algorithms

MDPI

Article

Branch and Price Algorithm for Multi-Trip Vehicle Routing with a Variable Number of Wagons and Time Windows

Leila Karimi *,† and Chowdhury Nawrin Ferdous †

Department of Mathematics and Computer Science, University of Lethbridge, Lethbridge, AB T1K 3M4, Canada
* Correspondence: l.karimi@uleth.ca
† These authors contributed equally to this work.

Abstract: Motivated by the transportation needs of modern-day retailers, we consider a variant of the vehicle routing problem with time windows in which each truck has a variable capacity. In our model, each vehicle can bring one or more wagons. The clients are visited within specified time windows, and the vehicles can also make multiple trips. We give a mathematical programming formulation for the problem, and a branch and price algorithm is developed to solve the model. In each iteration of branch and price, column generation is used. Different subproblems are created based on the different capacities to find the best column. We use CPLEX to solve the problem computationally and extend Solomon's instances to evaluate our approach. To our knowledge, ours is the first such study in this field.

Keywords: vehicle routing problem; multi-trip; time windows; column generation; branchand price

Citation: Karimi, L.; Nawrin Ferdous, C. Branch and Price Algorithm for Multi-Trip Vehicle Routing with a Variable Number of Wagons and Time Windows. *Algorithms* **2022**, *15*, 412. https://doi.org/10.3390/a15110412

Academic Editor: Roberto Montemanni

Received: 24 August 2022
Accepted: 1 November 2022
Published: 4 November 2022

1. Introduction

The Vehicle Routing Problem (VRP) initially emerged when Dantzig and Ramser formulated and resolved the problem of supplying fuel to service stations around the end of the fifties of the last century [1].

The VRP definition states that n customers with discrete quantities of goods must be served by m vehicles initially located at a depot. A VRP is to determine the optimal routes taken by a group of vehicles while serving a group of users. The objective is to minimize the overall transportation cost. The solution to the classical VRP problem is a set of routes visiting all the customers exactly once that all begin and end in the depot. The transportation cost is improved by reducing the total traveled distance [2].

The Multi-Trip Vehicle Routing Problem with Time Windows ($MTVRPTW$) is a type of the classical Vehicle Routing Problem with Time Windows ($VRPTW$) with more than one trip for a vehicle during a workday. A trip is a timed route when more than one route can be allocated to a vehicle. The multi-trip feature is needed when the vehicle fleet size is limited. In this case, a benefit is a reduced number of drivers and vehicles. Besides, in practice, industries cannot provide an unlimited number of vehicles to serve all customers, and they tend to prefer a limited number of vehicles to do more than one trip. Despite its apparent practical relevance, this variant of the classical VRP has not been the subject of a large number of studies. Refs. [3–5] are a few papers worked on Multi trip vehicle routing problem ($MTVRP$).

Multi-trip vehicle routing problem with a variable number of wagons and time windows defines a variant of the classical vehicle problem in which the capacity of vehicles can be determined given the total demand of the route when a vehicle is prepared to leave the depot. In this situation, one, two, or three wagons can be attached to make a vehicle ready to service the customers. The number of wagons and vehicles is limited, and the vehicle configuration will stay the same during all vehicle trips. This new methodology is suitable to decrease time and cost by reducing the number of vehicles, drivers, and fuel

consumption, which is specifically more critical in distributing goods over large distances like two different cities or from large cities to rural areas.

The main contribution of this work is to introduce a mathematical model for Multi-trip Vehicle Routing Problem with a Variable number of Wagons and Time Windows (MTVRP-VW-TW). We develop a Branch and Price method to find an optimal solution. Column generation is used for each iteration of the branch and pricing. Various subproblems are formed to choose the appropriate column based on the various capacities. We used modified Solomon's instances [6] to test the algorithm. It is the first time the model has been presented and solved with exact methods.

The rest of the paper is organized as follows. We review the relevant research in Section 2. We give the definition and mathematical model for MTVRP-VW-TW in Section 3. Column generation for MTVRP-VW-TW is presented in Section 4. MTVRP-VW-TW is solved using the branch and price algorithm in Section 5. The detailed experimental study is discussed in Section 6. Section 7 concludes this paper with a discussion of the limitations of this work and possible future extensions.

2. Literature Review

Multi-trip vehicle routing problem ($MTVRP$) is an essential type of vehicle routing problem in the real world that is studied less than other versions of vehicle routing problem, specifically, for exact methods. Ref. [7] is a survey that categorizes and examines urban logistic flows. As a result, it outlines the three main scientific issues that must be resolved: time dependency, the arrangement of the distribution on multiple levels and trips, and dynamic information. Fleischmann [8] proposed a modification of the savings heuristic and used a Bin Packing Problem heuristic to assign the routes to vehicles with multiple uses. Taillard, Laporte, and Gendreau [9] presented a tabu search algorithm with three phases to solve the problem. Brando and Mercer [10] proposed another tabu search algorithm with a variable neighborhood to find a solution with the least cost. The algorithm is a three-phase algorithm that creates an initial solution by a heuristic and then uses tabu search (reinsertion and exchange of customers) to improve the solution and restore feasibility. Brandão and Mercer [11] also presented a more complex problem when mixed fleets and maximum overtime constraints are allowed. Salhi [12] proposed the many-to-many location-routing problem. Campbell and Savelsbergh [13] described insertion heuristics that can be used effectively when time windows constraints are added to the problem.

Petch and Salhi [5] developed a multi-phase constructive heuristic for the $MTVRP$, which in phase one generates a VRP solution using a savings approach, and phase 3 generates a VRP solution by route population approach. Phase 2 is a $VRPM$ construction and improvement stage in which an $MTVRP$ solution is constructed using bin-packing with the minimization of overtime as the objective. Salhi and Petch [14] improved their previous method to a hybrid Genetic Algorithm with the same objective. Olivera and Viera [15] presented an adaptive memory approach to minimize total routing cost. Cattaruzza et al. [16] used a hybrid genetic algorithm with a new local search operator that is a combination of standard VRP moves and swaps between trips to minimize total traveling time. Wassan et al. [17] proposed a two-level variable Neighborhood Search to generate an $MT\text{-}VRPB$ initial solution to minimize the total cost. Tirkolaee et al. [18] formulated a new model for a robust multi-trip vehicle routing problem with intermediate depots and time windows to address the uncertain nature of the demand. Anggodo et al. [19] presented a genetic algorithm for multi-trip vehicle routing problems with time windows. More heuristic approaches are in [20–23], in which a hybrid genetic algorithm, a simulated annealing, a hybrid particle swarm optimization algorithm, and a hybrid genetic algorithm are used respectively. A new impact integer programming formulation for the multi-trip vehicle routing problem with time windows is developed in [24].

A limited number of papers on the exact methods for $MTVRP$ exist. Desrosiers and Solomon [25] were the first to use column generation in a Dantzig-Wolfe decomposition framework. Halse [26] implemented Lagrangean decomposition. After that, Kohl and

Madsen [27] extended Lagrangean relaxation. These approaches were further developed using Dantzig-Wolfe decomposition, including cutting planes or parallel platforms in Kohl, Desrosiers, Madsen, Solomon, and Soumis [28]; Larsen [29]; Cook and Rich [30]. A hybrid algorithm, a combination of Lagrangean relaxation and Dantzig-Wolfe decomposition, was presented by Kallehauge [31]. Chabrier, Danna and Le Pape [32]; Feillet, Dejax, Gendreau and Gueguen [33]; Rousseau, Gendreau and Pesant [34]; Larsen [29]; Chabrier [35]; Irnich and Villeneuve [36]; Danna and Le Pape (2005) [37] presented algorithms based on the subproblem methods. Hernandez et al. [3], and Nabila Azi et al. [38] suggested the branch and price algorithm with two phases. In the first phase, all paths are generated. In the second phase, the problem is solved by column generation. Macedo et al. [39] proposed an approach using a pseudo-polynomial model. Munari and Morabito [40] presented a branch-price-cut for a multi-trip vehicle routing problem; Faiz et al. [41] has two integer programs for the open vehicle routing problem and uses column generation to solve them; Azi et al. [4] gave column generation embedded in branch and price algorithm to solve multi-trip vehicle routing problem with time windows using dynamic programming to generate all non-dominated paths by label correcting algorithm which are used in the subproblem; Seixas and Mendes [42] presented a branch and price algorithm to solve the multi-trip vehicle routing problem with time windows and driver work hours. Bettinelli et al. [43] used a branch-and-cut-and-price algorithm to solve the multi-trip separate pickup and delivery problem with time windows. A branch-cut-and-price algorithm is developed in [44] for the single and multi-trip two-echelon vehicle routing problem with time windows. A new variant of the multi-trip vehicle routing problem for the case of being in a queue while the unloading capacity is full is presented in [45] which is solved using a branch-and-price-and-cut algorithm.

