Journal of Computers Vol. 28, No. 6, 2017, pp. 1-12
doi:10.3966/199115992017122806001

MN_GLS for VRP with Simultaneous Delivery and Pickup

Haodong Zhu[1], Jiamei Feng[1] and Hongchan Li[1*]

[1] School of Computer and Communication Engineering
Zhengzhou University of Light Industry
Zhengzhou, Henan, 450002, China
{zhuhaodong80, ketizulunwen, lihongchan126}@163.com

Received 1 July 2016; Revised 21 February 2017; Accepted 21 February 2017

Abstract. This Paper proposed a Multiple Neighborhood Guided Local Search Algorithm (MN_GLS) to solve vehicle routing problem (VRP) with simultaneous delivery and pickup. Firstly, it used the nearest neighbor method to build the initial solution. Secondly, it did the local search in multi-operator from the initial solution, and found the bow which had the biggest utility of punishment value when the solution fail into the local optimal solution, then changed the punishment of features value and objective function value. Thirdly, it selected the current optimal solution from the local optimal solutions, and then did the local optimization in multi-operator again from the current optimal solution which has the new objective function value. By means of 54 examples, the simulation results illustrate that MN_GLS is an effective and stabilize method for Vehicle Routing Problem with Simultaneous Delivery and Pickup.

Keywords: guided local search, multiple neighborhood, penalty strategy, vehicle routing problem

1 Introduction

Vehicle routing problem (Capacitated Vehicle Routing Problem, VRP) generally exists in logistics transportation of postal service, manufacturing, electric, and so on, which aims to reduce logistics costs and improve customer satisfaction [1]. In recent decades, the problem has been put forward as a research hot spot in computational science, operational, and so on. Experts and scholars at home and abroad in recent years usually use heuristic algorithm to solve the problem, and a lot of achievements have been gotten.

The VRP that is concerned with the optimal set of routes for a fleet of vehicles to traverse in order to deliver to a given set of clients. A widely studied generalization of VRP, the vehicle routing problem with simultaneous pickup and delivery (VRPSPD), proposed by Min [2], requires simultaneous consideration of both pickup and delivery demands. This causes a fluctuation in the current load of the vehicle which results in increased difficulty in checking the feasibility of the solutions. Therefore, a key aspect is to check the current load of the vehicle(s) at each client since the vehicle capacity cannot not be exceeded [3]. A well-known example occurs in the soft drink industry where the operations of delivering full bottles and picking empty ones up are performed by the same vehicle [4]. A further extension of VRPSPD, vehicle routing problem with simultaneous pickup and delivery with time limit (VRPSPDTL), additionally requires the vehicles to return to the central depot before a time deadline is reached [5]. A real life example occurs in milk transportation since there is a limited time to carry such sensitive goods on the route before turning bad. Mathematical formulation of VRPSPD and VRPSPDTL can be found in Alfredo Tang Montané and Galvão [6] and Polat, Kalayci, Kulak, and Günther [7], respectively.

Since VRPSPD works were started by Min [2] with a case study of a small sized book distribution problem that requires book delivery and pick-up operations between a central library and twenty-two local libraries, several studies have been published on this problem. These studies in the literature can be categorized as exact, heuristic, single solution based and population based metaheuristic algorithms.

Vehicle Routing Problem with Simultaneous Delivery and Pickup (VRPSDP) is the extension of the

* Corresponding Author

VRP problem. Because VRPSDP and the VRP belong to NP hard problem, very few exact approaches have been developed for this problem [8]. The first exact solution methodology based on branch-and-price approach with exact dynamic programming and state space relaxation procedures was developed by Dell'Amico, Righini, and Salani [9] in which only instances up to 40 customers could be solved to optimality despite high computational time. Other exact solution approaches have been proposed by Subramanian, Uchoa, Pessoa, and Ochi [10] based on branch-and-cut algorithm and Subramanian, Uchoa, Pessoa, and Ochi [11] based on branch-cut-and-price algorithms which were able to solve instances up to 100 clients. Although exact methodologies are very useful to prove optimal solutions and to provide new lower bounds for well-known data instances, since the solution space grows exponentially when the problem size increases due to computational complexity [12] which has a significant influence on the running time of algorithms, exact solution procedures may not be adequately efficient within reasonable time. Therefore, heuristic and metaheuristic approaches are commonly developed as solution methodologies.

Dethloff [13] proposed a mathematical formulation for VRPSPD and developed an insertion-based heuristic by addressing the problem in reverse logistics operations. Salhi and Nagy [14] studied a similar approach that combined Clarke and Wright saving heuristic [15] with an insertion based heuristic that insert two customers at a time in the route. Nagy and Salhi [16] developed a composite heuristic approach that temporarily allows a certain degree of infeasibility to occur at VRPSPD solutions and eliminates capacity infeasibilities using different sub-routines such as 2-opt, 3-opt, shift, exchange, reverse and perturb structures adapted for the problem. Heuristic approaches usually do not include an improvement routine and thus ends up with the same result for every run of the procedure unless a probabilistic mechanism is used [17]. Therefore, heuristics are either used for initial solution construction in order to provide a better starting point compared to pure randomness or as an embedded constructive sub procedure of iterative search algorithms.

