned} \quad (4)$$

Each city can be visited exactly once by one and only properly colored salesman:

$$\begin{aligned} &\sum_i \sum_k x_{ijk} = 1, \\ &\sum_i \sum_k x_{jik} = 1, \quad j \neq i, \forall i, j \in I_k, \forall k \in Z_m. \end{aligned} \quad (5)$$

A salesman may visit a multicolor city, and a pair of entry and exit is required if it is the case:

$$\sum_l x_{jlk} = \sum_i x_{ijk}, \quad i \neq j \neq l, \forall j \in V, \forall l \in V, \forall i \in V. \quad (6)$$

Any solution consisting of several disconnected subtours for a salesman must be forbidden:

$$u_{ik} - u_{jk} + n \times x_{ijk} \leq n - 1, \quad j \neq i, \forall i, j \in I_k \backslash \{d_k\}, \forall k \in Z_m, \quad (7)$$

where u_{ik} is the number of nodes visited in the tour of salesman k from $d_k$to the node i.

Lemma 1. *CTSP is a special case of CTSP-VCC.*

Proof. In CCM, $a_{ij}(t)$ denotes the variable color of city in CTSP-VCC. If $a_{ij}(t)$ is an invariant constant (i.e., $a_{ij}(t) = a_{ij}(t_0)$), it means that the model turns to be a stationary one, i.e., CTSP. □

Lemma 2. *The solution space size of CTSP-VCC is $\prod_{k=1}^{m} |R_k|!$, where $|R_k|$ denotes the number of cities visited by salesman k, and $\sum_{k=1}^{m} |R_k| = n$.*

Proof. Let R_k denote the route of salesman k. $|R_k|$ represents the number of cities in route k, and it is a variable, $|\Gamma_k| \leq |R_k| \leq |I_k|$. Therefore, the solution space of CTSP-VCC with salesman k is $|R_k|!$. Finally, the solution space of CTSP-VCC with all salesmen is $\prod_{k=1}^{m} |R_k|!$, where $\sum_{k=1}^{m} |R_k| = n$. □

Theorem 1. *CTSP-VCC is NP-hard.*

Proof. CTSP is NP-hard [1], and it has been proved to be a special case of CTSP-VCC by Lemma 1. The solution space size of CTSP-VCC is $\prod_{k=1}^{m} |R_k|!$ given by Lemma 2, and its optimal solution varies over time, that is, with the variable city color $a_{ij}(t)$, the optimal solution varies in its solution space. Therefore, CTSP-VCC is also NP-hard.

For this type of dynamic optimization problem, it is challenging to develop algorithms to track the optimal solution changes. In particular, the solution feasibility changes represent changes in city colors, (i.e., a feasible original solution path may become no longer feasible because of changes in city colors). □

3. Enhanced VNS for CTSP-VCC

VNS has been proved to be an efficient method to address CTSP [2], so we adopt it to resolve CTSP-VCC. Considering the optimal solution of CTSP-VCC is no longer a fixed solution and varies over time, we introduce a best-population immigrant scheme and greedy-initialization method into VNS to enhance global searching ability. Furthermore, given that the varying city colors can change the solution feasibility, we introduce a verification operation into VNS to judge the solution feasibility. Some solutions must be deleted if it is infeasible after changes in city colors.

3.1. Solution Encoding. The direct-route encoding is adopted to encode each salesman's route directly for CTSP-VCC. A coding example of CTSP-VCC with $n = 10$ and $m = 3$ (see Figure 1), namely, Cities 2-8-7-1 and 10-4-3, 9-5-6 are visited by salesmen 1–3, respectively. Let R_k denote the route of salesman k, excluding his depot. The solution is denoted by $R = (R_1, R_2, \ldots, R_m)$. The permutation of cities in the search space size is $|R_k|!$ for salesman k. For all the salesmen in CTSP-VCC, their total solution space's size is $\prod_{k=1}^{m} |R_k|!$, where $\sum_{k=1}^{m} |R_k| = n$.

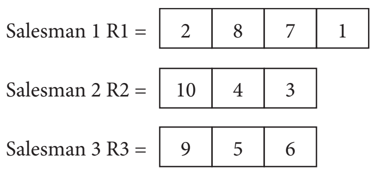

Figure 1: Direct-route encoding for CTSP-VCC.

3.2. Basic VNS. The core thought of VNS is to search a better solution in the neighborhood of a high-quality solution. First, to ensure the population diversity of VNS, a random initialization method is used in VNS to generate initial individuals, that is, all salesmen select randomly the cities carrying the same color to establish their routes.

Then, shaking and local search operations are performed in each generation of VNS, and the current best solution is updated if a better one is obtained. On one hand, Interchange and Relocation are adopted to perturb the current incumbent solution to generate its neighborhood solution set in the shaking operation (see Figures 2–4). The detailed operations of them are given in [2]. On the other hand, a local search explores the neighborhood of the solution selected by shaking to obtain a better one. Particularly, two operations, that is, city-removal and reinsertion operations, carry the deep neighborhood search. Once a better solution is obtained in the local search, it is saved for the next iteration.

In addition, if the environment changes, VNS needs to analyze the feasibility of the original solution. If the solution is feasible and better than the current solutions, it is used to update the best solution; otherwise, it is deleted. The flowchart of basic VNS is given in Figure S1 in the Supplementary Material.

3.3. Improved VNS. A greedy-initialization scheme and a best-population immigrant scheme are introduced to improve the performance of basic VNS. Greedy-initialization means that a two-stage greedy is adopted to generate a better initial solution, that is, the greedy search with the minimum insertion for single color cities first and the multicolor cities next. An immigrant strategy with the best population is used to prevent the premature convergence of VNS. First, the current best individual is selected to perform immigrants after VNS's local search. After the immigrant operation, several new individuals are generated, and they are used to replace those original individuals which are of poor quality. The detailed operations of the best-population immigrant scheme are shown in Algorithm 1.

On one hand, the greedy operation can be used to generate an initial solution of high quality during the initialization of VNS, and a good initial solution can benefit the quick search for a satisfactory solution. On the other hand, the best-population immigrant scheme can prevent premature convergence in a single search direction, which can help VNS to jump out of a local optimum and search more space. The flowchart of greedy ability VNS with a best-population immigrant scheme is illustrated in Figure 5.

3.4. Time Complexity of Improved VNS. We name the improved VNSs as VNS, VNSB, GVNS, and GVNSB. They represent in turn the random-initialized VNS, VNS with the best population immigrant, greedy-initialization VNS, and greedy-initialization VNS with a best population immigrant scheme.

