4OR-Q J Oper Res (2010) 8:407–424
DOI 10.1007/s10288-010-0130-z

EDUCATION PAPER

A tutorial on column generation and branch-and-price for vehicle routing problems

Dominique Feillet

Received: 28 April 2009 / Revised: 4 April 2010 / Published online: 5 June 2010

Abstract This paper provides a tutorial on column generation and branch-and-price for vehicle routing problems. The main principles and the basic theory of the methods are first outlined. Some additional issues, including reinforcement of the relaxation or stabilization, complete the paper. For the sake of simplicity, this material is illustrated with the case of the vehicle routing problem with time windows.

Keywords Column generation · Branch-and-price · Vehicle routing problem

MSC classification (2000) 9001 · 90B06 · 90C06

1 Introduction

Due to its many successes, branch-and-price has rapidly become an approach that cannot be overlooked when solving transportation problems and, especially, vehicle routing problems. Yet, carrying out an accurate implementation of the method or solely achieving a good understanding of it, is often a long and difficult task for at least three reasons:

- the inherent complexity of the method and the abundant literature on the subject,
- the very different angles from which the method can be looked at,
- the lack of simple and comprehensive descriptions of the method.

The main objective of this tutorial is to help the reader to better understand the method. For the sake of simplicity, branch-and-price is illustrated in the case of the Vehicle Routing Problem with Time Windows (VRPTW). Even so, most of the material of this tutorial remains valid for the vast majority of vehicle routing problems, for

D. Feillet (✉)
Ecole Nationale Supérieure des Mines de Saint-Etienne,
CMP Georges Charpak, 13541 Gardanne, France
e-mail: feillet@emse.fr

many other types of freight or person transportation problems and, hopefully, it can also be helpful when addressing branch-and-price in other domains, as telecommunication network design or production scheduling for example.

Section 2 introduces the VRPTW. Section 3 describes the principles and main issues of column generation and branch-and-price methodologies in this context. Some additional topics, opening to more evolved issues, are discussed in Sect. 4, with related references.

This paper does not insist on the many computational tricks that could help achieving a more efficient implementation of the method. Many of them can be found in Desaulniers et al. (2001). The reader is also referred to Desaulniers et al. (2005) for a recent and important book on the subject.

2 Description of the VRPTW

The VRPTW can be described as follows. Given a set of customers, a set of vehicles, and a depot, the VRPTW is to find a set of routes of minimal total length, starting and ending at the depot, such that each customer is visited by exactly one vehicle to satisfy a specific demand. Customer visit has to respect a requested time window. A vehicle can wait in case of early arrival, but late arrival is not allowed. In connection with customer demands, a capacity constraint restricts the load that can be carried by a vehicle.

More formally, the VRPTW is defined on a complete directed graph $G = (V, A)$, where $V = \{v_0, \ldots, v_n\}$ is the set of nodes and A is the set of arcs. Vertex v_0 is a special node called the depot, vertices v_1 to v_n represent customers. A cost c_{ij} and a travel time t_{ij} are defined for every arc $(v_i, v_j) \in A$. Every customer $v_i \in V\backslash\{v_0\}$ has a positive demand d_i, a time window $[a_i, b_i]$ and a positive service time $serv_i$. A fleet of U vehicles of capacity Q is available for serving the customers. Vehicles must begin and end their routes at the depot within a time horizon $[a_0, b_0]$. The cumulative demand of the customers visited by a route is limited by Q. The service of a customer has to start within its time window, but a vehicle is allowed to arrive earlier and to wait. The VRPTW consists in finding a minimum cost set of at most U routes visiting exactly once each customer with respect to the previous constraints.

In the following, we make these additional common assumptions: the cost and the travel time matrices are supposed to be equal, nonnegative and to satisfy the triangle inequality. For the sake of simplicity, we also define $d_0 = 0$ and $serv_0 = 0$.

