Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 152494, 13 pages
http://dx.doi.org/10.1155/2014/152494

Research Article

A Framing Link Based Tabu Search Algorithm for Large-Scale Multidepot Vehicle Routing Problems

Xuhao Zhang,[1] Shiquan Zhong,[1] Yiliu Liu,[2] and Xuelian Wang[3]

[1] *College of Management and Economics, Tianjin University, Tianjin 300072, China*
[2] *Department of Production and Quality Engineering, Norwegian University of Science and Technology, 7491 Trondheim, Norway*
[3] *School of Management, Hebei University of Technology, Tianjin 300401, China*

Correspondence should be addressed to Shiquan Zhong; shiquan_tju@163.com

Received 21 April 2014; Accepted 29 June 2014; Published 23 July 2014

Academic Editor: Baozhen Yao

A framing link (FL) based tabu search algorithm is proposed in this paper for a large-scale multidepot vehicle routing problem (LSMDVRP). Framing links are generated during continuous great optimization of current solutions and then taken as skeletons so as to improve optimal seeking ability, speed up the process of optimization, and obtain better results. Based on the comparison between pre- and postmutation routes in the current solution, different parts are extracted. In the current optimization period, links involved in the optimal solution are regarded as candidates to the FL base. Multiple optimization periods exist in the whole algorithm, and there are several potential FLs in each period. If the update condition is satisfied, the FL base is updated, new FLs are added into the current route, and the next period starts. Through adjusting the borderline of multidepot sharing area with dynamic parameters, the authors define candidate selection principles for three kinds of customer connections, respectively. Link split and the roulette approach are employed to choose FLs. 18 LSMDVRP instances in three groups are studied and new optimal solution values for nine of them are obtained, with higher computation speed and reliability.

1. Introduction

Nowadays, logistic cost is considered to be influential in fierce business competitions. The logistical delivery is a process from pickup to drop-off of goods, connecting vendors, shippers, and customers. In related studies, Vehicle Routing Problem (VRP) is always regarded as a significant issue, due to its impact on optimization of delivery vehicles scheduling and then on profit of logistic providers.

In common sense, Vehicle Routing Problem is defined as follows: how to determine appropriate delivery routes between series of collection and reception terminals and guarantee delivery vehicles in a proper order, so as to satisfy some requirements (e.g., shortest distance, minimum cost, delivery time, and needed vehicles) within kinds of constraints, such as amount of goods, sending time, vehicle capacity, mileage restriction, and time limitation. Currently, VRPs have been identified in many applications, such as products outbound distribution scheduling [1], home care crew scheduling [2], newspaper delivery [3], school bus routing [4], cargo routing [5], airline crew scheduling [6], waste collection scheduling [7], service system design [8], and computer system integration [9].

It is difficult to solve large-scale VRPs due to their complexities. Lenstra and Rinnooy Kan [10] proved that capacity constraint VRP (CVRP) was a NP-hard problem and recently Hassin and Rubinstein [11] verified the availability of polynomial-time algorithm in the cases where $k = 3$ or 4 for the k-VRP issue. Imai and his partners [12] highlighted that the Vehicle Routing Problem with full container load was also NP-hard. Furthermore, Solomon [13] realized that VRP with time window constraints was more complicated, and Hashimoto et al. [14] confirmed the NP-hard characteristics of VRP with soft time windows. In the paper of Savelsbergh [15], the author proved that it is a NP-complete problem to decide whether a feasible solution existed or not for the TSP with time windows. Lenstra and Rinnooy Kan [10] proved that all types of VRPs are NP-hard problems.

From basic capacity constraint VRP, researchers have recognized varied VRP problems, for example, VRP with time window (VRPTW), periodical VRP (PVRP), VRP with pickups and deliveries (VRPPD), and multidepot VRP (MDVRP). Compared with VRP with a single depot, MDVRP with more than one depot is more complicated. In MDVRP, not only the delivery sequence but also the vehicle type and amount for customers at each distribution center need to be determined.

Given that $G = (V, A)$ is a complete graph, $V = (1, \ldots, n)$ is the set of vertexes in the chart and A is the set of arcs. $V_c = (1, 2, \ldots, n)$ represents vertexes, $V_d = (n + 1, \ldots, n + m)$ represents depots, and q_i is the delivery amount to vertex i. The capacity of vehicles from depot j is W_j, $j = n + 1, \ldots, n + m$, and c_{ij} is the delivery cost from vertex i to j. In this paper, MDVRP is described as how to establish multiple paths in the chart G, so as to satisfy the following:

(a) each path starts and ends at the same depot;

(b) all vertexes are allocated to paths, and each vertex is allocated once;

(c) the total delivery amount of depot j is no more than W_j, $j = n + 1, \ldots, n + m$.

A heuristic approach is adapted by most researchers in dealing with MDVRP. It can be divided into the classical heuristic and the metaheuristic. Initially, the classical heuristic was more acceptable; for example, Gillett and Johnson [16] applied sweep heuristic in MDVRP and Golden et al. [17] used borderline search strategy and improved saving algorithm. A three-level heuristic algorithm was proposed by Salhi and Sari [18], where feasible solutions were generated on level 1 and delivery routes were optimized on levels 2 and 3. Sumichras and Markham [19] developed a C-W saving method. Wasner and Zäpfel [20] proposed a local search strategy with a series of feedback loops, and Nagy and Sahli [21] put forward multiple enhanced optimization strategies for MDVRP with pickups and deliveries problem (MDVRPPD). Lim and Wang [22] offered two solution methodologies—one-stage and two-stage approaches—to solve MDVRP with fixed distribution of vehicles. Recently, three hybrid heuristics were proposed by Mirabi et al. [23] for MDVRP, which were based on deterministic, stochastic techniques, and simulated annealing (SA) methods. Contardo and Martinelli [24] designed a new exact method to solve the MDVRP based upon the vehicle-flow formulation and the set-partitioning formulation.

Since the 1980s, some innovative optimization methods, such as genetic algorithm, simulated annealing, tabu search, and ant colony algorithm, have been developed greatly and acted as creative roles in tackling VRP. Cordeau et al. [25] proposed a tabu search heuristic effective for three well-known routing problems: PVRP, the periodic traveling salesman problem (PTSP), and MDVRP. Such a method generalized a tabu search in solutions to VRP. This author [26] then improved the above algorithm, to solve the periodic and the multidepot vehicle routing problems with time windows (PVRPTW and MDVRPTW). Similar methods occurred in the studies of Renaud et al. [27] and Crevier et al. [28]. Renaud et al. [27] divided the algorithm into two stages and employed a tabu search to optimize the feasible solutions generated with heuristic. Crevier et al. [28] combined adaptive memory principle, a tabu search method, and the integer programming. Belhaiza et al. [29] presented a new hybrid variable neighborhood-tabu search heuristic for VRP with multiple time windows. Genetic algorithm is also acceptable by researchers. Bae et al. [30] developed an integrated VRP solver based on an improved genetic algorithm. Ho et al. [31] designed two hybrid genetic algorithms: one generated initial routes randomly and another created initial routes with Wright saving method and the nearest neighbor heuristic. In addition, Wang et al. [32] used genetic algorithm to study more complicated MDVRPs with constraints of time windows, limited numbers of vehicles, and multitype vehicles. Besides, the ant colony algorithm [33] and the simulated annealing approach [34, 35] were also applied in solving MDVRP.

However, few existing literatures pay attention to large-scale MDVRP (LSMDVRP, with customers more than 150), and the effectiveness of its solving methods should be discussed further. Framing link (FL) introduced in the following parts is helpful to reduce the searching space effectively and is a new direction in optimizing LSMDVRP. In fact, some scholars have accepted similar concepts in studying VRP; for example, Tarantilis and Kiranoudis [36] presented an adaptive memory based method for solving the Capacitated Vehicle Routing Problem (CVRP), called bone route. Zhong [37] proposed the concept and principium of kernel route, and a tabu search algorithm was designed to solve open Vehicle Routing Problem (OVRP) with capacity and distance limits. In this paper, the authors will combine the FL with a tabu search in solving LSMDVRP.