Let g and p denote the generations and population size, respectively, and the time complexity of VNSs are investigated as follows: the greedy-initialization takes $O(|R_1|^2 + \cdots + |R_m|^2) = O(\sum_{k=1}^{m} |R_k|^2)$ time in constructing the initial routs. In VNS, the shaking operation needs $O(n)$ time. The least-cost insertion is adopted in the local search of VNS, and it requires $O(n)$ time. In addition, in the stage of Best-Population Immigrant, the rank operation and 2-opt operation take $O((p)\log(p))$ and $O(\sum_{k=1}^{m} |R_k|^2)$, respectively.

Therefore, we can obtain that GVNSB takes $O(p(\sum_{k=1}^{m} |R_k|^2)) + O(gpn) + O(gpn) + O(gp \log(p)) + O(gp(\sum_{k=1}^{m} |R_k|^2))$. Since $\sum_{k=1}^{m} |R_k|^2 > \sum_{k=1}^{m} |R_k| = n$, the time complexity of GVNSB is $O(gp(\sum_{k=1}^{m} |R_k|^2))$.

4. Experimental Study

4.1. Dynamic Environmental Simulations. The dynamics of CTSP-VCC are reflected by its city colors' change over time. First, let M change over time where$M = [a_{ij}(t)]_{n\times m}$. Then, to describe the change of city colors, each $a_{ij}(t)$ is changed every τ generations during the run of algorithms with the probability ρ, where $a_{ij}(t) = 1$, $0 < \rho < 1$, $0 < i < n$, $0 < j < m$, and $a_{ij}(t_0)$ is the initial value. The environmental dynamics can be easily tuned by two parameters, τ and ρ. The former controls the change speed, while the latter manipulates the

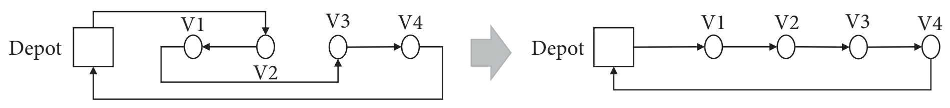

Figure 2: Interchange.

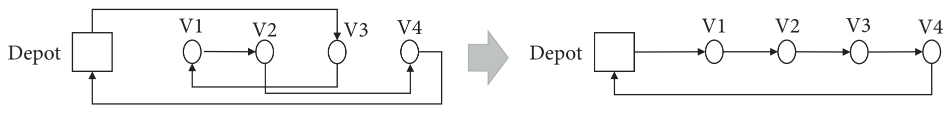

Figure 3: Relocation of a single route.

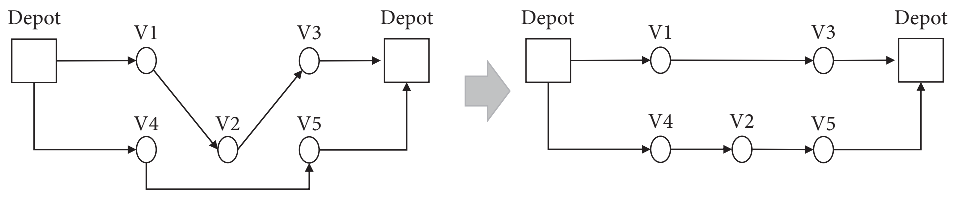

Figure 4: Relocation of a pair of routes.

```
(1) Begin
(2)   Input: $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{\mathbf{ps}}$, $k = 0$;
(3)     Rank the $ps$ solutions from small to large, where $x_1$ is the smallest one and $x_{ps}$ is the largest one. $x_1 = (R_1, R_2, \ldots, R_m)$.
(4)     While ($k < ps/2$) do
(5)       $l =$ random (1, $m$);
(6)       Do 2-opt for $R_l$ in $x_1, x_1' \leftarrow x_1$.
(7)       $x_{ps-k} = x_1'$.
(8)       $k$++;
(9)     End While
(10)  Output: $x_1, x_2, \ldots, x_{ps}$
(11) End
```

Algorithm 1: Best population immigrant scheme.

severity each time the environment changes. The dynamic environment of CTSP-VCC is generated by Algorithm 2.

4.2. Experiment Design. We have implemented all the algorithms in C++ with Microsoft Visual Studio 2012 and executed them on Dell Computer Optiplex 390 running Windows 7 with CPU Intel Core *i* 3 at 3.40 GHz and RAM 4 GB. The offline performance $\overline{F}_{\mathrm{BOG}}$ is adopted to evaluate the overall performance of an algorithm:

$$\overline{F}_{\mathrm{BOG}} = \frac{1}{G}\sum_{i=1}^{G}\left(\frac{1}{N}\sum_{j=1}^{N} F_{\mathrm{BOG}ij}\right). \tag{8}$$

Let $N = 50$ denote the total number of runs, and $G = 10 \times \tau$represents the total number of generations for a run. $F_{\mathrm{BOG}ij}$ is the best-of-generation fitness of generation i at run j, and $\overline{F}_{\mathrm{BOG}}$ is the average fitness of 50 runs.

To test the performances of the algorithms, our experiments are designed as follows: first, test cases, Eil101, A280, and Rat575 are derived from the datasets in TSPLIB [34] while their CCMs are generated randomly according to the number of cities and salesmen.

Then, the two parameters for dynamic environment control τ is set to be 30 or 50, and ρ is set at 4 different levels, i.e., 0.1, 0.2, 0.5 and 1.0. In addition, each VNS is executed independently 50 times for each test case. One-tailed t-test with 98 degrees of freedom at a 0.05 level of significance is adopted to compare the statistical results of VNSs in dynamic environments.

4.3. Comparison among the Proposed Algorithms. To compare the performances among the proposed algorithms, the statistical analysis of the experimental results is given as follows. Figures 6 and 7 show the offline performances, and Figures 8 and 9 illustrates the dynamic performances for each test. In the dynamic environments with different change speeds and intensities, the statistical results of compared algorithms by one-tailed t-test with 98 degrees of freedom at a 0.05 level of significance are given in Tables 1 and 2, where “s+“, “s-“, and “∼” denote that one