The VRPTW can then be described with the following model:

$$\text{minimize} \sum_{1 \le u \le U} \sum_{(v_i, v_j) \in A} c_{ij} x_{ij}^u \tag{1}$$

subject to

$$\sum_{1 \le u \le U} \sum_{\{v_j \in V | (v_i, v_j) \in A\}} x_{ij}^u \ge 1 \quad (v_i \in V\backslash\{v_0\}), \tag{2}$$

$$\sum_{\{v_j \in V | (v_i, v_j) \in A\}} x_{ij}^u - \sum_{\{v_j \in V | (v_j, v_i) \in A\}} x_{ji}^u = 0 \quad (v_i \in V, 1 \le u \le U), \tag{3}$$

$$\sum_{\{v_i \in V | (v_0, v_i) \in A\}} x_{0i}^u \leq 1 \quad (1 \leq u \leq U), \tag{4}$$

$$\sum_{(v_i, v_j) \in A} d_i x_{ij}^u \leq Q \quad (1 \leq u \leq U), \tag{5}$$

$$s_i^u + serv_i + c_{ij} - s_j^u + M x_{ij}^u \leq M \quad ((v_i, v_j) \in A, v_j \neq v_0, 1 \leq u \leq U), \tag{6}$$

$$s_i^u + serv_i + c_{i0} - b_0 + M x_{i0}^u \leq M \quad ((v_i, v_0) \in A, 1 \leq u \leq U), \tag{7}$$

$$a_i \leq s_i^u \leq b_i \quad (v_i \in V, 1 \leq u \leq U), \tag{8}$$

$$x_{ij}^u \in \{0, 1\} \quad ((v_i, v_j) \in A, 1 \leq u \leq U), \tag{9}$$

where x_{ij}^u and s_i^u are decision variables and M is a large number. Variables x_{ij}^u indicate whether arc (v_i, v_j) is used by vehicle u or not. For a customer v_i visited by a vehicle u_1, $s_i^{u_1}$ is the starting time of service, while s_i^u is meaningless for $u \neq u_1$. For the depot, s_0^u is the departure time of vehicle u.

Constraints (3, 4) define the route structure for the vehicles. Constraints (2) enforce the visit of every customer. Customers are allowed to be visited more than once for technical reasons explained later; this relaxation is valid since it is not optimal to visit a customer more than once (due to the triangle inequality); however, it is important to see that the model does not allow to visit a customer several times with a single vehicle. Constraints (5) and constraints (6–8) respectively concern vehicle capacity and time windows. Note that these constraints also forbid subtours in the solution. Note also that constraints (7) could be removed with a simple preprocessing on time window (preventing from reaching a costumer and not being able to return to the depot).

The VRPTW has been widely investigated in the literature. Many efficient metaheuristics have been proposed (Braysy and Gendreau 2005a,b). Most of the exact solution methods are based either on Lagrangian relaxation (Kallehauge et al. 2006), Branch and Price (Desrochers et al. 1992) or Branch and Cut (Bard et al. 2002). A detailed review can be found in Cordeau et al. (2001).

3 Branch-and-price methodology for the VRPTW

3.1 Motivation

The motivation for using a branch-and-price technique for solving the VRPTW is that the linear relaxation of model (1–9) is very weak. Hence, this model cannot be used directly with a branch-and-bound approach, except when fairly small instances are considered. To circumvent this difficulty, branch-and-price methods rely on a different model having a better linear relaxation. This new model can be obtained through a Dantzig-Wolfe decomposition from (1) to (9), as detailed in Sect. 4.3.

Let Ω be the set of feasible vehicle routes, *i.e.*, the set of paths in G issued from the depot, going to the depot, satisfying capacity and time window constraints and visiting at most once each customer. Let c_k be the cost of route $r_k \in \Omega$. Let $a_{ik} = 1$ if route

r_k visits customer v_i and 0 otherwise. Let $b_{ijk} = 1$ if route r_k uses arc (v_i, v_j) and 0 otherwise. Note that $c_k = \sum_{(v_i,v_j)\in A} b_{ijk}c_{ij}$ and that $a_{ik} = \sum_{\{v_j\in V|(v_i,v_j)\in A\}} b_{ijk}$.

The VRPTW can be described with the following set covering model:

$$\text{minimize} \sum_{r_k\in\Omega} c_k\theta_k \tag{10}$$

subject to

$$\sum_{r_k\in\Omega} a_{ik}\theta_k \geq 1 \quad (v_i \in V\backslash\{v_0\}), \tag{11}$$

$$\sum_{r_k\in\Omega} \theta_k \leq U, \tag{12}$$

$$\theta_k \in \mathbb{N} \quad (r_k \in \Omega). \tag{13}$$

where θ_k indicates whether route r_k is selected ($\theta_k = 1$) or not ($\theta_k = 0$) in the solution.

