Sigma J Eng Nat Sci, Vol. 39, No. 3, pp. 226–236, September, 2021

Sigma Journal of Engineering and Natural Sciences
Web page info: https://sigma.yildiz.edu.tr
DOI: 10.14744/sigma.2021.00012

Research Article

A metaheuristic solution approach with two construction heuristics for vehicle routing problem with simultaneous Linehauls and Backhauls

Mustafa DEMİRBİLEK[1,*]

[1]*Department of Industrial Engineering, Gaziantep Islam Science and Technology University, Gaziantep, Turkey*

ARTICLE INFO

Article history
Received: 10 July 2020
Accepted: 07 January 2021

Key words:
Simulated annealing approach; Construction heuristics; Vehicle routing problem with simultaneous Linehauls and Backhauls; Mixed integer linear programming

ABSTRACT

Since highly competitive business environments and strict governmental regulations, logistic companies have started to handle pickup and delivery activities at the same time to satisfy demands of customers. This problem is called as Vehicle Routing Problem with Simultaneous Linehauls and Backhauls (VRPSLB). The objective is to minimize total travel times of vehicles that visit a predetermined number of customers in a service area. When delivering and collecting orders of customers, we must also make sure not to exceed capacity of vehicles. Since VRPB is in NP Hard class, exact solution methods do not work for large problem sets. In this study, we proposed a simulated annealing (SA) metaheuristic method with two construction heuristics, Clark and Wright (CW) algorithm and the nearest neighbourhood (NN) search algorithm. Since quality of final solutions are directly related to the quality of initial solutions, we propose two heuristics to generate initial solutions. Daily working times for vehicles and service times for customers are considered as hard constraints in this problem. Results of SA with two heuristics are compared to results of the mixed integer linear programming (MILP) for relatively small problem sets where the numbers of customers and vehicles vary 20 to 40 and 2 to 5, respectively. Results demonstrate that SA provides up to %7 longer total travel times compared to MILP. SA with NN shows better performance compared to SA with CW. For large problems, we do not observe a significant difference between the performance of SA with NN and SA with CW.

Cite this article as: Mustafa D. A metaheuristic solution approach with two construction heuristics for vehicle routing problem with simultaneous linehauls and backhauls. Sigma J Eng Nat Sci 2021;39(3):226–236.

INTRODUCTION

Vehicle Routing Problem (VRP) has been highly studied by many researchers since the study of Dantzig and Ramser [1], "The Truck Dispatching Problem", 1959. They found optimal routes for vehicles that deliver gasoline from a bulk terminal to many gasoline stations. Many different variations of VRP, multi-depot, capacity, time windows, stochastic service and travel times as well as solution

***Corresponding author.**
*E-mail address: mustafa.demirbilek@gibtu.edu.tr

This paper was recommended for publication in revised form by Regional Editor Umut Tuzkaya

methods such as the linear programming, mixed integer programming, saving heuristics, tabu search, genetic algorithm have been proposed since then [2]. One of the popular variations, Vehicle Routing Problem with Simultaneous Linehauls and Backhauls (VRPSLB), has been attracted my researchers` attentions for long years and many real-life applications such as online shopping, parcel delivery and reverse logistic can be modelled as VRPSLB [3]. The main aim is to minimise travel times of vehicles that visit customers distributed in a service area by satisfying their pickup and delivery demands. This problem is divided to three different subproblems. The first problem is to consider satisfying delivery demands first and pickup demands next. The second problem is that delivery and pickup demands of customers are satisfied with mixed sequences (VRPMLB). The final problem, also considered in this study, is simultaneous linehauls and backhauls that pickup and delivery demands of each customer must be satisfied at the same time. One advantage of this problem is to decrease operation related costs since delivery and pickup activities are taken place together at the same time. Other advantage is to provide environment friendly solutions by reducing carbon emissions of vehicles thanks to decrease the number of vehicles during daily operations [4].

VRPSLB studies began with the study of Min [5], "The Multiple Vehicle Routing Problem with Simultaneous Delivery and Pickup Points", 1989. He found optimum routes for vehicles that delivered library materials from a main library to 22 branches and pick up the materials from branches to carry to the main library. Hezer and Kara [6] developed a metaheuristic method, bacterial foraging optimization algorithm, for VRPSLB. Proposed algorithm provided superior results for 24 out of 40 test sets compared to the heuristic solution method developed by Dethloff [7]. Nagy and Salhi [8] developed a heuristic solution method to be able to solve both VRPSLB and VRPMLB. They also extended the solution method to solve the multi-depot extension. Dell'Amico et al. [9] proposed a branch and price algorithm to solve the VRPSLB for 40 requests. Wassan et al. [10] developed a reactive tabu search metaheuristic that could control feasibility of predetermined moves quickly and acted to iterations to guide the search. Ai et al. [11] presented a particle swarm optimization method. Subramanian et al. [12] presented a parallel algorithm based on the sequential heuristics. They embedded the algorithm with a multi-start heuristic containing of the VND integrated in an iterated local search framework. Avci and Topaloglu [13] proposed adaptive local search algorithm for solutions of both VRPSLB and VRPMLB. Li et al. [14] considered Multi-Depot VRPSLB and developed an iterated local embedded adaptive neighbourhood selection approach. Results showed that the proposed algorithm was superiorthan large neighbourhood search, particle swarm optimization, and ant colony optimization approach. Zachariadis et al. [15] considered two-dimensional loading constraints beside simultaneous linehauls and backhauls. Kalayci and Kaya [16] developed an ant colony system empowered variable neighbourhood search algorithm, a hybrid algorithm that combines ant colony system and variable neighbourhood approach. Experiments showed that the proposed algorithm provided better results compared to individual performance of two methods. Kececi et al. [17] considered heterogeneous vehicle fleet consisted of different type of vehicles with costs. They proposed a mathematical model and an insertion-based heuristic for the solution. Yazgan and Buyukyilmaz [18] developed a greedy heuristic solution method and carried out a regression analysis to find the relationship between travel distances and the number of customers and capacity of vehicles. Goksal et al. [19] presented a discrete particle swarm optimization method and variable neighbourhood descent algorithm for VRPSLB. The algorithm kept swarm diversity by improving randomly selected solutions in each iteration. Montane and Galvao [20] modified a tabu search algorithm to become compatible with VRPSLB. They developed three types of movements to observe inter-route adjacent solutions: the relocation, interchange and crossover movements. Table 1 shows publications in terms of problem types, years, and solution methods. We advise the review study of Koc and Laporte [21] in that VRPSLB studies were classified in terms of models, solution methods, variants, industrial applications and case studies to interested readers.