We are unaware of any work on exact methods for the multi-trip vehicle routing problem with time windows and the flexibility of having different wagons attached to service customers, as in this paper. In contrast to the previous works like [3,4,38] which give branch and price algorithms for $MTVRP$, we formulate a new variant of $MTVRP$ in which the capacity can be different. This requires a modification of the branch and price algorithms. We have a different master problem compared to the previous work. There are three sub-problems with three different objective functions, and each subproblem is solved by constructing a new route graph based on the capacity of the vehicle in which we look for all the non-dominated tours.

3. Mathematical Model for Multi-Trip Vehicle Routing Problem with a Variable Number of Wagons and Time Windows

An instance of this problem is defined by a set of customers $C = \{1, 2, \cdots, n\}$, and the depot is represented by the vertices 0 and $n + 1$. Depot 0 is the start depot, and $n + 1$ is the return depot.

The set $\{0, 1, \ldots, n + 1\}$ is denoted N in a complete directed graph $G = (N, A)$, where A is a set of arcs $\{(i, j) : i \neq j, i, j \in N\}$. A traveling time of t_{ij} is associated with each arc $(i, j) \in A$, which we consider as the distance of two vertices i and j, where $i, j \in N$. A fleet of wagons W, with identical capacities q which can be connected as one wagon, two wagons or three wagons to organize a set of vehicles V. Vehicles in V are used to serve the customers. $|V|$ and $|W|$ are the number of vehicles and wagons, respectively. The set of arcs A represents all the connections between customers and the depot. There are no arcs ending at vertex 0 or originating from vertex $n + 1$. A traveling time of t_{ij} is associated with arc (i, j) , where $i \neq j$. Any customer $i \in C$ has a demand d_i, a service time s_i, and a time window $[a_i, b_i]$, which means that a vehicle must arrive at the customer before b_i. If it arrives before the time window opens, it has to wait until a_i to service the customer. The time windows for both depots are assumed to be $[a_0, b_0]$, representing the scheduling horizon. The vehicles may not leave the depot before a_0 and must return at the latest time b_0. A route of a vehicle is a closed path that starts and ends at the depot. The vehicle starts at the depot, visits several customers in a specific order, and returns to the depot again.

At every point on the route, the time windows and capacity constraints are satisfied. The workday of each vehicle is a sequence of routes where each route starts and ends at the depot and we call it a tour. Multiple routes can be performed by a vehicle during one day, and these routes are collectively called a tour. These routes are denoted by the set R and the maximum number of routes in any tour (for any vehicle) is fixed in our model.

Each vehicle can be configured to use one, two, or three wagons giving it different capacities. The configuration of each vehicle must stay the same on all trips. Next, we define some of the mathematical model's decision variables. The decision variable s_{ir}^k denotes the time that the vehicle k starts to service customer i in route r. If the vehicle k does not service customer i in route r, s_{ir}^k has no meaning; consequently, its value is considered irrelevant. Variable x_{ijr}^k is one if vehicle k drives directly from customer i to customer j and zero otherwise in route r. Variable z_n^k is used to determine how many wagons vehicle k needs, where n is in $\{1, 2, 3\}$. If $z_2^3 = 1$, vehicle 3 has 2 wagons. Therefore, z_1^3 and z_3^3 must be 0, which means vehicle 3 does not have 1 or 3 wagons. Moreover, finally, q^k is the k^{th} vehicle's capacity, depending on the number of wagons attached.

$$x_{ijr}^k = \begin{cases} 1 & \text{if vehicle } k \text{ drives directly from vertex } i \text{ to vertex } j \text{ on route } r \\ 0 & \text{otherwise} \end{cases}$$

$$z_m^k = \begin{cases} 1 & \text{if the } m \text{ wagons are attached for vehicle } k \\ 0 & \text{otherwise} \end{cases}$$

We assume $a_0 = 0$ and therefore $s_{0r}^k = 0$, for all k and r. The goal is to design a set of routes that minimizes the total distances of all routes and

- Each customer is serviced exactly once;
- Every route starts at vertex 0 and ends at vertex $n + 1$;
- The time windows of the customers are satisfied;
- The total demand on a route can not exceed the capacity of the vehicle, which depends on the number of wagons attached to it (1, 2, or 3);
- Total number of the wagons used should be less than $|W|$;
- A vehicle is assigned only one configuration;
- Each vehicle must leave the depot 0 ;
- All vehicles must return to the depot $n + 1$;
- The start time of the next route by the same vehicle should be after the finishing time of its previous route;

An unused vehicle is modeled by driving the empty route $(0, n + 1)$.
The mathematical model is described next.

$$\min \sum_{k \in V} \sum_{r \in R} \sum_{i \in N} \sum_{j \in N} d_{ijr}^k x_{ijr}^k \tag{1}$$

s.t.

$$\sum_{k \in V} \sum_{r \in R} \sum_{j \in N} x_{ijr}^k = 1 \qquad \forall i \in C \tag{2}$$

$$\sum_{i \in C} d_i \Big(\sum_{j \in N} x_{ijr}^k \Big) \leq q^k \qquad \forall k \in V,\ \forall r \in R \tag{3}$$

$$q^k = \sum_{m=1}^{3} m q z_m^k \qquad \forall k \in V \tag{4}$$

$$\sum_{m=1}^{3} \sum_{k \in V} m z_m^k \leq |W| \tag{5}$$

$$\sum_{m=1}^{3} z_m^k = 1 \qquad \forall k \in V \tag{6}$$

$$\sum_{j \in N \setminus \{0\}} x_{0jr}^k = 1 \qquad \forall k \in V,\ \forall r \in R \tag{7}$$

$$\sum_{i \in N} x_{ihr}^k - \sum_{j \in N} x_{hjr}^k = 0 \qquad \forall h \in C,\ \forall k \in V,\ \forall r \in R \tag{8}$$

$$\sum_{i \in N \setminus \{n+1\}} x_{i,n+1,r}^k = 1 \qquad \forall k \in V,\ \forall r \in R \tag{9}$$

$$s_{ir}^k + s_i + t_{ij} - M(1 - x_{ijr}^k) \leq s_{jr}^k \quad \forall i \in N \setminus \{n+1\},\ \forall j \in N \setminus \{0\},\ \forall k \in V,\ \forall r \in R \tag{10}$$

$$a_i \sum_{j \in N - \{0\}} x_{ijr}^k \leq s_{ir}^k \leq b_i \sum_{j \in N - \{0\}} x_{ijr}^k \qquad \forall i \in C,\ \forall k \in V,\ \forall r \in R \tag{11}$$

$$s_{0r}^k \geq s_{n+1,r-1}^k \qquad \forall r \in R,\ \forall k \in V \tag{12}$$

$$x_{ijr}^k \in \{0,1\} \qquad \forall i,j \in N,\ \forall k \in V,\ \forall r \in R \tag{13}$$

$$z_m^k \in \{0,1\} \qquad \forall k \in V,\ m \in \{1,2,3\} \tag{14}$$

$$s_{ir}^k \geq 0 \qquad \forall i \in C,\ \forall k \in V,\ \forall r \in R \tag{15}$$

$$q^k \geq 0 \qquad \forall k \in V \tag{16}$$

The objective function (1) minimizes the total distances of tours. The constraints (2) ensure that each customer is visited exactly once. Equations (3) and (4) state that the total demand on a route can not exceed the capacity of each vehicle depending on the number of wagons attached. The constraint in (5) shows the number of wagons in total. Constraints (6) ensure that a vehicle is assigned only one configuration. Equations (7)–(9) indicate that each vehicle must leave the depot 0; flow conservation constraints; finally, all vehicles must return to the depot $n+1$. The inequalities (10) establish the relationship between the vehicle departure time from a customer and its immediate successor. Constraints (11) assert that the time windows are observed. Constraints (12) ensure a proper trip sequencing for the workday of a vehicle that the starting time of the next trip of the vehicle must be after the finishing time of its previous trip. Equations (13) and (14) are integer variables.

