Vehicle Routing Problems with Time Windows and Convex Node Costs

Qie He* Yongjia Song†

August 30, 2016

Abstract

We consider a variant of the vehicle routing problems with time windows, where the objective includes the inconvenience cost modeled by a convex function on each node. We formulate this mixed integer convex program using a novel set partitioning formulation, by considering all combinations of routes and block structures over the routes. We apply a branch-and-price framework to solve this formulation. To solve the pricing problem, we leverage a recursive algorithm that solves the optimal service time scheduling problem given a fixed route, and develop a labeling algorithm with a dominance rule that is easy to check. In our computational study, we show the effectiveness of the labeling algorithm compared to state-of-the-art mixed integer convex programming solvers on an extensive set of vehicle routing problems with time windows and quadratic inconvenience costs.

Keywords: Vehicle routing problem, Branch-and-price, Labeling algorithm, Convex node cost.

1 Introduction

In many practical routing and scheduling problems, the major goal is to provide high-quality services. In the current competitive online retailing and food delivery industry, fast and on-time delivery plays an important role in attracting and retaining customers (Savelsbergh and Van Woensel, 2016; Klapp et al., 2016; Voccia et al., 2015). Another example is the popular ride-hailing services in urban areas provided by companies such as Uber and Lyft, which need to process real-time requests in a timely fashion (Gschwind and Irnich, 2014; Liu et al., 2015). In public transport sectors, service quality also receives increasing attention from practitioners and managers (de Oña and de Oña, 2014). There are many different ways to evaluate the service quality. In situations where a customer is expected to receive the service within a preferred time period, service quality can be modeled by an inconvenience function, transforming the difference between the actual and expected delivery time to the scale of economic cost (Jaw et al., 1986; Toth and Vigo, 2014). In this paper, we study a variant of the vehicle routing problem (VRP) where the objective is to minimize the sum of transportation cost and customer inconvenience costs. In particular, we assume that there is a unique inconvenience function associated with each customer. Instead of assigning a specific form to the inconvenience function, we assume it is a general convex function and its value is zero if the actual service delivery time matches customer's expectation. We call this problem the vehicle routing problem with time windows and convex node costs (VRPTWCNC).

*University of Minnesota-Twin cities, qhe@umn.edu
†Virginia Commonwealth University, ysong3@vcu.edu

The goal of the VRPTWCNC is to select a set of routes and schedules over each route to minimize the weighted sum of transportation cost and inconvenience cost, while respecting the vehicle capacity and time window constraints. When the inconvenience cost is always zero at each customer, the problem reduces to the classical vehicle routing problem with time windows (VRPTW). However, the VRPTWCNC is more challenging to solve than the VRPTW in general. The main reason is that the optimal service start times in the VRPTW can be trivially determined once a route is fixed, since each customer will always be served as early as possible. This is not the case in VRPTWCNC, in which a convex optimization problem needs to be solved to determine the service start times over a fixed route. From a theoretical perspective, the VRPTW is modeled as a mixed-integer linear program (MILP), while the VRPTWCNC can be modeled as a mixed-integer convex program (MICP). In terms of solver capability, although the state-of-the-art MILP solvers are able to routinely solve problems with tens of thousands of integer variables within a reasonable time limit, the development of MICP solvers is significantly lagging behind. The off-the-shelf MICP solvers cannot solve many MICP instances with just a few hundreds of integer variables (Belotti et al., 2013), including the VRPTWCNC instances as shown later. We should also point out that for many vehicle routing problems, the most successful exact algorithms are based on the set partitioning formulation solved in a branch-and-price framework (Desrochers et al., 1992; Barnhart et al., 1998; Desaulniers et al., 1998; Pecin et al., 2016), with customized branching rules and pricing algorithms. As illustrated later in this section, a naive set partitioning formulation of the VRPTWCNC will not lead to an efficient algorithm. A general explanation is that the resource extension function defined for the pricing problem based on the naive set partitioning formulation is not monotonically non-decreasing, which renders inefficient implementations of dominance checking.

The main contribution of this paper is that we propose a non-traditional set partitioning formulation for the VRPTWCNC, and develop an efficient branch-and-price algorithm based on the proposed formulation. To the best of our knowledge, this is the first exact algorithm able to solve large-size VRP with general convex node costs. We test our algorithm on the Solomon instances with convex quadratic node costs. The computational results show that our customized algorithm is much more efficient in solving large-size instances than the state-of-the-art MICP solver.

We now give a detailed description of the VRPTWCNC based on the naive set partitioning formulation. Consider a directed graph $G = (V, A)$ with $V = \{0, 1, \ldots, m\}$ and node 0 being the depot. Let $V_0 = \{1, \ldots, m\}$ denote the set of customers. Let q_i be the demand of customer $i \in V_0$, the vehicle capacity constraint restricts that the total demand carried by a vehicle (i.e., on a single route) cannot exceed the vehicle capacity, denoted by Q. Without loss of generality, we assume a "delivery" version of the vehicle routing problem, i.e., the load of a vehicle will decrease by q_i after serving customer $i \in V_0$. Each customer i is associated with a time window $[l_i, u_i]$ within which the service start time t_i must lie and an inconvenience function $c_i(t_i)$. We assume $c_i(t_i)$ is a general convex function for each $i \in V_0$. Let p_{ij} be the travel time over arc $(i, j) \in A$ and c_{ij} be the cost of traveling on arc $(i, j) \in A$. For simplicity, we do not consider service time or any limit on the number of available vehicles. We assume that a vehicle has to wait if it arrives at a customer earlier than the earliest available service time. Given a route r, let c_r^p denote the total traveling cost of a route r, i.e., $c_r^p = \sum_{(i,j)\in r} c_{ij}$, let c_r^n be the minimum inconvenience cost over route r, and let Ω be the set of all routes that correspond to a finite c_r^n and satisfy the vehicle capacity constraint, a set

partitioning formulation for VRPTWCNC can be written as:

$$\min \sum_{r\in\Omega}(c_r^p + \rho c_r^n) z_r \tag{1a}$$

$$\text{s.t.} \sum_{r\in\Omega} \alpha_{ir} z_r = 1, \ \forall i \in V_0 \tag{1b}$$

$$z_r \in \mathbb{Z}_+, \ \forall r \in \Omega, \tag{1c}$$

where $\alpha_{ir} \in \{0, 1\}$ indicates whether or not route r visits customer i and constraints (1b) stipulate that each customer needs to be visited exactly once. Parameter $\rho \geq 0$ represents a trade-off between the traveling cost and inconvenience cost. The objective function of this form is motivated by the scalarization method for multi-objective optimization problems, as the objective of minimizing the traveling cost and minimizing the inconvenience cost may not be aligned with each other.

The inconvenience cost c_r^n over a fixed route r is calculated as follows. Given a fixed route $r = (r_0, r_1, \ldots, r_n, r_{n+1})$, where $r_0 = r_{n+1} = 0$ and $\{r_1, r_2, \ldots, r_n\} \subseteq V_0$ is the sequence of customers visited on the route, the scheduling problem that aims to minimize the total inconvenience cost can be formulated as the following $(n + 1)$-dimensional convex program:

$$\min \sum_{i=1}^{n} c_{r_i}(t_{r_i}) \tag{2a}$$

$$\text{s.t.} \ t_{r_i} + p_{r_i r_{i+1}} \leq t_{r_{i+1}}, i = 0, 1, \ldots, n \tag{2b}$$

$$t_{r_0} \geq 0, t_{r_i} \in [l_{r_i}, u_{r_i}], i = 1, 2, \ldots, n \tag{2c}$$

$$t_{r_{n+1}} \in [l_0, u_0]. \tag{2d}$$

For any pair of indices i and j, let $p_{i,j}$ be the traveling time between customer r_i and r_j on route r, i.e., the total traveling time from r_i to r_j by visiting the sequence of customers $r_{i+1}, r_{i+2}, \ldots, r_{j-1}$ in between.