This paper is organized as follows. In Section 2, the authors propose the principle of FL for LSMDVRP and and the structure of tabu algorithm, and procedure of the method is described in Section 3. Then, the authors measure algorithm sensitivity and the relationship between FL and optimization results and then compare the result of the proposed method with those in references, which shows that the former is performed. Conclusions occur at the end.

2. LSMDVRP Framing Link

2.1. Principle of FL and Structure of Optimization. When an intelligent algorithm is applied in the optimization of vehicle routes, iteration is conducted based on the previous generation solutions. As a result, there are numerous links between solutions of neighbor generations, and these links are updated continuously with iterations; good links (in optimized solutions) are kept and bad ones are decomposed or combined with others. In the procedure of a so-called "good" algorithm to VRP, more good links are generated. As shown in Figures 1 and 2, the routes are achieved through iterations of a tabu search for MDVRP case p10 in Gillett and Johnson [16], and their solution values are 3714.28 and 3647.22, respectively.

The two solutions shown above occur at the generation 1233 and generation 1671, respectively, in the tabu search. Although there are over 400 generations between the two

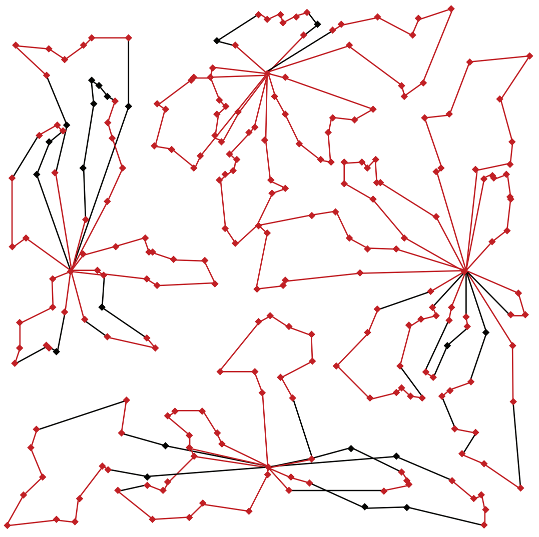
Figure 1: The route with a solution value 3731.38.

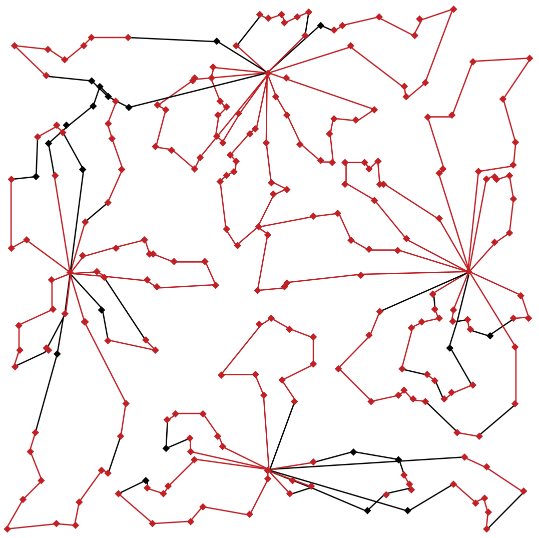
Figure 2: The route with a solution value 3647.22.

solutions, their structures have many similar parts as illustrated with red lines in Figures 1 and 2. In those routes with less generation gap, more similar links can be found. It is possible to optimize the VRP solutions based on these kinds of links as the skeleton, so as to obtain better results. In this paper, links that occur in both the optimal solution and suboptimal solutions are named as FLs.

Generation of framing links and update are keys of the FL based algorithm for LSMDVRP. The authors construct a FL base, which consists of links with higher frequencies in good routes in iterations. Addition and deletion of links are determined with the update condition. It is necessary to allocate vertexes to depots so as to generate LSMDVRP FLs, but, for those collection nodes close to several depots, possible allocations of them to different depots in optimization can result in generating unstable FLs and hence prevent them from entering the FL base. Consequently, the authors specify a FL tabu area among depots, where the generating principles of FLs are more rigorous. The basic structure of FL based tabu algorithm for LSMDVRP is described in Figure 3.

2.2. Update Condition of Framing Links. If the requirement is satisfied, an update occurs in the FL base and the generated links are added into the base; meanwhile, those links without high usage frequency are deleted. In this study, the update condition of FLs includes:

(a) the update of FLs has iterated N_d times;

(b) the local optimal solution keeps unchanged for N_i generations;

(c) all local and global optimums are less than B_q after N_q generations since the last update.

2.3. Generation of New-Entry Links

2.3.1. Generation of Candidate Links. FLs are generated with the continuous and large-scale optimization of current solutions in the procedure of the tabu algorithm. Compared with the premutation route, the postmutation one has different potential FLs. If these links are in the optimal solution during the current optimization period (one optimization period means the duration from updating the parameters of FLs to satisfying the next FL update condition), they are candidate links to enter the FL base. The entry procedure includes the following steps.

(a) Identify the optimal vertex and those local optimal vertexes which have difference of less than $\Delta s_{\max}$ with the current optimal vertex in solution value. Figure 4 shows the profile of parts of current solutions in an optimization period, where the optimal values of vertexes A, B, and C are s_A, s_B, and s_C, respectively. B is the optimal vertex in this period, and $(s_A - s_B)/s_B < \Delta s_{\max}$ and $(s_C - s_B)/s_B < \Delta s_{\max}$.

(b) Recognize all local optimal vertexes and the local worst solution before the optimum occurs. As shown in Figure 4, the local worst solution vertexes before A, B, and C are A', B', and C', respectively.

(c) Compare routes of local optimums and the periodical optimum with routes of their corresponding worst vertexes, and then choose different links among them. Define $\Omega_{k,i}$, $i = 1, 2, \ldots, n_L^k$, as the set of different links between the ith local optimal route and its corresponding worst route in the optimization period k and n_L^k as the number of local optimal solutions (including the periodical optimum) in the period k. For example, in Figure 4, comparing routes of A and A', different links between them can be found. Similarly, different links between B and B' and between C and C' also can be obtained. Consider $\Omega_k = \{\Omega_{k,1}, \Omega_{k,2}, \ldots, \Omega_{k,n_L^k}\}$.

(d) Set the corresponding link set of the optimal solution in the optimization period k as Ω_B^k and the set of

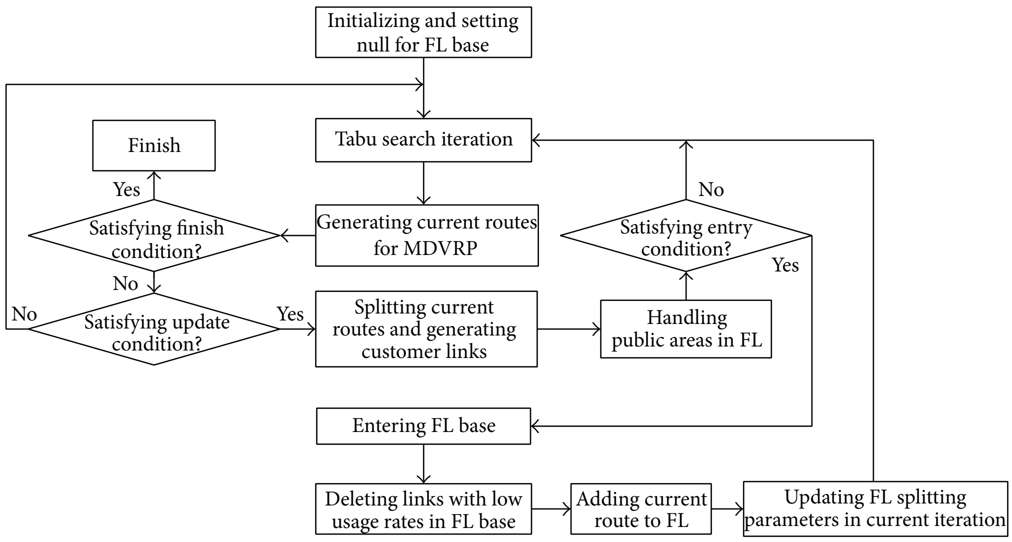
Initializing and setting null for FL base
Tabu search iteration
Finish
Yes
Satisfying finish condition?
No
Satisfying update condition?
No
Yes
Generating current routes for MDVRP
Splitting current routes and generating customer links
Handling public areas in FL
Satisfying entry condition?
No
Yes
Entering FL base
Deleting links with low usage rates in FL base
Adding current route to FL
Updating FL splitting parameters in current iteration

FIGURE 3: The basic structure of FL based tabu algorithm for LSMDVRP.

