A Time-Constrained Vehicle Routing Problem with a Heterogeneous Fleet: Algorithms and Analysis

Young-Chae Hong and Amy Cohn

August 18, 2014

Abstract

In this paper, we consider a new variant of the Time-Constrained Heterogeneous Vehicle Routing Problem (TCHVRP). In this problem, the cost and travel time of any given arc vary by vehicle type within a heterogeneous fleet. In particular, we make no assumptions about Pareto dominance across vehicles; nor do we assume that cost and time are correlated. Our research is motivated by situations in which existing fleets are evolving to incorporate hybrid vehicles in addition to their existing vehicle types; for many vehicles, the cost per mile depends heavily on the type of driving (such as highway versus city). We formulate TCHVRP as a path-based model, which we solve using column generation. We introduce several different methods to solve the pricing problem. We conclude by conducting empirical analyses to assess both the performance of the proposed approach.

1 Introduction

In this paper, we present models and algorithms to solve a variant of the Time-Constrained Heterogeneous Vehicle Routing Problem (TCHVRP). In this variant of the classical Vehicle Routing Problem (VRP), first posed by Dantzig and Ramser [12], we allow vehicles to vary not only by capacity (in our case, corresponding to a limit on total travel and service time), but by cost and time on each arc as well. This research is motivated by the gradual introduction of hybrid vehicles into existing fleets. In such cases, one vehicle (e.g. with a traditional combustion engine) might get better mileage (and thus have lower cost) on an arc primarily comprised of highway driving, whereas a hybrid vehicle might dominate in cost over an arc primarily comprised of city driving. Likewise, highway distances can be longer between two points than a city route, but travel times shorter due to less traffic and higher speeds. Thus, of particular importance to this research is the fact that we do not assume Pareto dominance of one vehicle type over another. Nor do we assume any correlation between arc distances and arc costs or arc times.

We consider two ways to model TCHVRP, one an arc-based formulation and the other a path-based formulation. We find the arc-based ("explicit") formulation to be intractable for all but very small problem instances. We therefore focus on a path-based approach, where *delayed column generation* [33] can be used to address the very large number of columns. We introduce and analyze several variants of a dynamic programming (DP) approach for generating these columns effectively in the pricing problem.

The contribution of this paper is in developing methods for solving a variant of the VRP with the novel aspect that arc costs and times are not correlated with distance, and that there is no assumed Pareto dominance across vehicle types. We demonstrate the tractability of our approach using data based on test instances commonly used in the literature.

The paper is organized as follows. Section 2 formally defines the problem, highlights our proposed solution approaches, reviews the related literature, and summarizes the data sets to be used for computational experiments. Section 3 presents the arc-based approach and demonstrates challenges of tractability. Section 4 formulates a path-based model. Section 5 presents a column generation method to solve the path-based model and, in particular, several dynamic programming approaches to solve the pricing problem. Finally, Section 6 summarizes our findings and concludes by proposing opportunities for future research.

2 Time-Constrained Heterogeneous Vehicle Routing Problem

2.1 Problem Description

The classical Vehicle Routing Problem (VRP) is to find an optimal (i.e. minimum-cost) set of routes for a fleet of vehicles so as to serve a given set of customers; the Traveling Salesman Problem (TSP) is a special case of VRP, in which only one vehicle must serve all customers [11]. Many other variants of the VRP have been considered, including: Capacitated VRP (CVRP), in which the vehicles have limited capacity [7, 8, 17], VRP with Time Windows (VRPTW), where customers have time windows within which the deliveries must be made [9, 14], and Heterogeneous Fleet VRP (HVRP), in which there are different types of vehicles characterized by different capacities and costs [25].

Within HVRP, problems can be further classified. For example, some have finite numbers of vehicles for each type and some are unlimited. Some have fixed costs for each vehicle and some do not. In some cases, the capacity is constant across all vehicle types and in some cases it varies across types. Furthermore, arc-costs may be constant or may vary by vehicle type. These problems are further described, and the literature reviewed, in Baldacci, Battarra, and Vigo [2].

We consider another variant of HVRP, which we label the Time-Constrained Heterogeneous Vehicle Routing Problem. In this variant, the cost and travel time of any given arc may vary by vehicle type within a heterogeneous fleet. In particular, we make no assumptions about Pareto dominance across vehicles. Thus, one vehicle may be the least-cost option on certain arcs but the most costly on others. Similarly, one vehicle may be fastest over some arcs, but slowest over others. Furthermore, we do not assume that cost and time are correlated. This allows us to incorporate characteristics of mixed fleets that combine vehicles with both traditional combustion and hybrid engines.

Specifically, TCHVRP is defined as follows:

- We are given a fixed depot and a known set of customers.
- We are given a set of vehicle types, with a finite number of vehicles within each type.
- We are given arcs connecting each customer with the depot and each pair of customers. Each vehicle type/arc pair has a given cost. There is no fixed cost per vehicle.
- Each vehicle type has its own time limit on the amount of work that vehicles of that type can perform. Each vehicle type/arc pair has a given time duration, and there is a given time associated with servicing each customer. Note that our "capacity" is time, not volume, and this resource is consumed over the arcs as well as at the nodes (i.e. customers).
- The goal is to find the minimum-cost assignment of vehicle types to routes such that each customer is visited exactly once, no vehicle exceeds its time limit, and we do not use more vehicles of a given type than are available.

2.2 Literature Review

Exact approaches to HVRP, and to VRP problems in general, are typically based on integer programming (IP) formulations. These are usually either arc-based or path-based models.

Arc-based models use binary decision variables to indicate whether a given vehicle (or vehicle type) travels between two customers in the optimal solution. Formulations of this type for the VRP can be first found in Garvin et al. [18], and Gavish and Graves [19, 20]. The heterogeneous VRP was formulated in Gheysens et al. [22] and Golden et al. [25] by using three-index binary variables x_{ij}^k as vehicle flow variables that take value 1 if a vehicle of type k travels directly from customer i to customer j, and 0 otherwise. Such formulations must include subtour elimination constraints, however, which often lead to heavily fractional solutions to the linear programming (LP) relaxation and, in turn, weak lower bounds on the optimal objective value.

VRP is also often modeled, therefore, with a set partitioning formulation, as originally proposed by Balinski and Quandt [4]. The set partitioning formulation for heterogeneous VRPs associates a binary variable x_r^t with each feasible route r and vehicle type t. However, given that the number of candidate routes is exponentially large, it is often impractical to explicitly enumerate all routes in the model. Instead, *delayed column generation* algorithms are used, using a pricing problem such as those posed by Rao and Zionts [33], Foster and Ryan [16], and Agarwal, Mathur, and Salkin [1].

Both of these types of exact approaches, however, can suffer from significant tractability issues. Even pure VRP problem instances with more than 100 nodes can typically not be solved to optimality in a reasonable amount of time [3, 17]. Variations such as HVRP are even more complex to solve. Therefore, many heuristics have also been developed in order to find high-quality solutions in acceptable run times.

Dating back as far as Dantzig and Ramser [12], many heuristics have been published. The savings heuristic by Clarke and Wright [10] iteratively merges partial routes to form a set of feasible routes, and the sweep algorithm by Gillett and Miller [23] sequentially generates non-overlapping feasible routes by rotating a half-line rooted at the depot. In addition to classical VRP heuristics, several meta-heuristic approaches have been developed as well. The best-known examples of meta-heuristics include tabu search [24], simulated annealing [28], and genetic algorithms [27]. Specific meta-heuristics for VRP are provided in Taillard [37], Gendreau [21], Nagata [31], and Prins [32]. Many of the most successful VRP heuristics, however, rely on the use of distance as a surrogate for cost, assuming a constant cost per mile for each vehicle and a given distance matrix. Because we are specifically interested in the case where distance and cost may not be correlated, we cannot build directly on these heuristics for TCHVRP. We instead present an alternative approach based on a path-based model, as we discuss later in the paper.

There is a rich and extensive literature on VRP in general. We refer the reader to Laporte [29], Toth and Vigo [38] and Golden, Raghavan, and Wasil [26] for comprehensive surveys. Additional studies and algorithms focused specifically on HVRPs can be found in Baldacci [2].