4. Column Generation for *MTVRP-VW-TW*

Multi-trip vehicle routing problem with a variable number of wagons and time windows and the mathematical model was described in Section 3. In this section, we solve the problem using column generation. The master problem, pricing subproblem, and the techniques for solving the pricing subproblem will be explained.

4.1. Master Problem for MTVRP-VW-TW

A tour is the set of all the routes a vehicle performs during a day. So, the decision variable y_w, which refers to a column corresponding to a tour. Ω is the set of all feasible tours, d_w is the total distance of tour $w \in \Omega$, $|W|$ is the total number of wagons, $|V|$ is the number of the vehicles and n_w is the number of wagons used for a vehicle that is used in tour w and $n_w \in \{1,2,3\}$. a_{iw} is one if customer i is in tour w, 0 otherwise. We use the following decision variable:

$$y_w = \begin{cases} 1, & \text{if tour } w \text{ is chosen} \\ 0, & \text{otherwise} \end{cases} \tag{17}$$

The master problem is:

$$\min \sum_{w \in \Omega} d_w y_w \tag{18}$$

s.t.

$$\sum_{w \in \Omega} a_{iw} y_w \geq 1 \quad \forall i \in C \tag{19}$$

$$\sum_{w \in \Omega} y_w \leq |V| \tag{20}$$

$$\sum_{w \in \Omega} n_w y_w \leq |W| \tag{21}$$

$$y_w \in \{0,1\} \qquad \forall w \in \Omega \tag{22}$$

The objective function (18) is to minimize the total distances of all tours. Constraints (19) ensure that each customer is visited at least once. The constraint (20) states that the number of all tours must be less than the number of vehicles. Constraint (21) states that the number of wagons used in all the vehicles must be less than the total number of wagons.

The solution is a subset of Ω. As the number of columns is exponential in the number of customers, we solve the restricted master problem (RMP) with a limited number of columns for the initial solution. The columns are progressively added into RMP. The LP-relaxation of RMP ($RLMP$) is solved with an LP solver to obtain the dual variables associated with the optimal solution of the $RLMP$. These dual values are sent to the subproblem to determine new tours with a negatively reduced cost, and these new tours are added to the master problem. The process will continue until there are no more tours with a negatively reduced cost. This guarantees an optimal solution to the $RLMP$.

4.2. Method for Constructing Tours/Columns

Once the LP relaxation of the restricted master problem is solved, Three sub-problems are defined below based on the definition of the $MTVRP$-VW-TW. The first sub-problem is to find tours of vehicles with one wagon attached; the second and third sub-problems are to find tours of vehicles with two and three wagons attached, respectively. Each sub-problem is an elementary shortest-path problem with resource constraints. The path starts at the artificial start node (depot) and to the artificial end node (depot) in the route graph. The route graph and its construction are explained in the following sections. Solving each sub-problem gives a new tour with the most negatively reduced cost. Following are the steps to solve the sub-problems.

4.3. Generating All Non-Dominated Paths

All non-dominated routes must first be generated to solve the sub-problems. The label correcting algorithm [33] is used to create all these routes. To keep track of previously visited nodes, elementary paths must be generated. A path p from an origin node $o \in N$ to a node $j \in N$ is labelled with $R_p = (C_p, t_p^1, t_p^2, s_{pv}, V_p^1, \ldots, V_p^n)$. Time consumption, t_p^1 is the time used in the path till customer v_j which sets to 0 at the depot, and after extending a path by visiting a new customer, it is updated as $t_p^1 = t_p^1 + t_{ij} + s_j$, where v_i and v_j are two adjacent customers on the route and s_j represents the service time for the customer v_j. Load consumption, t_p^2 is the capacity used for the path and sets to 0 at the beginning when the vehicle is at the depot, then when a customer is added to the path, it is updated by $t_p^2 = t_p^2 + d_j$ where d_j is the quantity that must be delivered to the customer v_j. A time interval $[a_i, b_i]$ representing the time window is associated with each customer v_i and a load interval $[0, Q]$, where Q is the vehicle's capacity. C_p is the length of the path and is made negative by replacing the distance t_{ij} of each arc with $t_{ij} - a$, such that $a > max_{(i,j) \in A} t_{ij}$. We make the distances negative so that vehicles leave the depot. Otherwise, it would be optimal to stay at the depot; s_{pv} is the number of unreachable nodes and $V_p^i = 1$ if

node i is unreachable, 0 otherwise. The following dominance relation is used to determine non-dominated routes:

Dominance Relation: If two paths, p, and p', extend from origin o to node j with labels R_p and $R_{p'}$, respectively, then path p dominates p' if and only if $C_p \leq C_{p'}, s_{pv} \leq s_{p'v}$, $t_p^k \leq t_{p'}^k, \forall k = 1, \ldots, l, V_p^i \leq V_{p'}^i, \forall i = 1, \ldots, n$ [38].

In other words, a path p dominates a path p' if (a) it is no longer, (b) it does not use more resources for each resource taken into account, and (c) every node that is unreachable for path p is also unreachable for path p' [38].

Using this relation will keep only the labels for non-dominated elementary paths.

To implement the label correcting algorithm for our problem, we need to create a label $(C_p, t_p^1, t_p^2, s_{pv}, V_p^1, \ldots, V_p^n)$ which represents a path p from the depot to the customer j. All feasible non-dominated routes are generated using these labels.

During the path extension, we need to see if the current time consumption (t_p^1) plus the distance d_{ij} is less than a_i, then t_p^1 is replaced by a_i, and the extension of the path continues. Each time we extend one node or the node is unreachable, s_{pv} increases by one. We also eliminate the partial routes when we are extending the paths. So, at the end of the algorithm, we will have all non-dominated routes.

4.4. Creation of the Route Graph

After creating all non-dominated routes, each can be looked at as a node in a new graph called the route graph. The route graph includes these routes as nodes and two artificial nodes for the start and end of the vehicle workday. To create the route graph, if there is an edge between nodes r and r', route r and r' must not visit the same customer, and the feasibility of servicing route r' after route r is determined through departure time windows as explained below.

The latest departure and arrival times and the earliest departure and arrival times need to be calculated to satisfy the second condition. To have an edge (r, r'), the latest departure of route r' must be larger than the latest arrival of route r. There are edges between the artificial start node and all routes and from all routes to the artificial end node.

There are two time windows for each route node r which are earliest and latest departure times $[\underline{t}_0^r, \bar{t}_0^r]$ and earliest and latest arrival times $[\underline{t}_{n+1}^r, \bar{t}_{n+1}^r]$. Routes must be started and completed in these intervals. These time windows are determined as shown below [38].