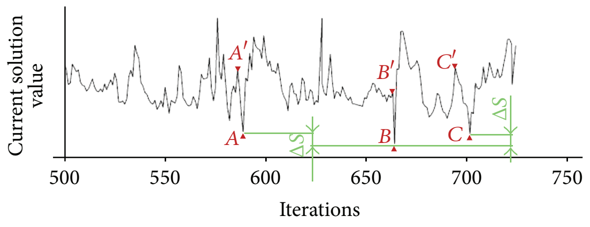

FIGURE 4: Generation of framing links.

candidate links to the FL base as Ω'_k, $\Omega'_k = \{l \mid l \in \Omega_k, l \in \Omega^k_B\}$. Different parts between A and A', B and B', and C and C' are extracted and compared, and links in the route of B are the candidate links to be added in the FL base.

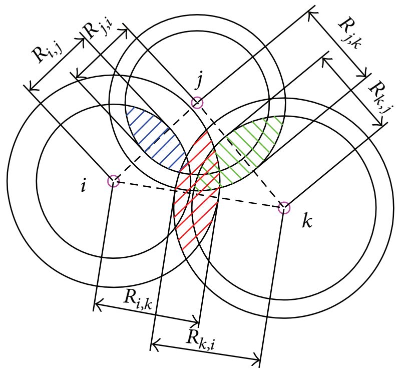

FIGURE 5: Illustration of the sharing area.

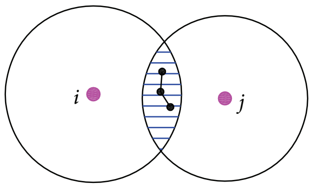

FIGURE 6: Link Type 1.

2.3.2. Selection of Links in the Sharing Area for the Base. In Figure 5, the area A with depot i as the center and $R_{i,j}$ ($i, j \in V_d, i \neq j$) as the radius is defined, and V_d is the total collection of all depots. The sharing area of two depots is $A_i \cap A_j$, $i, j \in V$, $i \neq j$, as shown in blue, red, and green shadow area. For vertexes outside the sharing area of two depots, the rule of generating candidate links in Section 2.3.1 is inappropriate.

Principles to generate candidate links for the FL base in the sharing area include the following.

(a) If all vertexes in a link are located in the sharing area of $A_i \cap A_j$, $i, j \in V_d$, $i \neq j$ (Link Type 1), this link cannot be taken as a candidate link, as shown in Figure 6.

(b) If vertexes of a link are spread in all following areas including $A_i \cap A_j$, A_i, A_j, $i, j \in V_d$, $i \neq j$ (Link Type 2), this link cannot be taken as a candidate link, as shown in Figure 7.

(c) If vertexes of a link are located in A_i and $A_i \cap A_j$, respectively, $i, j \in V_d$, $i \neq j$ (Link Type 3), this link can be a candidate link, as shown in Figure 8. If a link of Type 3 is added into the FL base, it can survive until the next N_cth generation and then will be inspected whether to be kept in the FL base or not. Meanwhile, the borderline of the sharing area will be adjusted with the rules in Section 2.3.3.

2.3.3. Adjusting Rules of Sharing Area. After the sharing area is initialized, its size and location will change with iterations, following the adhering rules.

(a) Sharing area initialization: $R_{i,j} = \delta_{i,j} d_{i,j}$, $i, j \in V_d$, $i \neq j$, $\delta_{i,j}$ is the borderline parameter of the sharing

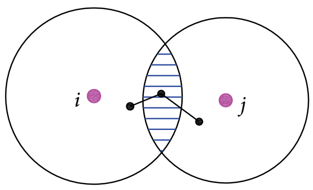

FIGURE 7: Link Type 2.

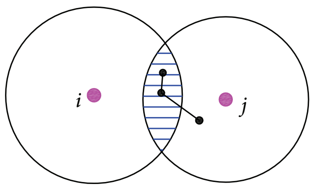

FIGURE 8: Link Type 3.

area, $d_{i,j}$ is the distance from depot i to depot j, and V_d is the collection of all depots.

(b) Adjusting rules of sharing area: since $d_{i,j}$ and $d_{j,i}$ remain constant, the area of $A_i \cap A_j$ is actually determined by $\delta_{i,j}$ and $\delta_{j,i}$ together. If there is a link of Type 3, the borderline of the sharing area will be adjusted. That means that if vertexes are located in the two areas A_i and $A_i \cap A_j$, $i, j \in V_d$, $i \neq j$, and this link is added to the current route after N_c generations and the current solution value is better than that before the N_cth generation, $\delta_{j,i} := \delta_{j,i} + \Delta\delta$ and $\Delta\delta$ is the updating step of the sharing area borderline coefficient; if the link is added to the current route after N_c generations but the current solution value is no better than that before the N_cth generation, $\delta_{j,i} := \delta_{j,i} - \Delta\delta$, as shown in Figure 9.

2.3.4. Selection of Qualified Links into the FL Base. Selection of qualified links into the FL base follows several principles.

(a) Amount restriction: a number of vertexes to enter the FL base are restricted into the range of $[L_{\min}, L_{\max}]$; ones out of this range are excluded.

(b) Minimized split: for those routes to be added in the base, except that the complete routes are kept, they also should be decomposed into links connecting $L_{\min}$ vertexes. Considering the case 3-7-4-1, $L_{\min} = 2$, this route needs to be decomposed into six FLs: 3-7-4-1, 3-7-4, 7-4-1, 3-7, 7-4, and 4-1.

(c) Backward generation: links in the FL base occur in pairs with opposite directions, so, once a link is selected into the FL base, its opposite link is generated synchronously. For example, if the link 3-7-4-1 is added, then a link 1-4-7-3 is generated.

(d) Entry: based on the minimized split results of an optimized route, If this route does not exist in the FL base, it will be added in.

(e) Update: once an achieved FL has more than $L_{\min}$ vertexes, it should be updated, as well as its subroutes.

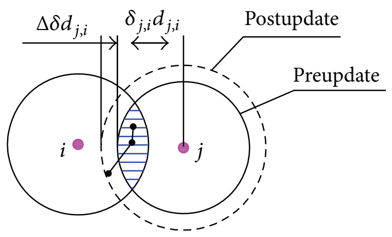

FIGURE 9: Update of the public area.

In above case, the occurrence frequencies and corresponding optimal solution values of 3-7-4-1, 3-7-4, 7-4-1, 3-7, 7-4, and 4-1 should be updated.

2.4. Deletion Principle in the Framing Link Base

(a) Comparison between links split from the current route is conducted every η iterations. After k generations, links before generation d have been compared for $c = [k/\eta]$ times. Set the usage frequency of a link as $u(c)$. If $u(c) < u_s(c)$, this link is deleted from the FL base. Here, $u_s(c)$ is the lower limit of FL usage, as a function of comparison times c.

(b) If the link is included in the current optimal solution, even if its usage is less than $u_s(c)$, it cannot be deleted.

2.5. Adding Framing Links into the Current Route. In the FL base, some link parameters are set: the optimal solution value of the route generated based on current links, the value of the worst solution, the average value of solutions, the usage times, the usage frequency, and the depot. Steps of adding FLs into the current route include the following.

