Jurnal Teknik Industri, Vol. 23, No. 2, December 2021
ISSN 1411-2485 print / ISSN 2087-7439 online
DOI: 10.9744/jti.23.2.161-170

Comparison Study of Neighborhood Structures in Local Search for Vehicle Routing Problem with Multiple Trips and Time Windows

Saskia Puspa Kenaka[1]*, Suprayogi[1]

Abstract: This paper deals with the vehicle routing problem with multiple trips and time windows (VRPMTTW). The problem combines two features to the basic vehicle routing problem (VRP): multiple trips and time windows. The basic VRP and its variants fall into hard combinatorial optimization problems. Therefore, many heuristics have been developed to solve large problems. Local search (LS) is one of the heuristics using a neighborhood structure in order to find a better solution. This paper focuses on the comparative study of neighborhood structures applied in LS for the VRPMTTW. Two classes of neighborhood structures are compared: Tour-based and permutation-based neighborhood structures. The tour-based neighborhood structures are the neighborhoods in which the moves are performed on the solution represented by a set of tours. The permutation-based neighborhood structures are the neighborhood structures in which the moves are carried out on the solution represented by a permutation of customers. Comparing the performance of the neighborhood structures uses two responses: (1) relative improvement in the objective function value and (2) computation time. The first response is used to measure effectiveness, while the second is used as efficiency. Based on the computational experiment results, it is generally revealed that the permutation-based neighborhood structures used in LS are more effective than the tour-based neighbourhood structures. However, the permutation-based neighborhood structures are less efficient because they give higher computation times.

Keywords: Vehicle routing problem, multiple trips, time windows, local search, neighborhood structure.

Introduction

The vehicle routing problem with multiple trips and time windows (VRPMTTW) is a variant of the basic vehicle routing problem (VRP) combining the following aspects: multiple trips and time windows. In the VRPMTTW, there are two main features added to the basic VRP: (1) each vehicle may perform more than one route or trip during a planning horizon, and (2) each customer has a time window indicated by the earliest and latest times to start the service.

The VRPMTTW has been investigated intensively. Fleischmann [1], who introduced the VRP with multiple trips (VRPMT), has considered time windows in his work. Brandão and Mercer [2] included additional features such as heterogeneous vehicles, site-dependent, maximum legal driving, maximum, and legal time breaks. Other studies on the VRPMTTW were as follows: Suprayogi [3], Suprayogi and Imawati [4], Suprayogi *et al.* [5], Suprayogi *et al.* [6], Azi *et al.* [7], Macedo *et al.* [8], Suprayogi *et al.* [9], Wang *et al.* [10], Hernandez *et al.* [11], Cattaruzza *et al.* [12], and Neira *et al.* [13]. Suprayogi and Mahaputra [14] and Suprayogi and Priyandari [15] discussed the VRPMTTW with simultaneous delivery and pickup. Ong and Suprayogi [16] and Suprayogi *et al.* [17] studied the VRPMTTW with backhauls.

The VRPMTTW discussed in this paper is associated with a delivery service consisting of a depot and a set of customers. There is an unlimited number of vehicles stationed at the depot where they are homogeneous. The number of goods to be delivered to each customer is known during the planning horizon. The service-dependent times (i.e., the loading times at the depot for a particular route depend on the quantity to be delivered on the route) are considered. In addition, the time to start the service for each vehicle is made as latest as possible. Therefore, the time to start the service at the depot (loading activity) for the first route is not necessary to begin at time 0. A solution of the VRPMTTW consists of a set of tours where a tour is defined as a set of consecutive routes. A route composes a subset of customers where it starts and ends at the depot. Each tour is serviced exactly by one vehicle. The objective function to be minimized is the total cost, including vehicle cost and travel cost. In this paper, a mixed-integer linear programming (MILP) model is formulated to represent the VRPMTTW.

[1] Faculty of Industrial Technology, Industrial System and Technology Economics Research Group, Institut Teknologi Bandung, Jl. Ganesha 10, Bandung 40116 Indonesia.
Email: saskia@mail.ti.itb.ac.id, yogi@mail.ti.itb.ac.id

*Corresponding author

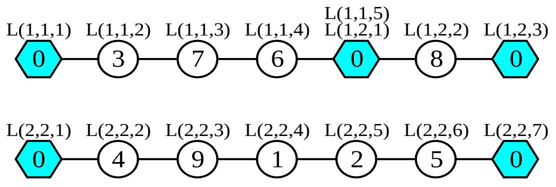

L(1,1,1) L(1,1,2) L(1,1,3) L(1,1,4) L(1,1,5) L(1,2,1) L(1,2,2) L(1,2,3)
0 3 7 6 0 8 0
L(2,2,1) L(2,2,2) L(2,2,3) L(2,2,4) L(2,2,5) L(2,2,6) L(2,2,7)
0 4 9 1 2 5 0

