A Column Generation Based Heuristic for the Dial-A-Ride Problem

Nastaran Rahmani, Boris Detienne, Ruslan Sadykov, François Vanderbeck

A Column Generation Based Heuristic for the Dial-A-Ride Problem

Nastaran Rahmani [1], Boris Detienne [2,3], Ruslan Sadykov [3,2], François Vanderbeck [2,3]

[1] Kedge Business School, 680 Cours de la Libération, 33405, Talence, France
[2] University of Bordeaux, 351 Cours de la Libération, 33405, Talence, France
[3] INRIA Bordeaux-Sud-Ouest, Talence, France
{nastaran.rahmani@kedgebs.com, boris.detienne@math.u-bordeaux1.fr, ruslan.sadykov@inria.fr, fv@math.u-bordeaux1.fr}

Abstract. The Dial-a-Ride Problem is a variant of the pickup and delivery problem with time windows, where the user inconvenience must be taken into account. In this paper, ride time and customer waiting time are modeled through both constraints and an associated penalty in the objective function. We develop a column generation approach, dynamically generating feasible vehicle routes. Handling ride time constraints explicitly in the pricing problem solver requires specific developments. Our dynamic programming approach for pricing problem makes use of a heuristic dominance rule and a heuristic enumeration procedure, which in turns implies that our overall branch-and-price procedure is a heuristic. However, in practice our heuristic solutions are experimentally very close to exact solutions and our approach is numerically competitive in terms of computation times.

Keywords: static dial-a-ride problem, column generation, dynamic programming.

1 Introduction

The Dial-a-Ride Problem (DARP) consists in optimizing the transport of persons from their origin to their destination. The objective is not only to minimize the operational costs, but also the user inconvenience. The latter is modeled either through constraints or via the objective function. In the literature, constraints that formulate the user inconvenience are known as ride time constraints. The door-to-door transportation services for elderly and disabled people together with the shuttle bus services connecting airports and customer hotels are some applications of DARP.

In 2007, Cordeau and Laporte [1] provided a complete review for the dial-a-ride problem. In this review, one can notice that among 25 contributions, only 5 contributions are devoted to exact approaches, the work of Cordeau [2] being the most cited. He proposed a three-index formulation and solved instances with up to 4 vehicles and 32 requests using a branch-and-cut approach. Later, Robke et al. [3] applied a branch-and-cut approach based on a two-index formulation. Their work resulted in solving instances with up to 8 vehicles and 96 requests. Although the latter formulation leads to better performances, it only applies in the case of a homogeneous fleet of vehicles.

In 2011, Parragh et al. [4] studied the dial-a-ride problem with heterogenous users and fleet. They successfully adapted the state-of-the-art branch-and-cut algorithm proposed by Cordeau [2]. Up to 2012, there exists no contribution dealing with the DARP using a column generation approach. Indeed, handling the ride time constraint in the process of a dynamic programming pricing approach is far from being trivial. In 2012, Parragh et al. [5] studied a more complex heterogeneous dial-a-ride problem, where two types of vehicles and four different transportation models where considered together with driver related constraints. They proposed a three-index formulation and a set partitioning formulation for their problem, and the linear programming relaxation of the set partitioning formulation was solved using a column generation approach. To manage the difficulty caused by the ride time constraint for the dynamic programming approach, Parragh et al. [5] considered the maximum ride time implicitly in terms of time windows both at the pickup and the delivery node. Note that their scheme implements necessary, but not sufficient conditions to obey the ride time constraints.

In Section 2.1, we present a polynomial size compact formulation for the heterogeneous dial-a-ride problem, where ride time and customer waiting time are modeled through both constraints and an associated

penalty in the objective function. We assume that the fleet size is given. Then, the Dantzig-Wolfe reformulation of the original compact formulation is outlined in Section 2.2. The reformulation is then solved using a branch-and-price approach enhanced using Strong Degree Cuts. To the best of our knowledge this is the first work that solves the dial-a-ride problem using a branch-and-price approach, while ride time constraints are considered explicitly in the process of a forward labeling approach. In Section 3, we fully discuss the difficulty that arises, when ride time constraints are taken into account explicitly in the forward labeling dynamic programming based heuristic. We adapt the latest technologies developed for the vehicle routing problem to accelerate column generation approaches such as the ng-set paradigm to avoid cycles. In Section 4, the efficiency of the presented methodology is numerically tested using the BaPCod[1] platform. Finally, Section 5 outlines some conclusion, observing that the overall approach is proved efficient in dealing with instances of the size that have been considered as challenging in the literature.