2.3 Data Sets

Throughout the paper, we present computational experiments to assess the performance of different approaches to solving TCHVRP. Our computational experiments are loosely based on the data sets of Golden et al. [25]. Specifically, we use the network topology of these data sets, defined by coordinates in a Euclidean space for each node in the network. Our arc costs and times cannot be drawn from the data sets, however, as it is the variation in these parameters that we are specifically exploring.

Our data sets range in size from 20 nodes to 35 nodes, with exhaustive sets of arcs connecting all node pairs. These correspond to Problem Instances 3, 4 and 15 through 18 of Golden et al. [25]. Specifically, we use the first 20 nodes from Problem Instances 3 and 4, the first 25 nodes from Problem Instances 17 and 18, the first 30 nodes from Problem Instances 15 and 16, and the first 35 nodes from Problem Instances 17 and 18.

We consider three different cases of problem instances derived from these original data sets. In the first case, which serves as a base case, we construct instances where there is only one vehicle type and arc costs and times are fully correlated with distance. Specifically, we define a constant cost per mile and a constant speed, and apply these to all arcs in the network for all vehicle types, using the Euclidiean distance between a given pair of nodes as their arc length.

In the second case, we allow cost per mile and speed to vary by vehicle type, but arc costs and speeds are still fully correlated with distance, and there is thus Pareto dominance with respect to cost, i.e. whichever vehicle type is least-cost on one arc will be least-cost on all arcs.

In the third case, we consider the case of interest, in which costs and times are neither Pareto-dominant nor fully correlated with each other (and thus, implicitly, with the arc distance), although we do recognize that there is at least some degree of correlation to be reasonably expected. To generate arc costs and times, we again start with the same arc lengths between each pair of customers as in the first two cases. We then assign each vehicle type a baseline cost per mile and a baseline speed, but we also randomly generate a perturbing error factor for each arc to avoid Pareto dominance and full correlation between cost and time. Given that we are randomly generating the perturbation factors, we create ten instances for each set of network topologies in this case. [See Appendix A for full details.]

Finally, we assume three types of vehicles for problem instances in the second and third cases. For 20-node instances, we assume two vehicles of each type; for 25-node instances, we assume three vehicles of each type; and for 30- and 35-node instances, we assume four vehicles of each type. For the sake of exposition, we assign the same service time (thirty minutes) to each customer and the same capacity constraint (eight hours) to each vehicle type. It is trivial, however, to specify vehicle- and customer-specific values within our proposed model.

All computational experiments were performed on an Intel Xeon 3.20 GHz processor with 32 GB of memory using CPLEX 12.2 C++ API with an optimality gap of 0.01%.

3 Arc-Based Approach

As with other variants of VRP, TCHVRP can be formulated as an arc-based network flow model, using subtour elimination constraints such as those found in Golden et al. [25], which in turn builds on the work of Miller, Tucker, and Zemlin (MTZ) for the TSP [30]. Such an approach to TCHVRP, however, suffers from poor computational performance, as we will demonstrate below. Thus, this section motivates us to instead pose a path-based approach to solving TCHVRP.

3.1 Formulation

Parameters and Sets

N	set of customers in the network, where index 0 represents the depot
T	set of vehicles types
c_{ij}^t	cost to travel from customer i to customer j for vehicle type t,$\forall i, j \in N, \forall t \in T$
d_{ij}^t	travel time from customer i to customer j for vehicle type t, $\forall i, j \in N, \forall t \in T$
M_t	number of available vehicles of type t, $\forall t \in T$
Q_t	time limit for vehicle type t, $\forall t \in T$
L_i	service time at customer i, $\forall i \in N \setminus 0$

Variables

x_{ij}^t	binary variable that takes value 1 if a vehicle of type t is assigned to travel between customers i and j, $\forall i, j \in N, \forall t \in T$
y_i	cumulative travel and service time up through customer i, $\forall i \in N$

Arc-Based Model (ABM):

$$\min \sum_{t \in T} \sum_{i \in N} \sum_{j \in N} c_{ij}^t x_{ij}^t \tag{1a}$$

$$\text{subject to: } \sum_{t \in T} \sum_{i \in N} x_{ij}^t = 1 \quad \forall j \in N \setminus 0 \tag{1b}$$

$$\sum_{i \in N} x_{ij}^t - \sum_{i \in N} x_{ji}^t = 0 \quad \forall j \in N, \forall t \in T \tag{1c}$$

$$\sum_{j \in N} x_{0j}^t \leq M_t \quad \forall t \in T \tag{1d}$$

$$Q_t \sum_{t \in T} x_{ij}^t + \sum_{t \in T} (d_{ij}^t + L_j) x_{ij}^t + y_i \leq y_j + Q_t \quad \forall i, j \in N, i \neq j \tag{1e}$$

$$y_j + d_{j0} \sum_{t \in T} x_{j0}^t \leq Q_t \quad \forall j \in N \setminus 0 \tag{1f}$$

$$x_{ij}^t = \begin{cases} \in \{0,1\} & , \quad i \neq j, \forall t \in T \\ 0 & , \quad i = j, \forall t \in T \end{cases} \tag{1g}$$

$$y_j = \begin{cases} \in \mathbb{R}^+ & , \quad j \in N \\ 0 & , \quad j = 0 \end{cases} \tag{1h}$$

The objective (1a) minimizes the total routing cost, which is the sum of the costs associated with each arc used.

Constraint set (1b) ensures that exactly one vehicle type and one incoming arc is assigned to customer j.

Constraint set (1c) specifies flow conservation at each customer.

Constraint set (1d) specifies that the number of arcs out of the depot (and thus the number of routes) assigned to a vehicle type does not exceed the available number of vehicles of that type.

Constraint set (1e) provides the subtour elimination constraints. The variable y_i is the accumulated time associated with whatever vehicle services customer i. Constraint set (1e) defines the relationship between any two customers that are sequential on a route to have respective y values that differ by at least the travel and service time associated with moving between them. This ensures that a subtour cannot exist, otherwise the y variable

associated with any customer appearing in a subtour would ultimately be required to be strictly greater than itself. In particular, when $\sum_{t\in T} x_{ij}^t = 1$, some vehicle travels from customer i to customer j. Therefore, y_j must be at least equal to y_i plus the time between them and the service time at customer j ($\sum_{t\in T}(d_{ij}^t + L_j)x_{ij}^t$). Conversely, when $\sum_{t\in T} x_{ij}^t = 0$, there is no known relationship between y_i and y_j. The equation reduces to $y_i \leq y_j + Q_t$ which will also hold because Q is the limit on accumulated time for any single vehicle, i.e. any single route.

Constraint (1f) ensures that the time incurred by a vehicle through servicing of customer j, plus the time that it would take to return to the depot if this were the last customer on the route, cannot exceed the total time limit for vehicles of the associated type. This enforces the capacity constraint.

3.2 Computational Experiments and Analysis

ABM is difficult to solve in practice, both because it is highly fractional and because the LP relaxation provides an inherently weak lower bound on the optimal objective value. To demonstrate this, we evaluated a randomly chosen subset of our non-Pareto data instances, limiting each run to at most two hours of run time. Tables 1 and 2 show our computational results. Specifically, for each of the four instances considered, Table 1 provides the number of nodes in the network, the realization number (recall that for each of the original networks, we randomly generated 10 sets of arc distances, costs, and times), the number of constraints, and the number of variables. This table also includes information about the branch-and-bound results after two hours of run time. Specifically, we see the number of branch-and-bound nodes solved, the number of pending nodes remaining in the branch-and-bound tree, and the optimality gap at the end of two hours. Table 1 shows that ABM failed to solve any of the instances to provable optimality (the optimality tolerance is 0.01%) within the two hour time limit, and in the case of the 35-node instances, could not even find an integer-feasible solution. In addition, the non-trivial optimality gaps and remaining high numbers of pending nodes suggest that the algorithm is far from terminating at the two-hour time limit.

Data Set				ABM (2hours)		
Size	Realization	Number of constraints	Number of variables	Number of nodes solved	Number of pending nodes	Optimality gap
20	8	~500	~1200	914,203	789,919	4.66%
25	5	~750	~1900	314,918	305,205	12.26%
30	7	~1050	~2700	190,431	188,429	13.27%
35	5	~1400	~3700	73,857	72,455	N/A

Table 1: ABM after two hours

Table 2 enables us to compare the run times and best objective values for three variations of the problem - the LP relaxation of the integrality constraints (ABM-LPR), the original problem with a two-hour limit on run time, and the original IP with the relaxation of the subtour elimination constraints (ABM-SER).