In this study, we proposed a simulated annealing solution approach for VRPSLB. Our contributions in this study are two folded. From the application side, daily working times for vehicles and service times for customers are defined and considered as hard constraints in the problem. These two real-life conditions are rarely considered in previous studies. From the methodological side, we take two construction heuristics into consideration to produce initial solutions for the simulated annealing approach. Metaheuristic solution methods must start iterations from initial solutions generated by heuristic algorithms or random feasible solutions. If the quality of initial solution is good, the chance of finding optimal solution in predetermined number of iterations increases. Therefore, we find initial solutions with the Clark and Right and the nearest neighbourhood search algorithms and run the simulated annealing approach with these two construction heuristics to observe the best results.

SOLUTION METHODS

We first describe mathematical formulation for VRPSLB. Next, we explain two construction heuristics, Clark and Wright algorithm and the nearest neighbourhood search. Finally, we present the simulated annealing metaheuristic approach.

Table 1. Publications in terms of problem types and solution methods

Year	Author	Problem	Solution Methods
1989	Min [5]	Book Delivery and Pickup	Clustered Based Heuristic Method
2001	Dethloff [7]	VRPSLB	Mathematical Model and Heuristic Method
2005	Nagy and Salhi [8]	VRPSLB, Multi-Depot VRPSLB	Mathematical Model and Insertion-based Heuristic
2006	Montane and Galvao [20]	VRPSLB	Tabu Search Algorithm
2006	Dell'Amico et al. [9]	VRPSLB	Branch and Price Algorithm
2007	Wassan et al. [10]	VRPSLB	Reactive Tabu Search Algorithm
2009	Ai et al. [11]	VRPSLB	Particle Swarm Optimization Algorithm
2010	Subramanian et al. [12]	VRPSLB	Parallel Algorithm Based on the Sequential Heuristics
2013	Hezer and Kara [6]	VRPSLB	Bacterial Foraging Optimization Algorithm
2013	Goksal et al. [19]	VRPSLB	Discrete Particle Swarm Optimization
2015	Avci and Topaloglu [13]	VRPSLB and VRPMLB	Adaptive Local Search Algorithm
2015	Li et al. [14]	Multi-Depot VRPSLB	Adaptive Neighbourhood Selection
2015	Kececi et al. [17]	VRPSLB with heterogeneous vehicle fleet	Mathematical Model and Insertion-based Heuristic
2015	Nagy and Salhi [8]	VRPSLB	Mathematical Model and Variable Neighbourhood Search
2016	Zachariadis et al. [15]	VRPSLB with two-dimensional loading constraints	Local Search Algorithm
2016	Kalayci and Kaya [16]	VRPSLB	Ant Colony Optimization
2017	Yazgan and Buyukyilmaz [18]	VRPSLB	Insertion-based Heuristic
2020	This study	VRPSLB	Simulated Annealing Approach with Two Construction Heuristics

Mathematical Model

We have n customers that must be visited by k homogeneous vehicles. Each customer has a certain amount of pickup and delivery. When a vehicle visits a customer, the amount of pickup and delivery demands of a customer cannot exceed capacity of the vehicle. Vehicles must start from and return to the depot. The mathematical formulation for VRPSLB is mostly derived from the study of Montane and Galvao [20]. On the other hand, the maximum distance constraint is removed, and the maximum daily working time and service time constraints are inserted into the model. Followings show notations and decision variables:

Notations:
V: set of customers
V_0: set of customers and the depot
K: set of vehicles
c_{ij}: distance between locations of customers i and j
p_j: pickup demand of customer j, $\forall j \in V$
d_j: delivery demand of customer j, $\forall j \in V$
Q: vehicle capacity
T: working time limit for vehicles
S: Service time

Decision Variables:
x_{ijk} = 1, if vehicle k moves from customer i to j. 0, otherwise.
y_{ij} = Cumulative amount of pickup at node i and transported in arc (i, j)
z_{ij} = Cumulative amount of delivery at node i and transported in arc (i, j)

The corresponding mathematical formulation is given by
Minimize:

$$\sum_{k}^{K}\sum_{i}^{V_0}\sum_{j}^{V_0} c_{ij}x_{ijk} + \sum_{k}^{K}\sum_{i}^{V_0}\sum_{j}^{V_0} Sx_{ijk} i \neq j \quad (1)$$