Latest departure and arrival times:

If the route r is shown as a sequence $(0 = i_0, i_1, i_2, \ldots, i_{n_r}, i_{n_r+1} = n+1)$ where the first and last points are the depot as well as other customers (n_r ones) in the middle, first the latest feasible time $\bar{t}_{i_j}^r$ of each customer must be calculated using a back-ward sweep of route r starting from i_{n_r+1} to i_0. Therefore:

$$\bar{t}_{i_{n_r+1}} \leftarrow b_{i_{n_r+1}}$$

$$\bar{t}_{i_j}^r \leftarrow \min\{\bar{t}_{i_{j+1}}^r - t_{i_j i_{j+1}} - s_{i_j}, b_{i_j}\}, \quad \forall j = i_{n_r}, \ldots, i_0$$

Finally, we will have $\bar{t}_{i_0}^r$ which is the latest departure of the route r and again in a similar way we obtain the latest arrival time of the route as well as the latest feasible schedules ($\bar{t}_{i_j}^r$) at each customer using a forward sweep, so each $\bar{t}_{i_j}^r$ can be calculated as:

$$\bar{t}_{i_j}^r \leftarrow \max\{\bar{t}_{i_{j-1}}^r + t_{i_{j-1} i_j} + s_{i_j}, a_{i_j}\}, \quad \forall j = i_1, \ldots, i_{n_r+1}$$

Earliest departure and arrival times:

Suppose we calculated the latest departure time ($\bar{t}_0^r$), the earliest departure time($\bar{t}_{n+1}^r$) and the latest feasible schedules to begin service at each customer($\bar{t}_{i_j}^r$) in the route r, two cases can happen:

Case 1: there is no waiting time (vehicle doesn't arrive before time windows) in the latest feasible time of customers, then we can shift the latest times by the minimum of $(\bar{t}^r_{i_j})$ and a_{i_j}, so we can calculate it for each route as:

$$\delta^r = min_{j=0,\ldots,n_r+1}(\bar{t}^r_{i_j} - a_{i_j})$$

By deducting these units from the latest departure and arrival times, the earliest departure and arrival can be obtained. So, we have:

$$\underline{t}^r_0 = \bar{t}^r_0 - \delta^r$$

and,

$$\underline{t}^r_{n+1} = \bar{t}^r_{n+1} - \delta^r$$

Having all the latest departure and arrival times as well as the earliest departure and arrival times, it is possible to write the time windows for all routes as: $[\underline{t}^r_0, \bar{t}^r_0]$ and $[\underline{t}^r_{n+1}, \bar{t}^r_{n+1}]$.

Case 2: If there are some waiting times in the latest feasible time for customers. Then, we can not leave the depot earlier, when the latest arrival times are before the time windows.

So, the earliest departure and arrival times will be the same as the latest departure and arrival times respectively.

$$\underline{t}^r_0 = \bar{t}^r_0$$

and,

$$\underline{t}^r_{n+1} = \bar{t}^r_{n+1}$$

In case 2, the time windows for departure and arrival will be a single point. So, the route r' can be served after route r if $\underline{t}^r_{n+1} + \delta^{r'} \leq \bar{t}^{r'}_0$.

Given all this information, now to create the route graph, we must note that there must not be any common customer between routes r and r' and the latest arrival time of the route r must be less than the latest departure time of the next route r'. If both conditions are met, then there is an edge from r to r'.

4.5. Sub-Problem for MTVRP-VW-TW

The subproblem is defined on the route graph $G^T = (V^T, A^T)$ where V^T is the set of all non-dominated routes generated by the label correcting algorithm [33] plus two artificial nodes for the start and end of the tour. A^T is the set of edges in the route graph with the time windows on each route, $[\underline{t}^r_0, \bar{t}^r_0]$ and $[\underline{t}^r_{n+1}, \bar{t}^r_{n+1}]$.

Dual variables associated with the master problem constraints are needed to formulate the subproblem on the route graph. Let π_i be dual the variables associated with constraints (19) in the master problem and μ_0 and μ_1 are the dual variables associated with (20) and (21) constraints respectively.

Let $c_{rs} = d_s$, where d_s is the total distance of route s, and the reduced cost of arc (r, s) is: $\bar{c}_{rs} = c_{rs} - \sum_{i \in V_s} \pi_i$.

Using the binary variable X_{rs} which is one if the route (r, s) is used and zero otherwise, and the continuous variable T_r which is the departure time of the route r, we formulate the subproblem as follows:

$$\min \sum_{(r,s)\in A^T} \bar{c}_{rs} X_{rs} - \mu_0 - \mu_1 \tag{23}$$

s.t.

$$\sum_{(r,h)\in A^T} X_{rh} - \sum_{(h,s)\in A^T} X_{hs} = 0 \quad \forall h \in V^T \tag{24}$$

$$\sum_{r\in A^T} X_{0r} = 1 \tag{25}$$

$$\sum_{r\in A^T} X_{r,n+1} = 1 \tag{26}$$

$$T_r + (\bar{t}^r_{n+1} - \bar{t}^r_0) - M(1 - X_{rs}) \leq T_s \quad \forall (r,s) \in A^T \tag{27}$$

$$\underline{t}^r_0 \leq T_r \leq \bar{t}^r_0 \quad \forall r \in V^T \tag{28}$$

$$X_{rs} \in \{0,1\} \quad \forall (r,s) \in A^T \tag{29}$$

$$T_r \geq 0 \quad \forall (r,s) \in A^T \tag{30}$$

The objective function (23) is to reduce the cost of the tour. Constraint (24) indicates that the vehicle must leave a route and go to the next one. Constraints (25) and (26) ensure that the tour starts and ends at the depot. The inequalities (27) establish the relationship between the vehicle departure time from a route and its immediate successor. Constraints (28) assert that the time windows of routes are observed.

The following flowchart shows the process of generating all non-dominated tours for the sub-problem with one wagon in Figure 1. The same procedure is used for two other sub-problems.

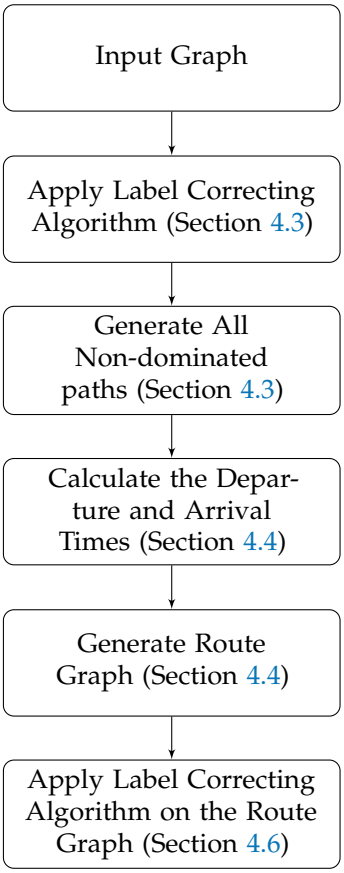

Figure 1. Process of generating all non-dominated tours.

4.6. Solving the Pricing Subproblems

For three different capacities of the vehicles, we have three route graphs. Consequently, three subproblems will be solved. The label correcting algorithm can be applied to each of them again to find the new tour with the negative reduced cost to be added to the master problem.

Now, the label correcting algorithm can be implemented to determine all non-dominated tours of the route graph. The method described in [4] is used to implement the label-correcting algorithm on the route graph.

Using the extend and dominate function in the label correcting algorithm, all non-dominated tours will be generated. The algorithm first generates all non-dominated routes and then generate the route graph, and finally creates all non-dominated tours. The algorithm is described next.

Description of the Algorithm:

The Algorithm 1 finds all non-dominated tours on the route graph from the origin node p (depot).