(a) Computation of the link adaptability $f_i = 1/s_i^*$, $i = 1, 2, \ldots, N_l$: N_i is the amount of links in the base, and s_i^* is the optimal solution value of the route where the ith link lies in. Set H as the collection of links in the FL base, and $H' = H$, $H^* = \Phi$.

(b) Judgment on H': if it is equal to Φ, the procedure ends.

(c) Selection of link l_j with a roulette according to the value of f_i, $l_j \in H'$: set $H^* = H^* + \{l_j\}$.

(d) Calculation with $L = \{l_h \mid l_h \in H'$, there is $v \in l_j$ and $v \in l_h\}$, where v is a vertex included in the link: if $H' = H' - L$, go back to step 2.

3. Framing Link Based LSMDVRP Tabu Search Algorithm

In this study, the algorithm for MDVRP consists of two parts: initial optimization and follow-up optimization. Initial optimization is based on links extracted from the FL base, and extracted link is regarded as a vertex, not to be decomposed, so as to maximize FLs' advantages in generating optimized routes. On the other hand, follow-up optimization splits links individually, excluding non-FLs so as to avoid inferior solutions.

3.1. Initial Optimization

3.1.1. p-Neighborhood. The p-neighborhood of a vertex: The p vertexes or links in the nearest collection A of vertex v are the p-neighborhood of v, denoted by $N_p(v, A)$. If the element in $N_p(v, A)$ is a vertex, it is illustrated with v', and if it is a link, it is described with the link $v_l^s(z_{l,v}^s) - v_l^2(z_{l,v}^2) - v_l^3(z_{l,v}^3) - \cdots - v_l^e(z_{l,v}^e)$ in the neighborhood, where v_l^s represents the start vertex of link l and v_l^e represents the ending vertex. Furthermore, that $z_{l,v}^s = 0$ means that v_l^s is not in the p neighborhood of vertex v; otherwise, $z_{l,v}^s = 1$. As showed in Figure 10, if $p = 2$, $N_2(4) = \{8, 3(1) - 2(1) - 1(0)\}$ and $N_2(7) = \{9(1) - 10(1) - 11(0), 12(1) - 13(0)\}$.

The p-neighborhood of a link: p vertexes or links in the nearest collection A of the start vertex v_l^s of link l are the p-neighborhood of link l, represented with $N_p^s(l, A)$. At the same time, p vertexes or links in the nearest collection A of link l are the p-neighborhood of the ending vertex v_l^e of link l, represented with $N_p^e(l, A)$.

3.1.2. Insertion Method. For MDVRP with the coexistence of vertexes and links, there are three kinds of insertions: (a) insertion between two vertexes, (b) insertion between two links, and (c) insertion between a vertex and a link. The insertion method is similar to traditional insertion method and the only difference is that this method treats the link as a node. The specific content can refer to Solomon [38].

3.1.3. Generation of the Initial Solution. The generation of the initial solution includes the following steps:

Step 1. Allocate vertexes and links to initial depots. For a point, the nearest depot is regarded as the initial depot; for a link, the initial depot is the one where its initial route is included. $N_r(c_h)$ represents the collection of all unallocated vertexes of the depot h, $G(c_h, r)$ represents the total delivery amount of the rth route of the hth depot c_h, $G_{\max}(c_h, r)$ represents the delivery limit of vehicles in the rth route of the depot c_h, $g(v_k(\text{or } l_k))$ represents the delivery amount of vertex v_k or link l_k, and N_C is the amount of depots.

Step 2. Set $h = 0$.

Step 3. Consider $h := h + 1$.

Step 4. If $h > N_C$, go to Step 8.

Step 5. Set A_{c_h} as the collection of all vertexes or links belonging to the depot c_h, $\forall v_i\,(\text{or } l_i) \in N_p(c_h, A_{c_h})$, and $v_j\,(\text{or } l_j) \in N_p(c_h, A_{c_h})$. For the rth route $R(c_h, r)$ of the depot c_h, c_h-$v_i\,(\text{or } l_i)$-$v_j\,(\text{or } l_j)$-c_h. $G(c_h, r) = g(v_i\,(\text{or } l_i)) + g(v_j\,(\text{or } l_j))$, $N_r(c_h) := N_r(c_h) - v_i\,(\text{or } l_i) - v_j\,(\text{or } l_j)$.

Step 6. For all two neighboring vertexes and links $v_i\,(\text{or } l_i)$-$v_j\,(\text{or } l_j)$ in the route r, $v_k\,(\text{or } l_k) = \min\{v\,(\text{or } l) \mid \Delta d(v_i\,(\text{or } l_i), v\,(\text{or } l), v_j\,(\text{or } l_j))$. If $G(c_h, r) + g(v_k\,(\text{or } l_k)) > G_{\max}(h, r)$, update $r := r + 1$ and turn to Step 5. Otherwise, insert v_k (or l_k) to form a route of v_i (or l_i)-v_k (or l_k)-v_j (or l_j).

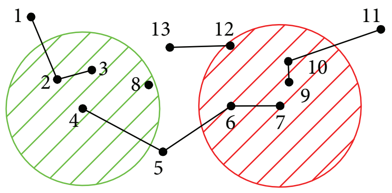

FIGURE 10: p-neighborhoods of vertexes and links.

Step 7. Update $G(c_h, r) := G(c_h, r) + g(v_k\,(\text{or } l_k))$, $N_r(c_h) := N_r(c_h) - v_k\,(\text{or } l_k)$. If $N(c_h) = \Phi$, turn to Step 3; otherwise, turn to Step 6.

Step 8. End.

3.1.4. Construction of Neighborhood. According to the characteristics of FL MDVRP, the authors introduce three neighborhood operators: insertion, interchange, and crossover.

Insertion. α is a random number in the range of $[\alpha_{\min}, \alpha_{\max}]$. In the route $R(c_h, r)$, if capacity constraint of $R(c_h, r)$ is satisfied, α vertexes or links are randomly selected in r, as v_i (or l_i) to v_j (or l_j). $A_{\overline{R(c_h,r)}}$ is set as the collection of vertexes or links outside the route $R(c_h, r)$, and, according Section 3.1.2, $v_k\,(\text{or } l_k) \in A_{\overline{R(c_h,r)}}$ is selected.

Interchange. θ is a random number in the range of $[\theta_{\min}, \theta_{\max}]$, as well as φ in $[\varphi_{\min}, \varphi_{\max}]$. Choose two routes: $R(c_h, r_1)$ and $R(c_h, r_2)$, and set $A_{R(c_h,r_1)}$ and $A_{R(c_h,r_2)}$ as the collection of vertexes and links in $R(c_h, r_1)$ and $R(c_h, r_2)$. θ vertexes and links in $R(c_h, r_1)$ are selected randomly, and then $v_m\,(\text{or } l_m) \in A_{R(c_h,r_2)}$ is obtained based on the method in Section 3.1.2, $m = 1, 2, \ldots, \theta$. Similarly, φ vertexes and links in $R(c_h, r_2)$ are selected. If neither capacity constraint of $R(c_h, r_1)$ nor that of $R(c_h, r_2)$ is unsatisfied after insertions, v_m (or l_m) can be inserted into the corresponding place in $A_{R(c_h,r_1)}$, or v_n (or l_n) in $A_{R(c_h,r_2)}$. Otherwise, v_m (or l_m) and v_n (or l_n) will be selected again.

Crossover. γ_1 and γ_2 are random numbers in the range of $[\gamma_{\min}, \gamma_{\max}]$, where $\gamma_1 < \gamma_2$, and the random number λ is in the range of $[\lambda_{\min}, \lambda_{\max}]$. In the route of $R(c_h, r_1)$, a segment s_1 from γ_1 to γ_2 is extracted. Meanwhile, segment $s^* = \{s \mid \min d(c_{s_1}, c_s)\}$ with a distance of λ is selected in a neighboring route of $R(c_h, r_1)$, where $d(c_{s_1}, c_s)$ represents the distance from the center of segment s_1 to the center of the whole route s. If the exchange of s_1 and s^* to the counterpart routes cannot lead to the capacity unconstraint of these two routes, this transform is allowed; otherwise, new γ_1 and γ_2 need to be selected.