Subject to:

$$\sum_{k}^{K}\sum_{i}^{V} x_{ijk} = 1, \forall j \in \mathrm{V} \quad (2)$$

$$\sum_{i}^{V_0} x_{ijk} - \sum_{i}^{V_0} x_{jik} = 0, \forall j \in V_0, \forall k \in K, i \neq j \quad (3)$$

$$\sum_{j}^{V_0} x_{0jk} \leq 1, \forall k \in K \quad (4)$$

$$\sum_{i}^{V_0} y_{ij} - \sum_{i}^{V_0} y_{ji} = d_j, \forall j \in \mathrm{V} \quad (5)$$

$$\sum_{i}^{V_0} z_{ij} - \sum_{i}^{V_0} z_{ji} = d_j, \forall j \in \mathrm{V} \quad (6)$$

$$y_{ij} + z_{ij} \le Q \sum_{k}^{K} x_{ijk}, \ \forall i, \ \forall j \in \mathrm{V} \quad (7)$$

$$\sum_{i}^{V_0}\sum_{j}^{V_0} c_{ij} x_{ijk} + S \sum_{i}^{V_0}\sum_{j}^{V_0} x_{ijk} \le T, \forall k \in K, i \ne j \quad (8)$$

$$x_{ijk} \in \{0,1\}, \forall i, \forall j \in \mathrm{V}_0, \ \forall k \in K, \ i \ne j \quad (9)$$

$$y_{ij}, z_{ij} \ge 0, \forall i, \forall j \in \mathrm{V}_0, \ i \ne j \quad (10)$$

The objective function (1) is to minimize total travel and service times. Constraint (2) makes sure that each customer is visited by exactly one vehicle. Constraint (3) ensures that the same vehicle arrives and departs from each customer. Constraint (4) shows the maximum number of vehicles allowed. Constraints (5) and (6) are flow equations for pick-up and delivery, respectively; they guarantee that both are satisfied for each customer. Constraint (7) ensures that pickup and delivery demands do not exceed capacity of the vehicle moving from customer *i* to *j*. Constraint (8) shows travelling and service times for a vehicle do not exceed the allowed working time. Constraints (9) and (10) demonstrate boundaries of decision variables.

Clark and Wright Algorithm

Clark and Wright algorithm, also called the saving algorithm, has been used to solve VRPs since 1964. It is a greedy heuristic and does not guarantee optimal solutions because it might get stuck at local optima. Clark and Wright algorithm is usually used as a construction heuristic to provide initial solutions to metaheuristic algorithms. The main idea behind the algorithm is to calculate insertion costs of each visit point (customer, patient, retailer, etc.) into intervals in the existing routes. There can be more than one insertion cost for a visit point if we have more than two assigned points in the schedule. The algorithm records the cheapest insertion cost for the visit point and iterates same procedure for other visit points. After calculating the cheapest insertion costs for all visit points, the visit point with the lowest cost compared to other points is selected and inserted into the selected interval if and only if the insertion satisfies the capacity and time constraints. If it does not, the algorithm evaluates the point with second lowest cost and so on. This procedure continues until no unassigned point exist or the routes are infeasible.

Let us assume that there are two customers, 1 and 3, and Euclidian distance between these two customers is C_{13}. If a new customer, 2, arrives to system and Euclidian distances between customer 1 and 2, and 2 and 3 are represented by C_{12} and C_{23} respectively, the cost of inserting customer 2 between 1 and 3 is calculated as following:

$$\text{Insertion cost} = C_{12} + C_{23} - C_{13} \quad (11)$$

Figure 1 shows the pseudo-code for Clark and Wright algorithm configurated based on VRPSLB. Each iteration in while loop (3–23), insertion cost of each customer (5–20) is calculated for each vehicle (7–15) and check whether they satisfy capacity and time constraints. Next, the customer with the cheapest insertion cost and satisfying constraints is inserted into the tour of vehicle and same procedure is repeated for the remaining customers.

The Nearest Neighbourhood Search

The nearest neighbourhood search is a constructive heuristic and one of the first heuristic algorithms to employ solving travelling salesman problem. The idea behind it is to find unvisited closest visit point to a reference point, the depot at the beginning of the day, add it into the existing tour, and continue this procedure till there is no unvisited node. The proximity between two nodes is calculated with the Euclidean distance. Because many vehicles exist in the problem, the distance for each visit point is calculated for the tour of each vehicle. Since we also deal with pickup and delivery demands of customers (visit points), not exceeding capacity of vehicles and satisfying working and service times are strict constraints.

Simulated Annealing Algorithm

Simulated annealing approach was developed by Kirkpatrick et al. [22] in the study, "Optimization by simulated annealing", 1983. The idea was to mimic the cooling of material in a heat bath. The process is known as "annealing". When allowing materials for controlled cooling, defects in materials significantly decrease. The method is a global optimization method consisted of exploration and exploitation procedures to be able to approximate optimal solution by preventing to get stuck at a local optimum. Simulated annealing method can accept worse solutions with a certain probability while heuristic methods such as the saving algorithm, nearest neighbourhood algorithm, hill climbing algorithm, etc. only move to better solution. Simulated annealing is highly popular method for solutions of continuous as well as discrete problems such as VRP. In following subsections, we explain simulated annealing method in detail.