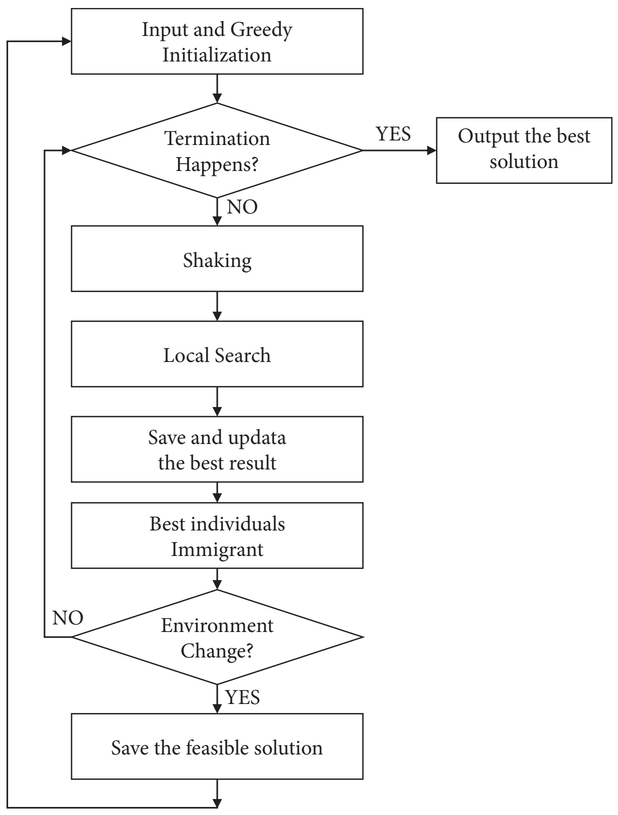
Input and Greedy Initialization
Termination Happens?
YES
Output the best solution
NO
Shaking
Local Search
Save and updata the best result
Best individuals Immigrant
NO
Environment Change?
YES
Save the feasible solution

Figure 5: The flowchart of the greedy-initialization VNS with the best population immigrant scheme.

```
(1) Begin
(2) Input: gen;
(3)     If gen = = 1
(4)     $a_{ij}(t) = a_{ij}(t_0)$
(5) Else
(6)         If gen%$\tau$ == 0
(7)             $a = random(0,1)$
(8)             If $a < \rho$
(9)                 $a_{ij}(t) = 1$
(10)            Else
(11)            $a_{ij}(t) = a_{ij}(t_0)$
(12)            End If
(13)        End If
(14)    End If
(15) Output: $\mathbf{a_{ij}(t)}$.
(16) End
```

Algorithm 2: Dynamic Environment.

algorithm is significantly better than, significantly worse than, and similar to another algorithm, respectively. The standard deviations of 50 results of the algorithms for each test are given in Tables 3 and 4.

4.3.1. Effectiveness of Greedy-Initialization. The offline performances illustrated in Figures 6 and 7 show that GVNS outperforms VNS.

For example, the result from GVNS in Eil101–6, at $\tau = 50$ and $\rho = 0.1$, is 2515.8 and it is 15.1% less than that from VNS. In A280-15, the result from GVNS is 5836.6 and it is only 22.4% of that from VNS.

In Figure 9, the dynamic performances show that, the greedy-initialization can effectively improve the initial solution quality of VNS. For example, the results obtained by GVNS in the environment at $\tau = 50$and $\rho = 0.2$ are 2357.6,

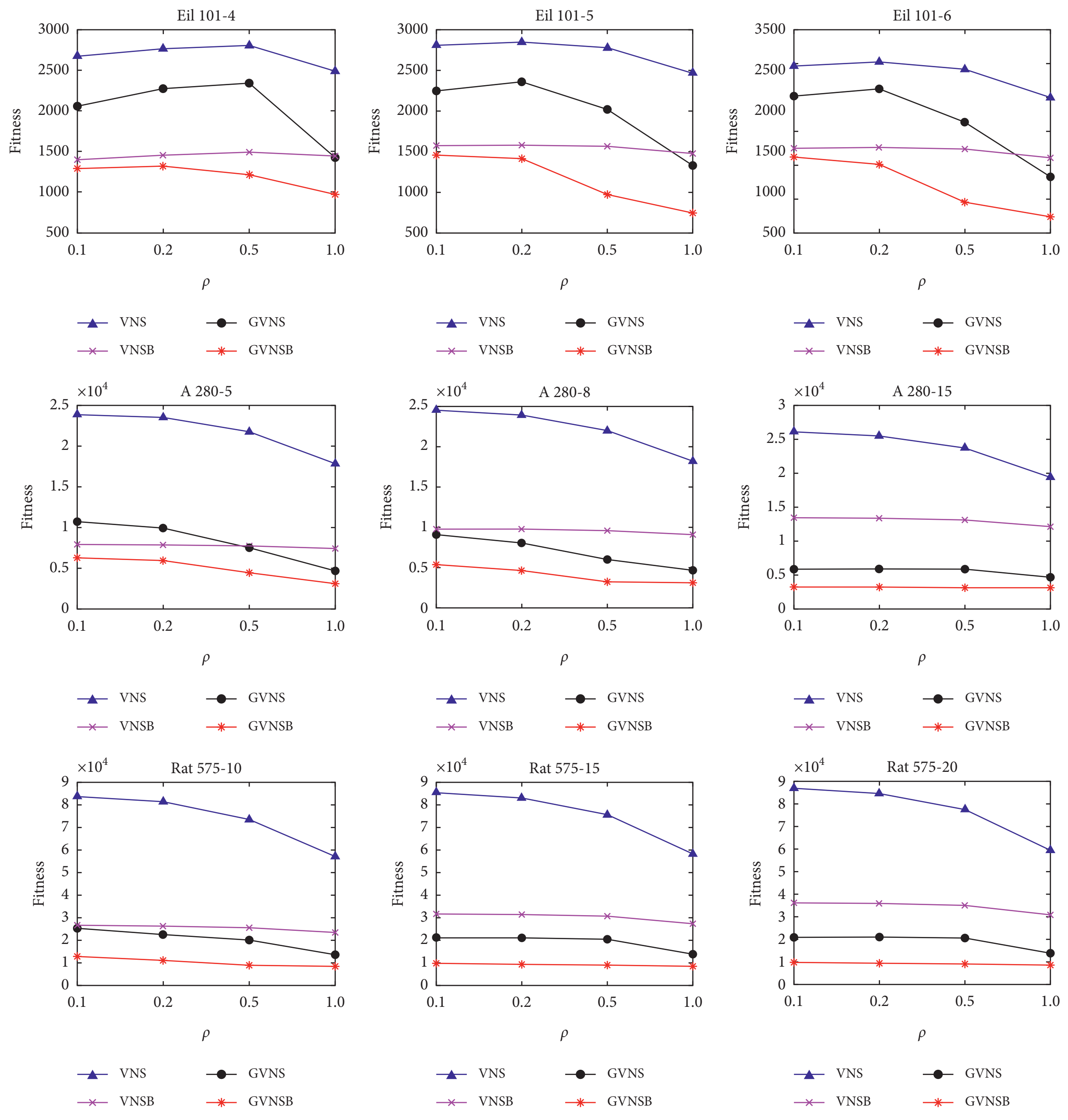

Figure 6: The results of VNSs at environment $\tau = 50$.

8029.4, and 22478.7, that is, 82%, 34%, and 28% of those by VNS in Eil101–5, A280-8, and Rat575-10, respectively.

Similarly, the t-test results illustrated in Tables 1 and 2 show that GVNSB outperforms VNS, VNSB, and GVNS significantly.