We need the following notation to describe the algorithm:

- $G = (N, A)$: The input graph.
- N: Set of customers and vertices 0 and $n + 1$ as the depot.
- A: Set of all edges between vertices in N
- H_i: List of labels on node v_i
- $Succ(v_i)$: Set of successors of node v_i.
- E: List of nodes waiting to be processed.
- $Extend(L_i, v_j)$: Function that returns the label resulting from the extension of label $L_i \in H_i$ towards node v_j when the extension is possible, nothing otherwise.
- $Dominated(A_j)$: Procedure that removes dominated labels in the list of labels H_j.
- F_{ij}: List of labels extended from v_i to v_j
- $Routes$: To save all non-dominated routes.
- $(Ld - time)_k$: latest departure time of $R_k \in Routes$.
- $(La - time)_k$: latest arrival time of $R_k \in Routes$.
- $(Ed - time)_k$: earliest departure time of $R_k \in Routes$.
- $(Ea - time)_k$: earliest arrival time of $R_k \in Routes$.
- $G^T = (V^T, A^T)$: Route graph
- V^T: Set of all non-dominated routes which are vertices in the route graph.
- A^T: Set of edges in the route graph
- H_k^T: List of labels on node R_k
- $SuccT(R_k)$: Set of successors of route R_k.
- E^T: List of routes waiting to be processed.
- $ExtendT(L_k^T, R_k)$: Function that returns the label resulting from the extension of label $L_k^T \in H_i^T$ towards node R_h when the extension is possible, nothing otherwise.
- $DominatedT(H_h^T)$: Procedure that removes dominated labels in the list of labels H_h^T.
- FT_{kh}: List of labels extended from R_k to R_h

Algorithm 1 Generating all non-dominated tours

Input: $G(N, A)$ {All notation used are written above}
output: all non-dominated tours
Initialization {Generate all non-dominated routes}
$H_p \leftarrow \{(0, ..., 0)\}$
for all $v_i \in V - \{p\}$ **do**
  $F_{ij} \leftarrow \emptyset$
end for
$E = \{p\}$
while $E! = \emptyset$ **do**
  Choose $v_i \in E$
  for all $v_j \in Succ(v_i)$ **do**
    $F_{ij} \leftarrow \emptyset$
    for all $L_i \in H_i$ **do**
      if $V_i^j = 0$ **then**
        $F_{ij} \leftarrow F_{ij} \cup Extend(L_i, V_j)$
      end if
    end for
    $H_j \leftarrow Dominated(F_{ij} \cup H_j)$
    if H_j has changed **then**
      $E \leftarrow E \cup \{v_j\}$
    end if
  end for
  for all $L_i \in H_i$ **do**
    if L_i is not extended to any $v_j \in Succ(v_i)$ **then**
      $Routes \leftarrow Extend(L_i, n + 1)$
    end if
  end for
  $E \leftarrow E - \{v_i\}$
end while
for all $R_k \in Routes$ **do** {Generate the route graph}
  Add R_k to V^T
  Calculate $(Ld - time)_k$, $(La - time)_k$,$(Ed - time)_k$, and$(Ea - time)_k$.
end for
for all $R_k, R_h \in Routes$ **do**
  if R_k and R_h don't have common customer and $(La - time)_k \leq (Ld - time)_h$ **then**
    Add an edge from vertex R_k to vertex R_h in A^T
  end if
end for
for all $R_k \in Routes$ **do**
  Add an edge from vertex R_k to p and from p to R_k
end for

Algorithm 1 *Cont.*

Initialization {Generate all non-dominated tours}
$H_k^T \leftarrow \{(0, ..., 0)\}$
for all $R_k \in V^T - \{p\}$ **do**
  $FT_{kh} \leftarrow \emptyset$
end for
$E^T = \{p\}$
while $E^T != \emptyset$ **do**
  Choose $R_k \in E^T$
  for all $R_h \in SuccT(R_k)$ **do**
    $FT_{kh} \leftarrow \emptyset$
    for all $L_k^T \in H_k^T$ **do**
      if $R_k^h = 0$ **then**
        $FT_{kh} \leftarrow FT_{kh} \cup ExtendT(L_k^T, R_h)$
      end if
    end for
    $H_h^T \leftarrow DominatedT(FT_{kh} \cup H_h^T)$
    if H_h^T has changed **then**
      $E^T \leftarrow E^T \cup \{R_h\}$
    end if
  end for
  $E^T \leftarrow E^T - \{R_k\}$
end while

All non-dominated tours will be generated using the extend and dominate function in the label-correcting algorithm. There are three subproblems based on the various capacity. All tours will be generated for these three subproblems. First, we will see if there is a new tour with a negative reduced cost in the subproblem with one wagon. If so, the column will be added to the master problem. If not, the second subproblem will be checked. Suppose a new tour with a negative reduced cost is added to the master problem. If not, we will check the third subproblem, which uses three wagons. We solve the subproblems until all tours with the negative reduced cost are found and added to the master problem. The following flowchart shows the procedure of the algorithm in Figure 2.

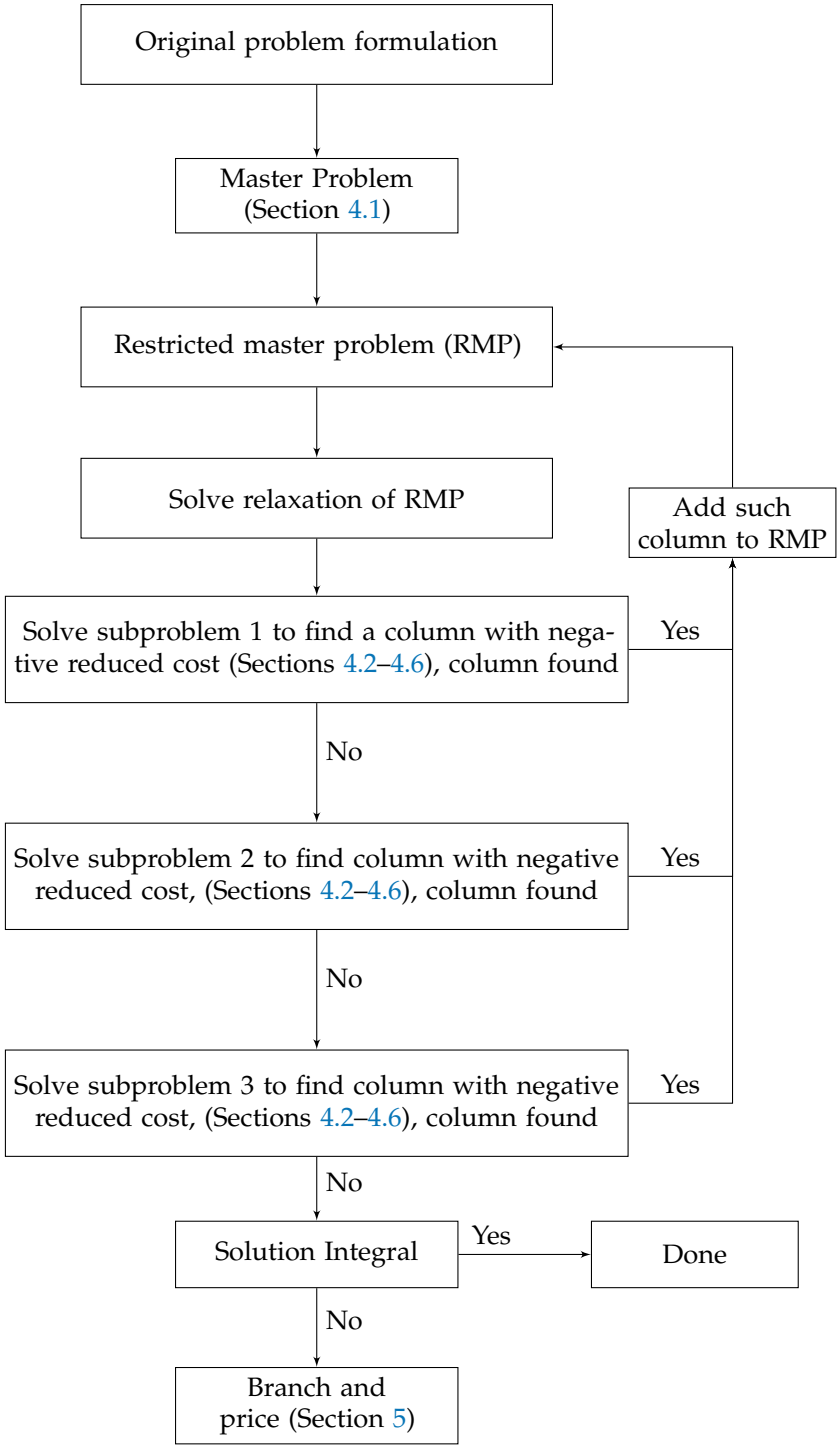

Figure 2. Column generation procedure for various capacities.