Steps of Simulated Annealing Algorithm

Simulated annealing approach begins with an initial solution (S_i), generated with construction heuristics. In each step, the algorithm moves to a neighbour solution (S_n) with a predefined operator. If the objective function value of S_n is smaller than S_i for a minimization problem or larger for a maximization problem, the algorithm can accept the

```
Algorithm 1 Clark and Wright Agorithm
 1: Set Customers (V)
 2: Set Vehicles (K)
 3: while Customers list is not empty do
 4:     MinCustomerCost ← ∞
 5:     for n= 1 To V  do
 6:         MinVehicleCost ← ∞
 7:         for k= 1 To K  do
 8:             Calculate insertion cost of Customers[n] for Vehicles[k]
 9:             if insertionCost =< MinVehicleCost then
10:                 if n satisfies capacity and time constraints for Vehicles[k]
    then
11:                     MinVehicleCost←insertion cost
12:                     Index←k
13:                 end if
14:             end if
15:         end for
16:         if MinVehicleCost =< MinCustomerCost then
17:             MinCustomerCost←MinVehicleCost
18:             CustomerIndex←n
19:         end if
20:     end for
21:     Vehicles [Index]← Customers[CustomerIndex]
22:     Remove Customers[CustomerIndex] from Customers list
23: end while
```

Figure 1. Pseudo-code for Clark and Wright algorithm configurated based on VRPSLB.

solution and replace the initial solution with the neighbour solution. If the objective function value of S_n is greater than S_i for a minimization problem or smaller for a maximization problem, the algorithm can accept the solution with a certain probability. The acceptance probability, p_a, is calculated as following:

$$p_a = \frac{1}{e^{\Delta/t}} \tag{12}$$

T is a predefined control parameter and represents temperature during an annealing process. T continuously reduces by a constant multiplier, α, iteration by iteration. Δ value demonstrates the difference between objective function value of S_n and S_i. If a randomly generated number, r, is less than p_a, the algorithm accepts a worse neighbour solution for the next step. Note that the probability of accepting worse solutions is higher when T value is relatively high. In other words, at beginning of the process, when T value gets around the highest level, the algorithm tends to accept worse solutions. This is where the algorithm uses exploration steps mostly. On the other hand, T value starts to decrease when iterations proceed and the chance of accepting worse solutions significantly decreases. There are some important parameters for simulated annealing approach. These parameters, M, N, α, and aforementioned T, must be defined at the beginning. M and N show the number of iterations for exploration and exploitation steps, respectively.

Figure 2 shows pseudo code for simulated annealing approach. The algorithm assigns an initial solution generated by one of construction heuristics as in Line 1. After initializing M, N, α, and T, the algorithm starts iterations as Line 3–5. In each iteration, the algorithm moves a neighbour solution with a randomly selected operator and calculates objective function value. If the calculated value is less than the objective function value of initial solution, the new solution is accepted and replace the previous solution. Otherwise, the acceptance probability, pa, is calculated and checked whether it is greater than a random number, r. If so, the worse solution is accepted and iterations continue as shown in Line 5–20. T value is updated after each exploitation step (N) as in Line 21. In final step, the algorithm prints the best solution and corresponding objective function value.

Neighbourhood Structure

We have three operators, swap-in-vehicle, swap-between-vehicles, and take-and-insert, used for moving neighbour solutions. The probability of selecting an operator equals for these three.

Swap-in-vehicle Operator

Two of customers in the tour of a vehicle are selected and swapped with this operator as it is represented in Figure 3. Since we have multiple vehicles, one of vehicles that we apply swap-in-vehicle operator is randomly selected. Next, the operator randomly selects two customers in the tour of chosen vehicle and swaps their visits. If this change is feasible in terms of capacity and daily working time of the vehicle, the change is applied in new tour as seen in Figure 3.

Algorithm 2 Simulated Annealing Approach

```
1: S_i ← initial solution by a construction heuristic
2: F_Si ← objective function value of S_i
3: Initialize M,N,T,α
4: for m= 1 To M do
5:     for n= 1 To N do
6:         Select an operator randomly
7:         Find a feasible neighbour solution S_n
8:         F_Sn ← objective function value of S_n
9:         p_a ← 1/exp(Δ/T)
10:        r ← random.uniform(0,1)
11:        if F_Si => F_Sn then
12:            S_i ← S_n
13:            F_Si ← objective function value of S_i
14:        else if p_a => r then
15:            S_i ← S_n
16:            F_Si ← objective function value of S_i
17:        else
18:            Keep S_i and F_Si as initial solution
19:        end if
20:    end for
21:    T ← T*α
22: end for
23: Print S_i and F_Si
```

Figure 2. Pseudo-code for Simulated Annealing Algorithm.

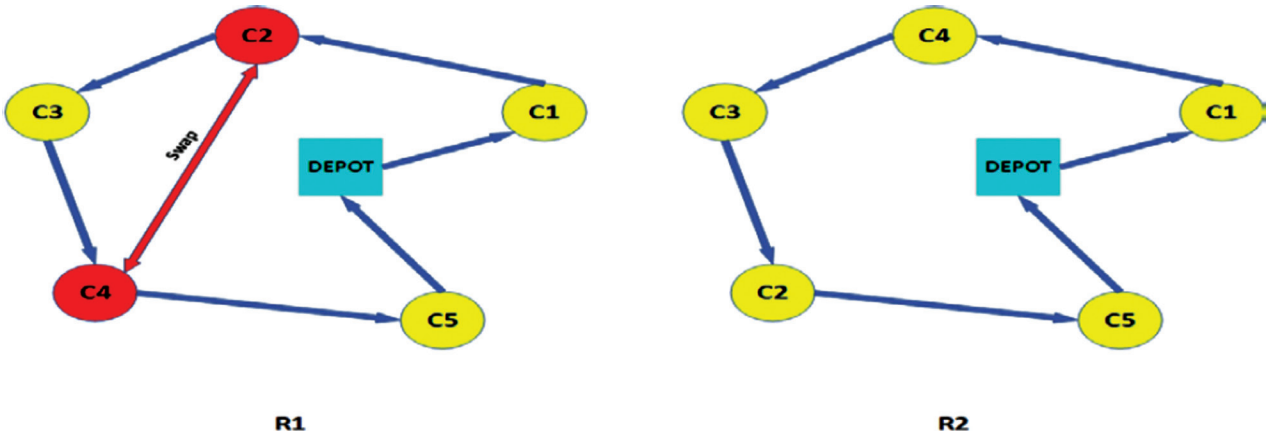

Figure 3. Swap-in-vehicle operator. R1 and R2 represent tours of the vehicle before and after the operator is applied.