3.2. Follow-Up Optimization. Although the FL method prompts to generate more desired solutions, FLs in the initial

Table 1: Column headings for Tables 2–10.

I	Instance ID
n	Number of customers
m	Number of depots
Q	Vehicle capacity
D	Maximum duration of a route
$c(s^*)$	Previous best known solution cost
$c(s^{**})$	Best known solution found by proposed tabu search algorithm on FL
$L_{\min}, L_{\max}$	Min and max amount limitation for vertexes to be added into the FL base
$\delta_{i,j}$	Borderline parameter of the sharing area
$\Delta\delta$	Updating step of $\delta_{i,j}$
$\Delta s_{\max}$	Max difference between optimal vertex and local optimal vertexes in an optimization period
n_Y	Number of optimization periods
n_L	The total number of local optimal vertexes
Ω'	The total number of candidate links
ρ_1^i	The proportion of customers in FLs to total customers in optimization period i
ρ_2^j	The proportion of customers in FLs to total customers in local optimal vertex j
ω_1^i	The optimal solution value of optimization period i
ω_2^j	The value local optimal solution j

Table 2: MDVRP instances and previous best known solution cost.

I	source	n	Q	D	m	GJ	CGW	CGL	$c(s^*)$
p08	GJ	249	500	310	2	4832.0	4511.6	4437.68	4437.68
p09	GJ	249	500	310	3	4219.7	3950.9	3900.22	3900.22
p10	GJ	249	500	310	4	3822.0	3727.1	3663.02	3663.02
p11	GJ	249	500	310	5	3754.1	3670.2	3554.18	3554.18
p15	CGW	160	60	0	4	—	2610.3	2505.42	2505.42
p16	CGW	160	60	200	4	—	2605.3	2572.23	2572.23
p17	CGW	160	60	180	4	—	2816.6	2709.09	2709.09
p18	CGW	240	60	0	6	—	3877.4	3702.85	3702.85
p19	CGW	240	60	200	6	—	3863.9	3827.06	3827.06
p20	CGW	240	60	180	6	—	4272.0	4058.07	4058.07
p21	CGW	360	60	0	9	—	5791.5	5474.84	5474.84
p22	CGW	360	60	200	9	—	5857.4	5702.16	5702.16
p23	CGW	360	60	180	9	—	6494.6	6095.46	6095.46
pr04	CGL	192	185	440	4	—	—	2072.52	2072.52
pr05	CGL	240	180	420	4	—	—	2385.77	2385.77
pr06	CGL	288	175	400	4	—	—	2723.27	2723.27
pr09	CGL	216	180	450	6	—	—	2153.1	2153.1
pr10	CGL	288	170	425	6	—	—	2921.85	2921.85

optimization are not necessarily parts of routes in the optimized solution; consequently, follow-up operation is needed to iterate continuously after splitting links into individual vertexes. Once the optimization solution is not yet updated ξ generations after the initial optimization, follow-up optimization should be introduced. The initial solution of the follow-up optimization is the optimal one of the initial optimization, and if this initial solution has occurred in previous periods, the second optimal solution of the initial optimization can be selected.

4. Computational Experiments

The algorithm proposed above is programmed in MS Visual C++ 6.0 and tested with a PC with a AMD Athlon (tm) X2 2.0 GHz CPU and 2 GB RAM. In this paper, the authors solve problems in existing literature with this new algorithm and then solved different LSMDVRPs with different parameters, in order to identify the coefficient valuing discipline.

The authors test the algorithm based on 18 instances from literature. Instances of p8–p11 were provided in Gillett and

TABLE 3: Sensitivity data of parameters $L_{\min}$ and $L_{\max}$.

I	n	m	$[L^*_{\min}, L^*_{\max}]$	Solution value % gap	CPU time % gap
p08	249	2	[3, 7]	3.37	403.23
p09	249	3	[3, 6]	4.6	476.87
p10	249	4	[3, 6]	7.62	322.33
p11	249	5	[3, 5]	6.26	398.74
p15	160	4	[2, 4]	5.06	366.86
p16	160	4	[2, 4]	3.01	485.78
p17	160	4	[2, 4]	3.1	573.27
p18	240	6	[2, 4]	6.53	397.2
p19	240	6	[2, 4]	4.61	676.61
p20	240	6	[2, 4]	7.85	238.97
p21	360	9	[2, 4]	8.18	291.3
p22	360	9	[2, 4]	6.76	334.59
p23	360	9	[2, 4]	6.14	618.93
pr04	192	4	[2, 5]	7.57	474.76
pr05	240	4	[3, 6]	3.17	356.43
pr06	288	4	[3, 6]	4.31	649.79
pr09	216	6	[2, 4]	7.55	414.62
pr10	288	6	[2, 4]	3.62	689.48

TABLE 4: Sensitivity data of parameter $\delta_{i,j}$ and its update step $\Delta\delta$.

I	n	m	$\delta_{i,j}{}^*$	$\Delta\delta^*$	Solution value		CPU time	
					% gap 1	% gap 2	% gap 3	% gap 4
p08	249	2	0.85	0.10	11.54	1.51	11.81	29.2
p09	249	3	0.75	0.08	9.82	1.79	19.13	34.48
p10	249	4	0.65	0.04	4.35	1.57	10.83	30.79
p11	249	5	0.65	0.04	17.68	1.2	16.61	51.14
p15	160	4	0.65	0.04	3.24	1.69	7.99	26.85
p16	160	4	0.65	0.06	16.32	1.95	8.68	41.14
p17	160	4	0.65	0.04	9.49	1.43	14.09	45.51
p18	240	6	0.7	0.06	9.59	2.59	12.67	48.92
p19	240	6	0.65	0.04	18.23	1.01	19.57	37.92
p20	240	6	0.65	0.06	13.66	1.99	17.61	44.74
p21	360	9	0.65	0.04	6.95	2.84	4.9	22.24
p22	360	9	0.65	0.04	8.49	1.9	19.68	57.11
p23	360	9	0.65	0.04	10.8	2.55	15.73	21.33
pr04	192	4	0.65	0.04	18.3	2.52	8.75	50.53
pr05	240	4	0.7	0.06	11.59	1.59	14.52	25.42
pr06	288	4	0.75	0.08	12.44	1.13	13.44	35.39
pr09	216	6	0.6	0.04	6.96	1.73	19.96	46.54
pr10	288	6	0.65	0.04	17.1	2.07	12.92	57.18

% Gap 1: % Gap between the worst and the best solution value with different $\delta_{i,j}$ and $\Delta\delta = 0.04$.
% Gap 2: % Gap between the worst and the best solution value with different $\Delta\delta$ and $\delta_{i,j}{}^*$.
% Gap 3: % Gap between the longest and the shortest CPU time with different $\delta_{i,j}$ and $\Delta\delta = 0.04$.
% Gap 4: % Gap between the longest and the shortest CPU time with different $\Delta\delta$ and $\delta_{i,j}{}^*$.

Johnson [16], p15–p23 by Chao et al. [39], and pr04–pr6, pr9, and pr10 by Cordeau et al. [25]. Table 1 is the column headings for Tables 2, 3, 4, 5, 6, 7, 8, 9, and 10.

The characteristics of the instances are summarized in Table 2 and the complete data sets and best known results are given in the website http://neo.lcc.uma.es/radi-aeb/WebVRP/index.html. The costs of the best solutions found by each method are listed in Table 2.

4.1. Sensitivity Analyses

4.1.1. $[L_{\min}, L_{\max}]$. Firstly, all other parameters are set as constants. $L_{\max}$ is selected from (3, 4, 5, 6, 7) and $L_{\min}$ is from (2, 3, 4), given $L_{\max} \geq L_{\min}$. Each of the 18 instances is calculated for 14 times, and the results are listed in Table 3.