5. Branch and Price Algorithm for *MTVRP-VW-TW*

In this section, we describe initialization, the search strategy, the branching strategy, and the upper bound that we use in our implementation of branch and price for the multi-trip VRP with variable wagons and time windows.

Initialization: At the root of the search tree, the *RLMP* is initialized with tours made of a single customer visit. The number of columns thus corresponds to the number of customers. For internal nodes in the search tree, the algorithm initializes the *RLMP* with the set of columns in the parent node considered after removing infeasible columns due

to branching [4]. The minimum capacity for all vehicles that service a single customer is one wagon.

Search Strategy: The branch and price tree is explored using a depth-first search.

Lower and Upper Bound: The solution to $RLMP$ at the root node gives a lower bound for the problem. We solve the master problem using CPLEX. The integer solution of the master problem given by the CPLEX 12.8 is used as the upper bound.

Branching Strategy: Two branching strategies are used. We branch on the number of vehicles and the arcs.

Branching on the Number of Vehicles: We sum the value of variables of the optimal solution of $RLMP$, so $k = \sum_{w \in \Omega'} y_w$, where $\Omega^{'} \subseteq \Omega$. If k is fractional, two branches are created. For each branch, one additional constraint is added to the master problem. These two constraints are:

$$\sum_{w \in \Omega^{'}} y_w \leq \lfloor k \rfloor$$

and

$$\sum_{w \in \Omega^{'}} y_w \geq \lfloor k+1 \rfloor.$$

The dual variable value corresponding to the new constraint is added to the subproblem, and the column generation is done again for this new child node.

Branching on Arcs: The branch on an arc happens when the flow on an arc (i, j) is fractional. We calculate the flow on any arc that is in some column. The sum of the y_w, $w \in \Omega^{'}$ on the columns that include an arc (i, j), will give the flow on the arc. The arc with fractional value is taken. So, these branches will be:

- Left branch: $x_{ij} = 1$, which means the customer j must be visited right after customer i in all tours of $RLMP$ and the route graph. To enforce it, all columns in the $RLMP$ and the route graph that contains arc (i, k) with $k \neq j$ and (k, j) with $k \neq i$ must be deleted. Also, if $x_i = \sum_{w \in \Omega'} a_{iw} y_w$, then the decision variables for vertices i and j are set to one in $RLMP$, $x_i = 1$ and $x_j = 1$.
- Right branch: $x_{ij} = 0$, which means the customer j must not follow the customer i immediately. So all tours must be removed, including the arc (i, j) in $RLMP$ and the route graph.

Branch and Price Process

We start at the root, and if $RLMP$ is feasible, all possible columns from three subproblems will be added to the $RLMP$. The LP solution of the root is set as the lower bound, and also integer solution given by the CPLEX is used as the upper bound. Two branches on the number of vehicles are created, and we use a stack in implementation DFS, so nodes are added to the front of the stack. For each node, column generation is used again to find the LP- solution of $RLMP$. We check if the node must be pruned or kept. If the node is not pruned, we update the upper bound and create two new branches. After processing all nodes, we see if the sum of the value of variables of the best bound is an integer, but variable values are not, then the branching on nodes is used to generate nodes. Then, we calculate the flows on all the arcs. In the same way, we continue with the last node added to the head of the list to see if the node must be pruned or kept after using column generation. If we keep the node, two branches are created, and the upper bound is updated. The process continues until the stack is empty.

6. Experimental Evaluation for *MTVRP-VW-TW*

6.1. Mathematical Model Test

The mathematical model of MTVRP-VW-TW with the objective function of the duration of the longest route is solved with CPLEX and C++, using modified Solomon instances [46]. The algorithm is implemented in C++. The instances are a subset of Solomon's set of CVRPTW test problems. They are type of $C1, C2, R1, R2$ with 25 customers. We

modified the number of vehicles and capacity in type $C1$, $R1$ to 25 vehicles, and the capacity of 200 changed to 15 vehicles, 30 wagons, 3 routes and a capacity of 150 for each wagon. For type $C2$, 25 vehicles and a capacity of 700 are changed to 15 vehicles, 30 wagons, 3 routes, and a capacity of 500 for each wagon. For type $R2$ with a capacity of 1000 changed to 15 vehicles, 30 wagons, 3 routes, and a capacity of 700 for each wagon. Tables 1–3 show the results.

Table 1. 25 Customers.

Instance	Gap (in %)	CPU Time (s)	Objective	Best Bound
C1	66.4	14,403	890	450
R1	67.2	14,402	665	292
R2	72.5	14,382	1033	308

Table 2. 50 Customers.

Instance	Gap (in %)	CPU Time (s)	Objective	Best Bound
C1	89	14,339	569	408
R1	76	14,420	680	408
R2	68	14,059	1065	414

Table 3. 10 Customers.

Instance	Gap (in %)	CPU Time (s)	Objective	Best Bound
C1	66	14,402	1137	630
R1	81	14,404	3340	630
R2	68	14,402	1562	630

All the instances have a large integrality gap and it takes a lot of time to solve the mathematical model.

6.2. Branch and Price Test

Solomon's (100 customers) instances [46] for Euclidean $VRPTW$ are modified to evaluate the model. We use the first ten customers of Solomon's instances to test the algorithm. The Euclidean distance between two customer locations determines the travel time for these instances.

The instances of Solomon [46] that we consider of six different types $C1$, $C2$, $R1$, $R2$, $RC1$, $RC2$. Each data set has eight to twelve 100-node problems. Sets $C1$ and $C2$ have clustered customers whose time windows were generated based on a known solution. Problem sets $R1$ and $R2$ randomly generate the customer's location over a square. Sets $RC1$ and $RC2$ combine randomly placed and clustered customers. Sets of type 1 have narrow time windows and small vehicle capacity. Sets of type 2 have large time windows and large vehicle capacity. Therefore, the optimal solutions for type 2 problems have very few routes and significantly more customers per route. It must be noted that the branch and price algorithm is an exact algorithm that could not solve more than 40–50 customers so far.

To use Solomon's instances, we modify them. So, the solution to $VRPTW$ instances are routes with one trip. We need to adjust them for the multi-trip and different capacity use. We use 10 of 25 customers for the instance. Instances of type 1 that have a capacity of 200 for each vehicle now have a capacity of 50 per wagon. Instances of type 2 with a capacity of 1000 per vehicle have been modified with a capacity of 100 per wagon. The number of vehicles is limited to 10. The number of wagons is 45. A tour's maximum number of routes is limited to three for these instances.

Experimental Results

The results of the branch and price algorithm are presented next. The program was implemented in the optimization programming language OPL, using ILOG CPLEX.

CEDAR, the Compute Canada cluster, was used for experimentation, with a limit of 4 h on each solve and a maximum memory requirement of 40GB. Our algorithm was able to find optimal solutions for $R1$, $R2$, $RC1$, and $RC2$.

The results are in Table 4 where the column named Instances is the type of the instance. The gap is between the LP-relaxation value at the root and the optimal integer value in %. CPU time is calculated as differences between the time recorded at the root and the end of the algorithm in seconds. Obj is the total distance. Iter is the number of iterations used to solve *RLMP* by CPLEX. Cols is the number of columns generated during the branch and price algorithm. Node is the number of nodes explored in the search tree. The route is the max number of routes used in all tours. Tour is the number of tours used to visit customers.

Table 4. Branch and Price for 10 Customers.