Single solution based metaheuristic algorithms have been commonly used for solving VRPSPD. Among the existing solution strategies, tabu search method is by far the most preferred approach applied for solving this problem [18]. The first tabu search study is proposed by Crispim and Brandao [19] which is a combination of tabu search and variable neighborhood descend approach with a sweep procedure that allows infeasibility to construct initial solution as well as insert and swap procedures to improve the incumbent solution until feasibility is established by penalizing according to the level of overloads. Chen and Wu [20] proposed a tabu search algorithm that obtains initial solutions by an insertion-based procedure based on distance and load based criteria and then improves solutions with 2-exchange, swap, shift, 2-opt and Or-opt procedures and record-to-record travel strategies. Alfredo Tang Montané and Galvão [6] proposed a tabu search algorithm that makes use of inter-route and intra-route neighborhood structures such as interchange, relocation, crossover and 2-opt procedures by controlling intensification and diversification scheme of the approach with a frequency penalization scheme. Ropke and Pisinger [21] developed a large neighborhood search approach that selects a neighborhood method according to a probability depending on its success for solving several variants of vehicle routing problems with backhauls involving VRPSPD. Bianchessi and Righini [22] also proposed a tabu search algorithm with a variable neighborhood structure that makes use of node and arc exchange based local search heuristics. Wassan, Wassan, and Nagy [23] proposed a tabu search algorithm with a mechanism that dynamically controls the tabu list size to achieve an effective balance between the intensification and diversification of the conducted search and makes use of neighborhood structures such as shift, swap, local shift and reverse procedures for improvement. Zachariadis, Tarantilis, and Kiranoudis [24] presented a tabu search algorithm combined with guided local search strategies that iteratively improves the initial solution generated by a saving based constructive heuristic using neighborhood structures such as customer relocation, customer exchange, route interchange procedures. Zachariadis, Tarantilis, and Kiranoudis (2010) [25] proposed a solution approach that stores the routes of high quality VRPSPD solutions in memory and makes use of an improvement procedure based on tabu search. A multi-start metaheuristic approach which consists of a variable neighborhood descent procedure, a random neighborhood ordering procedure and an iterated local search (ILS) framework is proposed by Subramanian, Drummond, Bentes, Ochi, and Farias [26] who performed the experiments in a cluster with a multi-core architecture using up to 256 cores utilizing the parallel structure of the algorithm. Zachariadis and Kiranoudis [27] utilize the weighted savings based heuristic approach to generate an initial solution and then a tabu search algorithm to improve the solutions. Jun and Kim [28] and Avci and Topaloglu [29] proposed iterated local search algorithms with inter-route and intra-route operators and perturbation mechanisms to solve VRPSPD.

Polat et al. [7] recently proposed a perturbation based variable neighborhood search algorithm for solving VRPSPD with and without time limit restrictions.

Population based metaheuristic algorithms have also been applied for solving VRPSPD. Ai and Kachitvichyanukul [30] and Goksal, Karaoglan, and Altiparmak [31] proposed particle swarm optimization algorithms for solving VRPSPD. While Ai and Kachitvichyanukul [30] used a real value encoding mechanism for representation of the solution, cheapest insertion heuristic and 2-opt procedures for improvement, Goksal et al. [31] implemented permutation encoding to represent a solution of the problem and a variable neighborhood descent algorithm as a local search to improve the randomly selected particles in each iteration. Tasan and Gen [32] applied genetic algorithm based metaheuristic approach to VRPSPD. Ant colony optimization algorithms for solving VRPSPD are proposed in the studies of Gajpal and Abad [33] and Çatay [34] that both make use of Clark and Wright's savings algorithm [15] and local search strategies such as 2-opt, customer insertion, interchange and sub-path exchange procedures.

It is worth to mention that since Salhi and Nagy [14] defined some benchmark problem instances for VRPSPDTL that impose time limit restrictions to VRPSPD, it has received less attention in the literature. Among the studies mentioned above, the ones which consider time limit restrictions for VRPSPD instances are as follows: Alfredo Tang Montané and Galvão [6], Polat et al. [7], Dethloff [13], Nagy and Salhi [16], Ropke and Pisinger [21], Wassan and Nagy [23], Jun and Kim [28], Ai and Kachitvichyanukul [30], Gajpal and Abad [33], and Çatay [34].

The interested reader is referred to the review paper of Berbeglia, Cordeau, Gribkovskaia, and Laporte [35] for more information about the vehicle routing problem with pickup and delivery as well as appropriate methods used to tackle this problem and its variants.

Currently, these algorithms have achieved certain achievements in VRPSDP, but research shows that the solution quality and algorithm performance to a large degree depend on the initial solution and the neighborhood structure. In order to avoid the above problems, this paper proposed a new local search algorithm based on multiple neighborhood orientation, which uses a variety of neighborhood to avoid dependence on single neighborhood, and expand neighborhood search. The proposed algorithm also employs an effective punishment strategy to help the search process to jump out of local optimum, and uses repeated search to reduce the influence on the algorithm of the initial solution. Through the 54 solver cases, simulation results show the proposed algorithm is effective and stable, and provides a new way of solution to solve vehicle routing problem.

2 Problem Description

2.1 Problem Introduction of VRPSDP

Vehicle routing problem with simultaneous delivery and pickup could be defined as: set the weighted directed graph $G=(V, A, C)$, in which $V=\{i|i=0, 1, ..., n\}$ is a set of nodes, node 0 is the starting node, node (1~n) are the client nodes. $A=\{(i, j)|i, j\in \text{V}\}$ expresses arc set, $C=\{c_{ij} \mid (i,j)\in A\}$ is the weight matrix, c_{ij} represents the distance of node i and node j.

Set each client node i has the unloading requirements d_i and loading requirements demand p_i and maximum cargo volume of transport vehicles is Q. VRPSDP can be described as: (1) Every car starts from node 0, after service some customer returns 0 point, comes into a solution S. (2) each customer would only be a service, but only by a vehicle to provide services. (3) the weight capacity of each of the vehicles are not more than Q. (4) cargo of all customers are not more than Q. (5) total transportation distance $f(S)$ is the minimum.