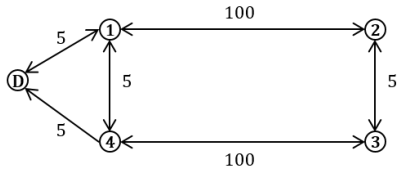
As Table 2 indicates, the run time of the LP relaxation is close to 0, suggesting that the problem size is not the source of computational challenge. The LP solutions, however, are highly fractional, as demonstrated in Table 1 by the large number of branch-and-bound nodes solved and the nearly equivalent number of pending nodes yet to be solved. Furthermore, the optimality gap at termination is quite weak, with three of the instances having gaps between 4.66% and 13.27% and one of the instances unable to find a single integer-feasible solution within the two-hour time limit.

The key to this fractionality appears to be the subtour elimination constraints. When we remove these, the remaining IP solves very quickly (almost as fast as ABM-LPR). As seen in Table 2, however, the objective value for ABM-SER is even lower than the objective value for the LP relaxation, providing an even worse optimality gap.

To help understand why ABM-LPR and ABM-SER provide such weak lower bounds, consider the example from Figure 1. In this example, we see a simple network with four nodes (plus the depot). Arc costs are depicted; for the sake of exposition, we assume all vehicles are identical and have infinitely long time limits. In this instance, the optimal IP solution is to follow the path D-1-2-3-4-D, with each corresponding arc having flow of one, and an objective value of 215. When the subtour elimination constraints are relaxed, however, it is valid to assign one unit of flow to the arcs in path D-1-4-D and likewise to the arcs in the path 2-3-2. This reduces the objective value by 88 percent, from 215 to 25.

Data Set		ABM-LPR		ABM (2hours)		ABM-SER	
Size	Realization	Run Time (sec)	Objective Value	Run Time (sec)	Objective Value	Run Time (sec)	Objective Value
20	8	< 1	375.46	7200	477.91	< 1	369.51
25	5	< 1	464.79	7200	544.89	1	462.04
30	7	< 1	472.99	7200	544.10	< 1	467.91
35	5	< 1	532.77	7200	613.84	< 1	526.73

Table 2: Optimal cost of relaxations



Models	Paths	Cost
ABM	r_{IP}: D-1-2-3-4-D (1)	$5 + 100 + 5 + 100 + 5 = 215$
ABM-SER	r_{SER}: D-1-4-D, 2-3-2 (1)	$(5 + 5 + 5) + (5 + 5) = 25$
ABM-LPR $(0 \leq \alpha \leq 1)$	r_{LPR}: D-1-2-3-4-D (α) D-1-4-D , 2-3-2 ($1 - \alpha$)	$\alpha \times 215 + (1 - \alpha) \times 25$ $= \alpha \times r_{IP} + (1 - \alpha) \times r_{SER}$

Figure 1: Example of weak lower bounds

Finally, when the subtour constraints are included but the integrality requirements are relaxed, we can assign α units of flow to each arc in the path D-1-2-3-4-D (the solution to ABM), and $1 - \alpha$ units to the arcs in the path D-1-4-D and 2-3-2 (the solution to ABM-SER), i.e. ABM-LPR is a convex combination of ABM and ABM-SER, where α approaches 0 as Q goes to infinity.

4 Path-Based Model

Motivated by the computational experiments described in Section 3, we seek an approach that is not hampered by the need for subtour elimination. In this section, we introduce a path-based model that, by explicitly constructing paths as an input to the model, eliminates the need for such constraints.

Notation

Parameters and Sets

N set of customers in the network, where index 0 represents the depot
T set of vehicles types
R_t set of all feasible routes (i.e. paths) for vehicle type $t \in T$
c_r^t travel cost of route r for vehicle type t, $\forall t \in T, \forall r \in R_t$
δ_{ir}^t binary coefficient that takes value 1 if customer i belongs to route r for vehicle type t, else 0, $\forall i \in N \setminus 0, t \in T, r \in R_t$
M_t number of available vehicles of type t, $\forall t \in T$
Q_t time limit for vehicle type t, $\forall t \in T$
L_i service time at customer i, $\forall i \in N \setminus 0$

Variables

x_r^t binary variable that takes value 1 if route r is assigned to a vehicle of type t, else 0, $\forall t \in T, \forall r \in R_t$

Path-Based Model (PBM):

$$\min \sum_{t \in T} \sum_{r \in R_t} c_r^t x_r^t \tag{2a}$$

$$\text{subject to: } \sum_{t \in T} \sum_{r \in R_t} \delta_{ir}^t x_r^t = 1 \quad \forall i \in N \setminus 0 \qquad (\pi_i) \tag{2b}$$

$$\sum_{r \in R_t} x_r^t \leq M_t \quad \forall t \in T \qquad (\mu_t) \tag{2c}$$

$$x_r^t \in \{0,1\} \quad \forall t \in T, \forall r \in R_t \tag{2d}$$

The objective (2a) minimizes the total routing cost.

Constraint set (2b) requires that each customer must be covered by exactly one route.

Constraint set (2c) specifies that at most M_t routes are assigned to vehicles of type $t \in T$.

In addition, we associate dual variables π_i and μ_t with (2b) and (2c), respectively.

5 Solving PBM

5.1 Column Generation Overview

PBM contains too many variables to solve explicitly except for very small problem instances. We therefore propose to use *column generation* [6, 13] to solve the LP relaxation (PBM-LPR). Specifically, we begin with an initial subset of the feasible columns (i.e. routes), which we call the *restricted master problem* (RPBM-LPR). We then use a *pricing problem* to identify promising new routes to add to RPBM-LPR, based on the dual values of the current optimal solution. If such columns (i.e. routes) are found, they are used to augment RPBM-LPR and the process repeats. An optimal solution to the LP relaxation is guaranteed when no new negative reduced cost routes can be found.

Reduced Cost:

For a given feasible route $r_t \in R_t$ where γ_{ijr}^t is a binary parameter specifying whether arc (i,j) is included in route r_t, the corresponding reduced cost equation is:

$$\begin{aligned} RC_r^t &= c_r^t - \sum_{i \in N} \pi_i \delta_{ir}^t - \mu_t \\ &= \sum_{i \in N} \sum_{j \in N} c_{ij}^t \gamma_{ijr}^t - \sum_{i \in N} \pi_i (\sum_{j \in N} \gamma_{ijr}^t) - \mu_t \\ &= \sum_{i \in N} \sum_{j \in N} (c_{ij}^t - \pi_i) \gamma_{ijr}^t - \mu_t \end{aligned}$$

To find a negative reduced cost column for vehicles of type t, or to prove that no such columns exist, we can formulate the following optimization problem:

$$\min \sum_{i \in N} \sum_{j \in N} (c_{ij}^t - \pi_i) \delta_{ijr}^t - \mu_t \tag{3a}$$

$$\text{subject to: } r_t \in R_t \tag{3b}$$

For the remainder of this section, we focus primarily on solving PBM-LPR and, in particular, the pricing problem. When we solve the integer version of the problem, we do so heuristically. That is, for the simplicity of implementation, we solve the LP to optimality and then find the best IP solution relative to the current set of columns in the restricted master. In practice, it would be necessary to generate new columns at each subsequent nodes of the branch-and-bound tree in order to ensure a provably optimal solution.

5.2 Dynamic Programming Approaches to Solving the Pricing Problem

The key challenge in solving PBM-LPR is in solving the pricing problem, i.e. in generating candidate routes (negative reduced cost pivot variables). In theory, this problem ((3a) and (3b)) could be formulated as a MIP. However this will result in the same difficulties observed in the arc-based formulation, such as fractionality in the LP relaxation and a weak lower bound, due to the sub-tour elimination (1e) and time limit constraints (1f), which must now be shifted into the pricing problem. We therefore instead take a dynamic programming (DP) approach in which, by constructing the routes dynamically, we naturally avoid sub-tours. Throughout the remainder of this section, we present and compare several DP algorithms for solving the pricing problem.