Figure 1. Tour-based representation

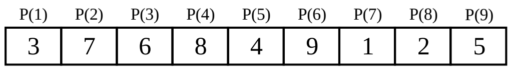


Figure 2. Permutation-based representation

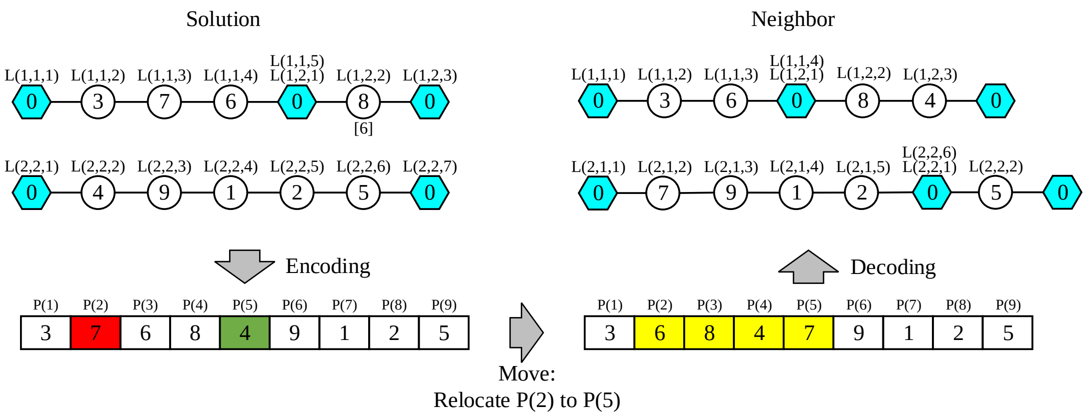

Solution
L(1,1,1) L(1,1,2) L(1,1,3) L(1,1,4) L(1,1,5) L(1,2,1) L(1,2,2) L(1,2,3)
0 3 7 6 0 8 0
[6]
L(2,2,1) L(2,2,2) L(2,2,3) L(2,2,4) L(2,2,5) L(2,2,6) L(2,2,7)
0 4 9 1 2 5 0
Encoding
P(1) P(2) P(3) P(4) P(5) P(6) P(7) P(8) P(9)
3 7 6 8 4 9 1 2 5
Move:
Relocate P(2) to P(5)
Neighbor
L(1,1,1) L(1,1,2) L(1,1,3) L(1,1,4) L(1,2,1) L(1,2,2) L(1,2,3)
0 3 6 0 8 4 0
L(2,1,1) L(2,1,2) L(2,1,3) L(2,1,4) L(2,1,5) L(2,2,6) L(2,2,1) L(2,2,2)
0 7 9 1 2 0 5 0
Decoding
P(1) P(2) P(3) P(4) P(5) P(6) P(7) P(8) P(9)
3 6 8 4 7 9 1 2 5

Figure 3. Process of constructing a neighbor solution

Several solution methods have been proposed for the VRPMTTW. One of them is local search (LS). LS is a solution approach based on the local improvement for the current best solution. LS requires an initial solution where it is set as the current best solution once the procedure starts. In the improvement phase, LS explores neighbor solutions based on the current best solution using all possible moves of a particular neighborhood structure. A feasible neighbor solution better than the current best solution is accepted as the new best solution. The procedure in LS terminates where there is no improvement for all possible moves of the neighborhood structure.

The solution method based on LS for the VRPMTTW is proposed in this paper. The customers in the solution are arranged in an array $L(t,r,k)$ where indices t, r, and k denote tour, route, and position, respectively. Index t is from 1 to NT where NT denotes the number of tours. Index r ranges from 1 to $NR(t)$ where $NR(t)$ is the number of routes of tour t. Index k starts from 1 to $NP(t,r)$ where $NP(t,r)$ is the number of positions on route r of tour t. Note that $L(t,r,1) = L(t,r,(NR(t,r)) = 0$ where it represents the depot.

To illustrate the solution, consider a problem consisting of one depot (denoted by number 0) and a set of customers (denoted by numbers 1 to 9). The quantity of delivery demand. One of the solutions is represented in Figure 1. The solution consists of two tours. The first tour constitutes two routes (0-3-7-6-0 and 0-8-0), while the second has only one route (0-4-9-1-2-5-0).

Suprayogi *et al.* [6] have proposed eleven neighbor structures applied for the VRPMTTW where they are extensions of the basic neighborhood structures applied for the basic VRP, including relocation, exchange, and cross. The neighborhood structures are based on the moves performed on the original representation of the solution consisting of a set of tours. In this paper, these neighborhood structures are called tour-based neighborhood structures. Figure 2 illustrates permutation where it is similar to the chromosome representation used in the genetic algorithm. To generate neighbor solutions using a particular neighborhood structure, the encoding procedure is used to convert the current best solution to its permutation representation. Then, moves are performed to generate new permutation representations. Finally, to obtain the neighbor solution, a decoding procedure is applied. In this paper, the neighborhood structures perform their moves on the permutation called permutation-based neighborhood structures. Figure 3 illustrates the process of constructing a neighbor solution.

In this paper, LS using tour-based neighborhood structure is called TBLS, while LS using permutation-based neighborhood structure is called PBLS. Both TBLS and PBLS have their strengths and weaknesses. Because the moves are carried out on the best current solution, the neighbor solutions obtained by TBLS may be restricted due to resulting infeasibilities in terms of vehicle capacity and time window constraints. On the other hand, in PBLS, there is no infeasibility issue of the resulting neighbor solutions due to the assumption of an unlimited number of vehicles. In this paper, a comparison study is conducted in order to analyze the performance of neighborhood structures applied in local search (LS) for the VRPMTTW. Comparing the performance uses two responses: (1) relative improvement in the objective function value and (2) computation time. The first response is used to measure effectiveness, while the second is used as efficiency.

Methods

Mixed Integer Linear Programming Model

The VRPMTTW considered in this paper consists of a set of n customers $C = \{1, 2, \cdots, n\}$ and a single depot denoted by number 0. The MILP model formulated in this paper is adapted from Azi *et al.* [7].

As in Azi *et al.* [7], to formulate into a MILP model, the VRPMTTW is represented in a complete directed graph $G = \{V, A\}$ where V is the set of nodes and A is the set of arcs. The depot 0 is duplicated by a virtual depot represented by number $n + 1$, Each route starts at depot 0 and ends at depot $n + 1$. Virtual depot $n + 1$ is included into V. Therefore, $V = \{0\} \cup C \cup \{n + 1\} = \{0, 1, \cdots, n, n + 1\}$.

According to Azi *et al.* [12], define $R = \{1, 2, \cdots, R'\}$ as a set of routes. Notation R' denotes the number of routes where it is determined in advance such that it is large enough to accommodate the maximum number of routes in the solution. The routes served by any vehicle are numbered in an increasing order, i.e., a vehicle serves route $s \in R$ after route $r \in R$ if and only if $r < s$.