Therefore, we can draw a conclusion that the greedy-initialization is very effective for VNS to solve CTSP-VCC. The reason is that a better initial solution is critically important for population evolution, and it can guide individuals to perform their search in a more promising solution space of its neighborhood. Indeed, VNS can find a better solution quickly and effectively at the initial stage by means of greedy-initialization, and it also can obtain other diverse solutions by random initialization. The greedy initialization can guarantee VNS not only an initial solution quality but also the diversity of the population. It is why GVNS and GVNSB perform better than VNS and VNSB.

4.3.2. Effectiveness of a Best Population Immigrant Scheme. The results of algorithms in a dynamic environment with $\tau = 50$ are illustrated in Figure 9. Specially, in Eil101-4 at $\rho = 0.1$, the results obtained by VNS and VNSB are 2670.2 and 1398.9, respectively. VNSB's result is improved about 47.6% after introducing a best population immigrant scheme. Furthermore, the result of GVNSB is 1293.8, just 62.8% of that with GVNS. In addition, the result in A280-8 shows that VNSB obtains 9783.9 and is 39.8% of that from

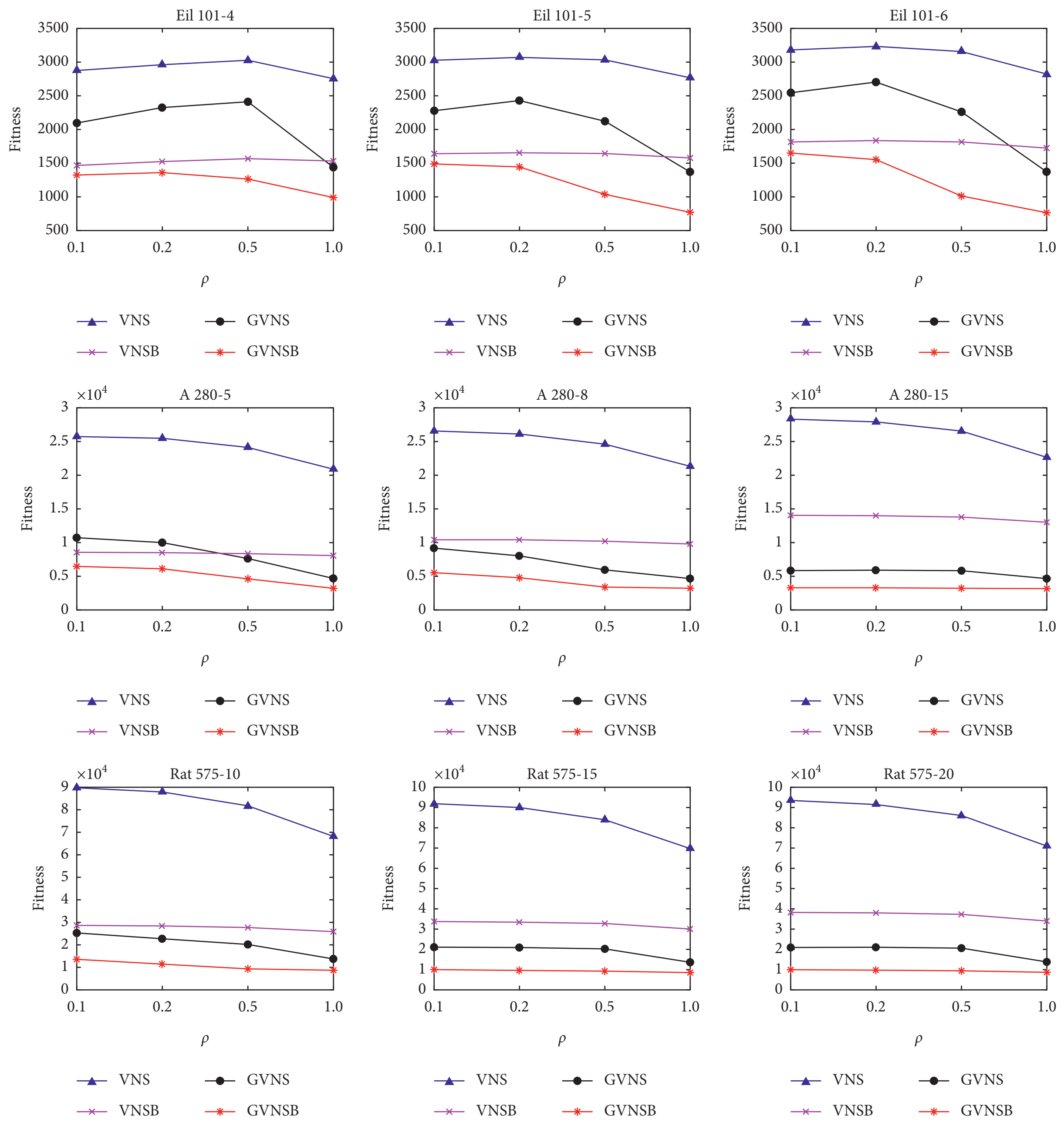

Figure 7: The results of VNSs in the environment with $\tau = 30$.

VNS, and GVNSB obtains 5328.5 and it is 58.7% of that from GVNS. Besides, the results in three cases of Rat 575 show that we obtain better results after introducing the best population immigrant scheme into VNS.

All the curves in Figure 6 show that the fitness decreases as ρ increases. The reason is that the increase of city colors with ρ enlarges the solution space of a problem. Thus, the algorithms may obtain better solutions in the new solution spaces. Furthermore, Figure 7 shows the results at $\tau = 30$, like Figure 6, where VNS and GVNS with a best population immigrant scheme can obtain better results than those of VNS and GVNS.

Figures 8 and 9 illustrate the dynamic performances in some test cases. The algorithms have different convergences. Obviously, the algorithms converge well when the best population immigrant scheme is applied. For example, the fitness value of the first generation in VNS is 34764 and 112337 in A280-15 and Rat575-15, at dynamic environment $\tau = 50$ and $\rho = 0.5$, respectively. After one generation, they are 33745.6 and 108974, respectively. Although the fitness of value in the first generation in VNSB is similar to that in VNS, after one-generation evolution, they are 56.4% and 60.2% of those in VNS, respectively.