5.2.1 Elementary Shortest Path Problem with Resource Constraints

The pricing problem for PBM-LPR can be posed as an Elementary Shortest Path Problem with Resource Constraints (ESPPRC). This problem identifies minimum-cost *elementary paths* (i.e. paths without any embedded cycles) that do not violate some sort of resource constraint. In our case, the resource is time (which is consumed over both arcs and nodes), and the cost is the reduced cost associated with the route when included in the restricted master. That is, we consider the true cost (which is the sum of all the arc costs) minus the dual value associated with each node minus the dual value associated with the specific vehicle type. In particular, one reduced cost problem can be solved for each vehicle type, in which the corresponding costs (true and dual) are assigned to individual arcs within the network.

Feillet et al. [15] were the first to propose an exact dynamic programming algorithm for solving the pricing problem of the Vehicle Routing Problem with Time Windows (VRPTW) where paths with embedded cycles are forbidden. Other refinements for the DP algorithm have also been suggested by Chabrier [5], and Righini and Salani [35, 36]. While DP can be a good candidate approach for solving the pricing problem for some VRPs [34], the exponential rise in the state-space can be burdensome in other cases. For example, the success of Feillet et al. [15] depends on the use of narrow time windows to facilitate pruning. On the other hand, when time windows are too wide, the computational performance suffers significantly.

In TCHVRP, we encounter tractability issues associated with the limited opportunity to prune. Although we can use the vehicle time limits and the fact that nodes cannot be repeated to prune, these are not sufficient. Ideally, we also want to prune whenever two partial paths meet at the same node but with different costs and consumed times. Specifically, given two partial routes p_1 and p_2 that terminate at the same node, p_1 is dominated by p_2 and therefore p_1 can be pruned, if: 1) p_1 is more costly than p_2; 2) p_1 takes longer than p_2. We also need a third criterion, however: 3) the set of nodes included in p_1 is a superset of the nodes included in p_2.

The third criteria is necessary to ensure equivalent remaining opportunities to capture the benefit of negative dual values. For example, consider partial paths p_1: $D - X - Y - Z$ and p_2: $D - W - Z$. We are tempted to prune p_1 if it is higher cost and consumes more time than path p_2. However, it may be the case that the optimal (i.e. most negative reduced cost) path is $D - X - Y - Z - W - D$. If we prune p_1, we will not encounter this path. Conversely, the path $D - W - Z - W - D$ (which should be both cheaper and less time consuming than $D - X - Y - Z - W - D$) is not actually a valid path because it repeats node W (and thus accumulates the dual value associated with the cover constraint for node W twice). Thus we can only prune p_1 if it is not only more costly and more time consuming than p_2 but also if it includes a superset of the nodes found in partial path p_2. This criteria greatly reduces the pruning opportunities, with significant negative impact on computational performance, as we demonstrate below.

To evaluate the ESPPRC approach, considering both run time and solution quality, we conducted two experiments, using the data sets described in Section 2.3 and detailed in Appendix A. First, we compared the optimal objective values of the LP relaxations of the arc-based model to the path-based model. Second, we compared the heuristic objective values found by (a) solving the LP relaxation of the path-based model to optimality and then finding the optimal integer solution with respect to the given columns in the restricted master problem versus (b) allowing the arc-based integer model to run for the same amount of time as the path-based model and taking the best integer feasible solution.

Consistently, we observed that the LP relaxation of the path-based model is substantially higher than that of the arc-based model, thus yielding a tighter lower bound. As seen in Columns ABM-LPR and ESPPRC-LPR of Table 3, the LP relaxation of the path-based model ranges from 18.33% to 32.75% higher than the arc-based model, with an average increase of 25.68%.

To compare the objective values, we solved the path-based LP relaxation to optimality, then found the best integer solution relative to those columns. We subsequently ran the arc-based integer model for the equivalent

amount of time and took the lowest integer solution found during that time. In the best case, the path-based approach found a solution that was 69.37% lower than the ABM solution. In addition, in many cases, the arc-based approach found no integer-feasible solution (i.e. N/A), while the path-based approach always did. Finally, in all but three instances, the PBM solution was strictly better than the ABM solution, and all three of those instances were instances of the homogeneous version of the problem. Results for this experiment can be seen in Table 3, Columns ESPPRC-IP and Column ABM-IP.

Index	Instance	Size	Vehicle Type	ABM-LPR	ESPPRC-LPR	%	ESPPRC-IP	time(sec)	ABM-IP	time(sec)	%
1	1	20	Single_Pareto	**438.072**	**548.778**	**25.27%**	**570.69**	**27**	**570.69**	**27**	**0.00%**
2	1	20	Multiple_Pareto	**383.637**	**488.224**	**27.26%**	**525.44**	**123**	**553.4**	**123**	**5.05%**
-	1-10	20	Multiple_NonPareto	**372.6257**	**466.8702**	**25.29%**	**499.287**	**65.5**	**531.31**	**66**	**6.03%**
3	1			371.78	466.91	25.59%	489.82	49	526.94	66	7.04%
4	2			374.5	473.504	26.44%	514.54	68	535.86	66	3.98%
5	3			373.955	471.317	26.04%	501.17	92	582.21	66	13.92%
6	4			375.452	460.342	22.61%	493.58	102	529.69	66	6.82%
7	5			364.397	464.748	27.54%	494.77	67	522.39	66	5.29%
8	6			382.248	479.287	25.39%	506.86	35	517.52	66	2.06%
9	7			372.486	461.86	23.99%	505.58	47	513.26	66	1.50%
10	8			375.455	470.413	25.29%	501.74	28	508.85	66	1.40%
11	9			371.301	466.656	25.68%	500.96	110	546.96	66	8.41%
12	10			364.683	453.665	24.40%	483.85	57	529.42	66	8.61%
13	1	25	Single_Pareto	**559.204**	**661.708**	**18.33%**	**674.79**	**47**	**674.79**	**47**	**0.00%**
14	1	25	Multiple_Pareto	**489.413**	**606.384**	**23.90%**	**616.48**	**255**	**922.24**	**255**	**33.15%**
-	1-10	25	Multiple_NonPareto	**475.7815**	**576.5563**	**21.18%**	**589.236**	**232.9**	**983.626**	**233**	**40.10%**
15	1			475.899	569.747	19.72%	598.79	160	764.29	233	21.65%
16	2			464.149	567.087	22.18%	572.85	154	631.79	233	9.33%
17	3			478.001	570.983	19.45%	595.24	192	683.24	233	12.88%
18	4			484.074	582.357	20.30%	594.7	281	929.81	233	36.04%
19	5			464.791	572.237	23.12%	577.96	245	829.94	233	30.36%
20	6			486.371	583.196	19.91%	595	305	1942.52	233	69.37%
21	7			475.794	574.853	20.82%	584.11	239	807.77	233	27.69%
22	8			481.448	584.172	21.34%	594.61	280	925.87	233	35.78%
23	9			463.837	566.54	22.14%	577.07	186	940	233	38.61%
24	10			483.451	594.391	22.95%	602.03	287	1381.03	233	56.41%
25	1	30	Single_Pareto	**556.388**	**690.789**	**24.16%**	**733.58**	**348**	**713.07**	**348**	**-2.88%**
26	1	30	Multiple_Pareto	**487.081**	**636.173**	**30.61%**	**650.24**	**1798**	**789.43**	**1798**	**17.63%**
-	1-10	30	Multiple_NonPareto	**476.5658**	**608.994**	**27.79%**	**636.354**	**2305.6**	-	**2306**	-
27	1			480.186	605.767	26.15%	642.18	2814	742.4	2306	13.50%
28	2			478.29	607.222	26.96%	631.58	2422	653.76	2306	3.39%
29	3			491.584	620.204	26.16%	659	1679	743.2	2306	11.33%
30	4			464.098	599.535	29.18%	643.53	2372	835.23	2306	22.95%
31	5			467.434	603.99	29.21%	603.99	2895	697.17	2306	13.37%
32	6			468.403	611.449	30.54%	646.09	3016	841.7	2306	23.24%
33	7			472.99	604.492	27.80%	644.12	1464	709.8	2306	9.25%
34	8			478.938	614.081	28.22%	623.81	2821	N/A	2306	N/A
35	9			480.524	609.289	26.80%	624.22	1820	746.59	2306	16.39%
36	10			483.211	613.911	27.05%	645.02	1753	780.29	2306	17.34%
37	1	35	Single_Pareto	**640.683**	**801.635**	**25.12%**	**821.23**	**1708**	**821.23**	**1708**	**0.00%**
38	1	35	Multiple_Pareto	**560.893**	**744.589**	**32.75%**	**766.09**	**8003**	**N/A**	**8003**	**N/A**
-	1-10	35	Multiple_NonPareto	**548.8337**	**703.8488**	**28.24%**	**728.123**	**8252.4**	-	**8253**	-
39	1			572.354	724.176	26.53%	750.18	8497	852.84	8253	12.04%
40	2			556.932	716.294	28.61%	749.49	11436	802.4	8253	6.59%
41	3			533.847	692.536	29.73%	707.57	7945	N/A	8253	N/A
42	4			565.321	712.613	26.05%	735.07	9706	N/A	8253	N/A
43	5			532.77	701.591	31.69%	725.4	6040	N/A	8253	N/A
44	6			546.034	699.645	28.13%	728.95	9672	795.19	8253	8.33%
45	7			529.464	680.76	28.58%	706.56	2800	N/A	8253	N/A
46	8			551.074	709.405	28.73%	732.58	8407	N/A	8253	N/A
47	9			546.031	697.333	27.71%	717.77	10155	N/A	8253	N/A
48	10			554.51	704.135	26.98%	727.66	7866	N/A	8253	N/A