To complete the formulation of the MILP model, the following notations are defined:
Sets:

V	:	set of nodes, $V = \{0, 1, \cdots, n, n + 1\}$
A	:	set of arcs
R	:	set of routes, $R = \{1, 2, \cdots, R'\}$

Parameters

ω_1	:	cost per vehicle unit
ω_2	:	cost per distance unit
ϕ	:	vehicle capacity
ξ_{ij}	:	distance on arc $(i, j) \in A$ ($\xi_{0j} = \xi_{i,n+1}$ for $i, j \in V$; $\xi_{0,n+1} = 0$)
τ_{ij}	:	travel time on arc $(i, j) \in A$ ($\tau_{0i} = \tau_{n+1,j}$ for $i, j \in V$; $\tau_{0,n+1} = 0$)
δ_i	:	quantity of delivery demand at node $i \in V$ ($\delta_0 = \delta_{n+1} = 0$)
α_i	:	earliest time to start the service at node $i \in V$ (α_0 and α_{n+1} are the opening times for the depot or the lower bounds of the planning period)
β_i	:	latest time to start the service at node $i \in V$ (β_0 and β_{n+1} are the closing times for the depot or the upper bounds of the planning period)
γ	:	loading time per unit
φ	:	unloading time per unit
M	:	big positive number

Decision variables:

X_{ijr}	:	binary variable indicating whether arc $(i, j) \in A$ is served by route $r \in R$ or not (note that if $X_{0,n+1,r} = 1$, then route r is empty)
Z_{rs}	:	binary variable indicating whether route $r \in R$ is followed immediately by route $s \in R$ or not
T_{ir}	:	time to start the service at node $i \in V$ on route $r \in R$
L_r	:	loading time at the depot for route $r \in R$
D_{ijr}	:	delivery load on arc $(i, j) \in A$ of route $r \in R$
K	:	number of vehicles deployed

The MILP model for the VRPMTTWSDP is formulated as follow:

Minimize

$$Z = \omega_1 K + \omega_2 \sum_{i \in V} \sum_{j \in V} \sum_{r \in R} \xi_{ij} X_{ijr} \quad (1)$$

subject to

$$\sum_{i \in V \setminus \{0\}} X_{0ir} = 1; r \in R \quad (2)$$

$$\sum_{i \in V \setminus \{n+1\}} X_{i,n+1,r} = 1; r \in R \quad (3)$$

$$\sum_{i \in V \setminus \{n+1\}, i \neq h} X_{ihr} = \sum_{j \in V \setminus \{0\}, j \neq h} X_{hjr}; \; h \in V \setminus \{0, n+1\}; r \in R \quad (4)$$

$$T_{ir} + \varphi \delta_i + \tau_{ij} \leq T_{jr} + M(1 - X_{ijr}); \; i \in V \setminus \{n+1\}, j \in V \setminus \{0\}, i \neq j, r \in R \quad (5)$$

$$T_{ir} \geq \alpha_i; i \in V \setminus \{0, n+1\}, r \in R \quad (6)$$

$$T_{ir} \leq \beta_i; i \in V \setminus \{0, n+1\}, r \in R \quad (7)$$

$$T_{0r} \geq \alpha_0; r \in R \quad (8)$$

$$T_{0r} + L_r \leq \beta_0; r \in R \quad (9)$$

$$T_{n+1,r} \geq \alpha_{n+1}; r \in R \quad (10)$$

$$T_{n+1,r} \leq \beta_{n+1}; r \in R \quad (11)$$

$$L_r = \gamma \sum_{i \in V \setminus \{0,n+1\}} D_{0ir}; r \in R \quad (12)$$

$$T_{n+1,r} \leq T_{0s} + M(1 - Z_{rs}); \; r \in R, s \in R, r < s \quad (13)$$

$$T_{n+1,r} \geq T_{0s} - M(1 - Z_{rs}); r \in R, s \in R, r < s \quad (14)$$

$$\sum_{r \in R} \sum_{s \in R, r < s} Z_{rs} \geq R' - K \quad (15)$$

$$\sum_{s \in R, s > r} Z_{rs} \leq 1; r \in R \quad (16)$$

$$\sum_{r \in R, s > r} Z_{rs} \leq 1; s \in R \quad (17)$$

$$D_{i,n+1,r} = 0; \; i \in V \setminus \{n+1\}; r \in R \quad (18)$$

$$\sum_{i \in V \setminus \{n+1\}, i \neq j} \sum_{r \in R} D_{ijr} - \sum_{i \in V \setminus \{n+1\}, i \neq j} \sum_{r \in R} D_{jir} = \delta_j; j \in V \setminus \{0, n+1\} \quad (19)$$

$$\sum_{i \in V \setminus \{0,n+1\}} \sum_{r \in R} D_{0ir} = \sum_{i \in V \setminus \{0,n+1\}} \delta_i \quad (20)$$

$$D_{ijr} \leq \phi X_{ijr}; \; i \in V \setminus \{n+1\}, j \in V \setminus \{0\}; \; i \neq j, r \in R \quad (21)$$

$$X_{ijr} \in \{0,1\}; i \in V \setminus \{n+1\}, j \in V \setminus \{0\}; \; i \neq j, r \in R \quad (22)$$

$$Z_{rs} \in \{0,1\}; r \in R, s \in R, r < s \quad (23)$$

$$T_{ir} \geq 0; i \in V, r \in R \quad (24)$$

$$D_{ijr} \geq 0; i \in V, j \in V, r \in R \quad (25)$$
$$L_r \geq 0; r \in R \quad (26)$$
$$K \geq 0 \quad (27)$$

The objective function to be minimized is the total cost represented in Eq. (1), where it is a sum of vehicle cost and travel cost. Constraint (2) guarantees that each route must start from the depot. Constraint (3) ensures that each route has to ends at the depot. Constraint (4) is used to ensure each customer is served by the same route. Constraint (5) defines the relationship between the times to start the service. Constraints (6)-(7) are time window constraints for each customer. Constraints (8)-(11) are time window constraints for the depot where they also represent the lower and upper bounds of the planning horizon). The loading times at the depot are defined by constraints (12). Constraints (13)-(17) are used to ensure the route sequence. Constraint (18) determines the delivery load when a vehicle returns at the depot. Constraint (19) is the flow conservation constraint for the delivery quantity at each customer. Constraint (20) is used to define the total load from the depot. The vehicle capacity constraint is given in constraint (21). Constraints (22)-(27) are constraints related to decision variable values.

Initial Solution

LS requires an initial solution. In this paper, sequential insertion (SI) algorithm is applied. Steps of the sequential insertion are given in algorithm SI.