It is not hard to find that the value of $[L_{\min}, L_{\max}]$ has not much influence on solutions, and average gap between

Table 5: The value range of $\delta_{i,j}{}^*$ and $\Delta\delta^*$.

n/m	$\delta_{i,j}{}^*$	$\Delta\delta^*$
≤50	[0.6, 0.7]	[0.04, 0.06]
≤100	(0.7, 0.8]	(0.06, 0.08]
>100	(0.8, 0.9]	(0.08, 0.10]

Table 6: The calculation results of parameter $\Delta s_{\max}$.

I	n	m	$\Delta s^*_{\max}$	n_Y	n_L	Ω'
p08	249	2	0.03	3	17	201
p09	249	3	0.04	5	18	193
p10	249	4	0.03	7	21	187
p11	249	5	0.03	7	42	422
p15	160	4	0.06	5	18	143
p16	160	4	0.07	5	18	159
p17	160	4	0.06	5	17	129
p18	240	6	0.05	8	31	255
p19	240	6	0.06	9	48	646
p20	240	6	0.05	10	47	531
p21	360	9	0.02	10	45	559
p22	360	9	0.02	11	46	315
p23	360	9	0.02	9	43	302
pr04	192	4	0.06	5	23	309
pr05	240	4	0.03	6	36	400
pr06	288	4	0.03	6	21	215
pr09	216	6	0.05	8	27	232
pr10	288	6	0.04	7	41	343

Table 7: The value of n_Y, n_L, and Ω' and the accuracies of Ω' with different $\Delta s_{\max}$ in the instance p09.

$\Delta s^*_{\max}$	n_Y	n_L	Ω'	Ω' accuracy
0.01	3	5	58	37.77%
0.02	3	8	101	38.37%
0.03	4	11	131	40.71%
0.04	5	18	193	42.40%
0.05	9	23	287	32.05%
0.06	10	26	402	23.77%
0.07	12	27	606	21.96%
0.08	12	40	726	15.59%
0.09	13	55	958	10.02%
0.1	15	101	974	11.23%

solutions is 5.52%. Inferior solutions always occur in the condition where $L_{\min}$ is relatively large and $L_{\min} = L_{\max}$; for example, $[L_{\min}, L_{\max}] = [5, 5]$. However, the value of $[L_{\min}, L_{\max}]$ greatly affects the computation speed of the algorithm, with an average gap of 453.88%, since $L_{\max}$ can control the upper limits of lengths of candidate links, and $L_{\min}$ determines their lower limits as well as split numbers. With the decrease in $L_{\min}$, the number of splits increases with an exponential distribution. According to this case, the relation of $[L^*_{\min}, L^*_{\max}]$ and n/m is revealed: a larger n/m leads to larger $L^*_{\min}$ and $L^*_{\max}$. When the value of n/m is less than 40, the commendation value of $[L^*_{\min}, L^*_{\max}]$ is $[2, 4]$, when $40 < n/m \leq 100$, the value is $[3, 5]$ or $[3, 6]$, and when n/m is more than 100, the value is $[3, 7]$.

4.1.2. Borderline Parameter $\delta_{i,j}$ and Update Step $\Delta\delta$. Keeping other parameters unchanged, the authors measure the effect of $\delta_{i,j}$ here. Firstly, $\Delta\delta = 0.04$, $\delta_{i,j}$ is in [0.5, 0.9], the interval between successive $\delta_{i,j}$ is 0.05, and then $\delta_{i,j}{}^*$ for 18 instances are obtained through 9 times of calculations for each instance. Next, $\Delta\delta$ is taken varied in [0.02, 0.12], $\delta_{i,j}{}^*$ are allocated to these instances, and the interval for $\Delta\delta$ is defined as 0.02. $\Delta\delta^*$ for 18 instances are achieved through 6 times of calculations for each one.

According to Table 4, given $\Delta\delta$ is a constant, difference between the best and the worst solution values of different $\delta_{i,j}$ is 11.48%. If $\delta_{i,j}$ is regarded as a constant, difference between the best and the worst solution values of different $\Delta\delta$ only is 1.84%. Such results illustrate that valuing of $\delta_{i,j}$ has a more obvious impact on the computation result of the algorithm and that of $\Delta\delta$. Similarly, valuing of $\delta_{i,j}$ is much more influential on the computation efficiency of the algorithm (average difference is 39.25%) than that of $\Delta\delta$ (average difference is 13.83%). The values of $\delta_{i,j}{}^*$ and $\Delta\delta$ are proportional to the customer numbers allocated in each depot (n/m). The approximate value range of $\delta_{i,j}{}^*$ and $\Delta\delta^*$ is shown in Table 5.

4.1.3. $\Delta s_{\max}$. Set $[L_{\min}, L_{\max}] = [L^*_{\min}, L^*_{\max}]$, $\Delta s_{\max}$ in [0.01, 0.10], and the interval between different $\Delta s_{\max}$ as 0.01. The authors calculate 10 times for each instance, the number of optimization periods is n_Y, the number of local optimal vertexes in optimization period k is n^k_L, $n_L = \sum_{k=1}^{n_Y} n^k_L$, Ω'_k is the number of candidate links produced in optimization period k, and $\Omega' = \sum_{k=1}^{n_Y} \Omega'_k$. The calculation results are shown in Table 6.

According to the analysis results, $\Delta s_{\max}$ can control the number of local optimal vertexes n_L, and the value of $\Delta s_{\max}$ is inversely proportional to n. This is because global optimization is difficult with the increase in n and more local optimal vertexes attained. As a result, a smaller $\Delta s_{\max}$ is needed to restrict the number of n_L, so as to control Ω'. Although a larger Ω' raises the possibilities to obtain FLs, it lowers the average quality of candidate links as well, and so some "bad" links corresponding to local optimal vertexes far from the periodical vertex are involved in the candidate link base. Once these unexpected links enter the FL base, the global optimization will be staggered and its results will be weakened. In this study, a proper $\Delta s_{\max}$ for computation scale and characteristics is needed. Table 7 lists the value of n_Y, n_L, and Ω' and the accuracies of Ω' with different $\Delta s_{\max}$ in the instance p09.

As shown in Table 7, when $\Delta s_{\max}$ is at its best value $\Delta s^*_{\max}$, Ω' is the most accurate, so as to guarantee the most efficient and effective optimization. Based on this study, the recommendation value of $\Delta s^*_{\max}$ is [0.06, 0.07] when $150 < n \leq 200$, [0.03, 0.06] when $200 < n \leq 300$, and [0.02, 0.03] when $300 < n \leq 400$.

Table 8: Numbers of FLs in every optimization period.

I	Optimization period										
	1	2	3	4	5	6	7	8	9	10	11
p08	20	34	38	—	—	—	—	—	—	—	—
p09	13	16	24	28	30	—	—	—	—	—	—
p10	11	15	19	26	30	33	34	—	—	—	—
p11	15	17	19	23	28	30	33	—	—	—	—
p15	12	18	22	26	27	—	—	—	—	—	—
p16	15	17	25	31	31	—	—	—	—	—	—
p17	16	21	25	27	27	—	—	—	—	—	—
p18	15	18	19	23	28	32	32	33	—	—	—
p19	13	18	22	27	32	37	37	38	40	—	—
p20	10	13	12	17	23	27	32	34	36	36	—
p21	11	15	19	27	31	35	38	45	48	50	—
p22	12	17	21	27	33	39	46	51	52	52	53
p23	12	15	19	25	33	43	47	54	56	—	—
pr04	14	21	25	24	25	—	—	—	—	—	—
pr05	16	25	34	39	39	40	—	—	—	—	—
pr06	15	19	25	32	36	38	—	—	—	—	—
pr09	13	18	23	27	31	32	33	35	—	—	—
pr10	19	22	29	34	34	35	37	—	—	—	—

Table 9: The correlation of ρ_1^i, ω_1^i, ρ_2^j, and ω_2^j.