(ΔLPR = [RSPPRC_LPR] − [ABM_LPR], ΔIP = [ABM_IP] − [ESPPRC_IP])

Table 3: Solution Quality (ESPPRC)

Although the first two experiments show that the path-based approach yields a tighter lower bound and better integer solution in equivalent time, we suggest that the approach is still too slow for practical use. For example, although the path-based approach can be solved in roughly a minute for the heterogeneous non-Pareto instance with twenty nodes, the computational time grows exponentially for this approach as the number of nodes increases, taking about two hours to solve the instances with 35 nodes. As seen in Table 4, Columns RPBM-LPR & Pricing

Problem and Branch & Bound for IP Heuristics, neither the LP relaxations of the restricted master nor the ultimate IP itself take long to solve; total run time depends primarily on the time required to solve the DPs at each iteration of the column generation. We therefore consider alternative approaches to solving the DP in the following sections.

Data Set				RPBM-LPR & Pricing Problem	Branch & Bound for IP	Total
Index	Instance	Size	Vehicle Type	time(sec)	time(sec)	time(sec)
1	1	20	Single_Pareto	**27**	**0**	**27**
2	1	20	Multiple_Pareto	**122**	**1**	**123**
-	1-10	20	Multiple_NonPareto	**65**	**0.5**	**65.5**
3	1			49	0	49
4	2			67	1	68
5	3			92	0	92
6	4			101	1	102
7	5			67	0	67
8	6			35	0	35
9	7			46	1	47
10	8			28	0	28
11	9			109	1	110
12	10			56	1	57
13	1	25	Single_Pareto	**47**	**0**	**47**
14	1	25	Multiple_Pareto	**254**	**1**	**255**
-	1-10	25	Multiple_NonPareto	**230.9**	**2**	**232.9**
15	1			158	2	160
16	2			152	2	154
17	3			189	3	192
18	4			280	1	281
19	5			244	1	245
20	6			304	1	305
21	7			237	2	239
22	8			278	2	280
23	9			184	2	186
24	10			283	4	287
25	1	30	Single_Pareto	**345**	**3**	**348**
26	1	30	Multiple_Pareto	**1786**	**12**	**1798**
-	1-10	30	Multiple_NonPareto	**2296.4**	**9**	**2305.6**
27	1			2811	3	2814
28	2			2414	8	2422
29	3			1662	17	1679
30	4			2352	20	2372
31	5			2894	1	2895
32	6			3004	12	3016
33	7			1451	11	1464
34	8			2816	5	2821
35	9			1813	7	1820
36	10			1747	6	1753
37	1	35	Single_Pareto	**1663**	**44**	**1708**
38	1	35	Multiple_Pareto	**7859**	**144**	**8003**
-	1-10	35	Multiple_NonPareto	**8050.6**	**200.9**	**8252.4**
39	1			8250	247	8497
40	2			11136	299	11436
41	3			7796	148	7945
42	4			9338	366	9706
43	5			5803	236	6040
44	6			9547	124	9672
45	7			2669	130	2800
46	8			8268	139	8407
47	9			9945	209	10155
48	10			7754	111	7866

Table 4: Computing Time (ESPPRC)

5.2.2 Relaxed Shortest Path Problem with Resource Constraints

The DP is hard to solve when elementary paths are required because the dominance requirements significantly limit pruning opportunities. As observed by Desrochers et al. [14], removing the no-cycles restriction can greatly reduce run times. Furthermore, even if cycles are allowed, the equality restriction in the cover constraints will prevent such routes from being included in the optimal integer solution [5]. In fact, prior to the work of Feillet et al. [15], most existing research relaxed the no-cycle constraint in order to simplify the pricing problem. For example, Desrochers et al. [14] presents a pseudo-polynomial primal-dual labeling algorithm to effectively solve this problem.

In this section, we consider alternative ways to solve the DP without requiring elementary paths, that is, by allowing cycles. We analyze the impact of this relaxation on both the time required to solve the pricing problem and also the strength of the LP relaxation. We refer to the variation of the pricing problem in which cycles are

allowed as the Relaxed Shortest Path Problem with Resource Constraints (RSPPRC). In our approach to solving RSPPRC, we recognize that when cycles are allowed, it is no longer required that the set of nodes in partial route p_1 be a superset of the nodes in partial route p_2 in order for p_2 to dominate p_1 and thus for p_1 to be pruned. It is trivial to modify our original DP algorithm for solving ESPPRC accordingly, by simply eliminating the third pruning criterion.

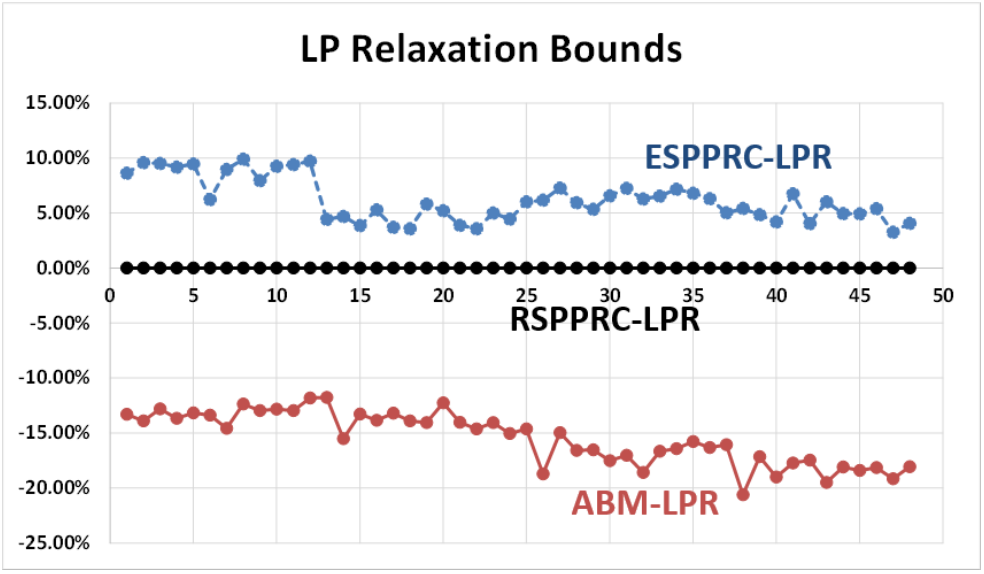
Data Set				ESPPRC	RSPPRC	Δ / ESPPRC
Index	Instance	Size	Vehicle Type	time(sec)	time(sec)	%
1	1	20	Single_Pareto	27	**4**	**-85.19%**
2	1	20	Multiple_Pareto	123	**10**	**-91.87%**
-	1-10	20	Multiple_NonPareto	65.5	**7.5**	**-88.55%**
3	1			49	7	-85.71%
4	2			68	7	-89.71%
5	3			92	8	-91.30%
6	4			102	8	-92.16%
7	5			67	9	-86.57%
8	6			35	6	-82.86%
9	7			47	8	-82.98%
10	8			28	7	-75.00%
11	9			110	7	-93.64%
12	10			57	8	-85.96%
13	1	25	Single_Pareto	47	**6**	**-87.23%**
14	1	25	Multiple_Pareto	255	**14**	**-94.51%**
-	1-10	25	Multiple_NonPareto	232.9	**12**	**-94.85%**
15	1			160	10	-93.75%
16	2			154	13	-91.56%
17	3			192	9	-95.31%
18	4			281	12	-95.73%
19	5			245	13	-94.69%
20	6			305	12	-96.07%
21	7			239	13	-94.56%
22	8			280	12	-95.71%
23	9			186	13	-93.01%
24	10			287	13	-95.47%
25	1	30	Single_Pareto	348	**9**	**-97.41%**
26	1	30	Multiple_Pareto	1798	**23**	**-98.72%**
-	1-10	30	Multiple_NonPareto	2305.6	**21**	**-99.09%**
27	1			2814	24	-99.15%
28	2			2422	20	-99.17%
29	3			1679	21	-98.75%
30	4			2372	20	-99.16%
31	5			2895	25	-99.14%
32	6			3016	21	-99.30%
33	7			1464	21	-98.57%
34	8			2821	21	-99.26%
35	9			1820	20	-98.90%
36	10			1753	17	-99.03%
37	1	35	Single_Pareto	1708	**15**	**-99.12%**
38	1	35	Multiple_Pareto	8003	**48**	**-99.40%**
-	1-10	35	Multiple_NonPareto	8252.4	**43.1**	**-99.48%**
39	1			8497	40	-99.53%
40	2			11436	48	-99.58%
41	3			7945	37	-99.53%
42	4			9706	43	-99.56%
43	5			6040	48	-99.21%
44	6			9672	42	-99.57%
45	7			2800	45	-98.39%
46	8			8407	35	-99.58%
47	9			10155	46	-99.55%
48	10			7866	47	-99.40%