In a branch-and-price framework, the linear relaxation of (1) is solved by column generation. Specifically, let μ_i be the dual multiplier associated with (1b) for each $i \in V_0$, with only a subset of columns in Ω appearing in the linear relaxation. Then the pricing problem to generate a new column in (1) is given by:

$$\min_{r\in\Omega} \bar{c}_r := c_r^p + \rho c_r^n - \sum_{i\in V_0} \mu_i \alpha_{ir}. \tag{3}$$

Without the term ρc_r^n, the pricing problem (3) can be formulated as an elementary shortest path problem with resource constraints (ESPPRC) (Irnich and Desaulniers, 2005; Pugliese and Guerriero, 2013), using the general framework introduced in Desaulniers et al. (1998) and Irnich (2008). The ESPPRC is strongly NP-hard. The ESPPRC is usually solved by a labeling algorithm, where a label denotes a path that starts from the depot and ends at some customer. A resource vector associated with each label is carefully defined to keep track of the travel time, remaining vehicle capacity, accumulated cost, etc. over the corresponding path. The resource vector is updated by the so-called resource extension function (Irnich, 2008) whenever a path is extended to a new customer. For the ESPPRC to be solved efficiently, the corresponding resource extension function needs to be monotonically non-decreasing (Desaulniers et al., 1998; Irnich, 2008). There are two issues applying this framework directly to solve the pricing problem (3) in the VRPTWCNC. First, in order to calculate the cost corresponding to a path P with n nodes, one has to solve an n-dimensional convex program (2). Second, the resource extension functions are not monotonically non-decreasing in the VRPTWCNC, since the inconvenience cost at each customer is a general

convex function. Thus it is difficult to check if a path P_1 "dominates" another path P_2 in terms of the possibility of extending to a lower cost route. In Section 3, we will introduce the idea of block structure and derive an alternative set partitioning formulation for VRPTWCNC that gives rise to a practical labeling algorithm for its pricing problem.

The rest of the paper is organized as follows. Section 2 reviews related literature. Section 3 presents a non-traditional set partitioning formulation for the problem and a recursive algorithm that solves (2) over a given route. Section 4 describes the labeling algorithm to solve the pricing problem and the new dominance rule that can be checked efficiently. Section 5 presents some computational results and we conclude in Section 6.

2 Literature review

We first survey some routing and scheduling problems with both time window constraints and node costs. Jaw et al. (1986) studied a multi-vehicle many-to-many dial-a-ride problem with service quality constraints and quadratic inconvenience costs related to the delivery time and ride duration. They developed a sequential insertion heuristic to assign customers to vehicles and to determine a time schedule of pickups and deliveries of each vehicle. Sexton and Bodin (1985a,b) studied a single-vehicle many-to-many dial-a-ride problem with one-sided time window constraints and a linear penalty cost. They solved the problem in a Benders decomposition framework, and used some heuristic to find good solutions for the master routing problem. Dumas et al. (1990) studied the scheduling problem over a fixed route to minimize the total inconvenience costs, where service times at nodes are constrained by time windows and inconvenience is modeled as a general convex function of the service times. This is essentially the subproblem (2) of the VRPTWCNC. The authors developed an algorithm that required solving n unidimensional convex minimization, where n is the number of nodes over the route; they also showed that the algorithm run in $O(n)$ time with quadratic or linear inconvenience cost. Ioachim et al. (1998) studied a shortest path problem with time windows and additional linear costs on the node service start times. This problem arises as a subproblem in decomposition approaches for time-constrained multicommodity network flow formulations for many aircraft routing and scheduling problems (Ioachim et al., 1999) and synchronization in vehicle routing problems (Drexl, 2012). They proposed a dynamic programming algorithm and showed that the algorithm outperformed the discretization approach on instances with wide time windows and many nodes with negative costs. For timing problems that arise in routing and scheduling, please refer to Vidal et al. (2015), which gave a comprehensive and excellent survey on this topic.

Another type of problems that are closely related to our problem are called the vehicle routing problem with soft time windows (VRPSTW) in the literature. In these problems, customers can be served outside the desired time windows and penalty costs will be incurred. Different penalty cost functions are considered in the literature: some only penalize late services, and others penalize both early and late services. The problems may also have additional hard time-window constraints that need to be satisfied. Sexton and Choi (1986) studied a pickup and delivery vehicle routing problem with soft time windows and a linear penalty cost for both early and late services, and proposed a heuristic based on Benders decomposition. Koskosidis et al. (1992) developed an optimization-based heuristic to decompose the problem into an assignment problem and a series of routing and scheduling subproblems. Balakrishnan (1993) proposed three different simple heuristics. Taillard et al. (1997) proposed a tabu search heuristic which also explored an adaptive memory that contains the routes and the best previously visited solutions. Many best-known solutions were reported. Fagerholt (2001) studied the pickup and delivery problem with soft time windows in the

context of maritime transportation. He solved the problem by a set partitioning formulation, in which each column represents a feasible route and the cost of the route is the minimum cost over that route. Chiang and Russell (2004) proposed a tabu search method that utilizes a mixed neighbourhood structure and an advanced recovery procedure to generate high-quality solutions. Ibaraki et al. (2005) proposed a local search algorithm for the VRPSTW in which the piecewise linear penalty cost function is allowed to be non-convex and discontinuous. Calvete et al. (2007) proposed a goal programming model to deal wth the VRPSTW with heterogeneous fleet. Fu et al. (2008) classified the VRPSTW into six types according to allowed time window violations, and presented a unified tabu search framework to solve these problems. Qureshi et al. (2010) considered late arrival penalties in the VRPSTW and developed a branch-and-price algorithm to solve the problem. Figliozzi (2010) proposed a new iterative route construction and improvement algorithm, which improved thirty benchmark problem solutions. Liberatore et al. (2011) developed a branch-cut-and-price exact algorithm for the VRPSTW with piecewise linear penalty cost for both early and late services. They designed a bi-directional dynamic programming approach for the pricing problem and employed dominance rules tailored to the piecewise linear structure of the penalty cost function. They also employed techniques such as decremental state space relaxation and several heuristic algorithms to speed up the pricing procedure. Recently, Mouthuy et al. (2015) developed a multistage variable large-scale neighborhood search method to solve the VRPSTW, in which the first stage is to minimize the total number of vehicles used, the second stage is to minimize the number of violated time windows, and the third stage is to minimize the total routing distance. Their algorithm improved best known solutions on 53% of the most studied instances. We should point out the two exact algorithms developed in Qureshi et al. (2009) and Liberatore et al. (2011) both explored the property of the penalty functions being piecewise linear. In contrast, our focus is to develop a branch-and-price algorithm that deals with general convex functions.

3 A non-traditional set partitioning formulation

In this section, we first introduce a critical concept called *block*, and then present a new set partitioning formulation based on the property of a block.

3.1 A formulation based on a block structure

Given a route $r = (r_1, r_2, \ldots, r_n)$, we define a sequence of customers $B = \{r_b, r_{b+1}, \ldots, r_{b+n_b}\}(1 \leq b \leq b + n_b \leq n)$ as *a block*, if there is no waiting between any of these customers, i.e.,

$$t_{r_{b+i}} = t_{r_{b+i-1}} + p_{r_{b+i-1}}, \ \forall i = 1, 2, \ldots, n_b. \tag{4}$$

Recall that t_i is the service start time at customer i. According to this definition, we can partition the sequence of customers visited by a route into blocks of customers. Given a route r, let $\mathcal{B} = \{B_1, \ldots, B_L\}$ be a sequence of blocks of the customers visited by route r, let $B_\ell(i)$ denote the i-th customer of block B_ℓ, and let $|B_\ell|$ denote the size of block B_ℓ. Then the service start time of a customer in block B_ℓ is completely determined by the service start time of the first customer in that block, i.e., $t_{B_\ell(1)}$. Furthermore, the node cost of each individual block B_ℓ, denoted by $C_{B_\ell}(t_{B_\ell(1)})$, can be calculated as follows:

$$C_{B_\ell}(t_{B_\ell(1)}) = \sum_{i=1}^{|B_\ell|} c_{B_\ell(i)}(t_{B_\ell(1)} + p_{B_\ell(1),B_\ell(i)}). \tag{5}$$

Given a route r and a block structure $\mathcal{B}$, the corresponding optimal node cost $c^n_{r\mathcal{B}}$ can computed by solving the following L-dimensional convex program:

$$c^n_{r\mathcal{B}} := \min \sum_{\ell=1}^{L} C_{B_\ell}(t_{B_\ell(1)}) \tag{6a}$$

$$\text{s.t. } t_{B_\ell(1)} + p_{B_\ell(1),B_{\ell+1}(1)} \le t_{B_{\ell+1}(1)}, \ell = 1,2,\ldots,L-1 \tag{6b}$$

$$t_{B_\ell(1)} + p_{B_\ell(1),B_\ell(i)} \in [l_{B_\ell(i)}, u_{B_\ell(i)}],\ \forall i = 1,2,\ldots,|B_\ell|,\ \forall \ell = 1,2,\ldots,L \tag{6c}$$

$$t_{B_1(1)} \ge p_{0,B_1(1)}, t_{B_L(1)} + p_{B_L(1),n+1} \in [l_0, u_0]. \tag{6d}$$