2 Problem Definition

In the dial-a-ride problem one has to cover a set of transportation requests, $\mathscr{R} = \{1,...,R\}$, defined over a physical network $G^s = (N^s, A^s)$ with a fleet of vehicles; G^s is a complete symmetric directed graph where nodes represent pickup or delivery sites and edge costs satisfy the triangle inequality. Each request $r \in \mathscr{R}$ entails a demand of a certain load for a transportation service from a pickup node to a delivery node within predefined time windows. Furthermore, for each pickup node (and similarly each delivery node) a service duration is defined. Thus, each request $r \in \mathscr{R}$ is characterized by:

- $p_r \in N^s$ (resp. $d_r \in N^s$), the pickup (resp. delivery) node of the request.
- $e_{p_r} \in \mathbb{R}^+$ (resp. $e_{d_r} \in \mathbb{R}^+$), the earliest time for starting service at the pickup (resp. delivery) node of the request.
- $l_{p_r} \in \mathbb{R}^+$ (resp. $l_{d_r} \in \mathbb{R}^+$), the due date for the start of the service at the pickup (resp. delivery) node; it can be violated at a cost.
- $\bar{l}_{p_r} \in \mathbb{R}^+$ (resp. $\bar{l}_{d_r} \in \mathbb{R}^+$), the deadline for starting service at the pickup (resp. delivery) node; it cannot be violated.
- $s_{p_r} \in \mathbb{R}^+$ ($s_{d_r} \in \mathbb{R}^+$), the service duration for the pickup (delivery) node.
- $q_r \in \mathbb{Z}^+$, the load of the request.
- $c_r^f \in \mathbb{R}^+$, the flow time cost of the request (= cost for the ride time).
- $b_r^f \in \mathbb{R}^+$, the flow time bound of the request (= bound for the ride time).
- $c_i^w \in \mathbb{R}^+$, the waiting cost that penalizes the violation of the due date when $i \in \{p_r, d_r\}$.

Now let $P = \{(p_r, r) : r \in \mathscr{R}\}$, $D = \{(d_r, r) : r \in \mathscr{R}\}$, and $O = \{o^+, o^-\}$, where o^+ and o^- are the duplication of the depot node that represent the depot on departure and the return to the depot node, respectively. To introduce the mathematical formulation of the problem, first we define network $G = (N, A)$, where $N = P \cup D \cup O$. Note that $P \cap D = \emptyset$, since the elements of P and D are distinguished by the request index r and for each request the pickup node and the delivery node cannot be equal. For simplicity, we use p_r to represent $(p_r, r) \in P$ and d_r to represent $(d_r, r) \in D$ along the text. Each vehicle $k \in \mathscr{K}$ is located in the depot. Vehicles are partitionned into classes indexed by i and all vehicles $k \in \mathscr{K}_i$ of a class $i \in \{1,...,K\}$ have the same maximum route duration $T^k = T^i$ and capacity $Q^k = Q^i$. Thus, $\mathscr{K}_i$ is the set of available vehicles of type i and $\mathscr{K} = \cup_{i=1}^{K} \mathscr{K}_i$. Let $Q = \max_{i=1,...,K} Q^i$.

2.1 Mathematical Formulation

Our formulation is an extension of the one proposed by Cordeau [2] for the homogenous DARP, where we penalize customer waiting time and ride time in the objective function. Moreover, we enforce bounds on customer waiting times and on their ride times. To model the problem as a mixed integer linear program, we use the following decision variables:

[1] BaPCod is a prototype code that solves Mixed Integer Programs (MIP) using a branch-and-price algorithm applied to a Dantzig-Wolfe reformulation.