2.2 Feasibility Definition of Solution of VRPSDP

For the solution of VRPSDP $S=\{r_j \mid j=1,...,n\}$, Which in $r_j=\{j \mid j\in[1,n]\}$ represents the path of the vehicle j, k is the maximum vehicles number of solution. The necessary and sufficient conditions that S is a strong solution for feasible solutions: any vehicle path r_j needs to satisfy the constraint in solution S after visited all customers: Transport vehicles visited all customers of r_j, its load does not exceed Q. For solution *S*, if $\exists r_j \in S$ does not satisfy the constraint, the solution S is a weak feasible solution, otherwise it a the strong feasible solution.

3 Related Work

3.1 Overview of GLS

Guided local search algorithm (Guided Local Search, GLS) was proposed by Voudouris and Tsang [36] in 1995, has the characteristics of compact structure, good general. It has already been applied to solving the classical TSP problem and other NP hard combinatorial optimization problem. Its basic principle is to keep the solution structure and neighborhood structure in the process of searching are unchanged, and the objective functions of dynamic is modified so that the current local extremum has no local optimality.

The objective function modification mechanism of GLS bases feature of solutions which can be used to distinguish certain features or characteristics of the different solutions. GLS usually adopts the arc length as a solution characteristic in the VRP problem. GLS mainly uses the penalty strategy to punish some of features of the approximate optimal solution to achieve local search solution. But the penalty strategy mainly includes the initial cost (distance between two points) and punishment times (i.e., the punished times of a feature).

Here punish feature is defined as *m*, its characteristic index function is defined as $I_m(S)$ for a solution S, the punish utility function is defined as $U_m(S)$, the objective function is defined as $h(S)$.

Characteristic index function $I_m(S)$ shows whether the solution S has punitive feature *m*:

$$I_m(S) = \begin{cases} 1, \text{ if } m \text{ } appears \text{ } in \text{ } the \text{ } S \\ 0, \qquad therwise \end{cases} \tag{1}$$

The characteristics m will be punished once if it appears in the S, here penalty coefficient P_{m} is be used to count. In order to avoid excessively punishing some important characteristics, the punishment times of characteristics need to be effectively evaluated. Therefore, the punish utility function $U_m(S)$ [7] is defined as follows:

$$U_m(S) = \frac{I_m(S)c_m}{1+P_m} \tag{2}$$

c_m is the arc length of punishment characteristics.

It selects the characteristic which its punish utility function is maximum to punish from solutions of S, the objective function $h(S)$ is:

$$h(S) = g(S) + \lambda \sum P_m I_m(S) \tag{3}$$

Where *g*(*S*) is the total arc length of *S* and represents the original objective function value of solution *S*, λis a normalization factor, usually taken from 0.01~0.10 [32]. By punishing the solutions with punishment characteristics, it can ensure search is carried out near better feasible solution. By continuously modifying the objective function, it makes the current local maxima are no longer a local optimum, thus it constantly iterates to find the optimal solution.

3.2 Constructing an Initial Feasible Solution of MN_GLS

Here, we use nearest neighbor method construct initial feasible solution of MN_GLS:

Step1: Selecting the node from no selected client node set which makes distance shortest between the starting node and no selected client node, and starting a new search path.

Step2: Selecting the node from no selected client node set which makes distance shortest between the last node of the path and no selected client node.

Step3: After adding a new node, if the path satisfies strong constraints of feasible solution, then goto Step2, otherwise goto Step1.

So, the constructed initial feasible solution is also a strong feasible solution.

3.3 Local Optima of Multi-neighborhood Search of MN_GLS

Obtained strong feasible solution as the initial solution from 3.2 section, we choose a neighborhood structure do local search, when it reaches the maximum number of iterations or successive iterations without improvement, the local search over. Here, we choice four neighborhood structures: Insert, Swap, 2-opt and 2-opt*.

Insert N_1**.** It moves the customer i of solution S from its current location l_i to another location l_j to generate a new solution S*. The node of location l_i and l_j on node belongs to any path of the solution S. The neighborhood structure N_1 is shown in Fig. 1.

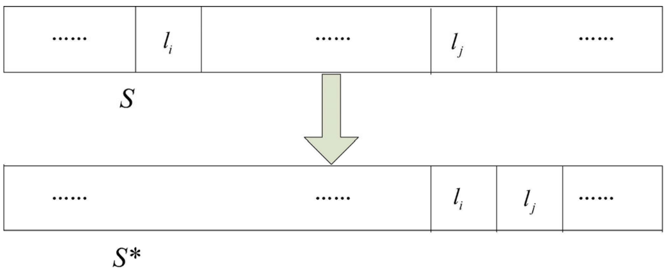

Fig. 1. The neighborhood structure N_1

Swap N_2**.** It swaps two customers i and j of solution S from their current location l_i and l_j. Customers i and j may belong to the same path, or may belong to different paths. The neighborhood structure N_2 is shown in Fig. 2.

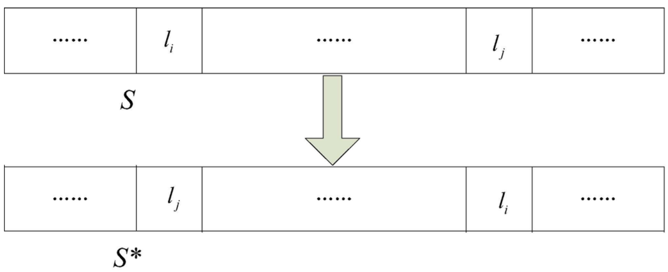

Fig. 2. The neighborhood structure N_2

2-opt N_3**.** For two customers of the solution S which locates l_i and l_j in the same path, it exchange node position from the node position l_{i+1} to l_j, reverses the node between l_{i+1} and l_j, so that neighborhood solution is produced. The neighborhood structure N_3 is shown in Fig. 3.