Let $\Omega^{\mathcal{B}}$ be the set of all combinations of routes and block structures that respect the capacity and admit a schedule satisfying the time window constraints. We propose a non-traditional set partitioning formulation for the VRPTWCNC as follows.

$$\min \sum_{(r,\mathcal{B})\in\Omega^{\mathcal{B}}} (c^p_r + \rho c^n_{r\mathcal{B}}) z_{r\mathcal{B}} \tag{7a}$$

$$\text{s.t. } \sum_{(r,\mathcal{B})\in\Omega^{\mathcal{B}}} \alpha_{ir} z_{r\mathcal{B}} = 1,\ \forall i \in V_0 \tag{7b}$$

$$z_{r\mathcal{B}} \in \mathbb{Z}_+,\ \forall (r,\mathcal{B}) \in \Omega^{\mathcal{B}}. \tag{7c}$$

Let μ_i be the dual multiplier associated with (7b) for each $i \in V_0$ for the linear programming relaxation of (7). The pricing problem to generate a new column in the linear programming relaxation of (7) is given by:

$$\min_{(r,\mathcal{B})\in\Omega^{\mathcal{B}}} \bar{c}_{r\mathcal{B}} := c^p_r + \rho c^n_{r\mathcal{B}} - \sum_{i\in V_0} \mu_i \alpha_{ir}. \tag{8}$$

The standard labeling algorithm can be extended to solve the above pricing problem (8), but the major advantage of our formulation (7) will be shown later after we design a more efficient labeling algorithm based on a recursive algorithm for (6). The general idea of our new labeling algorithm is as follows: when we extend a path that ends with customer i to a new customer j, we either treat j as the start of a new block, or we extend the current block to j. In this way, we implicitly enumerate all possible combinations $(r,\mathcal{B}) \in \Omega^{\mathcal{B}}$. At the first glance, although the optimal node cost $c^n_{r\mathcal{B}}$ of a route r with a block structure $\mathcal{B}$ is easier to solve than (2) (an L-dimensional convex program versus an n-dimensional convex program), the pricing problem (8) seems much harder than (3) since there are exponentially many block structures given the same route r. However, with the special block structure satisfied by the optimal service times (which we show later), the L-dimensional convex program (6) can be solved by recursively solving a one-dimensional convex program L times, which is much more efficient. More importantly, this result enables *separability* between blocks, and is crucial for efficient label updating and dominance rule checking in our proposed labeling algorithm.

3.2 A recursive algorithm for the optimal timing problem (6)

Our goal in this section is to present a recursive algorithm that solves the optimal timing problem (6) with a block structure $\mathcal{B} = \{B_1, B_2, \ldots, B_L\}$ in a greedy fashion. This is the foundation of our new labeling algorithm. We define the following one-dimensional convex program recursively:

$$(\mathcal{P}_\ell) \quad c_{rB_\ell} = \min_{t_{B_\ell(1)}} C_{B_\ell}(t_{B_\ell(1)}) \tag{9a}$$

$$\text{s.t. } t_{B_\ell(1)} \ge \hat{t}_{B_{\ell-1}(1)} + p_{B_{\ell-1}(1),B_\ell(1)} \tag{9b}$$

$$t_{B_\ell(1)} + p_{B_\ell(1),B_\ell(i)} \in [l_{B_\ell(i)}, u_{B_\ell(i)}],\ \forall i = 1,2,\ldots,|B_\ell|, \tag{9c}$$

where $\hat{t}_{B_{\ell-1}(1)}$ is the smallest optimal solution of $(\mathcal{P}_{\ell-1})$, $\forall \ell = 2, 3, \ldots, L$. We set $\hat{t}_{B_0(1)} := 0$ and $p_{B_0(1),B_1(1)} := p_{0,B_1(1)}$ in the first problem $\mathcal{P}_1$.

Given a route r associated with a block structure $\mathcal{B}$, we solve the problem $(\mathcal{P}_\ell)$ recursively for $\ell = 1, 2, \ldots, L$ and define a quantity $\check{c}^n_{r\mathcal{B}}$ in the following way: if $(\mathcal{P}_\ell)$ is infeasible for any $\ell = 1, 2, \ldots, L$, $\check{c}^n_{r\mathcal{B}} = \infty$; otherwise, $\check{c}^n_{r\mathcal{B}} = \sum_{\ell=1}^{L} c_{rB_\ell}$. It is clear that $\check{c}^n_{r\mathcal{B}}$ is an upper bound for $c^n_{r\mathcal{B}}$, since when $(\mathcal{P}_\ell)$ is feasible for all $\ell = 1, 2, \ldots, L$, the corresponding service start time vector gives a feasible solution to (6). We next show that when the block structure $\mathcal{B}$ is induced from an optimal solution t^* of (2), this recursive approach gives the optimal node cost $c^n_{r\mathcal{B}}$ defined in (6).

Proposition 1. *Given a fixed route r, let t^* be an optimal solution to (2) and z^* be the corresponding optimal objective value. Suppose t^* induces a block structure $\mathcal{B} = \{B_1, B_2, \ldots, B_L\}$ over route r, then t^* can be recovered by recursively solving (9) for $\ell = 1, \ldots, L$, and*

$$z^* = c^n_{r\mathcal{B}} = \check{c}^n_{r\mathcal{B}},$$

i.e., the optimal objective value z^* is recovered by recursively solving (9) L times and summing up their optimal objective values.

Proof. Our proof is based on induction on ℓ. We first show that $t^*_{B_1(1)}$ is an optimal solution of $(\mathcal{P}_1)$. In fact, if this does not hold, since $c_i(t_i)$'s are convex functions, $t^*_{B_1(1)}$ is not a local optimal solution of $(\mathcal{P}_1)$, i.e., there exists a solution $t^\epsilon_{B_1(1)}$ in an ϵ-neighborhood of $t^*_{B_1(1)}$ that yields a lower cost, $\forall \epsilon > 0$. According to the definition of a block, $t^*_{B_1(1)} + p_{B_1(1),B_2(1)} < t^*_{B_2(1)}$. Let $t^*_{B_\ell(1)}$ be the same for all $\ell = 2, 3, \ldots, L$, let ϵ be small enough: $\epsilon < t^*_{B_2(1)} - (t^*_{B_1(1)} + p_{B_1(1),B_2(1)})$, and replace $t^*_{B_1(1)}$ with $t^\epsilon_{B_1(1)}$, this new solution is feasible to (2) and yields a lower cost, which contradicts the fact that t^* is an optimal solution.

Assume $t^*_{B_k(1)}$ is an optimal solution of $(\mathcal{P}_k)$ for all $k = 1, 2, \ldots, \ell - 1$, we now consider $t^*_{B_\ell(1)}$. First, according to (2) and the definition of a block, $t^*_{B_\ell(1)} > t^*_{B_{\ell-1}(1)} + p_{B_{\ell-1}(1),B_\ell(1)}$, $t^*_{B_\ell(1)}$ is feasible to $(\mathcal{P}_\ell)$. If $t^*_{B_\ell(1)}$ is not optimal, using a similar argument as the case above, there exists a solution $t^\epsilon_{B_\ell(1)}$ in an ϵ-neighborhood of $t^*_{B_\ell(1)}$ that yields a lower cost, $\forall \epsilon > 0$. Let $t^*_{B_k(1)}$ be the same for all $k = 1, 2, \ldots, \ell - 1, \ell + 1, \ldots, L$, let ϵ be small enough that $\epsilon < t^*_{B_{\ell+1}(1)} - (t^*_{B_\ell(1)} + p_{B_\ell(1),B_{\ell+1}(1)})$ and $\epsilon < t^*_{B_\ell(1)} - (t^*_{B_{\ell-1}(1)} + p_{B_{\ell-1}(1),B_\ell(1)})$, and replace $t^*_{B_\ell(1)}$ with $t^\epsilon_{B_\ell(1)}$, this new solution is feasible to (2) and yields a lower cost, which contradicts the fact that t^* is an optimal solution. By induction, we obtain our conclusion. □