- $x_{ij}^k \in \{0,1\}$ is 1 if vehicle $k \in \mathscr{K}$ travels on arc $(i,j) \in A$ and 0 otherwise ($i \neq j$, $i \neq o^-$, $j \neq o^+$, and $i, j \in N$).
- $y_r^k \in \{0,1\}$ is 1 if vehicle k is taking care of request $r \in \mathscr{R}$ and 0 otherwise.
- $t_i^k \in [e_i, \bar{l}_i]$ is the start service time of vehicle k at node $i \in N$.
- $Q_i^k \in [0, Q]$ is the load of the vehicle k after visiting node $i \in N$.
- $w_i^k \in [0, \bar{l}_i - l_i]$ is the amount of time that customer $i \in N \setminus \{o^+, o^-\}$ waits for vehicle k; it is positive in the case where the vehicle arrives after due date l_i.
- $f_r^k \in [0, b_r^f]$ is the ride time for request $r \in \mathscr{R}$ on vehicle $k \in \mathscr{K}$.

The formulation is then:

$$\min \sum_{k\in\mathscr{K}} \{\sum_{i\in N}\sum_{j\in N} c_{ij}^k x_{ij}^k \quad + \quad \alpha \sum_{i\in N\setminus O} c_i^w w_i^k + \beta \sum_{r\in\mathscr{R}} c_r^f f_r^k\} \tag{1}$$

$$\sum_{k\in\mathscr{K}} y_r^k \quad = \quad 1 \quad \forall r \in \mathscr{R} \tag{2}$$

$\forall k \in \mathscr{K}$:

$$y_r^k - \sum_{j\in N\setminus\{o^+\}} x_{p_r j}^k \quad = \quad 0 \quad \forall r \in \mathscr{R} \tag{3}$$

$$y_r^k - \sum_{i\in N\setminus\{o^-\}} x_{i d_r}^k \quad = \quad 0 \quad \forall r \in \mathscr{R} \tag{4}$$

$$\sum_{j\in P} x_{o^+ j}^k \quad \leq \quad 1 \tag{5}$$

$$\sum_{j\in N\setminus\{o^-\}} x_{ji}^k - \sum_{j\in N\setminus\{o^+\}} x_{ij}^k \quad = \quad 0 \quad \forall i \in N \setminus O \tag{6}$$

$$\sum_{i\in D} x_{i o^-}^k \quad \leq \quad 1 \tag{7}$$

$$t_i^k + s_i + t_{ij} - M_{ij}^k(1 - x_{ij}^k) \quad \leq \quad t_j^k \quad \forall i \in N\setminus\{o^-\}, j \in N\setminus\{o^+\} \tag{8}$$

$$Q_i^k + q_j - W_{ij}^k(1 - x_{ij}^k) \quad \leq \quad Q_j^k \quad \forall i \in N\setminus\{o^-\}, j \in N\setminus\{o^+\} \tag{9}$$

$$t_{o^-}^k - t_{o^+}^k \quad \leq \quad T^k \tag{10}$$

$$e_{p_r}\, y_r^k \leq t_{p_r}^k \quad \leq \quad \bar{l}_{p_r}\, y_r^k \quad \forall r \in \mathscr{R} \tag{11}$$

$$e_{d_r}\, y_r^k \leq t_{d_r}^k \quad \leq \quad \bar{l}_{d_r}\, y_r^k \quad \forall r \in \mathscr{R} \tag{12}$$

$$w_i^k \quad \geq \quad t_i^k - l_i \quad \forall i \in N \setminus O \tag{13}$$

$$0 \leq w_i^k \quad \leq \quad \bar{l}_i - l_i \quad \forall i \in N \setminus O \tag{14}$$

$$q_r\, y_r^k \leq Q_{p_r}^k \quad \leq \quad Q^k\, y_r^k \quad \forall r \in \mathscr{R} \tag{15}$$

$$0 \leq Q_{d_r}^k \quad \leq \quad (Q^k - q_{d_r})\, y_r^k \quad \forall r \in \mathscr{R} \tag{16}$$

$$t_{d_r}^k - t_{p_r}^k - s_{p_r} \quad \leq \quad f_r^k \quad \forall r \in \mathscr{R} \tag{17}$$

$$t_{p_r d_r}\, y_r^k \leq f_r^k \quad \leq \quad b_r^f\, y_r^k \quad \forall r \in \mathscr{R} \tag{18}$$