Table 3 shows the standard deviation of 50 runs for each test case. The obtained results illustrate that the standard deviations of each algorithm are markedly different at the beginning. But they decrease significantly by introducing the best population immigrant scheme into the algorithms. For

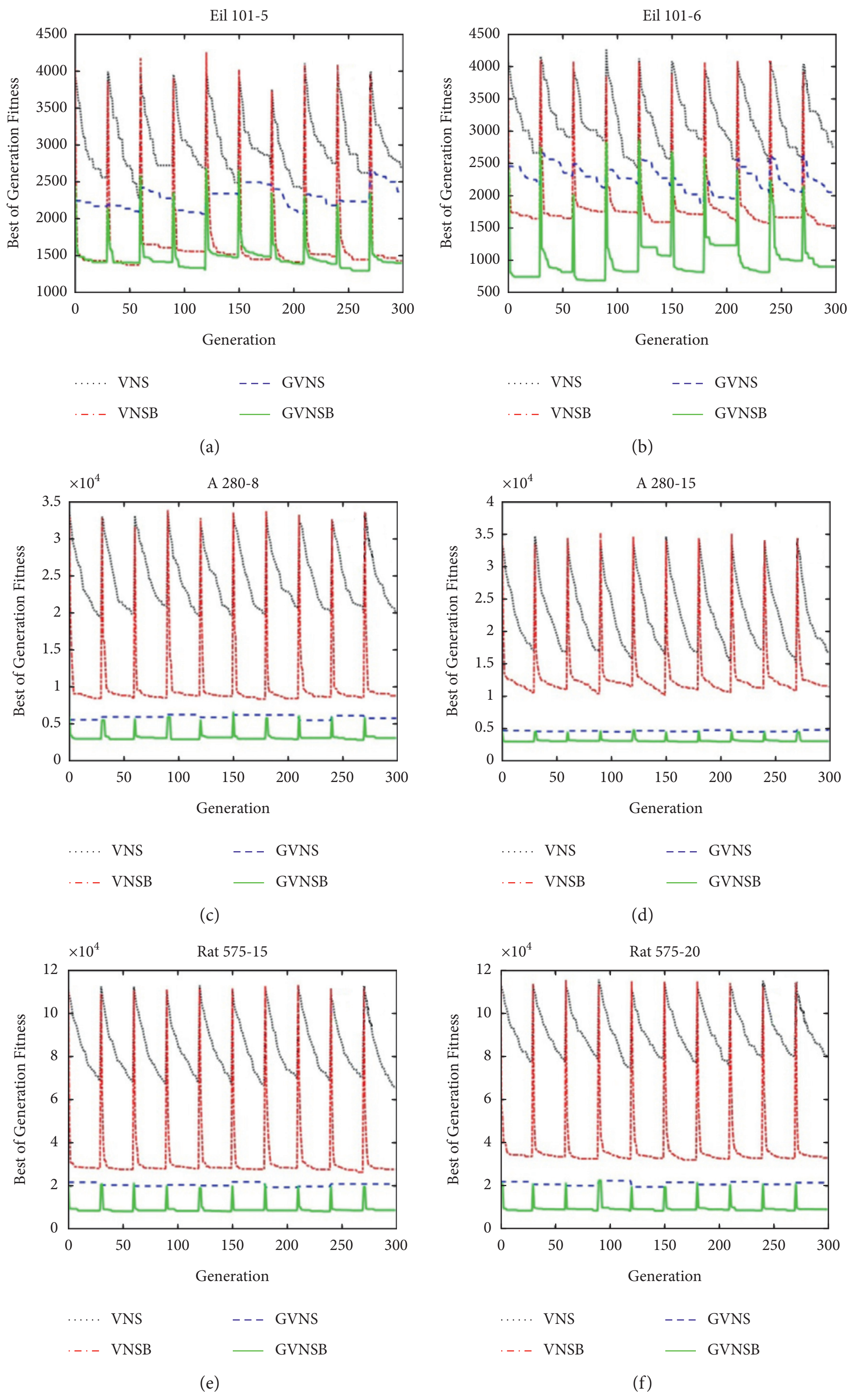

FIGURE 8: Dynamic performance of algorithms at different test cases at environment $\tau = 30$. (a) Eil 101–5, $\rho = 0.1$. (b) Eil 101–6, $\rho = 0.5$. (c) A280-8, $\rho = 0.5$. (d) A280-15, $\rho = 1.0$. (e) Rat 575–15, $\rho = 0.5$. (f) Rat 575–20, $\rho = 0.2$