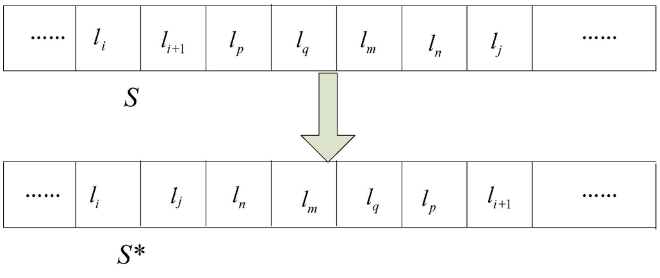

Fig. 3. The neighborhood structure N_3

2-opt* N_4**.** It selects two different paths r_x and r_y of the solution S, and deletes one of them, which get four sub-paths $r_{x1}, r_{x2}, r_{y1}, r_{y2}$. r_{x1}, r_{y2} and r_{x2}, r_{y1} are reassembled to get two new paths r_x' and r_y', so that neighborhood solution is produced. The neighborhood structure N_4 is shown in Fig. 4.

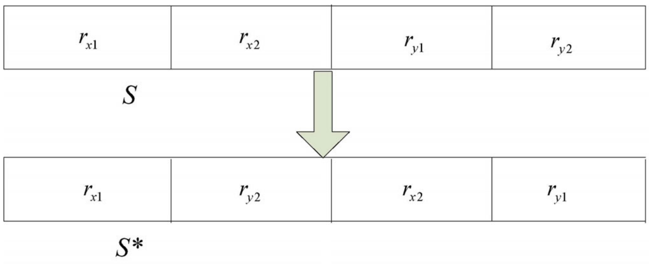

Fig. 4. The neighborhood structure N_4

A variety of neighborhood structure can combine with a variety of neighborhood searching capability to extend to a certain extent the search space of the algorithm.

3.4 Modified Penalty Coefficient of MN_GLS

After all the neighborhood searches get into local extremum, it need modify penalty coefficient of the solution characteristics. In traditional GLS algorithm based on single neighborhood, penalty coefficient just add 1. In MN_GLS, each neighborhood search local extremum solution can obtain maximum punishment which the punishment characteristics of utility function are poor arcs, may be the same, also may be different, so the penalty coefficient is updated by $p_a = p_a + 4c_a/(c_a + c_b + c_c + c_d)$.

In which, c_a, c_b, c_c, c_d respectively represent punishment characteristics with maximum punishment utility function values in 4 kinds of neighborhood search. When the search neighborhood is a single neighborhood, the penalty coefficient updates by means of $p_a = p_a + 1$. when the search neighborhood is multi- neighborhood more, it can reflect the punished punishment extent size of penalty characteristics.

3.5 Proposed MN_GLS

We adopt relevant strategies form section 3.2, section 3.3 and section 3.4, and proposed a Multiple Neighborhood Guided Local Search Algorithm (MN_GLS) to solve vehicle routing problem (VRP) with simultaneous delivery and pickup, the Pseudo code of MN_GLS procedure is given in Fig. 5:

Select n as the customer size
Select (x_i, y_i) as the customer geographical coordinates
Select d_i as the customer demand
Select p_i as the cargo demand
Select Q as vehicle transportation capacity
Select S_0 as current optimal solution
Select T as a termination condition (maximum number of iterations)
$S_0 \leftarrow$ an initial feasible solution.
While T is not met do
Starting local search in neighborhood $(N_1 \sim N_k)$ of the initial solution S_0
If the searh falls into local optima then
$S_0^{best}(N_1), S_0^{best}(N_2), ..., S_0^{best}(N_k) \leftarrow$ the local optimal solution
    End if
$U_m(S_0^{best}(N_i)), i \in [1,k]$ ←Each local optimal solution $S_0^{best}(N_1), S_0^{best}(N_2), ..., S_0^{best}(N_k)$
$p_m \leftarrow p_m + 1$
$h(S) \leftarrow g(S) + \lambda \sum P_m I_m(S)$
$S_{now}^{best} \leftarrow$ Select optimal solution $(S_0^{best}(N_1), S_0^{best}(N_2), ..., S_0^{best}(N_k))$
$S_{history}^{best} \leftarrow S_0^{best}(N_1), S_0^{best}(N_2), ..., S_0^{best}(N_k)$
End while

Fig. 5. The pseudo code of MN_GLS procedure

4 Simulation Experiments

4.1 Test Cases and Parameter Selection

Test cases with variety of problem size and characteristics available in the paper are used in order to evaluate the effectiveness of the proposed MN_GLS, we mainly adopt two types of benchmark data sets for comparison purposes: The first group that consists of 40 test problems involving 50 customers and a central depot, namely SCA and CON sets, is generated and presented by Dethloff [13]. SCA is uniform distribution of [0,100], CON is non-uniform distribution of [0,200]. The unloading demand d of customer point is the random value in [0,100], The loading demand $p=(k+0.5)*d$, Where k is a random value in [0,1]. The second group that consists of 28 test problems involving from 50 to 200 customers and a central depot, namely CMT sets, is generated and presented by Salhi and Nagy [14]. 14 problems of CMT sets do not impose time limit (a limit on the total route length) restrictions while the remaining 14 problems impose time limit restrictions. Each problem contains two examples, which *CMTxX* is the expanded the original *CMTx* which is increased unloading demand and loading demand, *CMTxY* is the opposite example of *CMTxX* for unloading demand and loading demand. The scales of *CMTxX* and *CMTxY are* (50,75,100,100,120,150,199).