The objective function (1) is to minimize routing and timing costs; c_{ij}^k is the transportation cost between node $i \in N$ and node $j \in N$ for vehicle $k \in \mathscr{K}$. Parameters α and β can be tuned to balance timing costs. Constraint (2) is the assignment constraint, which ensures that each request is covered exactly once. Constraints (3) and (4) impose that if a request is handled by a vehicle, both pickup and delivery must be done in the route. Constraints (5) to (7) guarantee that the route of each vehicle k starts at the source node o^+ and ends at the sink node o^-. The feasibility of time and load variables for arc (i,j) is checked by constraints (8) and (9). The M_{ij}^k and W_{ij}^k are "big-M" constants that are defined in Remark 1 below. In constraint (8), t_{ij} represents the travel time between node i and j. Constraint (10) defines an upper bound for route duration. The time window are imposed in constraints (11) - (12). The tardiness of the vehicle at each node can be measured by constraints (13) and (14). Constraints (15) and (16) formulate the capacity constraints. The ride time of each request is defined and bounded through constraints (17) - (18).

Remark 1 *For the "big-M" constant, one can set* $W_{ij}^k = \max\{Q^k, Q^k + q_j\}$ *and* $M_{ij}^k = \bar{l}_i + s_i + t_{ij}$ *for each* $i, j \in N$ *and* $k \in \mathscr{K}$.

2.2 The Dantzig-Wolfe Reformulation

The presence of big-M in the compact formulation is an indication of the poor quality of the Linear Programming (LP) dual bound that can be expected. On the other hand, the block diagonal structure of the constraint matrix points to a decomposition approach. The Dantzig-Wolfe reformulation of the above compact formulation is as follows:

$$[\text{MP}] : \min \quad \sum_{i=1}^{K} \sum_{g_i \in Z_i} c_{g_i} \nu_{g_i} \tag{19}$$

$$\text{s.t}$$

$$\sum_{i=1}^{K} \sum_{g_i \in Z_i} y_{pg_i} \nu_{g_i} \geq 1 \quad \forall p \in P \tag{20}$$

$$\sum_{g_i \in Z_i} \nu_{g_i} \leq n_i \quad i = 1, ..., K \tag{21}$$

$$\nu_{g_i} \in \mathbb{Z}_+ \tag{22}$$

where Z_i is the subproblem polyhedron that contains all feasible routes g_i satisfying constraints (3) to (18) for a vehicle of type i. As we have identical subproblems with a vehicle class i, the aggregate variables ν_{g_i} are used that represent the number of times route g_i is chosen in the solution. Finally, c_{g_i} is the cost of route g_i, and y_{pg_i} is one if route g_i contains node p; and $y_{pg_i} = 0$ otherwise.
Solving the LP relaxation of formulation (19) to (22), which is traditionally called the *Master Program* [MP], requires to handle its large number of variables. This is done using a column generation approach. A restricted master linear program [RMLP] is defined by considering a subset of feasible routes $\bar{Z}_i \subset Z_i$. Its dual formulation is

$$[\text{DRMLP}] : \max \quad \sum_{p \in P} \omega_p \quad + \quad \sum_{i=1}^{K} \sigma_i \tag{23}$$

$$\text{s.t}$$

$$\sum_{p \in P} y_{pg_i} \omega_p + \sigma_i \leq c_{g_i} \quad \forall i = 1, ..., K,\ g_i \in \bar{Z}_i \tag{24}$$