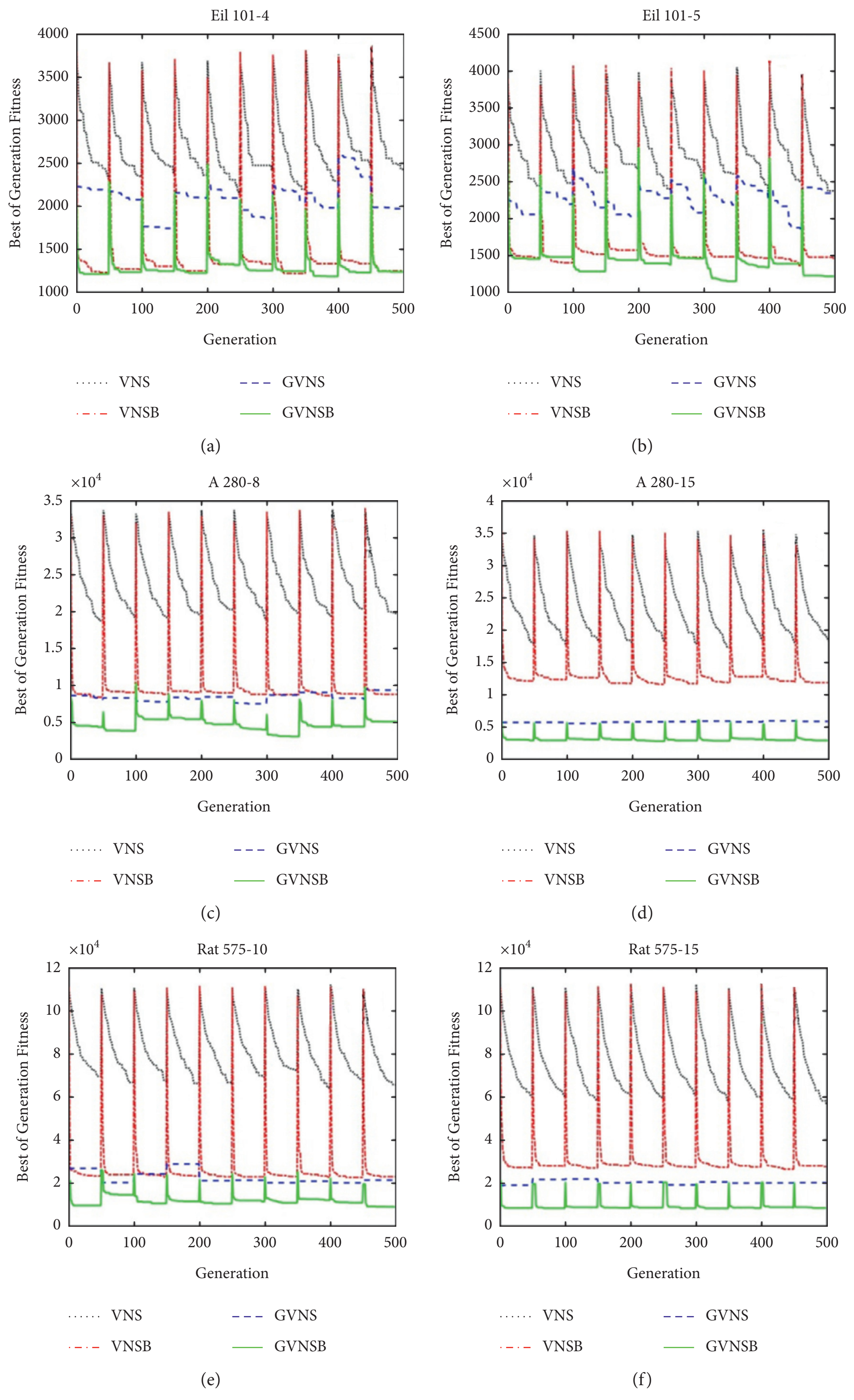

FIGURE 9: Dynamic performance of algorithms at different test cases in environment $\tau = 50$. (a) Eil 101–4, $\rho = 0.1$. (b) Eil 101–5, $\rho = 0.2$. (c) A280-8, $\rho = 0.2$. (d) A280-15, $\rho = 0.5$. (e) Rat 575–10, $\rho = 0.2$. (f) Rat 575–15, $\rho = 0.5$

Table 1: t-test result of comparing VNSs in environment $\tau = 50$.

t-test result	Eil 101–4				Eil 101–5				Eil 101–6			
$\tau = 30, \rho \Rightarrow$	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0
VNSB-VNS	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
GVNSB-GVNS	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
GVNS-VNS	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
GVNS-VNSB	s-	s-	s-	s+	s-	s-	s-	s+	s-	s-	s-	s+
GVNSB-VNSB	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
t-test result	A 280–5				A 280–8				A280-15			
$\tau = 30, \rho \Rightarrow$	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0
VNSB-VNS	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
GVNSB-GVNS	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
GVNS-VNS	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
GVNS-VNSB	s-	s-	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
GVNSB-VNSB	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
t-test result	Rat 575–10				Rat 575–15				Rat 575–20			
$\tau = 30, \rho \Rightarrow$	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0
VNSB-VNS	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
GVNSB-GVNS	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
GVNS-VNS	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
GVNS-VNSB	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
GVNSB-VNSB	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+

Table 2: t-test result of comparing VNSs in environment $\tau = 30$.

t-test result	Eil 101–4				Eil 101–5				Eil 101–6			
$\tau = 50, \rho \Rightarrow$	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0
VNSB-VNS	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
GVNSB-GVNS	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
GVNS-VNS	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
GVNS-VNSB	s-	s-	s-	s+	s-	s-	s-	s+	s-	s-	s-	s+
GVNSB-VNSB	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
t-test result	A 280–5				A 280–8				A 280–15			
$\tau = 50, \rho \Rightarrow$	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0
VNSB-VNS	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
GVNSB-GVNS	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
GVNS-VNS	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
GVNS-VNSB	s-	s-	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
GVNSB-VNSB	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
t-test result	Rat 575–10				Rat 575–15				Rat 575–20			
$\tau = 50, \rho \Rightarrow$	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0
VNSB-VNS	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
GVNSB-GVNS	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
GVNS-VNS	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
GVNS-VNSB	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
GVNSB-VNSB	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+

example, for Eil101–5, A280-5, and Rat575–15 in the environment with $\tau = 50$ and $\rho = 0.5$, the standard deviations of VNS are 23.9, 119, and 279.3, while those of VNSB are 20.6, 73.8, and 152.8, falling about 13.8%%, 37.9%, and 45.3%, respectively. Similarly, the standard deviations in GVNSB are all less than those of GVNS. The reason is that VNS can maintain the population evolution with high-quality genes and better stability by means of the best population immigrant scheme.

Tables 1 and 2 illustrate the t-test results of the algorithms in dynamic environments with ρ increasing from 0.1 to 1.0, while $\tau = 50$ and 30. Concretely, the t-test results of Eil101, A280, and Rat575 show that VNSB is significantly better than VNS, and GVNSB is significantly better than GVNS. The results are sufficient for us to validate that the best population immigrant scheme is very effective for VNS to solve CTSP-VCC.

4.3.3. Performance Comparison between Greedy-Initialization and Best-Population Immigrant. In Tables 1 and 2, the performance of GVNS is significantly worse than that of VNSB in three cases of Eil101 with $\rho = 0.1, 0.2$, and 0.5. This result is also illustrated more clearly in Figure 6. In particular, in dynamic environment at $\tau = 50$ and $\rho = 0.5$, the results in GVNS in Eil101–4, Eil101–5, and Eil101-6 are 2342, 2015, and 2132.1, respectively. Those results obtained by VNSB are 1494.5, 1561.8, and 1734.6, respectively, all less than their corresponding results by GVNS.

In addition, we find that the pros and cons of GVNS and VNSB swap with the increase of the number of cities. For example, in Figure 7, the results obtained by GVNS are better than those by VNSB in A280–8 and A280-15, but in A280-5 with $\rho = 0.1$ and 0.2, the result from GVNS is worse than that from VNSB. While in the dynamic environment at $\rho = 0.5$ and 1.0, the results have changed, that is, the result

Table 3: The standard deviations of VNS, VNSB, GVNS, and GVNSB in environment $\tau = 50$.

Case	Eil 101–4				Eil 101–5				Eil 101–6			
$\rho\Rightarrow$	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0
VNS	25.8	24.1	30.2	23.1	21.3	28.2	23.9	16.7	22.8	16.7	24.5	20.6
VNSB	12	17.4	12.8	15.9	14.4	14.1	20.6	18.1	13	18.7	19.5	17.7
GVNS	42.9	44.5	45.1	19.9	36	42	50.7	15.2	37.2	50	51.4	14.9
GVNSB	12.3	16.6	26.5	7.2	16.6	27.8	41.8	5.9	16.9	34.4	50.6	5.9
Case	Eil 280–5				Eil 280–8				Eil 280–15			
$\rho\Rightarrow$	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0
VNS	99.3	143.5	119	92.4	94.5	155.1	128.8	88.7	139.8	109.8	168.6	101.9
VNSB	65	58.5	73.8	65.3	52.9	61.9	58.9	62.8	69.1	81.1	82.4	84.7
GVNS	161	192.1	225.1	52.2	230.1	352.1	186.7	43.4	111.8	111.4	91.4	39
GVNSB	76.7	94.9	201.7	24.6	156.5	188.9	100.4	17.8	51.2	57.6	41.9	18.4
Case	Eil 575–10				Eil 575–15				Eil 575–20			
$\rho\Rightarrow$	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0
VNS	315.7	315.4	287.4	231.7	286.5	348.2	279.3	199.8	321.9	357.6	267.2	226.9
VNSB	110.7	150.2	127.1	129.1	153.5	146.6	152.8	144	167.9	201.1	168.3	150.5
GVNS	802.5	679.1	230.1	99.3	345	326.7	225.2	99.6	281.2	246.6	251.5	85.7
GVNSB	441.5	616.4	221.6	51.9	185.6	151.7	132.4	59.3	161.7	173.7	162.8	46.3

Table 4: Standard deviations of VNS, VNSB, GVNS, and GVNSB in environment $\tau = 30$.