We simulated MN_GLS algorithm in MATLAB7.0 and used C# in the. Net platform achieve MN_GLS algorithm, the operating environment is the win7 64x, CPU Intel core is i3-4150 for 3.5GHz, memory for 4G, parameter is the maximum number of iterations of algorithm, *max_ite*=1000. In terms of here standardized coefficient λ, it is calculated in SCA3-0, CON3-0, and CMT1X. The calculated results are shown in Table 1.

RPD is Relative Percentage Deviation for optimized results with the known best solution:

$$RPD = 100 \times (g(S) - g(S_b))/g(S_b) \tag{4}$$

Where S_b is the known best solution [18].

If the standardized coefficient ℓ value is smaller, the convergence is slower; otherwise, the solution is easy to fall into the local optimal solution. As can be seen from Table 1, when the standardized coefficient $\ell = 0.8$, search performance can achieve better results, so the following experiments standardized coefficient value is set 0.8.

Table 1. Search performance of different λ values

λ	SCA3-0		CON3-0		CMT1X	
	RPD	*t/s*	*RPD*	*t/s*	*RPD*	*t/s*
0.2	**0**	8.91	**0**	12.56	**0**	10.38
0.4	**0**	7.09	**0**	9.14	**0**	8.82
0.8	**0**	5.28	**0**	8.40	**0**	**7.19**
1.0	0.19	**5.19**	0.23	**8.35**	**0**	7.22
1.2	0.26	5.21	0.31	8.51	0.03	7.47

4.2 Experimental Results

In order to verify the feasibility and the stability of MN_GLS, GLS which was proposed by Voudouris and Tsang [36], and TS_GLS which was proposed Guided Local Search by Mustafa and Seyda [18] based on tabu search and hybrid algorithm, and the proposed algorithm MN_GLS are used in two types of benchmark data sets Dethloff [13], Salhi and Nagy [14]. After running 100 times, we recorded related experimental results. The Table 2 and the Table 3 show the experimental results (The best solution of each problem instance is highlighted in bold).

Table 2. Comparison experimental results on dethloff

Examples		GLS				TS_GLS				MN_GLS			
Test Cases	**BKPL**	**BPL**	**T^2/s**	**APL**	**Er**	**BPL**	**T^1/s**	**APL**	**Er**	**BPL**	**T^2/s**	**APL**	**Er**
SCA3-0	**635.62**	646.50	0.77	662.33	10.88	**635.62**	2.83	635.78	0.00	635.66	5.28	635.69	0.04
SCA3-1	**697.84**	703.42	0.85	719.45	5.58	**697.84**	2.12	698.01	0.00	**697.84**	5.10	697.99	0.00
SCA3-2	**659.34**	681.86	0.88	689.01	22.52	**659.34**	2.58	659.42	0.00	**659.34**	5.47	659.39	0.00
SCA3-3	**680.04**	690.51	1.01	691.09	10.47	**680.04**	3.13	680.67	0.00	**680.04**	5.29	680.23	0.00
SCA3-4	**690.50**	718.51	0.91	721.67	28.01	**690.50**	2.68	691.38	0.00	**690.50**	5.60	690.55	0.00
SCA3-5	**659.91**	671.18	0.85	685.61	11.27	**659.91**	2.56	660.28	0.00	659.96	5.56	660.07	0.05
SCA3-6	**651.09**	656.26	0.78	670.79	5.17	**651.09**	4.40	651.20	0.00	**651.09**	5.51	651.10	0.00
SCA3-7	**659.17**	666.10	0.86	690.47	6.93	**659.17**	2.98	659.17	0.00	**659.17**	5.69	659.19	0.00
SCA3-8	**719.48**	747.40	0.91	770.26	27.92	**719.48**	3.98	719.56	0.00	**719.48**	5.91	719.52	0.00
SCA3-9	**681.00**	699.16	0.79	721.48	18.16	**681.00**	3.86	681.21	0.00	**681.00**	5.87	681.01	0.00
SCA8-0	**961.50**	970.30	0.94	989.83	8.80	**961.50**	3.21	961.59	0.00	**961.50**	5.85	961.52	0.00
SCA8-1	**1049.65**	1054.56	1.28	1089.41	4.91	1050.20	3.55	1050.28	0.55	**1049.65**	6.13	1049.70	0.00
SCA8-2	**1039.64**	1050.57	1.73	1079.38	10.93	**1039.64**	4.67	1039.78	0.00	**1039.64**	6.85	1039.66	0.00
SCA8-3	**983.34**	1011.04	1.21	1047.37	27.70	**983.34**	3.29	983.39	0.00	**983.34**	6.10	983.37	0.00
SCA8-4	**1065.49**	1092.59	1.69	1131.29	27.10	**1065.49**	2.68	1065.51	0.00	**1065.49**	6.02	1065.50	0.00
SCA8-5	**1027.08**	1043.90	1.86	1061.61	16.82	**1027.08**	4.50	1027.10	0.00	**1027.08**	6.67	1027.09	0.00
SCA8-6	**971.82**	974.16	1.82	1009.71	2.34	**971.82**	2.67	971.90	0.00	**971.82**	6.69	971.83	0.00
SCA8-7	**1051.28**	1075.46	1.62	1109.83	24.18	1052.17	4.32	1052.93	0.89	**1051.28**	6.73	1051.30	0.00
SCA8-8	**1071.18**	1080.47	1.59	1101.48	9.29	**1071.18**	3.43	1071.24	0.00	**1071.18**	5.93	1071.20	0.00
SCA8-9	**1060.50**	1085.98	1.24	1097.56	25.48	**1060.50**	4.12	1060.57	0.00	1060.52	6.22	1060.55	0.02
CON3-0	**616.52**	629.00	1.90	653.39	12.48	**616.52**	3.89	616.56	0.00	**616.52**	8.40	616.55	0.00
CON3-1	**554.47**	574.42	1.64	591.97	19.95	**554.47**	2.97	554.50	0.00	**554.47**	7.91	554.49	0.00
CON3-2	**518.00**	542.35	1.53	580.35	24.35	**518.00**	3.32	518.14	0.00	**518.00**	6.28	518.01	0.00
CON3-3	**591.19**	597.91	1.87	619.31	6.72	**591.19**	2.78	591.27	0.00	**591.19**	6.10	591.21	0.00
CON3-4	**588.79**	609.62	2.10	640.03	20.83	**588.79**	3.12	588.82	0.00	**588.79**	7.47	588.80	0.00
CON3-5	**563.70**	574.07	1.82	589.28	10.37	**563.70**	3.45	563.78	0.00	**563.70**	7.29	563.73	0.00
CON3-6	**499.05**	512.30	1.55	554.92	13.25	500.01	2.98	500.05	0.96	**499.05**	8.60	499.06	0.00
CON3-7	**576.48**	592.01	2.01	624.73	15.53	**576.48**	2.40	576.50	0.00	**576.48**	7.56	576.49	0.00
CON3-8	**523.05**	538.89	0.86	571.44	15.84	**523.05**	5.02	523.38	0.00	**523.05**	6.51	523.06	0.00
CON3-9	**578.25**	599.34	1.73	631.56	21.09	**578.25**	3.14	578.30	0.00	**578.25**	8.69	578.27	0.00
CON8-0	**857.17**	867.24	1.34	889.26	10.07	**857.17**	3.40	857.29	0.00	857.23	7.91	857.25	0.06
CON8-1	**740.85**	755.60	1.94	791.73	14.75	**740.85**	3.73	740.91	0.00	**740.85**	8.09	740.89	0.00
CON8-2	**712.89**	742.00	1.16	789.92	29.11	**712.89**	2.87	712.90	0.00	**712.89**	9.38	712.90	0.00
CON8-3	**811.07**	825.09	1.52	858.37	14.02	**811.07**	3.82	811.12	0.00	**811.07**	9.81	811.08	0.00
CON8-4	**772.25**	779.03	1.77	805.74	6.78	**772.25**	2.98	772.28	0.00	**772.25**	9.90	772.26	0.00
CON8-5	**754.88**	760.58	1.62	793.24	5.70	754.92	5.76	754.99	0.04	**754.88**	8.85	754.89	0.00
CON8-6	**678.92**	703.67	2.15	731.06	24.75	**678.92**	4.00	678.95	0.00	**678.92**	9.30	678.92	0.00
CON8-7	**811.96**	841.04	1.35	878.39	29.08	**811.96**	2.46	811.96	0.00	**811.96**	10.59	811.96	0.00
CON8-8	**767.53**	778.27	1.11	809.81	10.74	**767.53**	4.21	767.57	0.00	**767.53**	10.21	767.53	0.00
CON8-9	**809.00**	835.27	1.38	866.32	26.27	**809.00**	3.87	809.48	0.00	**809.00**	9.49	809.04	0.00
TAvg	**758.54**	774.44	1.40	800.26	15.90	758.60	3.44	758.74	0.06	**758.54**	7.17	758.57	0.00
NBKSF	**40**	0				36				36			