Swap-between-vehicles Operator

Two customers from different vehicles' tours are selected and swapped with this operator as represented in Figure 4. The operator randomly selects two vehicles. Next, one customer is randomly chosen from the tour of first vehicle and another customer is randomly selected from the second vehicle's tour. Finally, visits of these two customers are swapped as seen in Figure 4. Of course, the algorithm checks whether capacity and daily working time constraints are satisfied after applying the operator.

Take-and-insert Operator

The algorithm randomly selects and removes a customer from one randomly selected vehicle's tour. After that, it inserts this customer into another randomly selected vehicle's tour. The interval where the customer is inserted is also

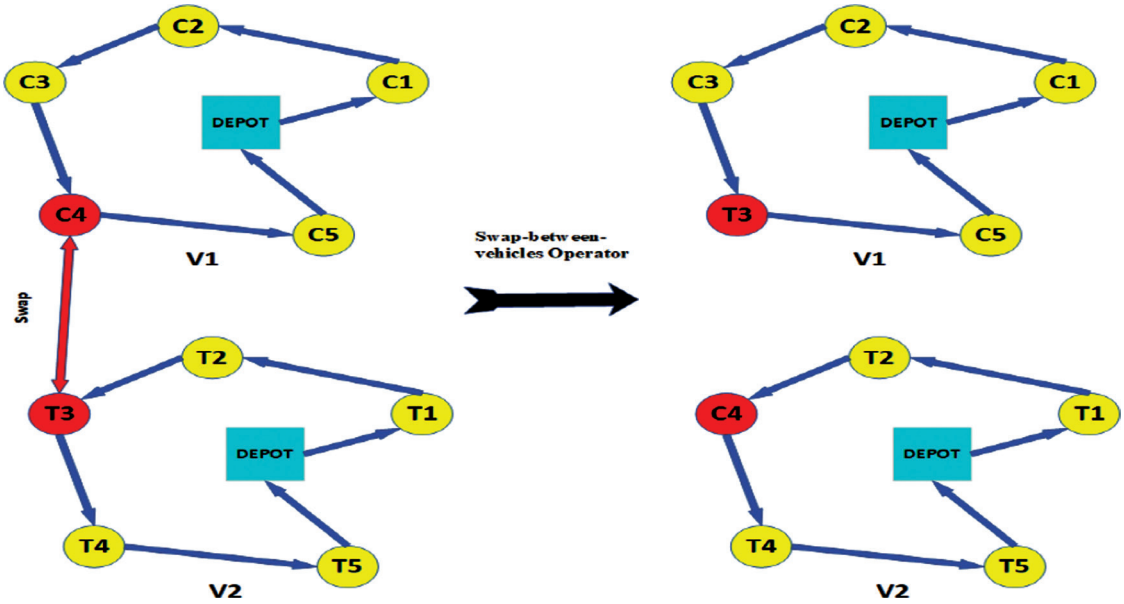

Figure 4. Swap-between-vehicles operator. V1 and V2 represent tours of two randomly selected vehicles.

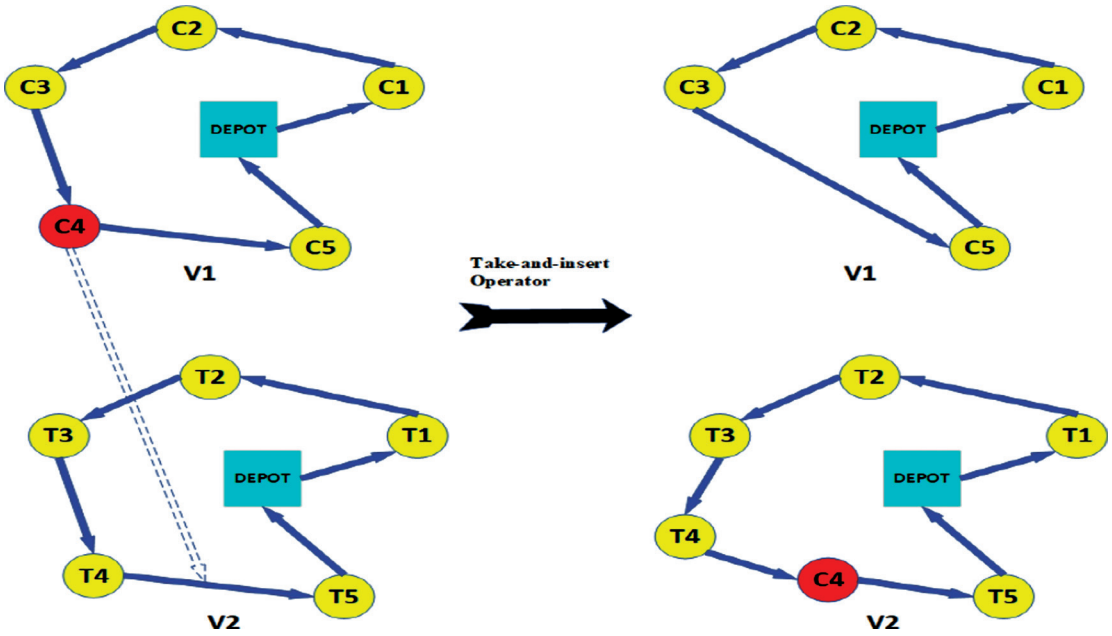

Figure 5. Take-and-insert operator. V1 and V2 represent tours of two randomly selected vehicles.

randomly selected by the operator. The change is applied if it does not violate capacity and daily working time constraints. Figure 5 represents how the operator works.