Algorithm SI

```
1. Set $t = 1$ and $NT = 1$.
2. Set $r = 1$ and $NR(t) = 1$.
3. Select an unassigned customer $jmin$ with the minimum latest time to start the service.
4. Set $L(t, r, 2) = jmin$. Update the solution.
5. If all customers have been assigned, then go to 13.
6. Try to insert unassigned customers to each position of the current route. If there are feasible insertions with respect to the vehicle capacity and time window constraints, then go to 7. Otherwise, go to 8.
7. Select an unassigned customer $jmin$ and a position of current route $kmin$ giving the minimum total distance. Set $L(t, r, kmin) = jmin$. Update the solution. Go to 5.
8. Set $r = r + 1$ and $NR(t) = NR(t) + 1$.
9. Try to assign unassigned customers to the current route.
10. If there are feasible assignment with respect to the time window constraint, then go to 11. Otherwise, set $r = r - 1$, $NR(t) = NR(t) - 1$ and go to 12.
11. Select an unassigned customer $jmin$ with the minimum total distance time. Set $L(t, r, 1) = jmin$. Update the solution. Go to 5.
12. Set $t = t + 1$ and $NT = NT + 1$. Go back to step 2.
13. Perform a backward procedure to determine the latest arrival times, the latest times to start the service, and the latest departure times
14. Calculate the objective function value. Stop.
```

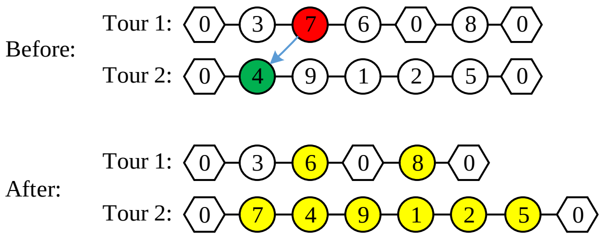


Figure 4. Inter-tour relocation 1-0

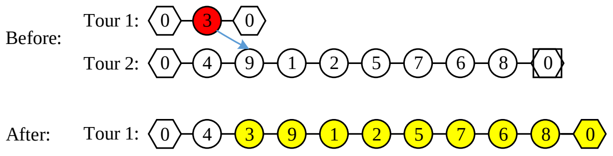


Figure 5. Inter-tour relocation 1-0 causing a tour elimination

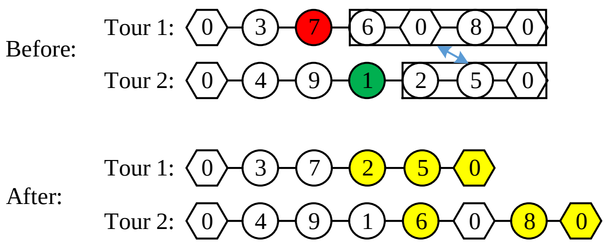


Figure 6. Cross

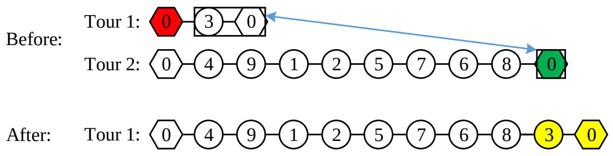


Figure 7. Cross causing a tour elimination

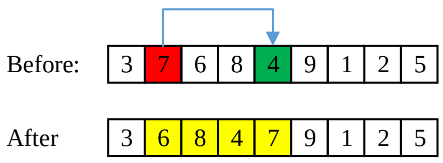


Figure 8. Relocation 1-0

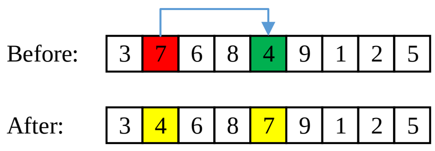


Figure 9. Exchange 1-1

Table 1. Neighborhood structured used by TBLS

Neighborhood structure	Abbreviation	Description
Inter-tour relocation 1-0	IRR 1-0	Relocate or insert one customer from position $k1$ on route $r1$ of tour $t1$ to position $k2$ on route $r2$ of tour $t2$ ($t2 \neq t1$)
Intra-tour relocation 1-0	IAR 1-0	Relocate or insert one customer from position $k1$ on route $r1$ of tour $t1$ to position $k2$ on route $r2$ of tour $t1$
Inter-tour exchange 1-1	IRE 1-1	Exchange or swap one customer at position $k1$ on route $r1$ of tour $t1$ with another customer at position $k2$ on route $r2$ of tour $t2$ ($t2 \neq t1$)
Intra-tour exchange 1-1	IAE 1-1	Exchange or swap one customer at position $k1$ on route $r1$ of tour $t1$ with another customer at position $k2$ on route $r2$ of tour $t1$
Cross	CRS	Exchange or swap a segment of customers starting at position $k1$ on route $r1$ of tour $t1$ with another segment of customers starting at position $k2$ on route $r2$ of tour $t2$ ($t2 \neq t1$)

Table 2. Neighborhood structures used by PBLS

Neighborhood structure	Abbreviation	Description
Relocation 1-0	REL 1-0	Move one customer at positon i to position j ($j \neq i$)
Relocation 2-0	REL 2-0	Move two consecutive customers starting at positon i to position j ($j \neq i$)
Relocation 3-0	REL 3-0	Move three consecutive customers starting at positon i to position j ($j \neq i$)
Exchange 1-1	EXC 1-1	Exchange one customer at position i with another customer at position j ($j > i$)
Exchange 2-2	EXC 2-2	Exchange two consecutive customers starting at position i with other two consecutive customers starting at position j ($j > i$)
Exchange 3-3	EXC 3-3	Exchange three consecutive customers starting at position i with other three consecutive customers starting at position j ($j > i$)
Inversion	INV	Invert the order of customers from position i to position j ($j > i$)
Shift	SHI	Shift customers starting at position i to position j ($j \neq i$)

Neighborhood Structures

As mentioned before, LS explores neighbor solutions based on the current best solution in order to seek the new best solution. The exploration utilizes a particular neighborhood structure. In this paper, five neighborhood structures used in TBLS are given in Table 1. Two neighborhood structures can reduce the number of tours from five neighborhood structures: inter-tour relocation 1-0 and cross. Figure 4 and Figure 5 show illustrations of inter-tour relocation 1-0 (IRR 1-0), while Figure 6 and Figure 7 depict illustrations of the cross (CRS).

There are eight neighborhood structures used in PBLS where they are shown in Table 2. Figure 8 and Figure 9 illustrate relocation 1-0 (REL 1-0) and Exchange 1-1 (EXC 1-1).

Local Search

In the search process, LS can apply either first improvement or best improvement strategies. In this paper, both proposed TBLS and PBLS apply the first improvement strategy.

Steps of TBLS are shown in Algorithm TBLS. In TBLS, the feasibility check is performed for each neighbor solution generated by a particular move. If a neighbor solution is infeasible for a particular move, the procedure searches for the next possible move.