(Δ = [RSPPRC] − [ESPPRC])

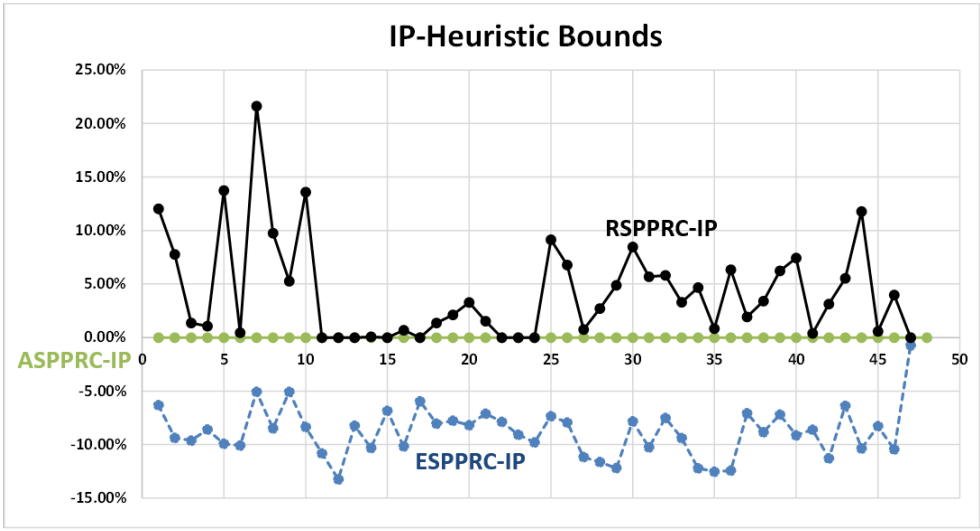
Table 5: Computing Time (RSPPRC)

We conducted computational experiments on the same data as in 5.2.1 to evaluate run time, strength of the LP relaxation, and IP-heuristic solution quality. Figure 2 presents the resulting LP relaxation bounds and objective values for the IP-heuristic approaches to RSPPRC, ASPPRC (Augmented RSPPRC defined later) and ESPPRC. Note that the first column of table 3, 4 and 5 provides a unique index for each instance and x axis in Figure 2 correspond to these indices. We observe that, by simply enabling a greater degree of pruning, we can solve RSPPRC much more quickly than ESPPRC, as demonstrated in Table 5. As seen in Figure 2, however, the LP relaxation is significantly worse relative to when using ESPPRC (although still significantly better than the LP relaxation of the arc-based model, ABM-LPR). In turn, as seen in Figure 2, the quality of the heuristic IP solution (found when branching just on those columns found when solving for the LP relaxation) is worse than the ESPPRC approach as well.

To understand why the LP relaxation when cycles are allowed is worse than ESPPRC, consider the following



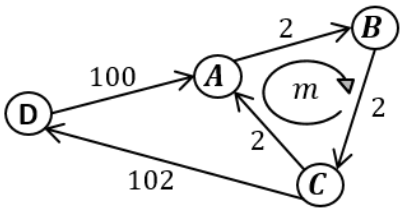
(Percentage difference of ESPPRC-LPR and ABM-LPR based on SPPRC-LPR)



(Percentage difference of RSPPRC-IP and ESPPRC-IP based on ASPPRC-IP)

Figure 2: Solution Quality (RSPPRC)

example, as depicted in Figure 3. The true cost of path $D-A-B-C-D$ is 206 and this path covers nodes A,B, and C. When cycles are allowed, however, we can replace this variable with a variable corresponding to the path $D-A-B-C-A-B-C-D$, with value 1/2. Again, this covers nodes A,B, and C, but now with cost 106. The reduction is because we have reduced the travel distance from the depot to the cluster of nodes. Extending this idea, consider traversing the cycle $A-B-C-A$ m times with corresponding value $x=\frac{1}{m}$. Again, in the LP relaxation, this would satisfy coverage of A,B, and C, but with a cost of $\frac{6m}{m+1}+\frac{206}{m+1}$. As m approaches infinity, the cost approaches six, the cost of the cycle $A-B-C-A$. The only limiting factor here is the total time limit Q allowed for the path.



	Route Cost(c_r)	Value(x)	Total Cost
No cycle ($m=0$)	$D-A-B-C-D$: 100+2+2+102 = 206	1 (A:1, B:1, C:1)	206
One cycle ($m=1$)	$D-(A-B-C)-(A-B-C)-D$: 100+(2+2+2)+2+2+102 = 212	$^1/_2$ (A:1, B:1, C:1)	106
⋮	⋮	⋮	⋮
m cycle	$D-(A-B-C)\cdots(A-B-C)-D$: 100+m × (2+2+2)+2+2+102 = 206+m × 6	$^1/_{(m+1)}$ (A:1, B:1, C:1)	$\frac{6m}{m+1}+\frac{206}{m+1}$

Figure 3: Weak lower bound by the no-cycle constraint relaxation

In addition, the IP heuristic objective is also worse because many of the columns identified when solving PBM-LPR using RSPPRC will not in fact be valid columns for the integer version of the problem because, as motivated in the previous example, many will contain cycles which are invalid under the equality constraints. Since we are not generating columns beyond the root node, this is particularly problematic.

Motivated by this, we augmented the RSPPRC approach to include the acyclic equivalent of each newly-generated column. That is, given a column where some node occurs more than once, we also create a new column in which we delete from the route all occurrences of that node except the first. We refer to this augmented RSPPRC as ASPPRC. Note that this does not impact the LP relaxation of RSPPRC (because these were not chosen to be included in the optimal LP solution), nor does it have any significant impact on the run time over solving RSPPRC. Therefore, ASPPRC is equivalent in run time to RSPPRC. For solution quality, as seen in Figure 2, ASPPRC is always better than RSPPRC (given that it has a superset of the columns of RSPPRC), however, it still has worse objective values than ESPPRC.

5.2.3 ESPPRC with Weak Dominance Rule

In the previous sections, we observed that allowing cycles made the DP approach much faster because we were not limited to pruning only when one path contained a superset of the nodes in another. On the other hand, allowing cycles greatly weakened the lower bound. Motivated by these facts, we propose a final alternative approach in which we do not allow cycles, but we eliminate the third pruning criterion. For example, if path p_1 is less costly and takes less time than path p_2, we allow path p_2 to be pruned, even if it does not contain a superset of the nodes in path p_1. We refer to this ESPPRC with weak dominance rule as WESPPRC. Note that this new approach no longer guarantees an optimal solution to the LP relaxation because we may prune a path that is in fact part of the optimal solution.