Proposition 1 serves as a basis for our proposed labeling algorithm. Our idea is to use the optimal node cost calculated by this recursive approach, $\check{c}^n_{r\mathcal{B}}$, instead of the true optimal cost $c^n_{r\mathcal{B}}$ in the set partitioning formulation (7). In other words, the formulation (7) is equivalent to the following set partitioning formulation.

$$\min \sum_{(r,\mathcal{B})\in\Omega^{\mathcal{B}}} (c^p_r + \rho \check{c}^n_{r\mathcal{B}}) z_{r\mathcal{B}} \tag{10a}$$

$$\text{s.t.} \sum_{(r,\mathcal{B})\in\Omega^{\mathcal{B}}} \alpha_{ir} z_{r\mathcal{B}} = 1, \ \forall i \in V_0 \tag{10b}$$

$$z_{r\mathcal{B}} \in \mathbb{Z}_+, \ \forall (r,\mathcal{B}) \in \Omega^{\mathcal{B}}. \tag{10c}$$

Proposition 2. *Formulation (10) is a valid formulation for VRPTWCNC.*

Proof. Let r^* be any route in an optimal solution of (7) and $\mathcal{B}^*$ be the corresponding optimal block structure in (7). Note that in (10), all possible combinations between route and block structures $(r, \mathcal{B})$ are enumerated. Therefore, although the recursive approach does not necessarily give the optimal node cost for every combination $c^n_{r,\mathcal{B}}$, $\forall (r, \mathcal{B}) \in \Omega^{\mathcal{B}}$, it gives the optimal node cost $c^n_{r^*\mathcal{B}^*} = \check{c}^n_{r^*\mathcal{B}^*}$ for the optimal combination $(r^*, \mathcal{B}^*)$, and gives an upper bound for others. This guarantees that (10) is a valid reformulation of (7). □

The pricing problem for (10) is the same as (8) except that $c^n_{r\mathcal{B}}$ is replaced by $\check{c}^n_{r\mathcal{B}}$. The recursive approach that calculates $\check{c}^n_{r\mathcal{B}}$ naturally enables separability in the labeling algorithm. It enables efficient dominance checking between paths by only solving one dimensional convex optimization problems instead of any higher dimensional convex optimization problems. We sketch the idea of our proposed labeling algorithm below, and formally describe the label definition, extension and dominance in Section 4. Roughly speaking, a label consists of a path $\bar{r}$ and a block structure $\bar{\mathcal{B}}$ associated with this path. When we extend this label $(\bar{r}, \bar{\mathcal{B}})$ ending with customer i, to a new customer j, we either treat j as the starting node of a new block, or we extend the current block to j. In the first case, since the current block ends, according to the recursive approach, we compute the optimal cost of this block, along with the optimal service start time $\hat{t}_i$ at customer i (the smallest optimal $\hat{t}_i$ is chosen when multiple optimal solutions exist). This optimal service start time induces a lower bound for the service start time t_j at customer j that $t_j \geq t_i + p_{ij}$, in addition to the time window constraints that need to be satisfied by t_j. In the second case, the service start time t_j satisfies $t_j = t_{B_1} + p_{B_1,j}$, where B_1 is the starting node of the current block, whose service start time t_{B_1} will be determined when the current label extends to a new block of customers or back to the depot. To make the labeling algorithm practical in solving the pricing problem, the key is to develop an effective dominance rule so that many labels can be dominated (and therefore be removed). We explain the label definition, initialization, extension and termination in more detail in the next section, along with a dominance rule.

4 The Labeling Algorithm

4.1 Label definition, initialization, extension and termination

Let $P = (i_0, i_1, \ldots, i_h)$ be a path from $i_0 = 0$ over a subset of customers. A label $\mathcal{L}$ is associated with a path P and contains the following attributes:

- w: the index of the first node i_w of the last block on P. For example, if $P = \{i_1, \ldots, i_5\}$ and the last block on P is i_3, i_4, i_5, then $w = 3$.
- W: set of forbidden nodes. When each route r is an elementary route, W is the set of nodes visited so far by P; when each route r is a q-route, W is just a singleton $\{i_h\}$.
- K: the remaining capacity of the vehicle.
- C^p: total arc cost along P up to node i_h.
- C^n: total node cost up to node i_w along P.
- $C(t_{i_w}) = \sum_{k=w}^{h} c_{i_k}(t_{i_w} + p_{i_w,i_k})$: total cost from node i_w to i_h as a function of t_{i_w}.
- S: set of feasible service start time t_{i_w} at node i_w such that all customers from i_w to i_h along P can be served within their corresponding time windows.

- $\mu(P)$: the sum of dual multipliers along P, i.e., $\mu(P) = \sum_{k=1}^{h} \mu_{i_k}$.

The initial label $\mathcal{L}_0$ is associated with a path $P^0 = (0)$. The corresponding attributes are initialized as $w = 0$, $W = \emptyset$, $C^p = C^n = K = 0$, the function C is set as zero everywhere, and $S = \mathbb{R}_+$. When a label $\mathcal{L}$ with $P = (i_0, i_1, \ldots, i_h)$ is extended to a new node $j = i_{h+1}$, we create two labels depending on whether the new node belongs to the last block of path P, or it is the starting node of a new block. Some attributes in the two labels are updated in the same way:

- For elementary routes, $W = W \cup \{j\}$; for q-routes, $W = \{j\}$.
- $K = K + d_j$.
- $C^p = C^p + c_{i_h,j}$.

Attributes w, $C^n, C(t_{i_w})$, and S will be updated differently for the two labels:

- For the first label $\mathcal{L}_1$ in which node j stays in the last block of path P,
  - w stays unchanged.
  - C^n stays unchanged.
  - $C(t_{i_w}) = C(t_{i_w}) + c_j(t_{i_w} + p_{i_w,j})$.
  - $S = S \cap \{t_{i_w} \mid t_{i_w} + p_{i_w,j} \in [l_j, u_j]\}$.
- For the second label $\mathcal{L}_2$ in which node j belongs to a new block,
  - $w = h + 1$.
  - $C^n = C^n + C^*_{i_w}$, where $C^*_{i_w}$ is obtained by solving $\min\{C(t_{i_w}) \mid t_{i_w} \in S\}$.
  - $C(t_{i_w}) = c_{i_w}(t_{i_w})$.
  - $S = \{t_{i_{h+1}} \in [l_{i_{h+1}}, u_{i_{h+1}}] \mid t_{i_{h+1}} \geq t^*_{i_w} + p_{i_w,i_{h+1}}\}$, where $t^*_{i_w}$ be the smallest optimal solution corresponding to $C^*_{i_w}$.

During a label extension, a label is discarded if set S becomes an empty set or $K > Q$. Label termination means extending a label back to the depot. We assume that the final arrival time at the depot needs to satisfy a time window constraint, but there is no cost associated with it.

4.2 Dominance rule

A dominance rule is crucial for a labeling algorithm to work effectively in practice. A dominated label can be discarded along with all its extensions, which helps to prevent enumerating all (exponentially many) labels. Given a path $P = (i_1, \ldots, i_h)$ and a path $P' = (j_1, \ldots, j_k)$ in G, we use $P \oplus P'$ to denote the path $(i_1, \ldots, i_h, j_1, \ldots, j_k)$. Given a label L associated with a path P, we call P' as a feasible extension of $\mathcal{L}$, if

- $P \oplus P'$ is a complete route;
- The total demand on $P \oplus P'$ does not exceed the vehicle capacity;
- $P \oplus P'$ admits a feasible schedule of service start times such that the customers from i_w to i_h belong to the same block and $t_{i_w} \in S$.

Let $E(L)$ be the set of all feasible extensions of label L.

Definition 1. *A label $\mathcal{L}_2$ is a* dominated *label (so it can be discarded), if there exists a label $\mathcal{L}_1$ such that the following conditions hold:*

1. *The path P_1 associated with $\mathcal{L}_1$ ends at the same customer as the path P_2 associated with $\mathcal{L}_2$.*
2. *$E(\mathcal{L}_2) \subseteq E(\mathcal{L}_1)$.*
3. *For any $P' \in E(\mathcal{L}_2)$, there exists a set of service start times over $P_1 \oplus P'$ that satisfy all the time window constraints, and its corresponding total cost over $P_1 \oplus P'$ is no larger than the optimal cost over $P_2 \oplus P'$.*

Dominance according to Definition 1 is difficult to check. We will replace the conditions on Definition 1 by a set of sufficient conditions that are easier to check.