Constraints (11) corresponds to constraints (2) and guarantee that the service of customers is performed. Constraint (12) limits the number of vehicles that are used. Note that variables θ_k are defined as being integer (instead of binary) in constraint (13). The reason is to avoid constraints $\theta_k \leq 1$ in the linear relaxation of (10–13). It is of course clear that any solution with $\theta_k \geq 2$ for a route r_k would not be optimal.

The main interest of this model is to have a better linear relaxation than (1–9). Indeed, it can easily be seen that every relaxed solution of (10–13) can be transformed into a solution of the linear relaxation of (1–9), but the opposite is not true. This second point is shown with the simple example of Fig. 1. With the data presented in the figure, the set of feasible routes is limited to routes $v_0 \rightarrow v_1 \rightarrow v_0$ and $v_0 \rightarrow v_2 \rightarrow v_0$ and the optimal solution of the linear relaxation of (10–13) is formed of these two routes with a coefficient 1.0. This solution has a cost 40 and also happens to be the optimal solution of the integer program (10–13). On the contrary, the second half of the figure exhibits a feasible solution of the linear relaxation of (1–9), whose cost is 2, thus showing a large gap between the values of the linear relaxations of the two models.

This property is demonstrated more formally in Sect. 4.4.

3.2 Column generation

Unfortunately, the set covering model (10–13) is not tractable with a standard branch-and-bound approach. This stems from the fact that the size of set Ω grows exponentially with the number of customers n. Hence, the linear programs that would be used to evaluate search tree nodes would contain too many variables to be solved in a classical manner. This evaluation can however be tackled with a column generation technique, thus transforming branch-and-bound into branch-and-price.

Column generation follows the subsequent principle. In the following, we call MP (Master Problem) the linear relaxation of (10–13). We introduce $MP(\Omega_1)$, the

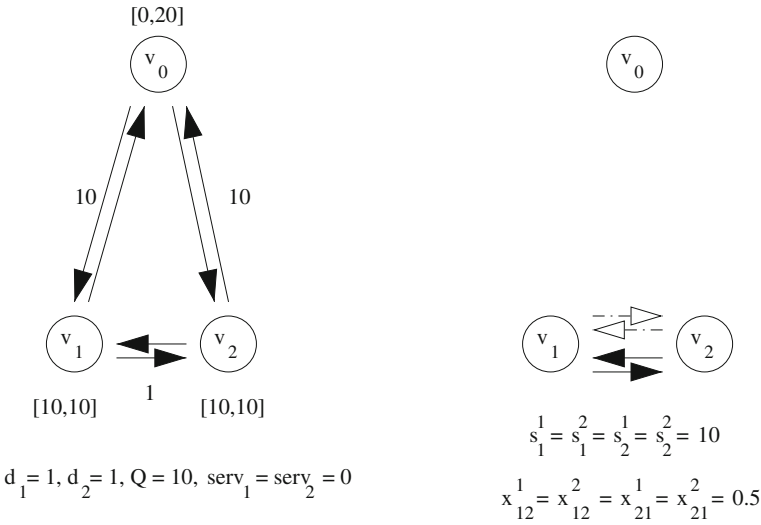

Fig. 1 Instance of the VRPTW (left part) and example of a feasible solution for the linear relaxation of (1–9) for this instance (right part)

restriction of the Master Problem MP to a subset of variables $\Omega_1 \subset \Omega$:

$$(MP(\Omega_1)) \qquad \text{minimize} \sum_{r_k \in \Omega_1} c_k \theta_k \tag{14}$$

subject to

$$\sum_{r_k \in \Omega_1} a_{ik} \theta_k \geq 1 \quad (v_i \in V \backslash \{v_0\}), \tag{15}$$

$$\sum_{r_k \in \Omega_1} \theta_k \leq U, \tag{16}$$

$$\theta_k \geq 0 \quad (r_k \in \Omega_1). \tag{17}$$

$MP(\Omega_1)$ is called the Restricted Master Problem. Let also $D(\Omega_1)$ be the dual program of $MP(\Omega_1)$:

$$(D(\Omega_1)) \qquad \text{maximize} \sum_{v_i \in V \backslash \{v_0\}} \lambda_i + U\lambda_0 \tag{18}$$

subject to

$$\sum_{v_i \in V \backslash \{v_0\}} a_{ik} \lambda_i + \lambda_0 \leq c_k \qquad (r_k \in \Omega_1), \tag{19}$$

$$\lambda_i \geq 0 \qquad (v_i \in V \backslash \{v_0\}), \tag{20}$$

$$\lambda_0 \leq 0. \tag{21}$$

Fig. 2 Column generation algorithm

```
Generate an initial set of columns Ω1
Do
    Solve MP(Ω1)
    Γ ← column(s) provided by the subproblem
    Ω1 ← Ω1 ∪ Γ
While Γ ≠ ∅
```

In this model, λ_i is the nonnegative dual variable associated with the visit of customer v_i (constraints (15)) and λ_0 is the nonpositive dual variable associated with the fleet size constraint (16). Note that keeping a set partitioning formulation for constraints (11), and equivalently an equality in constraints (2), would lead to free variables λ_i, which typically slow down the convergence of the column generation algorithm.