Instance	Gap	CPU Time (s)	Obj	Iter	Cols	Node	Route	Tour
RC102	994.364	869.56	563.562	20	17	3	2	2
RC103	994.364	899.163	563.562	20	17	3	2	2
RC105	994.548	398.842	545.237	18	17	1	2	2
RC106	993.178	111.626	682.168	13	12	1	2	3
RC107	994.189	202.428	581.069	11	10	1	2	2
RC108	994.121	670.193	587.86	17	14	3	2	2
RC202	992.434	470.885	756.612	5	4	1	1	2
RC203	992.434	470.885	756.612	5	4	1	1	2
RC204	993.722	872.675	627.785	3	2	1	1	2
RC206	794.258	139.613	724.979	8	7	1	2	1
RC207	991.067	446.75	893.255	15	14	1	2	1
RC208	797.627	800.939	388.128	3	2	1	1	1
R102	993.936	102.048	606.438	7	6	1	2	2
R103	993.936	106	606.438	7	6	1	2	2
R108	993.326	1287.93	667.381	16	13	3	2	2
R110	994.023	171.832	597.728	10	9	1	2	2
R111	993.629	2010.17	637.106	14	12	3	2	2
R202	991.33	455.858	867.046	5	4	1	2	1
R203	991.33	458.102	867.046	5	4	1	2	1
R204	993.722	881.522	627.785	3	2	1	1	2
R205	992.355	102.946	764.518	3	2	1	2	1
R206	993.992	581.478	600.814	3	2	1	1	2
R207	993.992	593.218	600.814	3	2	1	1	2
R208	993.993	612.737	600.683	3	2	1	1	2
R209	992.61	345.444	739.023	8	7	1	1	2
R210	993.672	1068.08	632.814	3	2	1	1	2
R211	996.263	443.173	373.736	2	1	1	1	1

6.3. Analysis

The mathematical model was tested, and it can take hours, or days to give solutions with gaps of around 60%, this is why we need to use a method like a branch and a price, which is an exact method to have the optimal solution of the problem.

The mathematical model is a new one in that the mathematical test confirms the validation of the model and the branch and price test shows solving the model optimally. The branch and price algorithm solves the model for the small instances, but it gives the optimal solution for these instances that the model can not provide.

During experimentation, we notice that the algorithm explores more nodes if a similar capacity (one wagon) is used for all tours. Giving the algorithm the option to check more tours with two wagons and three before starting a new branch makes it faster and explores a smaller number of nodes. The number of added columns before starting a new branch increases when different capacities are used. Varying capacity can significantly affect the number of explored nodes. The upper bound that the CPLEX provides us is a good upper

bound close to the optimal solution and it helps to explore fewer nodes and have a solution faster.

The algorithm solves type 2 instances where the time windows are more expansive on the horizon. It is faster and uses fewer iterations. For type C instances, branch and price could not find a solution to the time cut-off limit. All the instances listed in Table 4 were solved optimally within four hours.

We need to solve two resource constraints, the shortest path problem to determine a column with a negative reduced cost, and we need also to construct a route graph. The shortest path problem is solved using dynamic programming whose run time depends on the number of non-dominated paths which can be exponential. For large instances, this type of program run out of memory to store all the non-dominated paths. That is why we cannot solve large instances using column generation, as a part of future work we will look at other methods for solving the resource constraint shortest path problem.

7. Conclusions and Future Work

This work defines a new type of VRP, multi-trip vehicle routing problem with a variable number of wagons and time windows. The problem is serving clients' demands in a specific interval of time (time windows). At the same time, vehicles can make multiple daily trips, and the vehicle's capacity can be set at the beginning of the day by adding up to three wagons for each vehicle. First, a mathematical model of the problem is developed, and then we develop a branch and price algorithm to solve the problem. The approach to solving the problem is column generation embedded in a branch and bound algorithm. We implemented the branch and price algorithm for $MTVRP$-VW-TW on several Solomon's instances to show the algorithm's effectiveness. It can compute the optimal integer solution for limited customers.

We are interested in combining the column generation approach with metaheuristics to develop a faster solution for $MTVRP$-VW-TW. We will extend the model for split delivery with time windows, multiple wagons, and multiple trips per vehicle. The model can be extended to a multi-objective problem as well. In addition to these two types of the VRPs, the model can be extended to other variants of the VRPs. We can develop heuristic, exact and approximate approaches to solve the model.

Author Contributions: Conceptualization, L.K.; methodology, L.K.; software, validation, formal analysis L.K. and C.N.F.; writing—original draft preparation, L.K.; writing—review and editing, L.K. and C.N.F. All authors have read and agreed to the published version of the manuscript.

Funding: Funding support from NSERC, Canada, in the form of research assistantships, is gratefully acknowledged. Support for experimentation on a compute cluster from Compute Canada is also acknowledged.

Institutional Review Board Statement: Not Applicable.

Informed Consent Statement: Not Applicable.

Data Availability Statement: Solomon's (100 customers) instances for Euclidean VRPTW from https://www.sintef.no/projectweb/top/vrptw/100-customers/ (accessed on 20 August 2022) are modified to evaluate the model.

Acknowledgments: The authors would like to thank Robert Benkoczi and Daya Gaur for numerous discussions and helpful comments on a draft version of this manuscript.

Conflicts of Interest: There is no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

VRP	Vehicle Routing Problem
VRPTW	Vehicle Routing Problem with Time Windows
MTVRP	Multi-trip vehicle routing problem
MTVRPTW	Multi-Trip Vehicle Routing Problem with Time Windows
MTVRP-VW-TW	Multi-trip VRP with a Variable number of Wagons and Time Windows
RMP	Restricted Master Problem
RLMP	LP-relaxation of RMP