Proposition 3. *A label $\mathcal{L}_2$ is dominated, if there exists a label $\mathcal{L}_1$ such that the following conditions hold:*

- *The last node of P_1, $i^1_{h^1}$, is the same as the last node of P_2, $i^2_{h^2}$, i.e., $i^1_{h^1} = i^2_{h^2} = i$.*
- *$W_1 \subseteq W_2, K_1 \leq K_2$.*
- *For any $t_2 \in S_2$, there exists $t_1 \in S_1$ such that $t_1 + p_{i_{w^1},i} \leq t_2 + p_{i_{w^2},i}$, i.e., vehicle can arrive at node i through P_1 no later than through P_2, and*

$$D_1(t_1) - \mu(P_1) < D_2(t_2) - \mu(P_2), \tag{11}$$

where $D_1(t_1)$ and $D_2(t_2)$ are the total costs (including traveling cost and node cost) of $\mathcal{L}_1$ and $\mathcal{L}_2$ up to node $i^1_{h^1-1}$ and node $i^2_{h^2-1}$, respectively, and $\mu(P_1)$ and $\mu(P_2)$ are the summations of dual multipliers along the path P_1 and P_2, respectively.

Proof. The first two conditions guarantee that condition 1 in Definition 1 is satisfied. To see that condition 2 in Definition 1 is also satisfied, we consider all possible feasible extensions $P' \in E(\mathcal{L}_2)$. We then show that the third condition guarantees that the total cost over $P_1 \oplus P'$ is less than $P_2 \oplus P'$. In fact, the third condition here allows us to set the service start time at customer $i^1_{h^1}$ for $\mathcal{L}_1$ to be the same as the service start time at customer $i^2_{h^2}$ for $\mathcal{L}_2$ (note that $i^1_{h^1} = i^2_{h^2} = i$). In this case, the total cost from i back to the depot over P' is the same for $P_1 \oplus P'$ and $P_2 \oplus P'$, so we only need to compare the total cost up to $i^1_{h^1-1}$ for P_1 and the total cost up to $i^2_{h^2-1}$ for P_2. Equation (11) ensures that the former is smaller. □

The first two conditions for dominance can be verified easily. We next focus on the third condition. Assume $S_1 = [L_1, U_1]$, and $S_2 = [L_2, U_2]$. To check the third condition for dominance, we first check if for any $t_2 \in S_2$, there exists $t_1 \in S_1$ such that $t_1 + p_{i_{w^1},i} \leq t_2 + p_{i_{w^2},i}$. This can be done by checking if $L_1 + p_{i_{w^1},i} \leq L_2 + p_{i_{w^2},i}$. If this does not hold, then we conclude with no dominance. Otherwise, we check whether (11) holds. Consider the following four cases:

- Case 1: $w^1 = h^1, w^2 = h^2$, so that $i^1_{w^1} = i^2_{w^2} = i$, i.e., the last node of both labels, node i, is the starting node of a new block in both labels. In this case, $D^1(t_1) = C^p_1 + \rho C^n_1$, $D^2(t_2) = C^p_2 + \rho C^n_2$.
- Case 2: $w^1 < h^1, w^2 = h^2$. In this case, $D^1(t_1) = C^p_1 + \rho\left(C^n_1 + \sum_{j=w^1}^{h^1-1} c_{i_j}(t_1 + p_{i_{w^1},i_j})\right), t_1 \in S^1$, and $D^2(t_2) = C^p_2 + \rho C^n_2$.

- Case 3: $w^1 = h^1, w^2 < h^2$. In this case, $D^2(t_2) = C_2^p + \rho\left(C_2^n + \sum_{j=w^2}^{h^2-1} c_{i_j}(t_2 + p_{i_{w^2},i_j})\right), t_2 \in S^2$, and $D^1(t_1) = C_1^p + \rho C_1^n$.
- Case 4: $w^1 < h^1, w^2 < h^2$. In this case, $D^1(t_1) = C_1^p + \rho\left(C_1^n + \sum_{j=w^1}^{h^1-1} c_{i_j}(t_1 + p_{i_{w^1},i_j})\right), t_1 \in S^1$, and $D^2(t_2) = C_2^p + \rho\left(C_2^n + \sum_{j=w^2}^{h^2-1} c_{i_j}(t_2 + p_{i_{w^2},i_j})\right), t_2 \in S^2$.

Fix $t_2 \in S^2$. Let $S^1(t_2) = \{t_1 \mid 0 \le t_1 \le t_2 + p_{i_{w^2},i} - p_{i_{w^1},i}\} \cap S^1$. We consider the four cases in more detail:

1. Case 1: Checking dominance can just be done by checking if $C_1^p + \rho C_1^n - \mu(P^1) < C_1^p + \rho C_1^n - \mu(P^2)$.
2. Case 2: Given a fixed $t_2 \in S^2$, checking if there exists $t_1 \in S^1(t_2)$ such that

$$C_1^p + \rho\left(C_1^n + \sum_{j=w^1}^{h^1-1} c_{i_j}(t_1 + p_{i_{w^1},i_j})\right) - \mu(P^1) < C_2^p + \rho C_2^n - \mu(P^2),$$

   can be done by just checking if

$$C_1^p + \rho\left(C_1^n + \min_{t_1 \in S^1(t_2)} \sum_{j=w^1}^{h^1-1} c_{i_j}(t_1 + p_{i_{w^1},i_j})\right) - \mu(P^1) < C_2^p + \rho C_2^n - \mu(P^2).$$

   Let $F(t_2) = \min_{t_1 \in S^1(t_2)} \sum_{j=w^1}^{h^1-1} c_{i_j}(t_1 + p_{i_{w^1},i_j})$, the dominance can be verified by checking if

$$\rho \cdot \max_{t_2 \in S^2} F(t_2) < (C_2^p + \rho C_2^n) - \mu(P^2) - (C_1^p + \rho C_1^n) + \mu(P^1).$$

   Notice that the set $S^1(t_2)$ expands when t_2 increases, so $F(t_2)$ is a nonincreasing function and $\max_{t_2 \in S^2} F(t_2)$ is achieved at $t_2 = L_2$, we can just check if

$$\rho \cdot F(L_2) < (C_2^p + \rho C_2^n) - \mu(P^2) - (C_1^p + \rho C_1^n) + \mu(P^1).$$

   Calculating $F(L_2)$ involves solving a univariate convex function.
3. Case 3: The dominance rule can be checked by checking if $\forall t_2 \in S^2$,

$$C_1^p + \rho C_1^n - \mu(P^1) < C_1^p + \rho\left(C_1^n + \sum_{j=w^2}^{h^2-1} c_{i_j}(t_2 + p_{i_{w^2},i_j})\right) - \mu(P^2),$$

   which can be done by checking if

$$\rho \cdot \min_{t_2 \in S^2} \sum_{j=w^2}^{h^2-1} c_{i_j}(t_2 + p_{i_{w^2},i_j}) > (C_1^p + \rho C_1^n) - \mu(P^1) - (C_2^p + \rho C_2^n) + \mu(P^2).$$

   This again involves minimizing a univariate convex function.

4. Case 4: checking dominance rule in this case is more complicated that the above three cases. Using similar argument in Case 2, we check if

$$\rho \cdot \max_{t_2 \in S^2} \left[F(t_2) - \sum_{j=w^2}^{h^2-1} c_{i_j}(t_2 + p_{i_{w^2},i_j}) \right] < (C_2^p + \rho C_2^n) + \mu(P^1) - \mu(P^2) - (C_1^p + \rho C_1^n). \quad (12)$$

Unlike Case 2, function $F(t_2) - \sum_{j=w^2}^{h^2-1} c_{i_j}(t_2 + p_{i_{w^2},i_j})$ may not be monotone or even convex, we discuss the following three cases for checking (12).