Steps of PBLS are shown in Algorithm PBLS, which includes both encoding and decoding procedures. As mentioned before, the encoding procedure transforms the original representation of the current best solution to its permutation representation. It is simply performed by creating a permutation of customers starting from the customer appearing on the first route of the first tour of the solution. The decoding procedure is the opposite procedure of the encoding procedure, where it converts the permutation representation to the original solution representation of the neighbor solutions. The procedure is similar to the sequential insertion where customers are added one by one to the solution based on their order in the permutation. The vehicle capacity and time window constraints are considered in building routes and tours. Steps of the decoding procedure are given in **Algorithm Decoding**. By applying the decoding procedure, the neighbor solutions produced by PBLS are always guaranteed to be feasible.

Algorithm TBLS

```
1. Create $S^0$ using SI.
2. Set $S^* = S^0$ and $f(S^*) = f(S^0)$.
3. Set $S' = S^*$.
4. Set a move as the first move.
5. Generate neighbor solution $S$ based $S'$ on using the move.
6. If neighbor solution $S$ is infeasible then go to 8.
7. If $f(S) < f(S^*)$, then, set $f(S^*) = f(S)$ and $S^* = S$. Go to 3.
8. Go to next move.
9. If the last move has not been executed, go to 5. Otherwise, stop.
```

Algorithm PBLS

```
1. Create $S^0$.
2. Set $S^* = S^0$ and $f(S^*) = f(S^0)$.
3. Set $S' = S^*$.
4. Set a move as the first move.
5. Encode $S'$ to produce $X'$.
6. Generate $X$ based $X'$ on using the first move.
7. Decode $X$ to produce $S$.
8. If $f(S) < f(S^*)$, then, set $f(S^*) = f(S)$, $S^* = S$.
   Go to 3.
9. Go to next move.
10. Set a move is the next move.
11. If the last move has not been executed, go to 6.
    Otherwise, stop
```

Algorithm Decoding

```
1. Set $j = 1$.
2. Set $t = 1$ and $NT = 1$.
3. Set $r = 1$ and $NR(t) = 1$.
4. Set $k = 2$.
5. Set $L(t, r, k) = P(j)$. If it is feasible, then update
   the solution. Otherwise, go to 8.
6. If all customers have been assigned, then go to 9.
7. Set $j = j + 1$ and $k = k + 1$. Set $L(t, r, k) = P(j)$. If
   it is feasible, then update the solution and go to 6.
   Otherwise, set $r = r + 1$, $NR(t) = NR(t) + 1$. Go to 4.
8. Set $t = t + 1$ and $NT = NT + 1$. Go to 2.
9. Perform a backward procedure to determine the
   latest arrival times, the latest times to start the
   service, and the latest departure times.
10. Calculate the objective function value. Stop
```

Results and Discussions

Solver and Test Instances

LS is coded using Visual Basic 6 and it run on a PC with the following specifications: Processor Intel(R) Core(TM) i7-9700T CPU @ 2.00GHz, 1992 Mhz, 8 Core(s), 8 Logical Processor(s), Installed Physical Memory (RAM):16.0 GB, and Microsoft Windows 10 Home Single Language Operating system.

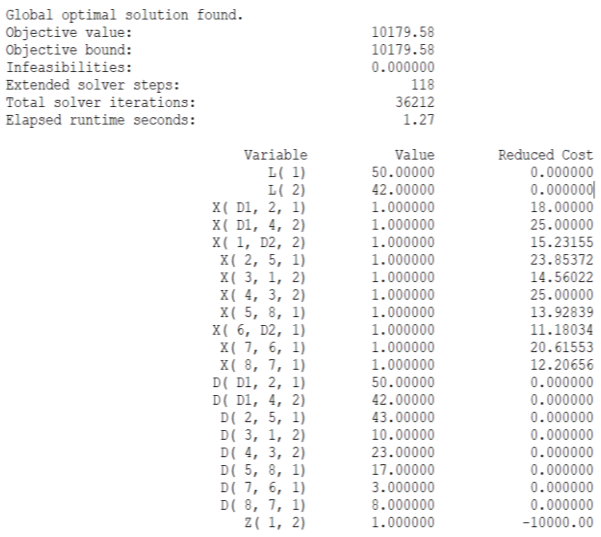
Test instances are generated from Solomon's instances consisting of 100 customers (Solomon [18]). There are six classes of Solomon's instances depending on the customer location and the length of the planning horizon. In this paper, Solomon's instances of C2 (C201-C208), R2 (R201-211), and RC2 (RC201-RC208) are used to generate large test instances by taking the first 25 and 40 customers. Some modifications are made from the original instances. The service times are discarded. The loading time per unit at the depot and the loading time per unit at the customers are set as 1. The cost per vehicle unit and the cost per distance unit are set as 10000 and 1, respectively. Two vehicle capacities are applied for each instance, i.e., 50 and 100. Therefore, the total test instances are 108.

In order to compare with the optimal solution obtained by solving the MILP model, a set of small test instances consisting of 18 instances are derived from R201, R205, and R211 by taking the first 5 to 10 customers. The MILP model is solved using LINGO 17 run on the same PC. In order to solve the MILP model, the number of routes R' has to be determined in advance for each instance. The computation time is limited to 7200 seconds.

Illustrative Example

The following subsection gives an illustrative example of the VRPMTTW. Instance R205-008 is used for the illustrative example. The optimal solution obtained by solving the MILP model is depicted in Figure 10 where only nonzero-valued variables are displayed. The total cost is 10179.58. Based on values of variable *X*, the solution consists of two routes. The first route (route 1) is D-2-5-8-7-6-D and the second one (route 2) is D-4-3-1-D. These routes are performed by a single vehicle (tour) where the second route follows the first route (denoted by variable *Z*). Loading times at depot for the first and second routes (denoted by variable *L*) are 50 and 42, respectively. The total loads for each customer pair on each route are given in values of variable *D*.