We observe, as seen in Table 6, that the result is a significant improvement in run time - in one instance, from more than three hours to roughly 34 seconds. We reiterate that the LP relaxation of WESPPRC is not guaranteed to be optimal (i.e. we may prune relevant columns). As seen in Table 6, however, the impact on the LP relaxation is very small, with a maximum increase of 0.94%. Finally, we note that the IP objective value of WESPPRC is sometimes worse and sometimes better than ESPPRC. However, we observe the difference in objective value to

be within 5.24% in all cases, and often much less. Furthermore, it is as good or better than ASPPRC in all but one of the large problem instances, and faster than that approach in most cases.

Data Set				ESPPRC	WESPPRC	WESPPRC-LPR	WESPPRC-IP		
Index	Instance	Size	Vehicle Type	time(sec)	time(sec)	% Ratio / ESPPRC-LPR	% Ratio / ESPPRC-IP	% Ratio / ASPPRC-IP	Time Ratio / ASPPRC-IP
1	1	20	Single_Pareto	**27**	**3**	**100.20%**	**100.00%**	**93.69%**	**0.43**
2	1	20	Multiple_Pareto	**123**	**5**	**100.00%**	**100.83%**	**91.39%**	**0.45**
-	1-10	20	Multiple_NonPareto	**65.5**	**4.9**	**100.39%**	**100.54%**	**91.53%**	**0.64**
3	1			49	5	100.41%	100.29%	90.65%	0.71
4	2			68	5	100.94%	99.56%	91.00%	0.71
5	3			92	4	100.33%	101.74%	91.65%	0.50
6	4			102	6	100.04%	101.82%	91.55%	0.86
7	5			67	6	100.20%	101.45%	96.31%	0.67
8	6			35	5	100.44%	100.86%	92.31%	0.83
9	7			47	5	100.81%	98.29%	93.32%	0.63
10	8			28	4	100.03%	99.43%	91.12%	0.50
11	9			110	5	100.19%	101.68%	90.68%	0.63
12	10			57	4	100.50%	100.33%	87.05%	0.44
13	1	25	Single_Pareto	**47**	**4**	**100.28%**	**102.61%**	**94.16%**	**0.80**
14	1	25	Multiple_Pareto	**255**	**7**	**100.00%**	**101.11%**	**90.69%**	**0.47**
-	1-10	25	Multiple_NonPareto	**232.9**	**7.3**	**100.21%**	**102.07%**	**93.82%**	**0.59**
15	1			160	10	100.00%	104.31%	97.18%	1.00
16	2			154	6	100.22%	101.24%	90.94%	0.43
17	3			192	8	100.11%	100.94%	94.94%	0.80
18	4			281	7	100.04%	101.06%	92.93%	0.54
19	5			245	7	100.05%	101.12%	93.26%	0.54
20	6			305	7	100.40%	102.11%	93.76%	0.58
21	7			239	7	100.00%	100.71%	93.56%	0.58
22	8			280	8	100.42%	103.50%	95.36%	0.62
23	9			186	6	100.62%	100.33%	91.23%	0.43
24	10			287	7	100.20%	105.21%	94.90%	0.54
25	1	30	Single_Pareto	**348**	**6**	**100.16%**	**99.97%**	**92.64%**	**0.67**
26	1	30	Multiple_Pareto	**1798**	**12**	**100.11%**	**105.24%**	**96.90%**	**0.48**
-	1-10	30	Multiple_NonPareto	**2305.6**	**12.1**	**100.32%**	**101.16%**	**90.29%**	**0.56**
27	1			2814	13	100.00%	101.07%	89.78%	0.57
28	2			2422	10	100.39%	103.21%	91.21%	0.48
29	3			1679	13	100.23%	98.19%	86.21%	0.59
30	4			2372	13	100.73%	98.66%	90.94%	0.59
31	5			2895	10	100.25%	100.25%	89.97%	0.40
32	6			3016	12	100.07%	101.35%	93.74%	0.57
33	7			1464	12	100.32%	97.73%	88.55%	0.55
34	8			2821	13	100.43%	105.24%	92.39%	0.59
35	9			1820	12	100.52%	102.46%	89.63%	0.60
36	10			1753	13	100.25%	103.64%	90.75%	0.76
37	1	35	Single_Pareto	**1708**	**11**	**100.02%**	**101.28%**	**94.12%**	**0.69**
38	1	35	Multiple_Pareto	**8003**	**25**	**100.06%**	**100.40%**	**91.53%**	**0.50**
-	1-10	35	Multiple_NonPareto	**8252.4**	**29.7**	**100.25%**	**100.16%**	**91.83%**	**0.67**
39	1			8497	32	100.11%	99.64%	92.46%	0.76
40	2			11436	34	100.16%	103.93%	94.44%	0.69
41	3			7945	25	100.28%	100.49%	91.85%	0.64
42	4			9706	43	100.05%	100.01%	88.74%	1.00
43	5			6040	28	100.07%	98.02%	91.78%	0.56
44	6			9672	26	100.36%	97.84%	87.72%	0.60
45	7			2800	21	100.83%	100.66%	92.34%	0.45
46	8			8407	30	100.47%	99.67%	89.25%	0.83
47	9			10155	30	100.07%	101.27%	100.54%	0.63
48	10			7866	28	100.16%	100.00%	90.18%	0.57
Max						**100.94%**	**105.24%**	**100.54%**	**1.00**
Min						**100.00%**	**97.73%**	**86.21%**	**0.40**
Average						**100.26%**	**101.05%**	**92.09%**	**0.61**

Table 6: Computing Time & Solution Quality (WESPPRC)

6 Conclusions

In this paper, we present a variation of VRP, TCHVRP, in which arc times and costs vary by vehicle type, and in particular, Pareto dominance is not assumed. We demonstrate the challenges of solving this problem with an arc-based model and consider a path-based model as a viable alternative. In particular, we focus on dynamic programming-based approaches to solving the pricing problem when solving the path-based formulation via column generation. We show that allowing cycles within the routes makes DP approaches to generating routes much faster, but at the expense of a weaker LP relaxation and poorer integer solutions when routes are generated only at the root node. We also show that prohibiting cycles, with fully-defined pruning, is prohibitive slow. As an alternative, we propose a heuristic where cycles are not allowed, but pruning is expanded to allow those cases where set of the nodes covered by one partial route is not necessarily a superset of the nodes covered by another. Although we can no longer guarantee the optimality of the LP relaxation, we show that this markedly improves solution time over the pure case and markedly improves solution quality over the case in which cycles are allowed.

A Problem Data

We generate TCHVRP instances based on the data sets of Golden et al. [25], which is popular benchmark test instances commonly used in the literature. Table 7 gives the characteristics of the TCHVRP instances we have generated. In this table, we give the number N of customers and, for each vehicle type k, the number M_k of vehicles available, the variable cost factor α_k and time factor β_k, error terms for cost ϵ_α and time ϵ_β, and the number of instances. As global parameter settings, we use the constant capacity Q and loading time L over all vehicles. Finally, we generate three different data sets for TCHVRP: single fleet type with Pareto cost, heterogeneous fleet with Pareto cost, and heterogeneous fleet with non-Pareto cost.

For Pareto instances, we have a fixed cost per mile and distance per hour for each vehicle type. For Non-Pareto instances, we generate the cost per mile (α_{ij}) and distance per hour (β_{ij}) from a normal distribution with the means and standard deviations in Table 7. More precisely, the travel cost c_{ij}^k and time t_{ij}^k between customers i and j for vehicles of type k are calculated by $c_{ij}^k = \alpha_{ij}^k d_{ij}$ and $t_{ij}^k = \frac{d_{ij}}{\beta_{ij}^k}$ when the travel is performed by a vehicle of type k. These are based off of a Euclidean distance matrix D_{ij} from the geographical data given in Golden et al. [25]. And then, we generate cost and time matrix such that cost $= \alpha_{ij} \times D_{ij}$ and time $= \frac{D_{ij}}{\beta_{ij}}$ given distance matrix D_{ij}.

For the time resources constraints, we have chosen to use a single value $Q = 8$ hours for all vehicles, and a loading time of $L = 30$ minutes for all customer nodes for all vehicle types. Our choice of Q and L is motivated by the fact that we wanted vehicle tours of approximately 9 to 12 nodes per route.

Finally, the error terms α_k and β_k have been chosen in such a way that no vehicle speed is less than 30 mph or more than 60 mph and cost is between 1.25 and 3.75 dollars per mile.