Case	Eil 101–4				Eil 101–5				Eil 101–6			
$\rho\Rightarrow$	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0
VNS	21.9	29.5	25.8	24	25	21.1	23.2	22.6	25.3	23.7	29.4	25
VNSB	15.7	19.5	14.9	12.6	13.3	15.8	17.1	16.2	16.8	19.1	15.1	18.2
GVNS	43	38.2	52.8	24.2	44.5	41	57.6	13.1	43.2	53.4	65.2	14.9
GVNSB	12.3	16.9	23.7	10.6	14.2	26.8	56.7	7.3	17	36.3	45.3	7.8
Case	Eil 280–5				Eil 280–8				Eil 280–15			
$\rho\Rightarrow$	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0
VNS	117	140.2	129.6	73.7	148.8	140.4	135	111.8	153.5	125.7	138.7	114.6
VNSB	89.3	73.6	84.8	79.7	87.3	77.9	80.9	65.7	73.5	79	72.3	76.2
GVNS	174.8	210.9	236.3	45.1	244.1	382.7	163.2	35.9	103.7	97.8	94.7	30.7
GVNSB	74.3	94.2	207.3	22.2	129.6	245.8	122.2	25.1	54	58.9	41.1	17.9
Case	Eil 575–10				Eil 575–15				Eil 575–20			
$\rho\Rightarrow$	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0
VNS	292.4	285.5	321.9	284.1	259	319.5	280.8	217	295.9	315.8	319.5	266.3
VNSB	219.1	220.3	203.3	210.6	156.2	214.8	179.7	178.9	143.9	196.5	193.6	151.8
GVNS	722.2	682	255.7	98.8	323.9	287.9	259.1	94.2	237.7	257.6	260.3	94.6
GVNSB	499.6	556.5	158.5	64.6	288.2	203.4	174.2	74.8	144.7	197.4	174	64.4

from GVNS is better than that from VNSB. Similarly, the results obtained by GVNS are better than those by VNSB in cases of Rat575–15 and Rat575–20.

Figures 8 and 9 show that GVNSB always achieves the best results among the algorithms. Furthermore, Tables 1 and 2 suggest that the introduction of the best population immigrant scheme into GVNS can guide the algorithm to track the change of the optimal solution rapidly in a dynamic environment. Hence, GVNSB outperforms significantly other three algorithms in terms of the offline performance, dynamic performance, and standard deviation.

In summary, both the best population immigrant scheme and greedy-initialization scheme can help VNS converge rapidly and obtain much better solutions, and they are beneficial for VNS to track the dynamic environment of CTSP-VCC.

5. Conclusions

This work proposes a dynamic CTSP in which city colors change with time. It can be applied to a supply decision problem of online retailers with multiwarehouses and varying supply availability. Then, a variable neighborhood search algorithm is proposed to address this new CTSP. It uses direct-route encoding and a two-stage greedy-initialization algorithm to obtain an optimal/near-optimal solution. In addition, a best population immigrant scheme is adopted to track the change of the optimal solution rapidly in a dynamic CTSP. Furthermore, to mimic the dynamic environment of CTSP-VCC, we construct a dynamic environment simulator that can be easily tuned via two parameters, i.e., change speed and severity.

To test the performances of the proposed algorithm, i.e., greedy-initialization VNS with the best population immigrant, extensive experiments are conducted in terms of offline performance, dynamic performance, and standard deviation.

The results show that VNS can obtain a desired solution in its initialization by means of a greedy-initialization scheme. Solution quality and stability can be significantly improved with the help of the best population immigrant scheme. Specially, because of the two schemes, the results of the baseline VNS can be improved about 30%–50%. In summary, the proposed best population immigrant scheme and greedy-initialization scheme are both effective for VNS to track the change of optimal solutions in the new CTSP and help the algorithm obtain better solutions. Our ongoing work is to apply CTSP-VCC and VNSs to some real-life problems [35, 36].

Data Availability

The data used to support the findings of this study have been deposited in the Github (Link:https://github.com/opt-mlearing/vehicle-routing-problems).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 61773115, the Natural Science Foundation of Anhui Province under Grant 2008085QF300, the Youth Foundation of Anhui University of Technology under Grant QZ201911, the Shenzhen Fundamental Research Program under Grant JCYJ20190813152401690, and the Fundo para o Desenvolvimento das Ciencias e da Tecnologia under Grant 0047/2021/A1.

Supplementary Materials

Figure S1 : the flowchart of VNS. (*Supplementary Materials*)

References

[1] J. Li, M. Zhou, Q. Sun, X. Dai, and X. Yu, "Colored traveling salesman problem," *IEEE Transactions on Cybernetics*, vol. 45, no. 11, pp. 2390–2401, 2015.

[2] X. Meng, J. Li, X. Dai, and J. Dou, "Variable neighborhood search for a colored traveling salesman problem," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 4, pp. 1018–1026, 2018.

[3] J. Li, X. Meng, M. Zhou, and X. Dai, "A two-stage approach to path planning and collision avoidance of multibridge machining systems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 7, pp. 1039–1049, 2017.

[4] J. Li, X. Meng, and X. Dai, "Collision-free scheduling of multi-bridge machining systems: a colored traveling salesman problem-based approach," *IEEE/CAA Journal of Automatica Sinica*, vol. 5, no. 1, pp. 139–147, 2018.

[5] P. He and H. Jin-Kao, "Iterated two-phase local search for the colored traveling salesmen problem," *Engineering Applications of Artificial Intelligence*, vol. 97, Article ID 104018, 2021.

[6] X. Xu, J. Li, and M. Zhou, "Delaunay-triangulation-based variable neighborhood search to solve large-scale general colored traveling salesman problems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 3, pp. 1583–1593, 2021.

[7] X. Xu, J. Li, M. Zhou, and X. Yu, "Bi-objective colored traveling salesman problems," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–11, 2021.