With these definitions, MP is identically $MP(\Omega)$ and $D(\Omega)$ is the dual program of MP. The column generation algorithm rests on the simple following property.

Property 1 *Let Ω_1 be a subset of Ω and λ^* be the optimal solution of $D(\Omega_1)$. If λ^* is feasible for $D(\Omega)$, λ^* is optimal for $D(\Omega)$.*

Proof The feasible space defined by $D(\Omega)$ is a subset of the feasible set defined by $D(\Omega_1)$, and the objective functions are identical for the two models. □

Consider the optimal solution of $MP(\Omega_1)$, obtained with the simplex algorithm for instance. This solution provides an optimal solution λ^* for $D(\Omega_1)$. If this solution is feasible for $D(\Omega)$, both $D(\Omega)$ and MP are solved. Otherwise, one or several constraints deriving from the routes of $\Omega \backslash \Omega_1$ are violated. The principle of the column generation method is to identify one or several of these constraints, by solving an appropriate optimization subproblem, in order to integrate the corresponding variables in the set Ω_1. Thus, solving alternately $MP(\Omega_1)$ and the subproblem allows to converge toward dual feasibility. The algorithm terminates when the subproblem solution attests that no violated constraint exists. Note that there is no guarantee that the generated columns improve the value of the optimal solution of $MP(\Omega_1)$. This is one of the origins of the stabilization issues discussed in Sect. 4.5. However, the algorithm is guaranteed to end in a finite number of steps, Ω being finite. The column generation algorithm is illustrated in Fig. 2.

Actually this algorithm can be viewed from different angles. First, a violated dual constraint exactly corresponds to a column with a negative reduced cost in MP, when the basic solution is the optimal solution of $MP(\Omega_1)$. Column generation then simply reproduces the simplex algorithm, except that many non-basic variables are not included in the model and that checking their reduced cost relies on a subproblem. Second, this algorithm can be interpreted as a Kelley algorithm (1960) applied to $D(\Omega)$: one searches for the optimal solution of a linear function over a convex set, defined by $D(\Omega)$; at each iteration the problem is solved on $D(\Omega_1)$; if the optimal solution is feasible for $D(\Omega)$ the algorithm stops, else one or several cuts (*i.e.*, dual constraints, *i.e.*, primal variables) are added.

An issue that have to be carefully handled is the definition of the inital set of columns. Besides the issues of efficiency, it is sometimes difficult to define a set of columns guaranteeing the feasibility of the Restricted Master Problem. A typical case is when the number of available vehicles is limited. A possibility is then to add dummy variables. It is important to see that any kind of variable can be added without interfering with the solution scheme. One just has to be sure when introducing these variables that they are going to be set to zero in the optimal integer solution (unless the problem is unfeasible). Preferably, they should also be equal to zero when linear relaxations of Restricted Master Problems are solved (unless no solutions with these variables equal to zero exist). Otherwise, the quality of the bounding scheme would be impacted negatively. Several ways can be used for defining these dummy variables. In the context of the VRPTW, one can for example add a single dummy variable with coefficient 1 in all constraints of type (11), coefficient 0 in constraint (12) and a high value in the objective function. One can rather prefer to add one variable per constraint of type (11), with similar characteristics (coefficient 1 and high value in the objective). The choice between these possibilities pertains to the convergence of the algorithm: different dummy variable definitions imply different dual constraints and possibly different dual optimal solutions (see Sect. 4.5 for a discussion on convergence issues). Basically, these variables can be interpreted as costly subcontracted routes added to Ω.

3.3 Subproblem

The purpose of the subproblem is to find routes $r_k \in \Omega \backslash \Omega_1$ such that

$$c_k - \sum_{v_i \in V \backslash \{v_0\}} a_{ik} \lambda_i^* - \lambda_0^* < 0. \tag{22}$$

This condition can be expressed as:

$$\sum_{(v_i, v_j) \in A} b_{ijk} (c_{ij} - \lambda_i^*) < 0. \tag{23}$$

Actually, it is equivalent, and more convenient as seen later, to extend the search to Ω. Indeed, every route of Ω_1 necessarily satisfies the above condition. In the following, for short, we call the routes satisfying this condition, routes with a negative reduced cost.