$$\omega_p \geq 0 \tag{25}$$

$$\sigma_i \leq 0 \tag{26}$$

where ω_p and σ_i are the dual variables corresponding to constraints (20) and (21), respectively. Let $\nu = (\nu_{g_i})_{\{g_i \in \bar{Z}_i : i=1,...,K\}}$ represent the solution to the primal problem obtained by applying the simplex algorithm on RMLP; $\omega = (\omega_p)_{p \in P}$ and $\sigma = (\sigma_i)_{\{i=1,...,K\}}$ denote the corresponding dual solutions. Suppose that (ω^*, σ^*) is the optimal solution of [DRMLP] associated to primal solution ν^*. If (ω^*, σ^*) is feasible for the unrestricted dual master program, then ν^* is optimal for linear relaxation of [MP]. If not, there exists a route in $Z_i \setminus \bar{Z}_i$ that violates constraint (24). To find such route one needs to solve, for each vehicle of type i, the following pricing problem: $\min\{c_{g_i} - \sum_{p \in P} y_{pg_i} \omega_p^* - \sigma_i^* \leq 0 : g_i \in Z_i\}$. If the solution has non-negative value, (ω^*, σ^*) is proved to be feasible for all $g_i \in Z_i$; otherwise, the route is added to $\bar{Z}_i$. In this way $\bar{Z}_i$ is gradually extended in such a way that it includes all the feasible routes that participate in the optimal solution. In next section, we propose a forward labeling approach to solve the pricing problem. As our approach generates ng-routes, we add to the restricted master problem the so-called Strong Degree Cuts (SDCs), introduced by Contardo et al. [6], to eliminate a solution where a fraction of a route cycles at a node to cover it: $\sum_{g_i \in \bar{Z}_i : r \in g_i} \nu_{g_i} \geq 1 \quad \forall r \in \mathscr{R}$. Our forward labeling algorithm is adapted in such a way that it manages the effect of non-robust strong degree cuts.

3 Forward Labeling Approach to Solve the Pricing Problem

Developing a forward labeling algorithm to solve pricing problem is challenging, as it requires deciding on not only the sequence of visits, but also the time of visits. Note that the issue of timing is also present in different variants of resource constraint shortest path problems that appears as sub-problems of vehicle

routing problems with time windows. However, there the timing decisions are trivial: nodes must simply be visited as early as they can feasibly be. For the dial-a-ride problem, such early start service times can result in the violation of the ride time constraint of some requests. Hence, the existence of ride time constraints induce an incentive for delaying pickup times. Moreover, waiting time and ride time costs might also induce that the optimal solution is not to start service as early as feasible. In the sequel, we make the **simplifying assumption** that those costs are zero and due dates are equal to deadlines: $c_r^w = c_r^f = 0$ and $\bar{l}_i = l_i$. So the only incentive that remains to consider not starting service as early as feasible is to meet the ride time constraints.

Our alogrithm is in fact **a heuristic based on a dynamic programming paradigm**. It is inexact in two ways: our dominance conditions eliminate more partial solutions that it should, and our transition procedure makes use of information related to the partial solution itself instead of using only what we define as state defining attributes of a partial solution. The latter "trick" can be understood as a state space relaxation: the full state being the explicit path so far, the aggregated state being what we have used as defining attributes of a partial solution. Below, we define labels (nodes of the state space graph) that are generated during the forward labeling approach.

Each label L represent a partial solution built so far that is summarized by a state. Associated with a label, we define two kinds of fields: defining attributes and auxiliary attributes.

Defining Attributes, define the current state and its charaterization in terms of the quantity of resource consumption in the partial path associated to label L:

- $\mathrm{i}^L \in N$ is the node at which the partial solution ends.
- z^L is the vehicle route duration up to arrival at this node.
- $\mathscr{A}^L$ is the vector of earliest start service times. Element j of this vector noted as a_j^L, represents the earliest start service time at node j; adding new nodes in the path might induce the requirement to further delay some start service time of previous node of the partial path.
- c^L is the accumulated cost.
- $\mathscr{V}^L$ is the set of served requests so-far. It is used to implement the ng-set paradigm to avoid cycles. The service of the elements of $\mathscr{V}^L$ is completed.
- $\mathscr{O}^L$ is the set of open requests at the node. The service of the elements of $\mathscr{O}^L$ is not completed.
- p^L is a pointer to the predecessor label of L.

Auxiliary Attributes, are quantities that could be computed using the defining attributes. The aim of introducing auxiliary attributes is to ease computations by avoiding to recompute quantities from scratch. Those attributes are not used when comparing partial solutions.

- r^L is the index of the request corresponding to node i^L.
- q^L is the load of the vehicle after visiting node i^L; it can be computed knowing $\mathscr{O}^L$.

When considering an arc extension from label L to a node $j \in N$ ($j \neq \mathrm{i}^L$), we make use of some necessary feasibility conditions to restrict the generation of solutions.