- If $t_2 + p_{i_{w^2},i} - p_{i_{w^1},i} \geq U_1$, $S^1(t_2) = [L_1, U_1]$, so $F(t_2)$ is a constant, i.e.,

$$F(t_2) = \min_{t_1 \in [L_1,U_1]} \sum_{j=w^1}^{h^1-1} c_{i_j}(t_1 + p_{i_{w^1},i_j}).$$

  Therefore, checking (12) can be done by minimizing the convex function $\sum_{j=w^2}^{h^2-1} c_{i_j}(t_2 + p_{i_{w^2},i_j})$.
- Otherwise,

$$F(t_2) = \min_{t_1 \in [L_1, t_2 + p_{i_{w^2},i} - p_{i_{w^1},i}]} \sum_{j=w^1}^{h^1-1} c_{i_j}(t_1 + p_{i_{w^1},i_j})$$

  Let $t^*_{\min} = \min\{\arg\min \sum_{j=w^1}^{h^1-1} c_{i_j}(t + p_{i_{w^1},i_j})\}$, and $t^*_{\max} = \max\{\arg\min \sum_{j=w^1}^{h^1-1} c_{i_j}(t + p_{i_{w^1},i_j})\}$, i.e., they are the smallest and biggest minimizer of function $\sum_{j=w^1}^{h^1-1} c_{i_j}(t+p_{i_{w^1},i_j})$ over the real line, respectively.
  - If $t^*_{\max} \leq L_1$, $F(t_2)$ is a constant:

$$F(t_2) = \sum_{j=w^1}^{h^1-1} c_{i_j}(L_1 + p_{i_{w^1},i_j}),$$

    so checking (12) can again be done by minimizing the convex function $\sum_{j=w^2}^{h^2-1} c_{i_j}(t_2 + p_{i_{w^2},i_j})$ for $t_2 \in S^2$.
  - Otherwise, $F(t_2)$ is a piecewise convex function with two pieces. If $t_2 + p_{i_{w^2},i} - p_{i_{w^1},i} \geq t^*_{\min}$,

$$F(t_2) = \sum_{j=w^1}^{h^1-1} c_{i_j}(t^*_{\min} + p_{i_{w^1},i_j}),$$

    which is a constant. Otherwise,

$$F(t_2) = \sum_{j=w^1}^{h^1-1} c_{i_j}(t_2 + p_{i_{w^2},i} - p_{i_{w^1},i} + p_{i_{w^1},i_j}),$$

    which is a convex function. In this case, checking (12) requires maximizing the difference of two convex functions. In general, this can be done using methodology known as Difference-of-Convex programming (Tao and An, 1997). In our case, since the functions involved are univariate functions, the maximum of a smooth function over a closed interval is achieved at either stationary points or the endpoints of the interval, and the maximum of a piecewise linear function is achieved at either break points, stationary points, or endpoints of the interval.

In our computational experiments, we use a quadratic function as the cost function for each node i in the form $c_i(t_i) = b_i(t_i - a_i)^2$. This significantly reduces the amount of information that needs to be stored with each label. In particular, the cost function

$$\begin{aligned} C(t_{i_w}) &= \sum_{k=w}^{h} c_{i_k}(t_{i_w} + p_{i_w,i_k}) \\ &= (\sum_{k=w}^{h} b_{i_k}) \left[t_{i_w} + \frac{\sum_{k=w}^{h} b_{i_k}(p_{i_w,i_k} - a_{i_k})}{\sum_{k=w}^{h} b_{i_k}} \right]^2 + \sum_{k=w}^{h} b_{i_k}(p_{i_w,i_k} - a_{i_k})^2 \\ &- \frac{[\sum_{k=w}^{h} b_{i_k}(p_{i_w,i_k} - a_{i_k})]^2}{\sum_{k=w}^{h} b_{i_k}}. \end{aligned}$$

Let $R_1 = \sum_{k=w}^{h} b_{i_k}, R_2 = \sum_{k=w}^{h} b_{i_k}(p_{i_w,i_k} - a_{i_k}), R_3 = \sum_{k=w}^{h} b_{i_k}(p_{i_w,i_k} - a_{i_k})^2$, then $C(t_{i_w})$ becomes:

$$C(t_{i_w}) = R_1(t_{i_w} + \frac{R_2}{R_1})^2 + R_3 - \frac{R_2^2}{R_1}.$$

Therefore, we just need to update these three parameters R_1, R_2 and R_3 in a label extension.

5 Computational Experiments

We present our computational experiments to test the performance of the branch-and-price algorithm using the proposed labeling algorithm for VRPTWCNC on instances with a quadratic node cost $c_i(t_i) = b_i(t_i - a_i)^2$ for each node $i \neq 0$. We do not impose any limit on the number of vehicles in the model. We implement our branch-and-price algorithm in C++ using SCIP Optimization Suite 3.2.0 with CPLEX 12.6 as the linear programming solver. We use the following set partitioning formulation for VRPTWCNC by introducing a binary variable x_a for each arc $a \in A$ to indicate whether a is traversed by any route.

$$\min \sum_{(r,\mathcal{B})\in\Omega} (c_r^p + \rho \check{c}_{r\mathcal{B}}) z_{r\mathcal{B}} \tag{13a}$$

$$\text{s.t.} \sum_{(r,\mathcal{B})\in\Omega} \alpha_{ir} z_{r\mathcal{B}} = 1, \ \forall i \in V_0 \tag{13b}$$

$$x_a \geq \sum_{(r,\mathcal{B})\in\Omega} e_{ar} z_{r\mathcal{B}}, \ \forall a \in A \tag{13c}$$

$$\sum_{a\in\delta^+(i)} x_a = 1, \ \forall i \in V_0 \tag{13d}$$

$$\sum_{a\in\delta^-(i)} x_a = 1, \ \forall i \in V_0 \tag{13e}$$

$$z_{r\mathcal{B}} \in \mathbb{R}_+, \ \forall (r,\mathcal{B}) \in \Omega \tag{13f}$$

$$x_a \in \{0,1\}, \ \forall a \in A, \tag{13g}$$

where α_{ir} indicates if route r goes through node i, e_{ar} indicates if route r goes through arc a, and $\delta^+(i)$ ($\delta^-(i)$) denotes the set of arcs leaving (entering) customer i. We use this formulation so that branching will be performed on the arc variables x_a, which will not affect the pricing problem (8). To ensure that (13) is always feasible, we add simple routes $(0, i, 0)$ with two different block

structures, $\{\{0\},\{i\}\}$, and $\{\{0,i\}\}$, for each $i \in V_0$ initially to the formulation. In our experiments, the pricing problem is solved using the proposed labeling algorithm over elementary routes.

Before we present our computational results, we note that our implementation of the branch-and-price algorithm is indeed preliminary. There are many computational enhancements that can be done to further improve the results. For example, one can: (i) incorporate valid inequalities in the x space such as rounded capacity cuts (Lysgaard, 2003), making it a branch-price-and-cut algorithm; (ii) price over more complicated route structures such as q-routes or ng-routes (Baldacci et al., 2011) rather than elementary routes; (iii) consider a bi-directional labeling algorithm (Righini and Salani, 2008) to solve the pricing problem. The purpose of showing the computational results here is simply to illustrate the effectiveness of the proposed new set partitioning formulation, the labeling algorithm, and the associated dominance rule. These computational results are by no means the best possible results that one could obtain after even a moderate effort of algorithmic fine tuning.

We compare the computational results of the proposed branch-and-price algorithm with the following standard flow-based two-index formulation solved by an MIQP solver in CPLEX 12.6.

$$\min \sum_{(i,j)\in A} c_{ij}x_{ij} + \rho \sum_{i\in V_0} c_i(t_i) \tag{14a}$$

$$\text{s.t.} \sum_{j\in\delta^+(i)} x_{ij} = 1,\ \forall i \in V_0, \tag{14b}$$

$$\sum_{j\in\delta^-(i)} x_{ji} = 1,\ \forall i \in V_0 \tag{14c}$$

$$\sum_{j\in\delta^-(i)} f_{ji} - \sum_{j\in\delta^+(i)} f_{ij} = q_i,\ \forall i \in V_0 \tag{14d}$$

$$q_j x_{ij} \le f_{ij} \le (Q - q_i)x_{ij},\ \forall (i,j) \in A \tag{14e}$$

$$x_{ij} \in \{0,1\}, f_{ij} \ge 0,\ \forall (i,j) \in A \tag{14f}$$

$$t_j - t_i - p_{ij} \ge -M_{ij}(1 - x_{ij}),\ \forall i,j \in V_0, i \ne j \tag{14g}$$

$$l_i \le t_i \le u_i,\ \forall i \in V_0 \tag{14h}$$

$$s_i - t_i - p_{i0} \ge -M_{i0}(1 - x_{i0}),\ \forall i \in V_0 \tag{14i}$$

$$0 \le s_i \le b_0,\ \forall i \in V_0. \tag{14j}$$

where $M_{ij} := p_{ij} + u_i - l_j$, and $M_{i0} := p_{i0} + u_i$.