Note. [1] Pentium IV 2.4G; [2] i3-4150 3.5GHz

Notice: BKPL: The best known path length of every test case [3-7,14-20]; **BPL**: The best path length of every test case; **APL:** The average path length of every test case; **T:** The average consumed time of every test case; **Er:** The difference between **BPL** and **BKPL**; **TAvg:** The average values of 40 test cases; N**BKSF**: The number of best paths found.

From Table 2 we can see: in terms of the average best path length, MN_GLS respectively reduces the 0.06 which is compared with TS_GLS and 15.9 which is compared with GLS. In terms of quality of the optimal solution, MN_GLS is significantly higher than GLS, is worse than TS_GLS on SCA3-0, SCA3-5, SCA8-9 and CON8-0, but is better than TS_GLS on SCA8-1, SCA8-7, CON3-6 and CON8-5. In terms of stability, the average path length of every test case and the average path length of 40 test cases on MN_GLS are better than TS_GLS's and GLS's. Because the different computing environments, we only rough compare the computation time: due to increased neighborhood, it makes that the computing time of MN_GLS is about 5 times than GLS, and about twice than TS_GLS, but the most consumed time and the average consumed time of MN_GLS are only 10.59 s and 7.17s which are acceptable. Table 2 indicates that MN_GLS is generally more feasible and more stable than TS_GLS and GLS.

Taking into account the inherent randomness of MN_GLS, in order to further validate the stability of MN_GLS, we tested MN_GLS and TS_GLS on some representative test problems of Salhi and Nagy. After running 100 times, we recorded related experimental results, they are shown in Table 3.