Case 1 : Assume that j is a pickup node, $j = p_r \in P$. Then, our feasibility conditions take the form:

$$\mathrm{a}_{\mathrm{i}^L}^L + s_{\mathrm{i}^L} + t_{\mathrm{i}^L j} \leq \bar{l}_j \tag{27}$$

$$\mathrm{z}^L + \max\{e_j, \mathrm{a}_{\mathrm{i}^L}^L + s_{\mathrm{i}^L} + t_{\mathrm{i}^L j}\} + s_j + t_{j d_{r_j}} - \mathrm{a}_{\mathrm{i}^L}^L \leq T^k \tag{28}$$

$$\mathrm{q}^L + q_j \leq Q^k \tag{29}$$

$$r \notin \mathscr{V}^L \tag{30}$$

$$r \notin \mathscr{O}^L \tag{31}$$

Constraints (27) and (29) covers the feasibility conditions for time window and load constraints, respectively. Constraint (28) assess feasibility condition for maximum vehicle duration. Conditions (30) and (31)

express the fact that arc extension (i^L, j) can happen only if request r has not been visited nor opened yet.
Case 2 : Let j be a delivery node, $j = d_r \in D$. The conditions include (27) and (28) together with:

$$r \in \mathscr{O}^L \tag{32}$$

$$\max\{e_j, \mathrm{a}^L_{\mathrm{i}^L} + s_{\mathrm{i}^L} + t_{\mathrm{i}^L j}\} - \mathrm{a}^L_{p_r} - s_{p_r} \leq b^f_r\ . \tag{33}$$

Conditions (32) and (33) cover the precedence constraints and the ride time constraints, respectively. In case condition (33) is not satisfied at d_r, we delay the earliest start service times by the value of a so-called "forward time slack" (the "forward time slack" is a concept introduced by Cordeau [2] to represent the delay that is required to meet the bounds on ride times) that we compute recursively. Then, we check the ride time constraints and maximum vehicle duration constraints for requests that appears in the partial path and we reset earliest starting times accordingly.
Case 3 : When $j = o^-$, conditions (27), (28) together with the following condition must be valid:

$$\mathscr{O}^L = \emptyset \tag{34}$$

Note that one can apply further feasibility conditions to detect the infeasibility of the potential label at node j early by verifying if

$$\max\{e_j, \mathrm{a}^L_{\mathrm{i}^L} + s_{\mathrm{i}^L} + t_{\mathrm{i}^L j}\} + s_j + t_{jo^-} \leq \bar{l}_{o^-}.$$

If feasibility conditions are satisfied, the new label L' can be generated at node j as follows:

$$\mathrm{i}^{L'} = j \tag{35}$$

$$\mathrm{z}^{L'} = \mathrm{z}^L + \max\{e_j, \mathrm{a}^L_{\mathrm{i}^L} + s_{\mathrm{i}^L} + t_{\mathrm{i}^L j}\} - \mathrm{a}^L_{\mathrm{i}^L} \tag{36}$$

$$\mathrm{a}^{L'}_j = \max\{e_j, \mathrm{a}^L_{\mathrm{i}^L} + s_{\mathrm{i}^L} + t_{\mathrm{i}^L j}\} \tag{37}$$

$$\mathrm{q}^{L'} = \begin{cases} \mathrm{q}^L + q_j & \text{if } j \in P \\ \mathrm{q}^L - q_j & \text{if } j \in D \end{cases} \tag{38}$$

$$\mathrm{r}^{L'} = r \tag{39}$$

$$\mathrm{c}^{L'} = \mathrm{c}^L + c_{\mathrm{i}^L j} \tag{40}$$

$$\mathscr{V}^{L'} = \begin{cases} \mathscr{V}^L & \text{if } j \in P \\ \mathscr{V}^L \cup \{r\} & \text{if } j \in D \end{cases} \tag{41}$$

$$\mathscr{O}^{L'} = \begin{cases} \mathscr{O}^L \cup \{r\} & \text{if } j \in P \\ \mathscr{O}^L \setminus \{r\} & \text{if } j \in D \end{cases} \tag{42}$$

$$\mathrm{p}^{L'} = L \tag{43}$$

Once a new label L' is created, we apply the concept of dominance. We consider that label L dominates label L' if:

$$\mathrm{i}^L = \mathrm{i}^{L'} \tag{44}$$

$$\mathrm{a}^L_j \leq \mathrm{a}^{L'}_j \tag{45}$$

$$\mathrm{c}^L \leq \mathrm{c}^{L'} - \textstyle\sum_r \pi_r \quad r \notin \mathscr{V}^{L'}, r \in \mathscr{V}^L \tag{46}$$

$$\mathscr{O}^L = \mathscr{O}^{L'} \tag{47}$$

$$|\mathscr{O}^L| = |\mathscr{O}^{L'}| = \begin{cases} 1 & \text{if } \mathrm{i}^L \in P \\ 0 & \text{if } O.W \end{cases} \tag{48}$$

where π_r is the dual variable corresponded to SDCs. If $\pi_r > 0$, its effect must be taken into account by subtracting the value of π_r from the cost of the new label L' at p_r if $r \notin \mathscr{V}^L \cup \mathscr{O}^L$, (i. e. $\mathrm{c}^{L'} - \pi_r$). Since $\pi_r > 0$, this particular extension is being rewarded. This **heuristic dominance rule** results in discarding more labels than an exact dominance rule. Hence, our overall scheme is heuristic.

Table 1: Numerical Results

	B&C [2]			B&P		B&P (MIP SP)	
Instance	DB	T	T/7.5	DB	T	DB	T
a2-16	**294.25**	0.6	0.08	**294.26**	4.20	**294.26**	81.58
a2-20	**344.83**	4.8	0.64	**344.86**	36.63	**344.86**	804.43
a2-24	**431.12**	16.2	2.16	**431.10**	80.68	**431.10**	2584.73
a3-18	**300.48**	12.0	1.60	**300.49**	11.17	**300.49**	548.05
a3-24	**344.83**	250.2	33.36	**344.85**	157.22	-	T_{lim}
a3-30	**494.85**	2543.4	339.12	**494.82**	273.87	-	T_{lim}
a3-36	**583.19**	747.6	99.68	**589.86**	2050.35	-	T_{lim}
a4-16	**282.68**	152.4	20.32	**282.67**	10.29	**282.67**	519.07
a4-24	**375.02**	1511.4	201.52	**375.02**	76.17	-	T_{lim}
a4-32	447.66	14400.0	-	**487.42**	1618.59	-	T_{lim}
a4-40	475.05	14400.0	-	**557.70**	3133.92	-	T_{lim}
a4-48	486.03	14400.0	-	-	T_{lim}	-	T_{lim}
b2-16	**309.41**	9.0	1.20	**309.39**	11.52	**309.39**	427.89
b2-20	**332.64**	0.6	0.08	**332.65**	4.23	**332.65**	168.64
b2-24	**444.71**	7.8	1.04	**444.72**	55.57	-	T_{lim}
b3-18	**301.64**	42.0	5.60	**301.64**	6.76	**301.64**	2531.81
b3-24	**394.51**	217.2	28.96	-	T_{lim}	-	T_{lim}
b3-30	**531.44**	409.2	54.56	**531.43**	140.95	-	T_{lim}
b3-36	**603.79**	3724.2	496.56	-	T_{lim}	-	T_{lim}
b4-16	**296.96**	47.4	6.32	**296.95**	0.86	**296.95**	135.18
b4-24	**371.41**	351.0	46.80	**371.39**	47.52	-	T_{lim}
b4-32	**494.82**	10609.2	1414.56	**494.86**	72.39	-	T_{lim}
b4-40	591.76	14400.0	-	-	T_{lim}	-	T_{lim}
b4-48	586.91	14400.0	-	-	T_{lim}	-	T_{lim}

4 Numerical Experiments

In this section, we numerically assess the efficiency of our proposed approach. The experiments are implemented in C++ , under the environment of BaPCod and run on an MacBookPro10.9.5 (8GB, 2.4 GHz) over the data set available on http://chairelogistique.hec.ca/data/darp/branch-and-cut/. We emphasize that our symplifying assumption whereby we set $c_r^f = 0$, $c_i^w = 0$, and $l_i = \bar{l}_i$ $(\forall r \in \mathscr{R}, \forall i \in N)$ reduces our model to the same one as that of Cordeau [2].