THE RESEARCH FINDINGS AND DISCUSSION

In this section, first we explain test settings. Next, results of tests are demonstrated and discussed.

Test Settings

As we mentioned before, VRP and its extensions are in NP Hard class and exact solution methods such as the MILP used in this study work for only small instance problems. To be able to show quality of proposed construction heuristics and the metaheuristic methods, we compare their results with the MILP for 20, 30, and 40 customers. Next, we test the simulated annealing method for 100, 150, and 200 customers where MILP cannot find optimal solutions in reasonable computational times. Delivery demands of customers are assigned uniformly between 4 and 10 packages, and pickup demands are assigned uniformly between 3 and 7 packages. We have different number of vehicles depending on the number of customers. Each vehicle capacity is 75 packages. The number of vehicles changes such a way that all customers must be visited by satisfying their pickup and delivery demands.

We consider three different service areas. Total area size is 3600 km^2 (60 km × 60km) and represented, $X \in [0, 60]$ and $Y \in [0, 60]$, in the coordinate system. In the first area type (R), customer requests can be raised from anywhere equally likely. Next, customer requests can be arrived from only three clusters in the service area (C). First cluster is in the area, $X_{C1} \in [5, 15]$ and $Y_{C1} \in [5, 15]$. Second one is the area, $X_{C2} \in [25, 35]$ and $Y_{C2} \in [45, 55]$. The last cluster is located in the area, $X_{C3} \in [50, 60]$ and $Y_{C3} \in [25, 35]$. Finally, 50% of customer requests can be arrived from three clusters and other requests can be raised from the remaining of the area equally likely (RC). The depot is located at the middle of the service area. Table 2 summarizes test settings.

Table 2. Test settings

Daily working time (min)	480
Service time (min)	5
Service area	$X \in [0, 60]$ and $Y \in [0, 60]$
Area type	Clustered, Random, Random-Clustered
Number of customers	20 to 200
Number of vehicles	2–21
Vehicle capacity (package)	75
Pickup demand (package)	Uniform (3, 7)
Delivery demand (package)	Uniform (4, 10)
α	0.92
T	1500
M	400
N	40

The mixed integer linear programming (MILP) is coded with AMPL (A mathematical programming language) IDE 3.5 software and solved with CPLEX 12.9. Construction heuristics and the simulated annealing algorithm are coded in Python programming language. Necessary parameters, M, N, α, and T, for the algorithm are taken from the literature and shown in Table 2. Test sets are solved in a PC with Intel i5 7200U 2.5 GHz CPU and 8 GB Ram.

RESULTS AND DISCUSSIONS

Table 3 shows total travel times for all vehicles under different test settings. MILP provides optimal solutions for each scenario and can be considered as a benchmark point. We generate initial solutions for the simulated annealing (SA) by Clark and Wright algorithm (CW) and the nearest neighbourhood algorithm (NN). In Table 3, we do not only show solutions of SA with the initial solution of one of construction heuristics, but also demonstrate solutions of heuristics separately. First, we mostly observe the closest travel times to MILP arriving from the combination of SA and NN (NN+SA). SA with NN results between 2.61% and 7.33% higher total travel times compared to MILP. Although NN provides initial solutions with higher travel times compared to CW in some cases, SA successfully approximates near optimal solutions when starting solutions NN generates. One reason can be that NN tends to assign customers to different vehicles at the beginning. Therefore, "swap-between-vehicles" might work better.

Table 4 shows execution times for solution methods under different test settings. As expected, computational times of MILP skyrocket when the number of customers increases. When we test methods for 40 customers, the execution times of MILP are around 2 hours. On the other hand, execution times of construction heuristics are under 1 second in each case. SA is also executed quickly compared to MILP under each scenario.

Table 5 shows total travel times for all vehicles for large instance problems and computational times of all methods. Since MILP can find optimal solutions around 2 hours for only 40 customers, it is impossible to find optimal solutions for more than 100 customers in reasonable computational times. For large problem sets, the combination of SA and NN seems almost equal performance to the combination of SA and CW. The improvements when comparing it to CW+SA are around 5%. For large instance problems, NN is very fast in terms of computational times. In most cases, NN generates initial solutions in less than one second. In some cases, execution times of CW are longer than SA.