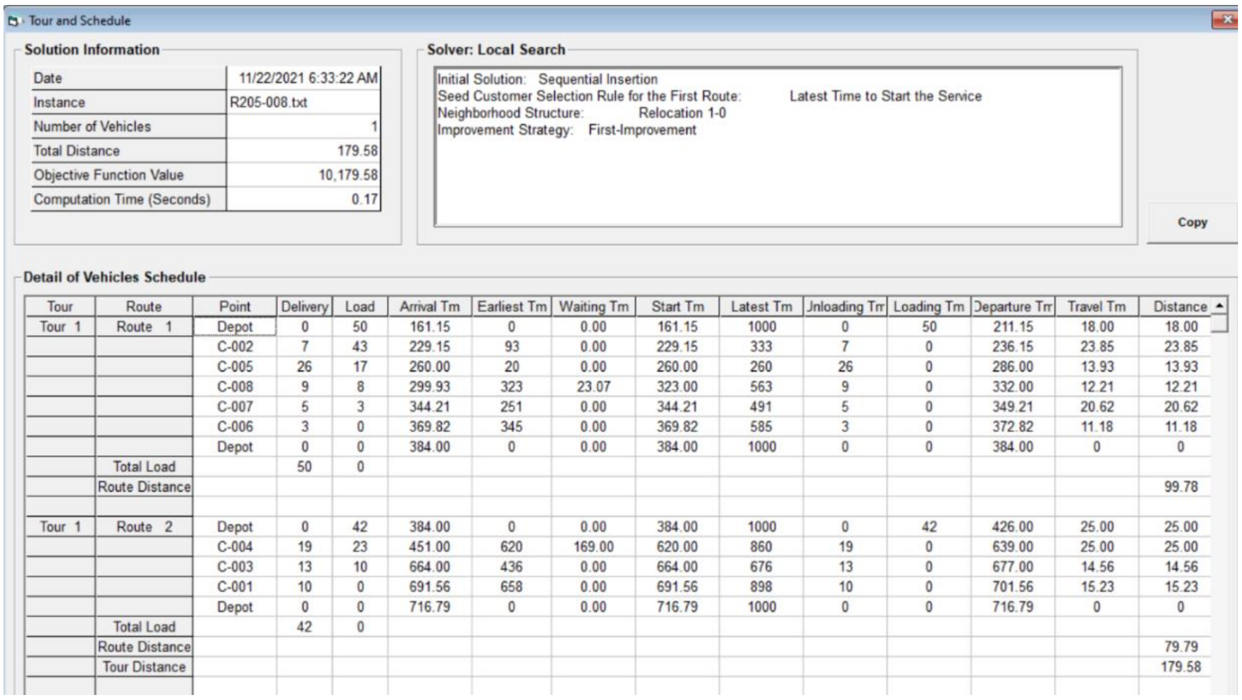
The solution produced by LS using relocation 1-0 (REL 1-0) is shown in Figure 11. This solution is optimal. Information including arrival times, times to start the service, and departure times are also given.



```
Global optimal solution found.
Objective value:                              10179.58
Objective bound:                              10179.58
Infeasibilities:                              0.000000
Extended solver steps:                             118
Total solver iterations:                         36212
Elapsed runtime seconds:                          1.27

                        Variable           Value        Reduced Cost
                           L( 1)        50.00000            0.000000
                           L( 2)        42.00000            0.000000
                     X( D1, 2, 1)        1.000000            18.00000
                     X( D1, 4, 2)        1.000000            25.00000
                     X( 1, D2, 2)        1.000000            15.23155
                      X( 2, 5, 1)        1.000000            23.85372
                      X( 3, 1, 2)        1.000000            14.56022
                      X( 4, 3, 2)        1.000000            25.00000
                      X( 5, 8, 1)        1.000000            13.92839
                     X( 6, D2, 1)        1.000000            11.18034
                      X( 7, 6, 1)        1.000000            20.61553
                      X( 8, 7, 1)        1.000000            12.20656
                     D( D1, 2, 1)        50.00000            0.000000
                     D( D1, 4, 2)        42.00000            0.000000
                      D( 2, 5, 1)        43.00000            0.000000
                      D( 3, 1, 2)        10.00000            0.000000
                      D( 4, 3, 2)        23.00000            0.000000
                      D( 5, 8, 1)        17.00000            0.000000
                      D( 7, 6, 1)        3.000000            0.000000
                      D( 8, 7, 1)        8.000000            0.000000
                         Z( 1, 2)        1.000000           -10000.00
```

Figure 10. MILP's solution



Solution Information

Date	11/22/2021 6:33:22 AM
Instance	R205-008.txt
Number of Vehicles	1
Total Distance	179.58
Objective Function Value	10,179.58
Computation Time (Seconds)	0.17

Solver: Local Search

```
Initial Solution:   Sequential Insertion
Seed Customer Selection Rule for the First Route:        Latest Time to Start the Service
Neighborhood Structure:           Relocation 1-0
Improvement Strategy:   First-Improvement
```

Detail of Vehicles Schedule

Tour	Route	Point	Delivery	Load	Arrival Tm	Earliest Tm	Waiting Tm	Start Tm	Latest Tm	Jnloading Trr	Loading Tm	Departure Trr	Travel Tm	Distance
Tour 1	Route 1	Depot	0	50	161.15	0	0.00	161.15	1000	0	50	211.15	18.00	18.00
		C-002	7	43	229.15	93	0.00	229.15	333	7	0	236.15	23.85	23.85
		C-005	26	17	260.00	20	0.00	260.00	260	26	0	286.00	13.93	13.93
		C-008	9	8	299.93	323	23.07	323.00	563	9	0	332.00	12.21	12.21
		C-007	5	3	344.21	251	0.00	344.21	491	5	0	349.21	20.62	20.62
		C-006	3	0	369.82	345	0.00	369.82	585	3	0	372.82	11.18	11.18
		Depot	0	0	384.00	0	0.00	384.00	1000	0	0	384.00	0	0
	Total Load		50	0										
	Route Distance													99.78
Tour 1	Route 2	Depot	0	42	384.00	0	0.00	384.00	1000	0	42	426.00	25.00	25.00
		C-004	19	23	451.00	620	169.00	620.00	860	19	0	639.00	25.00	25.00
		C-003	13	10	664.00	436	0.00	664.00	676	13	0	677.00	14.56	14.56
		C-001	10	0	691.56	658	0.00	691.56	898	10	0	701.56	15.23	15.23
		Depot	0	0	716.79	0	0.00	716.79	1000	0	0	716.79	0	0
	Total Load		42	0										
	Route Distance													79.79
	Tour Distance													179.58

Figure 11. Solution obtained by LS with relocation (1)

Computational Results for Small Test Instances

The results for the MILP model are shown Table 3. Column R' indicates the given number of routes. Columns OFV and CT under MILP represents the objective function value and computation time (in seconds). Symbol asterisk (*) indicates that the solution is optimal. PBLS are run only for REL 1-0 and EXC 1-1, while TBLS are run only for IRR-1-0 and CRS. Results for PBLS and TBLS are also given in Table 3. Gaps of OFV compared to MILP are also displayed. Based on average gaps, it is shown that PBLS (REL 1-0 and EXC 1-1) is more effective than TBLS (IRR 1-0 and CRS). Moreover, REL 1-0 outperforms compared to others (average gap is 10.97%).