I	n_Y	n_L	Correlation coefficient of ρ_1^i and ω_1^i	Correlation coefficient of ρ_2^j and ω_2^j
p08	3	17	−0.92	−0.91
p09	5	18	−0.91	−0.99
p10	7	21	−0.97	−0.96
p11	7	42	−0.97	−0.97
p15	5	18	−0.89	−0.89
p16	5	18	−0.97	−0.90
p17	5	17	−0.88	−0.93
p18	8	31	−0.95	−0.90
p19	9	48	−0.92	−0.95
p20	10	47	−0.93	−0.98
p21	10	45	−0.88	−0.95
p22	11	46	−0.90	−0.96
p23	9	43	−0.91	−0.93
pr04	5	23	−0.96	−0.91
pr05	6	36	−0.90	−0.89
pr06	6	21	−0.96	−0.95
pr09	8	27	−0.90	−0.92
pr10	7	41	−0.92	−0.90

4.2. FLs and Optimization Results. With the best parameters in Section 4.1, the generation process of FLs in the optimization is tracked. Set ρ_1^i as the proportion of customers in FLs to total customers in each optimization period, ω_1^i as the optimal solution value of optimization period i, $i = 1, 2, \ldots, n_Y$, ρ_2^j as the proportion of customers in FLs to total customers in each local optimal vertex, and ω_2^j as the value local optimal solution j, $j = 1, 2, \ldots, n_L$. Table 8 shows numbers of FLs in every optimization period.

In Tables 8 and 9, the following can be found. (a) Average numbers of FLs increase with the progress of optimization, to the most in the last optimization period, and the optimal solution is achieved at the same time. (b) Increasing speed of FL numbers falls down gradually in the optimization process. Due to the existence of sharing area, when the proportion of customers on FLs reaches a certain number, the increase in its absolute amount will be slowed. (c) The optimal solution keeps updating with optimization. Although FLs do not increase rapidly, the optimal search ability is not challenged.

Table 10: The results found by proposed tabu search algorithm on FL.

I	n	$c(s^*)$	$c(s^{**})$	Average CPU time (min)	% CPU time	Average solution value	% solution value	% above best known solution
p08	249	4437.68	**4417.46**	10.40	27.04	4465.69	1.08	0.46
p09	249	3900.22	**3886.82**	10.32	28.60	3937.76	1.29	0.34
p10	249	3663.02	**3647.22**	9.61	28.20	3684.78	1.02	0.43
p11	249	3554.18	**3547.70**	9.20	12.03	3611.16	1.76	0.18
p15	160	2505.42	2513.43	2.88	8.31	2528.42	0.59	−0.32
p16	160	2572.23	2578.88	4.04	9.03	2591.07	0.47	−0.26
p17	160	2709.09	2709.09	3.43	11.86	2710.19	0.04	0
p18	240	3702.85	3712.61	4.22	12.45	3719.78	0.19	−0.26
p19	240	3827.06	3827.06	3.73	12.58	3853.29	0.68	0
p20	240	4058.07	4066.85	5.33	18.03	4078.59	0.29	−0.22
p21	360	5474.84	5483.31	7.72	20.46	5518.1	0.63	−0.15
p22	360	5702.16	5708.12	8.05	11.57	5743.57	0.62	−0.10
p23	360	6095.46	6095.46	7.07	17.23	6136.23	0.66	0
pr04	192	2072.52	**2058.80**	9.63	32.54	2081.42	1.09	0.66
pr05	240	2385.77	**2369.00**	8.59	19.30	2405.35	1.51	0.70
pr06	288	2723.27	**2718.11**	13.25	19.24	2763.24	1.63	0.19
pr09	216	2153.1	**2147.40**	8.04	13.21	2182.06	1.59	0.26
pr10	288	2921.85	**2865.31**	10.88	22.79	2934.16	2.35	1.94

% CPU time: the gap in percentage between the average CPU time and the fastest CPU time among the ten runs.
% solution value: the gap in percentage between the average value of the solutions and the best solution value among the ten runs.

In the latter periods of optimization, low quality FLs are likely to be replaced by high quality ones. (d) the average correlation coefficient between ρ_1^i and ω_1^i reaches −0.924, and the number between ρ_2^j and ω_2^j is −0.933 for all 18 instances. Such a result apparently bridges the proportion of customers in FLs and their corresponding local optimal solutions, so as to prove that the improvement of ρ_1^i and ρ_2^j prompts to achieve better optimal solutions.

4.3. Optimization Results. The 18 instances are calculated for 10 times based on parameters recommended above, respectively, and the optimization results are shown in Table 10. Figure 11 describes the corresponding route of the optimal solution for p09.

In 9 of all 18 instances, new optimal solutions are obtained. The average fluctuation value of optimal solution in 18 instances is 0.97%. This fluctuation value consolidates the independence of optimization results to initial solutions in the proposed algorithm, as well as its effectiveness and robustness.

Scales of all 18 instances are large enough ($150 < n \leq 360$), but the average computation duration is only 7.58 min with a little fluctuation rate of 18.03% in 10 calculations. All these results show the improvement of this algorithm in optimization efficiency.

For p15–p23, no better route is attained, and values of the average computation time, fluctuation rate of computation time, and optimal solutions are lower than the average one in the total 18 instances. The reason is that regular distributions and similar requirements of customers in these instances reduce the optimization space and difficulty.

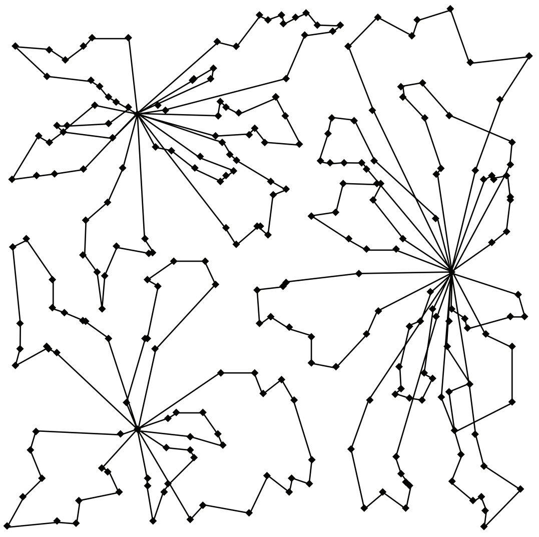
Figure 11: The corresponding route of the optimal solution for p09.

5. Conclusions

The authors propose a new tabu algorithm to optimize large-scale multidepot routes, using FLs as skeleton to improve search ability, speed up optimization, and then obtain better solutions. The validity of the new algorithm has been verified by example of verification.

FL is a new concept proposed in view of the VRP. However, in terms of the present study level, it is hard to get an algorithm and recommended parameters applying to all VRP constraints because of the complexity of VRP. Studies of this paper only aimed at large-scale VRP and capacity constraints. For the VRP in other conditions, this paper has certain reference value on the way of thinking; however, the specific operation of operator, particularly the recommended value parameters, needs analysis case by case. In principle, the effectiveness of FL is the result of joint action of many factors and the designs and parameter values of different operators are not unique. As the research of FL under different constraint conditions of VRP and influence factors continues, more complete processing methods and recommended parameters will gradually form.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

Data of 18 instances is from the website http://neo.lcc.uma.es/radi-aeb/WebVRP/index.html. This research is supported by National Natural Science Foundation of China (no. 71371134). Meanwhile, the authors appreciate the help and comments from editors and reviewers.

References

[1] C. A. Ullrich, "Integrated machine scheduling and vehicle routing with time windows," *European Journal of Operational Research*, vol. 227, no. 1, pp. 152–165, 2013.

[2] M. S. Rasmussen, T. Justesen, A. Dohn, and J. Larsen, "The home care crew scheduling problem: preference-based visit clustering and temporal dependencies," *European Journal of Operational Research*, vol. 219, no. 3, pp. 598–610, 2012.