Table 3. Comparison experimental results on Salhi and Nagy

Examples		TS_GLS				MN_GLS			
Test Cases	**BKPL**	**BPL**	T^1/s	**APL**	**Er**	**BPL**	T^2/s	**APL**	**Er**
CMT1X	**466.77**	469.80	2.89	469.81	3.03	**466.77**	2.13	466.78	0.00
CMT1Y	**466.77**	469.80	3.85	469.81	3.03	476.81	2.06	476.83	10.04
CMT2X	**684.21**	**684.21**	7.42	684.24	0.00	**684.21**	6.81	684.23	0.00
CMT2Y	**684.21**	**684.21**	8.02	684.25	0.00	**666.75**	7.03	666.76	-17.46
CMT3X	**721.27**	**721.27**	11.62	721.28	0.00	**715.51**	9.72	715.53	-5.76
CMT3Y	**721.27**	**721.27**	13.53	721.28	0.00	**721.27**	10.48	721.28	0.00
CMT4X	**852.46**	**852.46**	27.75	852.47	0.00	**852.46**	40.40	852.47	0.00
CMT4Y	**852.46**	**852.46**	31.20	852.48	0.00	**852.46**	40.98	852.47	0.00
CMT5X	**1029.25**	1030.55	51.67	1030.58	1.30	1031.92	93.51	1031.93	2.67
CMT5Y	**1029.25**	1030.55	58.81	1030.56	1.30	1031.92	93.88	1031.93	2.67
CMT11X	**833.92**	838.06	17.78	838.08	4.14	841.23	22.44	841.24	7.31
CMT11Y	**833.92**	837.08	14.26	837.09	3.16	837.08	28.79	837.09	3.16
CMT12X	**662.22**	**662.22**	11.80	662.23	0.00	**662.22**	10.90	662.23	0.00
CMT12Y	**662.22**	**662.22**	7.59	662.23	0.00	663.50	10.47	663.51	1.28
TAvg	**750.01**	751.15	19.16	751.17	1.14	750.29	27.11	750.31	0.28
N**BKSF**	**14**	8				8			

Note. [1] Pentium IV 2.4G; [2] i3-4150 3.5GHz

Notice: BKPL: The best known path length of every test case [3-7, 14-20]; **BPL**: The best path length of every test case; **APL:** The average path length of every test case; **T:** The average consumed time of every test case; **Er:** The difference between **BPL** and **BKPL**; **TAvg:** The average values of 14 test cases; N**BKSF**: The number of best paths found.

From Table 3 we can see: in terms of the average path length, MN_GLS reduces the 0.86 which is compared with TS_GLS. In terms of quality of the optimal solution, MN_GLS is mostly better than TS_GLS. In terms of stability, the average path length of every test case and the average path length of 14 test cases on MN_GLS are better than TS_GLS's. Because the different computing environments, we also rough compare the computation time: due to increased neighborhood, it makes that the computing time of MN_GLS is about 1.5 times than TS_GLS. As for CMT5 which the scale is 199, the average running time of MN_GLS is 93.88s, the time is acceptable. Table 3 indicates that MN_GLS is generally more feasible and more stable than TS_GLS and GLS.

5 Conclusions

Vehicle Routing Problem with Simultaneous Delivery and Pickup is an actual needs and widespread problem. In order to solve the problem, we proposed a Multiple Neighborhood Guided Local Search Algorithm (MN_GLS) to expand the search capabilities. Through a total of 54 of Dethloff [13], Salhi and Nagy [14] examples, we verified MN_GLS. The simulation results illustrate that MN_GLS can obtain better quality optimal solution and better stability than GLS and TS_GLS, MN_GLS is an effective and stabilize method for Vehicle Routing Problem with Simultaneous Delivery and Pickup. However, MN_GLS takes more computing time than other algorithms. How to improve the time performance of MN_GLS is the focus point of our future research.

Acknowledgements

The authors would like to thank the editors and the anonymous reviewers for their helpful comments and suggestions, which have improved the presentation. This work was supported in part by the Science and Technology Plan Projects of Henan Province of China under grant No.152102210357 and No.152102210149, the Youth Backbone Teachers Funding Planning Project of Colleges and Universities in Henan Province of China under grant No.2014GGJS-084, the Key Science Research Project of Colleges and Universities in Henan Province of China under grant No.16A520030, the Youth Backbone Teachers Training Targets Funded Project of Zhengzhou University of Light Industry of Henan Province of China under grant No. XGGJS02, the Ph.D. Research Funded Project of Zhengzhou University of Light Industry of Henan Province of China under grant No.2010BSJJ038 and No.2014BSJJ080, and the National Science Foundation of China under grant No.81501548.

References

[1] N. Soeffker, W.M. Ulmer, D.C. Mattfeld, Problem-specific state space partitioning for dynamic vehicle routing problems, in: V. Nissen, D. Stelzer, S. Straßburger, D. Fischer (Eds.), Proceedings of Multikonferenz Wirtschaftsinformatik, Universitätsverlag Ilmenaupp, 2016, pp. 229-240.

[2] H. Min, The multiple vehicle routing problem with simultaneous delivery and pick-up points, Transportation Research Part A: General 23(5)(1989) 377-386.

[3] O. Dominguez, D. Guimarans, A.A. Juan, I. Nuez, A biase d-randomised large neighbourhood search for the two-dimensional vehicle routing problem with backhauls, European Journal of Operational Research 255(2016) 442-462.

[4] D. Männel, A. Bortfeldt, A hybrid algorithm for the vehicle routing problem with pickup and delivery and three-dimensional loading constraints, European Journal of Operational Research 255(2016) 840-858.

[5] C. B. Kalayci, C. Kaya, An ant colony system empowered variable neighborhood search algorithm for the vehicle routing problem with simultaneous pickup and delivery, Expert Systems with Applications 66(2016) 163-175.

[6] F. Alfredo Tang Montané, R.D. Galvão, A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and delivery service, Computers & Operations Research 33(2006) 595-619.

[7] O. Polat, C.B. Kalayci, O. Kulak, H.-O. Günther, A perturbation based variable neighborhood search heuristic for solving the vehicle routing problem with simultaneous pickup and delivery with time limit, European Journal of Operational Research 242(2015) 369-382.