We use test instances proposed by Solomon (1987) as the basis for the VRPTW instances. We set the cost of traveling an arc as its traveling time, i.e., $c_{ij} = p_{ij},\ \forall (i,j) \in A$. For the node cost function $c_i(t_i)$, we set $a_i = \frac{1}{2}(l_i + u_i)$ and $b_i = 1$ for each $i \ne 0$. We consider three cases for ρ: $\rho = 0.2, 1, 5$, which correspond to low penalty, medium penalty, and large penalty, respectively. Solomon's instances are organized into six groups according to the geographical locations of the customers: Random (R), Clustered (C), and Randomly Clustered (RC), and the time window ranges: tighter (1) or wider (2). In all our tests, we set the time limit to two hours (7200 seconds).

Table 1 summarizes the computational performance of the proposed branch-and-price algorithm for solving the set partitioning formulation (13) and the two-index flow-based formulation (14). For each group of instances, we show the number of instances solved within the time limit out of all instances within the group (# solve), the average solution time (Avg nodes) and the average number of nodes processed in the branch-and-bound tree (Avg nodes) for the instances that get solved within the time limit in each group.

Instance	total #	BP			Two-index		
		# solved	Avg time	Avg nodes	# solved	Avg time	Avg nodes
c1-25	9	9	65.7	1	8	352.5	206K
c2-25	8	8	25.9	1	7	32.2	24K
c1-50	9	8	103.6	1	7	603.4	136K
c2-50	8	3	3077.9	3	6	91.7	15K
r1-25	12	12	0.1	1	10	541.4	501K
r2-25	11	11	0.2	1	9	408.8	431K
r1-50	12	12	5.1	2	3	923.3	151K
r2-50	11	11	52.2	1	6	626.8	207K
rc1-25	8	8	3.6	7	7	864.3	560K
rc2-25	8	8	4.7	3	7	143.3	112K
rc1-50	8	8	234.8	6	3	2572.6	893K
rc2-50	8	7	367.0	7	6	655.2	240K

Table 1: Summary of computational results of the set partitioning formulation (13) and two-index flow-based formulation (14) on VRPTWCNC with quadratic node cost functions for each class of instances, using medium penalty parameter $\rho = 1$.

We see from Table 1 that the BP algorithm for the set partitioning formulation (13) performs much better than the two-index flow-based formulation (14), in particular on instance classes R and RC, where customers are more likely to be distant from each other. This indicates that the dominance rule is very effective in these cases. It is less effective if customers are clustered together, because this makes it hard to differentiate between routes with similar costs. The root bound of the set partitioning formulation is so strong that only a small number of branch-and-bound nodes are explored, which is typical for a branch-and-price algorithm applied on a vehicle routing problem. On the other hand, the two-index flow-based formulation ends up with a huge number of branch-and-bound nodes as a result of a loose relaxation bound. In most cases, it solves less number of instances within the time limit, and for the instances that can be solved within the time limit, much more time is taken than the branch-and-price algorithm. The only exception is instance c2-50, where customers are clustered together, and the time windows are wide. This makes the dominance rule hardly satisfied and therefore the labeling algorithm may end up with a huge number of labels. We next show results that use low and high penalty parameter b_i in Table 2 and investigate its impact on the effectiveness of the dominance rule.

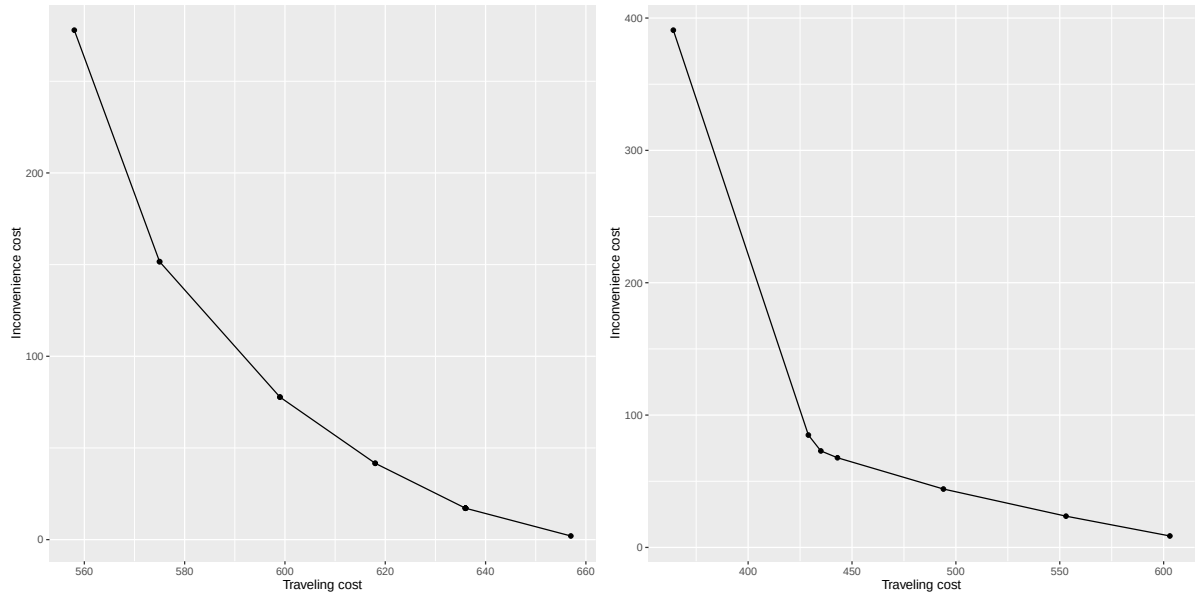
Comparing the two columns of Table 2 and the first column of Table 1, we see that as the penalty parameter ρ increases, more instances get solved, less computational time is spent, and less number of branch-and-bound nodes is obtained. This shows that in the cases with a larger penalty parameter, the dominance rule is more effective, which ends up with a smaller number of labels to store and process. In addition to the solution time, we see that a larger penalty parameter also yields a smaller number of branch-and-bound nodes, which implies that the root relaxation bound is tighter.

Finally, we show the trade off between the traveling cost and inconvenience cost (node cost) over a range of different penalty parameters ρ. Specifically, we take instance $r102$ and instance $rc102$ with 25 customers as examples, and plot the traveling cost and the inconvenience cost in the optimal solution corresponding to a range of ρ: $\{0.1, 0.2, \ldots, 6.0\}$ in Figure 1.

We see from Figure 1 that as the penalty parameter ρ increases, the inconvenience cost decreases

Instance	total #	low penalty ($\rho = 0.2$)			high penalty ($\rho = 5$)		
		# solved	Avg time	Avg nodes	# solved	Avg time	Avg nodes
c1-25	9	7	7.9	1	9	1.4	2
c2-25	8	8	362.6	2	7	6.1	1
c1-50	9	7	156.4	1	7	60.9	2
c2-50	8	1	6549.6	1	3	1671.1	3
r1-25	12	12	0.4	3	12	0.1	1
r2-25	11	11	0.8	3	11	0.1	1
r1-50	12	11	100.6	6	12	0.9	1
r2-50	11	8	430.5	2	11	28.8	1
rc1-25	8	8	147.7	7	8	0.3	3
rc2-25	8	8	66.6	2	8	0.7	2
rc1-50	8	5	1156.9	26	8	3.0	2
rc2-50	8	5	173.5	2	8	41.2	1

Table 2: Summary of computational results of the set partitioning formulation (13) for VRPTWCNC with quadratic node cost functions solved by the branch-and-price algorithm for each class of instances, using small and large penalty parameter ($\rho = 0.2$ and 5).



(a) Instance r102

(b) Instance rc102

Figure 1: Traveling cost vs. inconvenience cost at the optimal solutions of (13) with $\rho \in \{0.1, 0.2, \ldots, 6.0\}$.

and the traveling cost increases. This clearly shows the trade off between the two objectives. In a practical setting, a decision maker may experiment with the penalty parameter ρ as what has been shown in Figure 1 and chooses the best parameter according to their underlying utility.

6 Conclusion

Motivated by using service quality as a major performance index in many routing applications, we studied the VRPTW with a weighted sum of travel cost and customer inconvenience cost as the objective. We proposed a novel set partitioning formulation based on the combination of a route and blocks of nodes visited along the route. Given a fixed route and a block structure, an upper bound of the total cost can be calculated by recursively solving a univariate convex program. We showed that this upper bound is exact for the optimal block structure, that is, the block structure corresponding to the optimal service start times. We took advantage of this result and proposed a labeling algorithm to solve the pricing problem for the set partitioning formulation, and developed an effective dominance rule. Preliminary computational results illustrated the advantage of the proposed approach. We have identified some future research directions along this topic. First, how well the proposed approach performs on other types of inconvenience cost functions is worth investigating. Second, it is of interest to consider other applications in routing and scheduling that could fit into the proposed algorithm framework.