[8] X. Xu, J. Li, M. Zhou, and X. Yu, "Precedence-constrained colored traveling salesman problem: an augmented variable neighborhood search approach," *IEEE Transactions on Cybernetics*, pp. 1–12, 2021.

[9] L. Huang, M. C. Zhou, K. R. Hao, and E. Hou, "A survey of multi robot regular and adversarial patrolling," *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 4, pp. 865–874, 2019.

[10] J. Wang, Y. Sun, Z. Zhang, and S. Gao, "Solving multitrip pickup and delivery problem with time windows and manpower planning using multiobjective algorithms," *IEEE/CAA Journal of Automatica Sinica*, vol. 7, no. 4, pp. 1134–1153, 2020.

[11] S. Yang, H. Hui Cheng, and F. Fang Wang, "Genetic algorithms with immigrants and memory schemes for dynamic shortest path routing problems in mobile ad hoc networks," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 1, pp. 52–63, 2010.

[12] X. Wang, Z. Ning, M. Zhou et al., "Privacy-Preserving Content Dissemination for Vehicular Social Networks: Challenges and Solutions," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, 2nd Quarter, 2019.

[13] X. Bu, C. Liu, Q. Yu, L. Yin, and F. Tian, "Optimization on cooperative communications based on network coding in multi-hop wireless networks," in *Proceedings of the International Wireless Communications and Mobile Computing (IWCMC)*, pp. 384–387, Limassol, Cyprus, June 2020.

[14] G. Fortino, F. Messina, D. Rosaci, and G. M. L. Sarne, "ResIoT: an IoT social framework resilient to malicious activities," *IEEE/CAA Journal of Automatica Sinica*, vol. 7, no. 5, pp. 1263–1278, 2020.

[15] Q. Wu, Z. Du, P. Yang, Y.-D. Yao, and J. Wang, "Traffic-aware online network selection in heterogeneous wireless networks," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 1, pp. 381–397, 2016.

[16] C. Chen, B. Liu, S. Wan, P. Qiao, and Q. Pei, "An edge traffic flow detection scheme based on deep learning in an intelligent transportation system," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 3, pp. 1840–1852, 2021.

[17] G. Kim, Y. S. Ong, T. Cheong, and P. S. Tan, "Solving the dynamic vehicle routing problem under traffic congestion," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 8, pp. 2367–2380, 2016.

[18] X. Jin, H. Qin, Z. Zhang, M. Zhou, and J. Wang, "Planning of garbage collection service: an arc-routing problem with time-dependent penalty cost," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 5, pp. 2692–2705, 2021.

[19] L. Melo, F. Pereira, and E. Costa, "Extended experiments with ant colony optimization with heterogeneous ants for large dynamic traveling salesperson problems," in *Proceedings of the 2014 14th International Conference on Computational Science and its Applications*, pp. 171–175, Guimaraes, Portugal, July 2014.

[20] S. Wang, J. Zhang, Z. Zhang, and X. Yu, "A discrete particle swarm optimization algorithm for solving TSP under dynamic topology," in *Proceedings of the 2019 IEEE*

International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM), pp. 165–170, Bangkok, Thailand, 2019.

[21] H. Chen and Y. Zhang, "Dynamic path optimization in sharing mode to relieve urban traffic congestion," *Discrete Dynamics in Nature and Society*, vol. 2021, Article ID 8874957, 16 pages, 2021.

[22] R. Tinós, "Analysis of the dynamic traveling salesman problem with weight changes," in *Proceedings of the 2015 Latin America Congress on Computational Intelligence (LA-CCI)*, pp. 1–6, Curitiba, PR, Brazil, October 2015.

[23] M. Mavrovouniotis, F. M. Muller, and S. Yang, "Ant colony optimization with local search for dynamic traveling salesman problems," *IEEE Transactions on Cybernetics*, vol. 47, no. 7, pp. 1743–1756, 2017.

[24] A. Baykasoğlu and Z. Durmuşoğlu, "A multi-agent based approach to modeling and solving dynamic generalized travelling salesman problem," *Journal of Intelligent and Fuzzy Systems*, vol. 31, no. 1, pp. 77–90, 2016.

[25] U. Ritzinger, J. Puchinger, and R. Hartl, "A survey on dynamic and stochastic vehicle routing problems," *International Journal of Production Research*, vol. 54, no. No. 1, pp. 215–231, 2016.

[26] G. Kim, Y. Ong, T. Cheong, and P. Tan, "Solving the dynamic vehicle routing problem under traffic congestion," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 8, pp. 2367–2380, 2016.

[27] A. Juan, J. Faulin, S. Grasman, D. Riera, J. Marull, and C. Mendez, "Using safety stocks and simulation to solve the vehicle routing problem with stochastic demands," *Transportation Research Part C: Emerging Technologies*, vol. 19, no. 5, pp. 751–765, 2011.

[28] S. Vonolfen and M. Affenzeller, "Distribution of waiting time for dynamic pickup and delivery problems," *Annals of Operations Research*, vol. 236, no. 2, pp. 359–382, 2016.

[29] H. Jula, M. Dessouky, and P. A. Ioannou, "Truck route planning in nonstationary stochastic networks with time windows at customer locations," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 1, pp. 51–62, 2006.

[30] M. P. Fanti, A. M. Mangini, A. Favenza, and G. Difilippo, "An eco-route planner for heavy duty vehicles," *IEEE/CAA Journal of Automatica Sinica*, vol. 8, no. 1, pp. 37–51, 2021.

[31] M. Schilde, K. F. Doerner, and R. F. Hartl, "Integrating stochastic time-dependent travel speed in solution methods for the dynamic dial-a-ride problem," *European Journal of Operational Research*, vol. 238, no. 1, pp. 18–30, 2014.

[32] S. D. Bopardikar, S. L. Smith, and F. Bullo, "On dynamic vehicle routing with time constraints," *IEEE Transactions on Robotics*, vol. 30, no. 6, pp. 1524–1532, 2014.

[33] M. Pavone, E. Frazzoli, and F. Bullo, "Adaptive and distributed algorithms for vehicle routing in a stochastic and dynamic environment," *IEEE Transactions on Automatic Control*, vol. 56, no. 6, pp. 1259–1274, 2011.

[34] G. Reinelt, "Benchmark dataset of TSPLIB," Available: http://www.iwr.uni-heidelberg.de/groups/comopt/software /TSPLIB95/, 2002.

[35] Y. Li, F. Chu, C. Feng, C. Chu, and M. Zhou, "Integrated production inventory routing planning for intelligent food logistics systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 3, pp. 867–878, 2019.

[36] L. Wang and J. Lu, "A memetic algorithm with competition for the capacitated green vehicle routing problem," *IEEE/CAA Journal of Automatica Sinica*, vol. 6, no. 2, pp. 516–526, 2019.