Computational Results for Large Test Instances

The computational experiments for large test instances are carried out on 108 test instances. Because each neighborhood is run on different test instances, then the OFVs are different. In order to make fair comparisons, then a measure called the relative improvement (RI) is introduced. It is defined as $RI = |100\,(OFV_{NS} - OFV_{SI})/OFV_{SI}|$ where OFV_{NS} and OFV_{SI} represent the objective function value (OFV) for neighborhood structure and SI, respectively. It is used as the measure of effectiveness.

Table 4 summarizes the computational results for PBLS and TBLS. The values of RI are given in column RI, while the computation times are displayed in column CT. For each neighborhood structure, the results indicate the average values for the instances belonged to their associated group. Columns IS = 25 and IS = 40 denote the results for 25-customer and 40-customer instances. Columns VC = 100 and VC = 500 indicate the results for instances with the vehicle capacities of 100 and 500.

According to Table 4, it is seen that, except for shift (SHI), PBLS gives higher average RI. It means that PBLS, in general, is more effective compared to TBLS. Relocation 1-0 (REL 1-0) is the most effective neighborhood structure where its average RI is 13.32%. However, except for shift (SHI), PBLS has higher computation times. In addition, the computation time for PBLS increases more rapidly compared to TBLS. It is revealed that PBLS is less efficient compared to TBLS.

Conclusions

This paper has discussed the vehicle routing problem with multiple trips and time windows (VRPMTTW). The VRPMTTW discussed includes service-dependent times and allows the time to start the service for each vehicle to be made as latest as possible. A mixed-integer linear programming (MILP) model has been proposed to represent the VRPMTTW.

In this paper, LS with permutation-based neighborhood structure (PBLS) has been proposed. Furthermore, their performance has been compared against LS with tour-based neighborhood structure (TBLS).

Table 3. Comparison results for MILP, PBLS, and TBLS

		MILP		OFV				Gap (%)			
Instance	R'	OFV	CT	REL 1-0	EXC 1-1	IRR 1-0	CRS	REL 1-0	EXC 1-1	IRR 1-0	CRS
R201-005	2	10156.59 *	0.13	10173.67	20156.59	20157.26	20164.62	0.17	98.46	98.46	98.54
R201-006	2	10157.16 *	0.14	10180.19	10180.19	20157.83	20165.19	0.23	0.23	98.46	98.53
R201-007	2	10188.39 *	0.17	20188.39	20200.17	20189.06	20215.22	98.15	98.27	98.16	98.41
R201-008	2	10199.83 *	0.27	10199.83	10199.83	10199.83	10199.83	0.00	0.00	0.00	0.00
R201-009	3	10247.52 *	4.24	10286.89	20247.52	20232.51	20286.89	0.38	97.58	97.44	97.97
R201-010	3	10273.34 *	5.34	20259.00	20273.34	20259.00	20259.00	97.20	97.34	97.20	97.20
R205-005	2	10134.25 *	0.14	10142.26	20153.63	20157.26	20164.62	0.08	98.87	98.90	98.97
R205-006	2	10142.83 *	0.21	10142.83	20165.86	20157.83	20165.19	0.00	98.82	98.74	98.81
R205-007	2	10169.83 *	0.50	10173.47	10187.81	20188.39	20195.83	0.04	0.18	98.51	98.59
R205-008	2	10179.58 *	1.28	10179.58	10193.91	10193.91	10193.91	0.00	0.14	0.14	0.14
R205-009	2	10220.18 *	29.38	10220.18	10220.18	10220.18	10220.18	0.00	0.00	0.00	0.00
R205-010	3	10252.44 *	312.92	10253.08	10265.59	20271.17	20258.43	0.01	0.13	97.72	97.60
R211-005	2	10134.25 *	0.27	10156.35	10142.26	10164.62	10164.62	0.22	0.08	0.30	0.30
R211-006	2	10134.81 *	0.37	10142.83	10142.83	10165.18	10165.18	0.08	0.08	0.30	0.30
R211-007	2	10156.02 *	0.77	10196.44	10166.02	10221.75	10221.75	0.40	0.10	0.65	0.65
R211-008	2	10161.54 *	1.55	10171.54	10179.58	10179.58	10179.58	0.10	0.18	0.18	0.18
R211-009	3	10202.15 *	16.11	10212.15	10220.18	10220.18	10220.18	0.10	0.18	0.18	0.18
R211-010	3	10237.64 *	6823.56	10247.64	10255.68	10274.77	10274.77	0.10	0.18	0.36	0.36
Average		10186.02		11307.02	13530.62	15200.57	15206.39	10.96	32.82	49.21	49.26
Std. dev.		44.36		3244.19	4852.52	5141.13	5147.12	31.55	47.59	50.39	50.45

Table 4. Computational results using large test instances for PBLS and TBLS (average of 108 test instances)