[3] J. Wilck and T. Cavalier, "A genetic algorithm for the split delivery vehicle routing problem," *Physics & Mathematics*, vol. 2, pp. 207–216, 2012.

[4] J. Riera-Ledesma and J. Salazar-González, "Solving school bus routing using the multiple vehicle traveling purchaser problem: a branch-and-cut approach," *Computers and Operations Research*, vol. 39, no. 2, pp. 391–404, 2012.

[5] A. Lim, B. Rodrigues, and Z. Xu, "Transportation procurement with seasonally varying shipper demand and volume guarantees," *Operations Research*, vol. 56, no. 3, pp. 758–771, 2008.

[6] R. Agarwal and Ö. Ergun, "Ship scheduling and network design for cargo routing in liner shipping," *Transportation Science*, vol. 42, no. 2, pp. 175–196, 2008.

[7] B. Kim, S. Kim, and S. Sahoo, "Waste collection vehicle routing problem with time windows," *Computers and Operations Research*, vol. 33, no. 12, pp. 3624–3642, 2006.

[8] S. S. Syam, "A multiple server location-allocation model for service system design," *Computers & Operations Research*, vol. 35, no. 7, pp. 2248–2265, 2008.

[9] J. F. Campbell, A. T. Ernst, and M. Krishnamoorthy, "Hub arc location problems: part I: introduction and results," *Management Science*, vol. 51, no. 10, pp. 1540–1555, 2005.

[10] J. K. Lenstra and A. H. G. Rinnooy Kan, "Complexity of vehicle routing and scheduling problem," *Networks*, vol. 11, no. 2, pp. 221–227, 1981.

[11] R. Hassin and S. Rubinstein, "On the complexity of the k-customer vehicle routing problem," *Operations Research Letters*, vol. 33, no. 1, pp. 71–76, 2005.

[12] A. Imai, E. Nishimura, and J. Current, "A Lagrangian relaxation-based heuristic for the vehicle routing with full container load," *European Journal of Operational Research*, vol. 176, no. 1, pp. 87–105, 2007.

[13] M. M. Solomon, "On the worst-case performance of some heuristics for the vehicle routing and scheduling problem with time window constraints," *Networks*, vol. 16, no. 2, pp. 161–174, 1986.

[14] H. Hashimoto, T. Ibaraki, S. Imahori, and M. Yagiura, "The vehicle routing problem with flexible time windows and traveling times," *Discrete Applied Mathematics. The Journal of Combinatorial Algorithms, Informatics and Computational Sciences*, vol. 154, no. 16, pp. 2271–2290, 2006.

[15] M. W. P. Savelsbergh, "Local search in routing problems with time windows," *Annals of Operations Research*, vol. 4, no. 1–4, pp. 285–305, 1985.

[16] B. E. Gillett and J. G. Johnson, "Multi-terminal vehicle-dispatch algorithm," *Omega*, vol. 4, no. 6, pp. 711–718, 1976.

[17] B. Golden, T. Magnanti, and H. Nguyen, "Implementing vehicle routing algorithms," *Networks*, vol. 7, pp. 113–148, 1977.

[18] S. Salhi and M. Sari, "A multi-level composite heuristic for the multi-depot vehicle fleet mix problem," *European Journal of Operational Research*, vol. 103, no. 1, pp. 95–112, 1997.

[19] R. T. Sumichras and I. S. Markham, "A heuristic and lower bound for a multi-depot routing problem," *Computers & Operations Research*, vol. 22, no. 10, pp. 1047–1056, 1995.

[20] M. Wasner and G. Zäpfel, "An integrated multi-depot hub-location vehicle routing model for network planning of parcel service," *International Journal of Production Economics*, vol. 90, no. 3, pp. 403–419, 2004.

[21] G. Nagy and S. Salhi, "Heuristic algorithms for single and multiple depot vehicle routing problems with pickups and deliveries," *European Journal of Operational Research*, vol. 162, no. 1, pp. 126–141, 2005.

[22] A. Lim and F. Wang, "Multi-depot vehicle routing problem: a one-stage approach," *IEEE Transactions on Automation Science and Engineering*, vol. 2, no. 4, pp. 397–402, 2005.

[23] M. Mirabi, S. M. T. F. Ghomi, and F. Jolai, "Efficient stochastic hybrid heuristics for the multi-depot vehicle routing problem," *Robotics and Computer-Integrated Manufacturing*, vol. 26, no. 6, pp. 564–569, 2010.

[24] C. Contardo and R. Martinelli, "A new exact algorithm for the multi-depot vehicle routing problem under capacity and route length constraints," *Discrete Optimization*, vol. 12, pp. 129–146, 2014.

[25] J. Cordeau, M. Gendreau, and G. Laporte, "A tabu search heuristic for periodic and multi-depot vehicle routing problems," *Networks*, vol. 30, no. 2, pp. 105–119, 1997.

[26] J. Cordeau, G. Laporte, and A. Mercier, "A unified tabu search heuristic for vehicle routing problems with time windows," *Journal of the Operational Research Society*, vol. 52, no. 8, pp. 928–936, 2001.

[27] J. Renaud, G. Laporte, and F. F. Boctor, "A tabu search heuristic for the multi-depot vehicle routing problem," *Computers and Operations Research*, vol. 23, no. 3, pp. 229–235, 1996.

[28] B. Crevier, J. Cordeau, and G. Laporte, "The multi-depot vehicle routing problem with inter-depot routes," *European Journal of Operational Research*, vol. 176, no. 2, pp. 756–773, 2007.

[29] S. Belhaiza, P. Hansen, and G. Laporte, "A hybrid variable neighborhood tabu search heuristic for the vehicle routing problem with multiple time windows," in *Proceedings of the International Conference on Industrial Engineering and Systems Management (IESM '11)*, Metz, France, 2011.

[30] S. Bae, H. S. Hwang, G. Cho, and M. Goan, "Integrated GA-VRP solver for multi-depot system," *Computers and Industrial Engineering*, vol. 53, no. 2, pp. 233–240, 2007.

[31] W. Ho, G. T. S. Ho, P. Ji, and H. C. W. Lau, "A hybrid genetic algorithm for the multi-depot vehicle routing problem," *Engineering Applications of Artificial Intelligence*, vol. 21, no. 4, pp. 548–557, 2008.

[32] X. Wang, C. Xu, and H. Shang, "Multi-depot vehicle routing problem with time windows and multi-type vehicle number limits and its genetic algorithm," in *Proceedings of the International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM '08)*, pp. 1–5, Dalian, China, October 2008.

[33] Y. Li and X. Liu, "Modelling and its ant colony algorithm for multi-depot open vehicle routing problem with replenishment on the way," *Computer Integrated Manufacturing Systems*, vol. 14, no. 3, pp. 557–562, 2008.

[34] T. Wu, C. Low, and J. Bai, "Heuristic solutions to multi-depot location-routing problems," *Computers and Operations Research*, vol. 29, no. 10, pp. 1393–1415, 2002.

[35] S. Chen, A. Imai, and B. Zhao, "A SA-based heuristic for the multi-depot vehicle routing problem," *Journal of Japan Institute of Navigation*, vol. 113, pp. 209–216, 2005.

[36] C. D. Tarantilis and C. T. Kiranoudis, "BoneRoute: an adaptive memory-based method for effective fleet management," *Annals of Operations Research*, vol. 115, pp. 227–241, 2002.

[37] S. Zhong, "Open vehicle routing problem based on kernel route tabu search algorithm," *Computer Integrated Manufacturing Systems*, vol. 13, no. 4, pp. 827–832, 2007.

[38] M. M. Solomon, "Algorithms for the vehicle routing and scheduling problems with time window constraints," *Operations Research*, vol. 35, no. 2, pp. 254–265, 1987.

[39] I. Chao, B. Golden, and E. Wasil, "A new heuristic for the multi-depot vehicle routing problem that improves upon best-known solutions," *American Journal of Mathematical and Management Sciences*, vol. 13, pp. 371–406, 1993.