[8] A. Mustafa, T. Seyda, A hybrid local search algorithm for heterogeneous fleet vehicle routing problem with simultaneous pickup and delivery, in: Proc. the 44th International Conference on Computers and Industrial Engineering, 2014.

[9] M. Dell'Amico, G. Righini, M. Salani, A branch-and-price approach to the vehicle routing problem with simultaneous distribution and collection, Transportation Science 40(2006) 235-247.

[10] A. Subramanian, E. Uchoa, A.A. Pessoa, L.S. Ochi, Branch-and-cut with lazy separation for the vehicle routing problem with simultaneous pickup and delivery, Operations Research Letters 39(2011) 338-341.

[11] A. Subramanian, E. Uchoa, A.A. Pessoa, L.S. Ochi, Branch-cut-and-price for the vehicle routing problem with simultaneous pickup and delivery, Optimization Letters 7(2013) 1569-1581.

[12] C.A. Tovey, Tutorial on Computational Complexity, Interfaces 32(2002) 30-61.

[13] J. Dethloff, Vehicle routing and reverse logistics: The vehicle routing problem with simultaneous delivery and pick-up, OR-Spektrum 23(2001) 79-96.

[14] S. Salhi, G. Nagy, A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling, The Journal of the Operational Research Society 50(1999) 1034-1042.

[15] G. Clarke, J.W. Wright, Scheduling of vehicles from a central depot to a number of delivery points, Operations Research 12 (1964) 568-581.

[16] G. Nagy, S. Salhi, Heuristic algorithms for single and multiple depot vehicle routing problems with pickups and deliveries, European Journal of Operational Research 162(2005) 126-141.

[17] T. Zhang, W.A. Chaovalitwongse, Y. J. Zhang, Integrated ant colony and tabu search approach for time dependent vehicle routing problems with simultaneous pickup and delivery, Journal of Combinatorial Optimization 28(1)(2014) 288-309.

[18] A. Mustafa, T. Seyda, A hybrid metaheuristic algorithm for heterogeneous vehicle routing problem with simultaneous pickup and delivery, Expert Systems with Applications 53(2016) 160-171.

[19] J. Crispim, J. Brandao, Metaheuristics applied to mixed and simultaneous extensions of vehicle routing problems with backhauls, Journal of the Operational Research Society 56(2005) 1296-1302.

[20] J.F. Chen, T.H. Wu, Vehicle routing problem with simultaneous deliveries and pickups, Journal of the Operational Research Society 57(2005) 579-587.

[21] S. Ropke, D. Pisinger, A unified heuristic for a large class of vehicle routing problems with backhauls, European Journal of Operational Research 171(2006) 750-775.

[22] N. Bianchessi, G. Righini, Heuristic algorithms for the vehicle routing problem with simultaneous pick-up and delivery, Computers & Operations Research 34(2007) 578-594.

[23] N. Wassan, A.H. Wassan, G. Nagy, A reactive tabu search algorithm for the vehicle routing problem with simultaneous pickups and deliveries, Journal of Combinatorial Optimization 15(2008) 368-386.

[24] E.E. Zachariadis, C.D. Tarantilis, C.T. Kiranoudis, A hybrid metaheuristic algorithm for the vehicle routing problem with simultaneous delivery and pick-up service, Expert Systems with Applications 36(2009) 1070-1081.

[25] E.E. Zachariadis, C.D. Tarantilis, C.T. Kiranoudis, An adaptive memory methodology for the vehicle routing problem with simultaneous pick-ups and deliveries, European Journal of Operational Research 202(2010) 401-411.

[26] A. Subramanian, L.M.A. Drummond, C. Bentes, L.S. Ochi, R. Farias, A parallel heuristic for the vehicle routing problem with simultaneous pickup and delivery, Computers & Operations Research 37(2010) 1899-1911.

[27] E.E. Zachariadis, C.T. Kiranoudis, A local search metaheuristic algorithm for the vehicle routing problem with simultaneous pick-ups and deliveries, Expert Systems with Applications 38(2011) 2717-2726.

[28] Y. Jun, B.-I. Kim, New best solutions to VRPSPD benchmark problems by a perturbation based algorithm, Expert Systems with Applications 39(2012) 5641-5648.

[29] M. Avci, S. Topaloglu, An adaptive local search algorithm for vehicle routing problem with simultaneous and mixed pickups and deliveries, Computers & Industrial Engineering 83(2015) 15-29.

[30] T.J. Ai, V. Kachitvichyanukul, A particle swarm optimization for the vehicle routing problem with simultaneous pickup and delivery, Computers & Operations Research 36(2009) 1693-1702.

[31] F.P. Goksal, I. Karaoglan, F. Altiparmak, A hybrid discrete particle swarm optimization for vehicle routing problem with simultaneous pickup and delivery, Computers & Industrial Engineering 65(2013) 39-53.

[32] A.S. Tasan, M. Gen, A genetic algorithm based approach to vehicle routing problem with simultaneous pick-up and deliveries, Computers & Industrial Engineering 62(2012) 755-761.

[33] Y. Gajpal, P. Abad, An ant colony system (ACS) for vehicle routing problem with simultaneous delivery and pickup, Computers & Operations Research 36(2009) 3215-3223.

[34] B. Çatay, A new saving-based ant algorithm for the vehicle routing problem with simultaneous pickup and delivery, Expert Systems with Applications 37(2010) 6809-6817.

[35] G. Berbeglia, J.-F. Cordeau, I. Gribkovskaia, G. Laporte, Static pickup and delivery problems: a classification scheme and survey, TOP 15(2007) 1-31.

[36] C. Voudouris, E.P.K. Tsang, Guided local search, Technical Report CSM-247, Department of Computer Science, University of Essex, 1995.