References

Nagraj Balakrishnan. Simple heuristics for the vehicle routeing problem with soft time windows. *Journal of the Operational Research Society*, 44(3):279–287, 1993.

Roberto Baldacci, Aristide Mingozzi, and Roberto Roberti. New route relxation and pricing strategies for the vehicle routing problem. *Operations Research*, 59:1269–1283, 2011.

Cynthia Barnhart, Ellis L Johnson, George L Nemhauser, Martin W.P. Savelsbergh, and Pamela H Vance. Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, 46:316–329, 1998.

Pietro Belotti, Christian Kirches, Sven Leyffer, Jeff Linderoth, James Luedtke, and Ashutosh Mahajan. Mixed-integer nonlinear optimization. *Acta Numerica*, 22:1–131, 2013.

Herminia I Calvete, Carmen Galé, María-José Oliveros, and Belén Sánchez-Valverde. A goal programming approach to vehicle routing problems with soft time windows. *European Journal of Operational Research*, 177(3):1720–1733, 2007.

Wen-Chyuan Chiang and Robert A Russell. A metaheuristic for the vehicle-routeing problem with soft time windows. *Journal of the Operational Research Society*, 55(12):1298–1310, 2004.

Juan de Oña and Rocio de Oña. Quality of service in public transport based on customer satisfaction surveys: A review and assessment of methodological approaches. *Transportation Science*, 49(3): 605–622, 2014.

Guy Desaulniers, Jacques Desrosiers, Irina Ioachim, Marius M Solomon, François Soumis, and Daniel Villeneuve. A unified framework for deterministic time constrained vehicle routing and crew scheduling problems. In T. Crainic and G. Laporte, editors, *Fleet Management and Logistics*, pages 57–93. Kluwer, 1998.

Martin Desrochers, Jacques Desrosiers, and Marius Solomon. A new optimization algorithm for the vehicle routing problem with time windows. *Operations Research*, 40:342–354, 1992.

Michael Drexl. Synchronization in vehicle routing-a survey of vrps with multiple synchronization constraints. *Transportation Science*, 46(3):297–316, 2012.

Yvan Dumas, François Soumis, and Jacques Desrosiers. Technical noteoptimizing the schedule for a fixed vehicle path with convex inconvenience costs. *Transportation Science*, 24(2):145–152, 1990.

Kjetil Fagerholt. Ship scheduling with soft time windows: An optimisation based approach. *European Journal of Operational Research*, 131(3):559–571, 2001.

Miguel Andres Figliozzi. An iterative route construction and improvement algorithm for the vehicle routing problem with soft time windows. *Transportation Research Part C: Emerging Technologies*, 18(5):668–679, 2010.

Z. Fu, R.W. Eglese, and L.Y.O. Li. A unified tabu search algorithm for vehicle routing problems with soft time windows. *Journal of the Operational Research Society.*, 59(5):663–673, 2008.

Timo Gschwind and Stefan Irnich. Effective handling of dynamic time windows and its application to solving the dial-a-ride problem. *Transportation Science*, 49(2):335–354, 2014.

Toshihide Ibaraki, Shinji Imahori, Mikio Kubo, Tomoyasu Masuda, Takeaki Uno, and Mutsunori Yagiura. Effective local search algorithms for routing and scheduling problems with general time-window constraints. *Transportation science*, 39(2):206–232, 2005.

Irina Ioachim, Sylvie Gelinas, François Soumis, and Jacques Desrosiers. A dynamic programming algorithm for the shortest path problem with time windows and linear node costs. *Networks*, 31 (3):193–204, 1998.

Irina Ioachim, Jacques Desrosiers, François Soumis, and Nicolas Bélanger. Fleet assignment and routing with schedule synchronization constraints. *European Journal of Operational Research*, 119(1):75–90, 1999.

Stefan Irnich. Resource extension functions: Properties, inversion, and generalization to segments. *OR Spectrum*, 30(1):113–148, 2008.

Stefan Irnich and Guy Desaulniers. Shortest path problems with resource constraints. In Guy Desaulniers, Jacques Desrosiers, and Marius M Solomon, editors, *Column generation*, pages 33–65. Springer, 2005.

Jang-Jei Jaw, Amedeo R Odoni, Harilaos N Psaraftis, and Nigel HM Wilson. A heuristic algorithm for the multi-vehicle advance request dial-a-ride problem with time windows. *Transportation Research Part B: Methodological*, 20(3):243–257, 1986.

Mathias A Klapp, Alan L Erera, and Alejandro Toriello. The one-dimensional dynamic dispatch waves problem. *Transportation Science*, 2016.

Yiannis A Koskosidis, Warren B Powell, and Marius M Solomon. An optimization-based heuristic for vehicle routing and scheduling with soft time window constraints. *Transportation science*, 26 (2):69–85, 1992.

Federico Liberatore, Giovanni Righini, and Matteo Salani. A column generation algorithm for the vehicle routing problem with soft time windows. *4OR*, 9(1):49–82, 2011.

Mengyang Liu, Zhixing Luo, and Andrew Lim. A branch-and-cut algorithm for a realistic dial-a-ride problem. *Transportation Research Part B: Methodological*, 81:267–288, 2015.

Jens Lysgaard. *CVRPSEP: A package of separation routines for the capacitated vehicle routing problem*. Institut for Driftøkonomi og Logistik, Handelshøjskolen i Århus, 2003.

Sébastien Mouthuy, Florence Massen, Yves Deville, and Pascal Van Hentenryck. A multistage very large-scale neighborhood search for the vehicle routing problem with soft time windows. *Transportation Science*, 49(2):223–238, 2015.

Diego Pecin, Artur Pessoa, Marcus Poggi, and Eduardo Uchoa. Improved branch-cut-and-price for capacitated vehicle routing. *Mathematical Programming Computation*, 2016. doi: 10.1007/s12532-016-0108-8.

Luigi Di Puglia Pugliese and Francesca Guerriero. A survey of resource constrained shortest path problems: Exact solution approaches. *Networks*, 62(3):183–200, 2013.

AG Qureshi, E Taniguchi, and Tadashi Yamada. An exact solution approach for vehicle routing and scheduling problems with soft time windows. *Transportation Research Part E: Logistics and Transportation Review*, 45(6):960–977, 2009.

Ali Gul Qureshi, Eiichi Tanguchi, and Tadashi Yamada. Exact solution for the vehicle routing problem with semi soft time windows and its application. *Procedia - Social and Behavioral Sciences*, 2(3):5931–5943, 2010.

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

Martin Savelsbergh and Tom Van Woensel. 50th anniversary invited articlecity logistics: Challenges and opportunities. *Transportation Science*, 50(2):579–590, 2016.

Thomas R Sexton and Lawrence D Bodin. Optimizing single vehicle many-to-many operations with desired delivery times: I. scheduling. *Transportation Science*, 19(4):378–410, 1985a.

Thomas R Sexton and Lawrence D Bodin. Optimizing single vehicle many-to-many operations with desired delivery times: Ii. routing. *Transportation Science*, 19(4):411–435, 1985b.

Thomas R Sexton and Young-Myung Choi. Pickup and delivery of partial loads with soft time windows. *American Journal of Mathematical and Management Sciences*, 6(3-4):369–398, 1986.

Marius M Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations research*, 35(2):254–265, 1987.

Éric Taillard, Philippe Badeau, Michel Gendreau, François Guertin, and Jean-Yves Potvin. A tabu search heuristic for the vehicle routing problem with soft time windows. *Transportation science*, 31(2):170–186, 1997.

Pham Dinh Tao and Le Thi Hoai An. Convex analysis approach to dc programming: theory, algorithms and applications. *Acta Mathematica Vietnamica*, 22(1):289–355, 1997.

Paolo Toth and Daniele Vigo. *Vehicle routing: problems, methods, and applications*, volume 18. SIAM, 2014.

Thibaut Vidal, Teodor Gabriel Crainic, Michel Gendreau, and Christian Prins. Timing problems and algorithms: Time decisions for sequences of activities. *Networks*, 65(2):102–128, 2015.

Stacy A Voccia, Ann Melissa Campbell, and Barrett W Thomas. The same-day delivery problem for online purchases. Technical report, University of Iowa, 2015.