Table 3. Travel times (minute) for proposed solution methods under different service areas and number of customers (**MILP**: Mixed Integer Linear Programming, **CW**: Clark and Wright Algorithm, **SA**: Simulated Annealing, **NN**: Nearest Neighbourhood Algorithm)

Service Area	Customer	MILP	CW	CW+SA	NN	NN+SA	%
R	20	**345.72**	376.71	**356.16**	403.38	374.46	3.01%
	30	**466.94**	541.79	517.36	545.74	**491.96**	5.36%
	40	**599.89**	664.48	653.50	695.17	**643.88**	7.33%
C	20	**282.67**	313.90	312.90	298.60	**292.58**	3.51%
	30	**387.64**	431.77	**403.94**	428.26	414.90	4.23%
	40	**513.96**	538.32	534.14	537.46	**529.24**	2.97%
RC	20	**354.27**	374.13	370.42	408.73	**370.37**	4.83%
	30	**438.85**	472.52	459.90	454.19	**450.31**	2.61%
	40	**532.16**	596.09	564.89	583.40	**563.58**	5.90%

Table 4. Execution times (second) for proposed solution methods under different service areas and number of customers (**MILP**: Mixed Integer Linear Programming, **CW**: Clark and Wright Algorithm, **SA**: Simulated Annealing, **NN**: Nearest Neighbourhood Algorithm)

Service Area	Customer	MILP	CW	CW+SA	NN	NN+SA
R	20	2.89	0.03	6.67	0.02	6.35
	30	95.85	0.09	9.34	0.01	9.47
	40	7205.94	0.57	10.51	0.01	9.50
C	20	1.60	0.04	6.32	0.00	6.28
	30	41.79	0.12	6.26	0.00	8.37
	40	7204.81	0.44	13.85	0.00	8.96
RC	20	47.40	0.05	7.58	0.00	9.92
	30	92.76	0.20	9.62	0.00	7.84
	40	429.97	0.25	8.83	0.01	9.20

Table 5. Travel times (minute) for proposed solution methods under different service areas and number of customers (**CW**: Clark and Wright Algorithm, **SA**: Simulated Annealing, **NN**: Nearest Neighbourhood Algorithm)

		Daily Travel Times (minute)				Execution Times (second)			
	Customer	CW	CW+SA	NN	NN+SA	CW	CW+SA	NN	NN+SA
R	**100**	1603.04	1473.40	1486.18	**1391.53**	3.70	33.62	0.12	38.18
	150	2111.83	2031.21	2093.99	**1991.15**	12.68	44.67	0.62	53.62
	200	2666.88	**2609.88**	2733.65	2624.86	31.09	55.93	1.33	54.80
C	**100**	1279.37	**1157.62**	1338.66	1289.72	4.73	30.70	0.18	31.97
	150	1731.23	1651.85	1656.83	**1619.93**	11.09	42.32	0.51	38.66
	200	2364.59	2334.30	2259.25	**2204.36**	35.52	50.30	0.97	49.11
RC	**100**	1276.74	**1205.80**	1353.45	1242.43	3.60	32.71	0.12	30.26
	150	1813.21	**1806.55**	1828.36	1860.42	13.60	35.97	0.47	32.65
	200	2314.66	**2244.92**	2392.80	2293.36	29.99	43.34	1.02	53.12

Overall, results show that SA provides shorter travel times once it starts initial solutions generated by NN for small sized problems. However, this advantage vanishes for large sized problems. NN is also very fast in terms of computational times compared to CW. If one seeks a quick solution methodology, NN can be preferable with solutions as good as solutions of CW. Moreover, when we consider total execution times, it is reasonable to employ both combinations for finding best solutions.

CONCLUSION

Vehicle Routing Problem (VRP) and its extensions have been studied for long years to be able to model real-life problems. Because of the pressure on companies to decrease costs not to fall back in the industry, they start to plan their pickup and delivery activities together. VRP with Simultaneous Linehaul and Backhaul (VRPSLB) is taken into consideration since then. Since VRP and its variations are in NP hard class, exact solution methods are useful for small instance problems. When size of problems increases, exact solution methods are unable to find optimal solutions in reasonable computational times. In this situations, heuristic and metaheuristic approaches are mostly used to find optimal or near optimal solutions. Metaheuristic methods such as the genetic algorithm, simulated annealing approach, tabu search, etc., are more successful to approximate optimal solutions compared to heuristic methods since they can avoid local optima with their special operators. However, heuristic approaches are mostly employed as initial solutions where metaheuristic methods start to search optimal solutions.

In this study, we proposed a simulated annealing (SA) approach for VRPSLB. We considered daily working times for vehicles and service times for customers against many studies [4–12] that do not cover these constraints in the literature. The other contribution in terms of methodological side is to propose two construction heuristics, Clark and Wright (CW) and the nearest neighbourhood (NN) algorithms, generating initial solutions for SA. The idea behind this approach is to find the best solution by trying two good initial solutions generated by two construction heuristics since the solution quality of a metaheuristic approach is directly related to the initial solution. We compare results of SA combined with CW and NN to the mixed integer linear programming (MILP) for small size problems, the number of customers varies 20 to 40. According to results, SA provides 2.61% to 7.33% longer travel times compared to MILP. On the other hand, SA combined with NN outperforms SA with CW in 7 out of 9 cases. For large size problems, up to 200 customers and 21 vehicles, SA shows similar performances combined with CW and NN. SA with CW provides shorter travel times in the half of cases while SA with NN works well in the other half. We cannot see any strong evidence about the fact that good initial solutions provide the best final solutions. In some cases, SA with NN results the shortest travel times even though the initial solution generated by NN is not as good as the initial solution of CW or vice versa. Moreover, both construction heuristics work indistinguishably under different service areas.

Overall, any construction heuristic does not seem preferable to the other in terms of final travel times for each scenario. However, NN is fast compared to CW even for large problem sets and testing SA with both NN and CW for problems is still reasonable in terms of execution times. Therefore, the algorithm can be upgraded in order to consider both heuristics and provide better results. On the other hand, NN can be chosen if the execution time is really matter. Service times for customers and daily working times for vehicles are two important real-life aspects in VRPSLB. Therefore, consideration of these two constraints makes our study more realistic for practitioners. Moreover, using of two construction heuristic with simulated annealing approach and inspecting results can attract researchers` attentions.