Neighborhood structure	RI (%)					CT (Sec.)				
	VC = 100		VC = 500		All instances	VC = 100		VC = 500		All instances
	IS = 25	IS = 40	IS = 25	IS = 40		IS = 25	IS = 40	IS = 25	IS = 40	
REL 1-0	7.72	6.73	9.47	3.67	6.90	5.39	36.33	5.19	35.90	20.70
REL 2-0	9.73	9.78	9.31	8.57	9.35	3.30	27.75	3.48	25.47	15.00
REL 3-0	22.36	12.82	10.66	7.44	13.32	3.22	18.55	2.59	18.40	10.69
EXC 1-1	9.82	5.84	4.80	4.43	6.22	2.27	18.52	2.02	14.28	9.27
EXC 2-2	12.84	8.09	1.69	4.28	6.73	1.38	8.24	1.06	5.99	4.17
EXC 3-3	13.81	6.41	0.10	1.14	5.37	0.82	4.77	0.53	3.56	2.42
INV	8.68	4.16	1.80	1.28	3.98	2.50	16.00	2.17	12.66	8.33
SHI	0.01	0.80	0.04	0.02	0.22	0.47	1.96	0.48	1.88	1.20
IRR 1-0	0.18	0.27	0.34	1.33	0.53	0.82	2.12	0.93	2.86	1.68
IAR 1-0	0.05	0.05	0.08	0.05	0.06	0.34	0.82	0.37	1.04	0.65
IRE 1-1	0.03	0.03	0.02	0.02	0.02	0.17	0.32	0.13	0.28	0.22
IAE 1-1	0.02	0.02	0.00	0.01	0.01	0.19	0.37	0.11	0.36	0.26
CRS	2.25	0.94	1.67	1.19	1.51	0.34	0.75	0.37	1.04	0.63

Based on the computational experiment results, it is shown that PBLS is more effective than TBLS. However, compared to the optimal solutions, the effectiveness of PBLS is still low. PBLS is less efficient than TBLS. It needs higher computational time than the TBLS. In addition, the computation time for PBLS increases more rapidly as the problem size increases.

In this paper, the permutation-based neighborhood structures are applied for local search (LS). However, LS has a weakness where the solution will be trapped on the local optimal. Therefore, in order to improve their effectiveness, the permutation-based neighbourhood structures can be applied for the metaheuristics such as variable neighborhood search (VNS), simulated annealing (SA), and tabu search (TS). Of course, their reasonable computation time should be maintained in order to be competitive.

References

1. Fleischmann, B. *The Vehicle Routing Problem with Multiple Use of the Vehicles*, Working paper, Fachbereich Wirtschaftswissenschaften. Universität Hamburg, 1990.
2. Brandão, J. and Mercer, A., A Tabu Search Algorithm for the Multi-trip Vehicle Routing and Scheduling Problem, *European Journal of Operational Research*, 100(1), 1997, pp. 180–191.
3. Suprayogi, S., Algoritma Sequential Insertion untuk Memecahkan Vehicle Routing Problem dengan Multiple Trips dan Time Windows, *Jurnal Teknik dan Manajemen Industri*, 23(3), 2003, pp. 30–46.
4. Suprayogi, S., and Imawati, D., Algoritma Sequential Insertion dengan Forward dan Backward Pass untuk Memecahkan Vehicle Routing Problem with Multiple Trips and Time Windows, *Jurnal Teknik dan Manajemen Industri*, 25(1), 2005, pp. 41–54.
5. Suprayogi, S., Hardianto, A. and Yamato, H., Local Search and Genetic Algorithm Techniques for Solving Vehicle Routing Problem with Multiple Trips and Time Windows, in *Proceedings of the 1st International Conference on Operations and Supply Chain Management*, 2005, pp. J17–J24.

6. Suprayogi, S., Imawati, D., and Hardianto, A., Teknik Local Search untuk Pemecahan Vehicle Routing Problem with Multiple Trips and Time Windows, *Jurnal Teknik dan Manajemen Industri*, 27(2), 2007, pp. 57–75.
7. Azi, N., Gendreau, M., and Potvin, J.-Y., An Exact Algorithm for a Vehicle Routing Problem with Time Windows and Multiple Use of Vehicles, *European Journal of Operational Research*, 202(3), May 2010,pp. 756–763.
8. Macedo, R., Alves, C., Valério de Carvalho J. M., Clautiaux, F., and Hanafi, S., Solving the Vehicle Routing Problem with Time Windows and Multiple Routes Exactly using a Pseudo-Polynomial Model, *European Journal of Operational Research*, 214(3), Nov 2011, pp. 536–545.
9. Suprayogi, S., Hidayat, Y. A., and Imawati, D., Simulated Annealing untuk Pemecahan Masalah Rute dan Jadwal Kendaraan dengan Trip Majemuk dan Jendela Waktu, in *Proceedings of the 6th National Industrial Engineering Conference* (NIEC-6), 2011, pp. 242–249.
10. Wang, Z., Liang, W., and Hu, X., A Metaheuristic Based on a Pool of Routes for the Vehicle Routing Problem with Multiple Trips and Time Windows, *Journal of Operational Research Society*, 65(1), Jan 2014, pp. 37–48.
11. Hernandez, F., Feillet, D., Giroudeau, R., and Naud, O., Branch-and-price Algorithms for the Solution of the Multi-trip Vehicle Routing Problem with Time Windows, *European Journal of Operational Research*, 249(2), Mar. 2016, pp. 551–559.
12. Cattaruzza, D., Absi, N., and Feillet, D., The Multi-trip Vehicle Routing Problem with Time Windows and Release Dates, *Transportation Science*, 50(2), May 2016, pp. 676–693.
13. Neira, D. A., Aguayo, M. M., de la Fuente, R., and Klapp, M. A., New Compact Integer Programming Formulations for the Multi-trip Vehicle Routing Problem with Time Windows, *Computers & Industrial Engineering*, 144, January 2019, p. 106399.
14. Suprayogi, S., and Mahaputra, M. S., Pemecahan Masalah Rute Kendaraan dengan Trip Majemuk, Jendela Waktu dan Pengantaran-Penjemputan Simultan Menggunakan Algoritma Genetika, *J@ti Undip Jurnal Teknik Industri*, 12(2), Jul 2017, p. 95.
15. Suprayogi, S., and Priyandari, Y., Tabu Search for the Vehicle Routing Problem with Multiple Trips, Time Windows, and Simultaneous Delivery-Pickup, *Jurnal Teknik Industri*, 19(2), 2017, pp. 75–82.
16. Ong, J. O., and Suprayogi, S., Vehicle Routing Problem with Backhaul, Multiple Trips and Time Window, *Jurnal Teknik Industri*, 13(1), 2011, pp. 1–10.
17. Suprayogi, S., Cahyono, R. T., and Lubis A. T., Multi-trip Vehicle Routing Problem with Backhauls and Time Windows, in *Proceeding of the 9th International Conference on Operations and Supply Chain Management*, 2019.
18. Solomon, M.M, Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints, *Operational Research*, 35(2), 1987, pp. 254–265.