References

1. Dantzig, G.B.; Ramser, J.H. The Truck Dispatching Problem. *Manag. Sci.* **1959**, *6*, 80–91. [CrossRef]
2. Caric, T.; Gold, H. *Vehicle Routing Problem*; InTech: London, UK, 2008.
3. Hernandez, F.; Feillet, D.; Giroudeau, R.; Naud, O. An exact method to solve the multitrip vehicle routing problem with time windows and limited duration. *TRISTAN* **2010**, *7*, 366–369.
4. Azi, N.; Gendreau, M.; Potvin, J.Y. An exact algorithm for a vehicle routing problem with time windows and multiple use of vehicles. *Eur. J. Oper. Res.* **2010**, *202*, 756–763. [CrossRef]
5. Petch, R.J.; Salhi, S. A multi-phase constructive heuristic for the vehicle routing problem with multiple trips. *Discret. Appl. Math.* **2003**, *133*, 69–92. [CrossRef]
6. Solomon, M.M. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Oper. Res.* **1987**, *35*, 254–265. [CrossRef]
7. Cattaruzza, D.; Absi, N.; Feillet, D.; González-Feliu, J. Vehicle routing problems for city logistics. *EURO J. Transp. Logist.* **2017**, *6*, 51–79. [CrossRef]
8. Fleischmann, B. The Vehicle Routing Problem with Multiple Use of Vehicles. Ph.D. Thesis, Fachbereich Wirtschaftswissenschaften, Universität Hamburg, Hamburg, Germany, 1990.
9. Taillard, É.D.; Laporte, G.; Gendreau, M. Vehicle routeing with multiple use of vehicles. *J. Oper. Res. Soc.* **1996**, *47*, 1065–1070. [CrossRef]
10. Brandao, J.; Mercer, A. A tabu search algorithm for the multi-trip vehicle routing and scheduling problem. *Eur. J. Oper. Res.* **1997**, *100*, 180–191. [CrossRef]
11. Brandão, J.C.S.; Mercer, A. The multi-trip vehicle routing problem. *J. Oper. Res. Soc.* **1998**, *49*, 799–805. [CrossRef]
12. Nagy, G.; Salhi, S. The many-to-many location-routing problem. *Top* **1998**, *6*, 261–275. [CrossRef]
13. Campbell, A.M.; Savelsbergh, M. Efficient insertion heuristics for vehicle routing and scheduling problems. *Transp. Sci.* **2004**, *38*, 369–378. [CrossRef]
14. Salhi, S.; Petch, R. A GA based heuristic for the vehicle routing problem with multiple trips. *J. Math. Model. Algorithms* **2007**, *6*, 591–613. [CrossRef]
15. Olivera, A.; Viera, O. Adaptive memory programming for the vehicle routing problem with multiple trips. *Comput. Oper. Res.* **2007**, *34*, 28–47. [CrossRef]
16. Cattaruzza, D.; Absi, N.; Feillet, D.; Vigo, D. An iterated local search for the multi-commodity multi-trip vehicle routing problem with time windows. *Comput. Oper. Res.* **2014**, *51*, 257–267. [CrossRef]
17. Wassan, N.; Wassan, N.; Nagy, G.; Salhi, S. The multiple trip vehicle routing problem with backhauls: Formulation and a two-level variable neighbourhood search. *Comput. Oper. Res.* **2017**, *78*, 454–467. [CrossRef]
18. Tirkolaee, E.B.; Goli, A.; Bakhsi, M.; Mahdavi, I. A robust multi-trip vehicle routing problem of perishable products with intermediate depots and time windows. *Numer. Algebr. Control Optim.* **2017**, *7*, 417–433. [CrossRef]
19. Anggodo, Y.; Ariyani, A.; Ardi, M.; Mahmudy, W. Optimization of multi-trip vehicle routing problem with time windows using genetic algorithm. *J. Environ. Eng. Sustain. Technol.* **2017**, *3*, 92–97.
20. Tirkolaee, E.B.; Hosseinabadi, A.A.R.; Soltani, M.; Sangaiah, A.K.; Wang, J. A hybrid genetic algorithm for multi-trip green capacitated arc routing problem in the scope of urban services. *Sustainability* **2018**, *10*, 1366. [CrossRef]
21. Babaee Tirkolaee, E.; Abbasian, P.; Soltani, M.; Ghaffarian, S.A. Developing an applied algorithm for multi-trip vehicle routing problem with time windows in urban waste collection: A case study. *Waste Manag. Res.* **2019**, *37*, 4–13. [CrossRef]
22. Chen, D.; Pan, S.; Chen, Q.; Liu, J. Vehicle routing problem of contactless joint distribution service during COVID-19 pandemic. *Transp. Res. Interdiscip. Perspect.* **2020**, *8*, 100233. [CrossRef]
23. Zhen, L.; Ma, C.; Wang, K.; Xiao, L.; Zhang, W. Multi-depot multi-trip vehicle routing problem with time windows and release dates. *Transp. Res. Part E Logist. Transp. Rev.* **2020**, *135*, 101866. [CrossRef]
24. Neira, D.A.; Aguayo, M.M.; De la Fuente, R.; Klapp, M.A. New compact integer programming formulations for the multi-trip vehicle routing problem with time windows. *Comput. Ind. Eng.* **2020**, *144*, 106399. [CrossRef]
25. Desrochers, M.; Desrosiers, J.; Solomon, M. A new optimization algorithm for the vehicle routing problem with time windows. *Oper. Res.* **1992**, *40*, 342–354. [CrossRef]

26. Halse, K. Modeling and Solving Complex Vehicle Routing Problems. Ph.D. Thesis, Technical University of Denmark, Kongens Lyngby, Denmark, 1992.
27. Kohl, N.; Madsen, O.B.G. An optimization algorithm for the vehicle routing problem with time windows based on Lagrangian relaxation. *Oper. Res.* **1997**, *45*, 395–406. [CrossRef]
28. Kohl, N.; Desrosiers, J.; Madsen, O.B.; Solomon, M.M.; Soumis, F. 2-path cuts for the vehicle routing problem with time windows. *Transp. Sci.* **1999**, *33*, 101–116. [CrossRef]
29. Larsen, J. *Parallelization of the Vehicle Routing Problem with Time Windows*; Citeseer: Lyngby, Denmark, 1999.
30. Cook, W.; Rich, J.L. *A Parallel Cutting-Plane Algorithm for the Vehicle Routing Problem with Time Windows*; Technical Report; Rice University: Houston, TX, USA, 1999.
31. Kallehauge, B.; Larsen, J.; Madsen, O.B. *Lagrangean Duality and Non-Differentiable Optimization Applied on Routing with Time Windows-Experimental Results*; Relatório interno IMM-REP-2000-8; Department of Mathematical Modeling, Technical University of Denmark: Lyngby, Denmark, 2000.
32. Chabrier, A.; Danna, E.; Le Pape, C. Coopération entre génération de colonnes avec tournées sans cycle et recherche locale appliquée au routage de véhicules. In Proceedings of the Journées Nationales sur la Résolution Pratique de Problèmes NP-Complets, Nice, France, 27–29 May 2002; pp. 83–97.
33. Feillet, D.; Dejax, P.; Gendreau, M.; Gueguen, C. An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Netw. Int. J.* **2004**, *44*, 216–229. [CrossRef]
34. Rousseau, L.M.; Gendreau, M.; Pesant, G.; Focacci, F. Solving VRPTWs with constraint programming based column generation. *Ann. Oper. Res.* **2004**, *130*, 199–216. [CrossRef]
35. Chabrier, A. Vehicle routing problem with elementary shortest path based column generation. *Comput. Oper. Res.* **2006**, *33*, 2972–2990. [CrossRef]
36. Irnich, S.; Villeneuve, D. The shortest path problem with k-cycle elimination (k $\geqslant$ 3): Improving a branch and price algorithm for the VRPTW. *INFORMS J. Comput.* **2003**, *10*, 1–15.
37. Danna, E.; Le Pape, C. Accelerating branch-and-price with local search: A case study on the vehicle routing problem with time windows. In *Column Generation*; Desaulniers, G., Desrosiers, J., Solomon, M.M., Eds.; Kluwer Academic Publishers: Amsterdam, The Netherlands, 2005; Charpter 3.
38. Azi, N.; Gendreau, M.; Potvin, J.Y. An exact algorithm for a single-vehicle routing problem with time windows and multiple routes. *Eur. J. Oper. Res.* **2007**, *178*, 755–766. [CrossRef]
39. Macedo, R.; Alves, C.; de Carvalho, J.V.; Clautiaux, F.; Hanafi, S. Solving the vehicle routing problem with time windows and multiple routes exactly using a pseudo-polynomial model. *Eur. J. Oper. Res.* **2011**, *214*, 536–545. [CrossRef]
40. Munari, P.; Morabito, R. A branch-price-and-cut algorithm for the vehicle routing problem with time windows and multiple deliverymen. *Top* **2018**, *26*, 437–464. [CrossRef]
41. Faiz, T.I.; Vogiatzis, C.; Noor-E-Alam, M. A column generation algorithm for vehicle scheduling and routing problems. *Comput. Ind. Eng.* **2019**, *130*, 222–236. [CrossRef]
42. Seixas, M.P.; Mendes, A.B. A branch-and-price approach for a multi-trip vehicle routing problem with time windows and driver work hours. In Proceedings of the Congreso Latino-Iberoamericano de Investigación Operativa. Simpósio Brasileiro de Pesquisa Operacional, Rio de Janeiro, Brazil, 24–28 September 2012; pp. 3469–3480.
43. Bettinelli, A.; Cacchiani, V.; Crainic, T.G.; Vigo, D. A branch-and-cut-and-price algorithm for the multi-trip separate pickup and delivery problem with time windows at customers and facilities. *Eur. J. Oper. Res.* **2019**, *279*, 824–839. [CrossRef]
44. Marques, G.; Sadykov, R.; Dupas, R.; Deschamps, J.C. A branch-cut-and-price approach for the single-trip and multi-trip two-echelon vehicle routing problem with time windows. *Transp. Sci.* **2022** . [CrossRef]
45. Huang, N.; Li, J.; Zhu, W.; Qin, H. The multi-trip vehicle routing problem with time windows and unloading queue at depot. *Transp. Res. Part E Logist. Transp. Rev.* **2021**, *152*, 102370. [CrossRef]
46. Solomon's Benchmark Instances. Available online: https://www.sintef.no/projectweb/top/vrptw/100-customers/ (accessed on 20 August 2022).