N	Time & Cost	Vehicle 1	Vehicle 2	Vehicle 3	Instances
20 ~ 35	Single Pareto $(\boldsymbol{\alpha_1}, \boldsymbol{\beta_1})$	M_1: 6(20) 9(25) 12(30,35) $\boldsymbol{\alpha_1}$: 2 , $\boldsymbol{\beta_1}$: 45	None	None	1
	Multiple Pareto $(\boldsymbol{\alpha_t}, \boldsymbol{\beta_t})$	M_1: 2(20) 3(25) 4(30,35) $\boldsymbol{\alpha_1}$: 2.25 , $\boldsymbol{\beta_1}$: 40	M_2 : 2(20) 3(25) 4(30, 35) $\boldsymbol{\alpha_2}$: 2 , $\boldsymbol{\beta_2}$: 45	M_3 : 2(20) 3(25) 4(30, 35) $\boldsymbol{\alpha_3}$: 1.75, $\boldsymbol{\beta_3}$: 50	1
	Multiple Non-Pareto $(\boldsymbol{\alpha_t} + \boldsymbol{\epsilon_\alpha}, \boldsymbol{\beta_t} + \boldsymbol{\epsilon_\beta})$	$\boldsymbol{\epsilon_\alpha}$: N(0,0.25)[1.25,3.75] $\boldsymbol{\epsilon_\beta}$: N(0,7)[30,60]	$\boldsymbol{\epsilon_\alpha}$: N(0,0.25)[1.25,3.75] $\boldsymbol{\epsilon_\beta}$: N(0,7)[30,60]	$\boldsymbol{\epsilon_\alpha}$: N(0,0.25)[1.25,3.75] $\boldsymbol{\epsilon_\beta}$: N(0,7)[30,60]	10

Table 7: Parameters for instances

References

[1] Yogesh Agarwal, Kamlesh Mathur, and Harvey M Salkin. A set-partitioning-based exact algorithm for the vehicle routing problem. *Networks*, 19(7):731–749, 1989.

[2] Roberto Baldacci, Maria Battarra, and Daniele Vigo. Routing a heterogeneous fleet of vehicles. In *The vehicle routing problem: latest advances and new challenges*, pages 3–27. Springer, 2008.

[3] Roberto Baldacci, Nicos Christofides, and Aristide Mingozzi. An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. *Mathematical Programming*, 115(2):351–385, 2008.

[4] Michel L Balinski and Richard E Quandt. On an integer program for a delivery problem. *Operations Research*, 12(2):300–304, 1964.

[5] Alain Chabrier. Vehicle routing problem with elementary shortest path based column generation. *Computers & Operations Research*, 33(10):2972–2990, 2006.

[6] Eunjeong Choi and Dong-Wan Tcha. A column generation approach to the heterogeneous fleet vehicle routing problem. *Computers & Operations Research*, 34(7):2080–2095, 2007.

[7] Nicos Christofides. The vehicle routing problem. *RAIRO-Operations Research-Recherche Opérationnelle*, 10(V1):55–70, 1976.

[8] Nicos Christofides, Aristide Mingozzi, and Paolo Toth. Exact algorithms for the vehicle routing problem, based on spanning tree and shortest path relaxations. *Mathematical programming*, 20(1):255–282, 1981.

[9] Nicos Christofides, Aristide Mingozzi, and Paolo Toth. State-space relaxation procedures for the computation of bounds to routing problems. *Networks*, 11(2):145–164, 1981.

[10] G u Clarke and JW Wright. Scheduling of vehicles from a central depot to a number of delivery points. *Operations research*, 12(4):568–581, 1964.

[11] G Dantzig, R Fulkerson, and S Johnson. Solution of a large-scale traveling-salesman problem. *Operations Research*, 2(4):393–410, 1954.

[12] George B Dantzig and John H Ramser. The truck dispatching problem. *Management science*, 6(1):80–91, 1959.

[13] George B Dantzig and Philip Wolfe. Decomposition principle for linear programs. *Operations research*, 8(1):101–111, 1960.

[14] Martin Desrochers, Jacques Desrosiers, and Marius Solomon. A new optimization algorithm for the vehicle routing problem with time windows. *Operations research*, 40(2):342–354, 1992.

[15] Dominique Feillet, Pierre Dejax, Michel Gendreau, and Cyrille Gueguen. An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks*, 44(3):216–229, 2004.

[16] Brian A Foster and David M Ryan. An integer programming approach to the vehicle scheduling problem. *Operational Research Quarterly*, pages 367–384, 1976.

[17] Ricardo Fukasawa, Humberto Longo, Jens Lysgaard, Marcus Poggi de Aragão, Marcelo Reis, Eduardo Uchoa, and Renato F Werneck. Robust branch-and-cut-and-price for the capacitated vehicle routing problem. *Mathematical programming*, 106(3):491–511, 2006.

[18] WW Garvin, HW Crandall, JB John, and RA Spellman. Applications of linear programming in the oil industry. *Management Science*, 3(4):407–430, 1957.

[19] Bezalel Gavish and Stephen C Graves. The travelling salesman problem and related problems. 1978.

[20] Bezalel Gavish and Stephen C Graves. Scheduling and routing in transportation and distribution systems: formulations and new relaxations. *Management Science (accepted. subject to revision)*, 1982.

[21] Michel Gendreau, Gilbert Laporte, Christophe Musaraganyi, and Éric D Taillard. A tabu search heuristic for the heterogeneous fleet vehicle routing problem. *Computers & Operations Research*, 26(12):1153–1173, 1999.

[22] Filip Gheysens, Bruce Golden, and Arjang Assad. A comparison of techniques for solving the fleet size and mix vehicle routing problem. *Operations-Research-Spektrum*, 6(4):207–216, 1984.

[23] Billy E Gillett and Leland R Miller. A heuristic algorithm for the vehicle-dispatch problem. *Operations research*, 22(2):340–349, 1974.

[24] Fred Glover. Tabu search-part i. *ORSA Journal on computing*, 1(3):190–206, 1989.

[25] Bruce Golden, Arjang Assad, Larry Levy, and Filip Gheysens. The fleet size and mix vehicle routing problem. *Computers & Operations Research*, 11(1):49–66, 1984.

[26] Bruce L Golden, Subramanian Raghavan, and Edward A Wasil. *The Vehicle Routing Problem: Latest Advances and New Challenges: latest advances and new challenges*, volume 43. Springer, 2008.

[27] John H Holland. *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence.* U Michigan Press, 1975.

[28] Scott Kirkpatrick, MP Vecchi, et al. Optimization by simmulated annealing. *science*, 220(4598):671–680, 1983.

[29] Gilbert Laporte. Fifty years of vehicle routing. *Transportation Science*, 43(4):408–416, 2009.

[30] Clair E Miller, Albert W Tucker, and Richard A Zemlin. Integer programming formulation of traveling salesman problems. *Journal of the ACM (JACM)*, 7(4):326–329, 1960.

[31] Yuichi Nagata. Edge assembly crossover for the capacitated vehicle routing problem. In *Evolutionary Computation in Combinatorial Optimization*, pages 142–153. Springer, 2007.

[32] Christian Prins. A grasp $\times$ evolutionary local search hybrid for the vehicle routing problem. In *Bio-inspired algorithms for the vehicle routing problem*, pages 35–53. Springer, 2009.

[33] MR Rao and S Zionts. Allocation of transportation units to alternative trips-a column generation scheme with out-of-kilter subproblems. *Operations Research*, 16(1):52–63, 1968.

[34] Giovanni Righini and Matteo Salani. Dynamic programming algorithms for the elementary shortest path problem with resource constraints. *Electronic Notes in Discrete Mathematics*, 17:247–249, 2004.

[35] Giovanni Righini and Matteo Salani. Symmetry helps: bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. *Discrete Optimization*, 3(3):255–273, 2006.

[36] Giovanni Righini and Matteo Salani. New dynamic programming algorithms for the resource constrained elementary shortest path problem. *Networks*, 51(3):155–170, 2008.

[37] Éric D Taillard and Québec) Centre for Research on Transportation (Montréal. A heuristic column generation method for the heterogeneous fleet vrp. *Operations Research*, 33(1):1–14, 1999.

[38] Paolo Toth and Daniele Vigo. *The vehicle routing problem.* Siam, 2001.