There is a shortcoming caused by the solution methodology itself. The simulated annealing approach uses some parameters (T, M, N, α) for exploration and exploitation steps. In this study, these parameters are received from the literature. However, different parameters can change results. We also consider only one test set for pickup and delivery demands of customers. Our assumption is that we have sufficient number of vehicles to serve all customers without any rejection. However, companies have a limited number of vehicles as well as capacities to answer customer requests in real life. Therefore, the rejection possibility and demand sensitivity should be taken into account in future research.

DATA AVAILABILITY STATEMENT

No new data were created in this study. The published publication includes all graphics collected or developed during the study.

CONFLICT OF INTEREST

The author declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

ETHICS

There are no ethical issues with the publication of this manuscript.

REFERENCES

[1] Dantzig GB, Ramser JH. The truck dispatching problem. Management Science 1959;6:80–91 [CrossRef]

[2] Laporte G. Fifty years of vehicle routing. Transportation Science 2009;43:408–16. [CrossRef]
[3] Demirbilek, M. A tactical/strategic level cost analysis based on visit time preferences for vehicle routing problem with simultaneous pickup and delivery. European Journal of Technique 2020;10:301–312. [CrossRef]
[4] Parragh SN, Doerner KF, Hartl RF. A survey on pickup and delivery problems. Journal für Betriebswirtschaft 2008;58:21–51. [CrossRef]
[5] Min H. The multiple vehicle routing problem with simultaneous delivery and pick-up points, Transportation Research Part A: General 1989;23:377–86. [CrossRef]
[6] Hezer S, Kara Y. Eşzamanlı dağıtımlı ve toplamalı araç rotalama problemlerinin çözümü için bakteriyel besin arama optimizasyonu tabanlı bir algoritma. Gazi Üniversitesi Mühendislik-Mimarlık Fakültesi Dergisi 2013;28:373–82.
[7] Dethloff J. Vehicle routing and reverse logistics: the vehicle routing problem with simultaneous delivery and pick-up, OR-Spektrum 2001;23:79–96. [CrossRef]
[8] Nagy G, Salhi S. Heuristic algorithms for single and multiple depot vehicle routing problems with pick-ups and deliveries European Journal of Operational Research 2005;162:126–41. [CrossRef]
[9] Dell'Amico M, Righini G, Salani M. A branch-and-price approach to the vehicle routing problem with simultaneous distribution and collection, Transportation Science 2006;40:235–47. [CrossRef]
[10] Wassan NA, Wassan AH, Nagy G. A reactive tabu search algorithm for the vehicle routing problem with simultaneous pickups and deliveries. Journal of Combinatorial Optimization 2008;15:368–86. [CrossRef]
[11] Ai TJ, Kachitvichyanukul V. A particle swarm optimization for the vehicle routing problem with simultaneous pickup and delivery. Computers & Operations Research 2009;36:1693–702. [CrossRef]
[12] Subramanian A, Drummond LMDA, Bentes C, Ochi LS, Farias R. A parallel heuristic for the vehicle routing problem with simultaneous pickup and delivery. Computers & Operations Research 2010;37:1899–911. [CrossRef]
[13] Avci M, Topaloglu S. An adaptive local search algorithm for vehicle routing problem with simultaneous and mixed pickups and deliveries. Computers & Industrial Engineering 2015;83:15–29. [CrossRef]
[14] Li J, Pardalos PM, Sun H, Pei J, Zhang Y. Iterated local search embedded adaptive neighbourhood selection approach for the multi-depot vehicle routing problem with simultaneous deliveries and pickups, Expert Systems with Applications 2015;42:3551–61. [CrossRef]
[15] Zachariadis EE, Tarantilis CD, Kiranoudis CT. The vehicle routing problem with simultaneous pick-ups and deliveries and two-dimensional loading constraints. European Journal of Operational Research 2016;251:369–86. [CrossRef]
[16] Kalayci CB, Kaya C. An ant colony system empowered variable neighbourhood search algorithm for the vehicle routing problem with simultaneous pickup and delivery, Expert Systems with Applications 2016;66:163–75. [CrossRef]
[17] Kececi B, Altiparmak F, Kara I. Heterogeneous vehicle routing problem with simultaneous pickup and delivery: Mathematical formulations and a heuristic algorithm. Journal of the Faculty of Engineering and Architecture of Gazi University 2015;30:185–95.
[18] Yazgan HR, Büyükyilmaz RG. Eş zamanlı topla dağıt araç rotalama problemine sezgisel bir çözüm yaklaşımı. Sakarya Üniversitesi Fen Bilimleri Enstitüsü Dergisi 2017;22:436–49.
[19] Goksal FP, Karaoglan I, Altiparmak F. A hybrid discrete particle swarm optimization for vehicle routing problem with simultaneous pickup and delivery. Computers & Industrial Engineering 2013;65:39–53. [CrossRef]
[20] Montané FAT, Galvao RD. A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and delivery service. Computers & Operations Research 2006;33:595–619. [CrossRef]
[21] Koç Ç, Laporte G. Vehicle routing with backhauls: review and research perspectives, Computers & Operations Research 2018;91:79–91. [CrossRef]
[22] Kirkpatrick S, Gelatt CD, Vecchi MP. Optimization by simulated annealing. Science 1983;220:671–80. [CrossRef]