Our results are presented in Table 1 which is partitioned into three parts: the first part entitled as B&C reports the state-of-the-art results obtained by applying branch-and-cut approach proposed by Cordeau [2]. These results are directly extracted from their paper. The second part entitled as B&P reports the results of applying our heuristic branch-and-price approach. To assess the performance of our heuristic, we compare it to an exact branch-and-price approach where the pricing subproblem is solved using a MIP solver. These results are presented in the third part of the table entitled as B&P (MIP SP). All three parts are composed of two columns entitled as DB and T that report dual bound and CPU time. For all schemes when a dual bound is highlighted in bold it reports optimal value (dual bound is equal to primal bound); otherwise, only a lower bound to the optimal value is obtained. For the heuristic branch-and-price approach, our primal bound is the best integer solution encountered as an intermediate solution while solving the master LP by column generation through the branch-and-bound tree. We branch on the edge variables. Computational times are reported in seconds, and a time limit of one hour is set. Whenever the time limit is reached, it is reported by T_{lim}. Since the machine used by Cordeau [2] is not as recent as ours, we apply a correction factor 7.5 which is a very approximative factor that gives an order of performance improvement from a 2.5 GHz Pentium 4 computer with 512 Mb of memory to MacBookPro10.9.5 (8GB, 2.4 GHz) according to [7]. Thus, for a rough comparison of computing times we add a column entitled $T/7.5$.

Numerical results show that the exact branch-and-price approach solves 37.5% of instances to optimality in one hour time limit with average computational time of 866.82 seconds. For the same set of instances,

our heuristic branch-and-price approach manages to find the same dual and primal bound with no gap and hence solves these instances to optimality with average computational time of 18.48 seconds. The branch-and-cut approach proposed by Cordeau [2] solves this set of instances with average computational time of 31.67 seconds that decreases to 4.22 seconds using a correction factor of 7.5. On larger instances, where the branch-and-cut approach converges within the one hour time limit, our heuristic branch-and-price converges to the same dual and primal bounds. Overall, our heuristic branch-and-price converges for 79.17% of instances, and 37.5% of them are proved to be optimal for the restricted model. As the performance factor is approximative, we conclude that our approach can converge faster than the branch-and-cut approach for around 25% of instances.

5 Conclusion

In this paper, a generic compact formulation for heterogenous dial-a-ride problem is proposed, where customer waiting and ride times are modeled via a cost in the objective function and constraints. A column generation approach is then applied to solve the Dantzig-Wolfe reformulation under a simplifying assumption: waiting and ride time costs are set to zero and due dates are equal to deadlines. The pricing problem of the column generation considers explicitly the ride time constraints. To solve it, we develop a dynamic programming based heuristic approach where we apply a non-exact dominance rule and some implicit state space aggregation. The overall method is therefore a heuristic. Numerical results showed that our approach often lead to optimal solutions and that our computing times can be rather competitive.

References

[1] Cordeau, J.-F., Laporte, G.: The Dial-a-Ride Problem: Models and Algorithms, Annals of Operations Research, 153, 29–46 (2007)

[2] Cordeau, J.-F.: A Branch-and-Cut Algorithm for the Dial-a-Ride Problem, Operations Research, 54, 573–586 (2006)

[3] Ropke, S., Cordeau, J.-F., and Laporte, G.: Models and Branch-and-Cut Algorithms for Pickup and Delivery Problems with Time Windows, Networks, 49, 258–272 (2007)

[4] Parragh, S. N.: Introducing Heterogeneous Users and Vehicles into Models and Algorithms for the Dial-a-Ride Problem, Transportation Research Part C: Emerging Technologies, 19, 912–930 (2011)

[5] Parragh, S., Cordeau, J.-F., Doerner, K., and Hartl, R.: Models and Algorithms for the Heterogeneous Dial-a-Ride Problem with Driver-Related Constraints, OR Spectrum, 34, 593–633 (2012)

[6] Contardo, C., Gendron, B., and Cordeau, J.-F.: A branch-and- Cut-and-Price algorithm for the Capacitated Location-Routing Problem, CIRRELT, 2011.

[7] http://preshing.com/20120208/a-look-back-at-single-threaded-cpu-performance/