Expression (23) shows that the subproblem reduces to an elementary shortest path problem with resource constraints (ESPPRC). One has to find elementary paths from the depot to the depot, satisfying capacity and time constraints with a negative cost, where costs are set to $c_{ij} - \lambda_i^*$. A solution procedure based on dynamic programming is particularly well adapted to this context because it computes a set of Pareto optimal paths and might provide MP with several columns at a time. Here, we give a brief description of the algorithm used to solve the subproblem. A full description can be found in Feillet et al. (2004). Note that the requirement that the path is elementary is not trivially fulfilled here, as arc costs can be negative.

The algorithm is an extension of the classical Bellman's algorithm. The principle is to associate with each possible partial path a label and to extend these labels checking resource constraints until the best feasible paths are obtained. Dominance rules are used to compare partial paths arriving at a same location and to discard some of them. But, unlike Bellman's algorithm when no resources are considered, each vertex of the graph can maintain a large number of labels since the comparison of two labels takes into account their consumption level for each resource. Note that unfortunately this algorithm is not polynomial – actually, the problem is NP-hard in the strong sense (Dror 1994).

A common practice is to avoid computing the complete Pareto optimal path set by stopping the subproblem solution algorithm prematurely when a sufficient number of columns of negative reduced cost has been found. Actually, one has to remember that finding the path of minimal cost is only necessary at the last iteration of the algorithm, when the fact that no path with a negative reduced cost exists has to be proven. However, it is sometimes useful to compute the optimal path as it provides a so-called Lagrangian lower bound (see Lübbecke and Desrosiers 2005).

3.4 Branching scheme

Another important point of the branch-and-price method concerns the branching scheme. It is important for the constraints added through the branching rules to be compatible with the column generation phase.

The standard generic branching rule, selecting a fractional variable θ_k and deriving two branches where θ_k is respectively set to 0 and 1, would pose some difficulties. Imposing $\theta_k = 1$ is rather easy. A vehicle and the vertices visited in r_k just have to be removed. On the contrary, it is not trivial to forbid the use of a route $r_k (\theta_k = 0)$. Indeed, the subproblem would then have to restrict its search to $\Omega \backslash \{r_k\}$. This restriction could be handled by searching for the 2 shortest feasible paths in the graph, and, more generally, the $d + 1$ shortest feasible paths where d is the number of forbidden paths, which quite complicates the solution. A more efficient, and elegant, solution, detailed in, (Irnich S and Desaulniers 2005), would consist in adapting the topology of the graph, so that path r_k, and only path r_k, is removed from the set of feasible paths.

Anyway, the main reason for not using this rule is its apparent inefficiency. Indeed, it would cause a strong imbalance in the tree: while enforcing $\theta_k = 1$ is very strong, enforcing $\theta_k = 0$ only slightly limits the size of the search space. Furthermore, variables θ_k are in some way artificial and not representative of the local decisions taken by the vehicles. It is commonly admitted that the variables of the initial formulation (1–9) are much more representative and should be used for branching. The following rule has become a standard in the context of the VRPTW:

- select an arc $(v_i, v_j) \in A$ such that $0 < f_{ij} < 1$, where $f_{ij} = \sum_{r_k \in \Omega_1} b_{ijk} \theta_k$ (which corresponds to $\sum_{1 \leq u \leq U} x_{ij}^u$ in the initial formulation),
- derive two branches:
  - a branch where the arc cannot belong to the solution ($f_{ij} = 0$),
  - a branch where the arc is enforced in the solution ($f_{ij} = 1$).

For short, f_{ij} is subsequently called the flow on arc (v_i, v_j). The first type of constraints is very easy to address in the column generation scheme. At the Master Problem level, columns using arc (v_i, v_j) just have to be removed. In the subproblem phase, arc (v_i, v_j) also has to be removed, to be sure that no column using this arc is generated.

The second type of constraint is rather easily addressed too. Arcs (v_i, v_l) with $v_l \neq v_j$ or (v_l, v_j) with $v_l \neq v_i$ are to be forbidden. Indeed, constraints (11) impose that v_i and v_j are visited and we know that they will be visited exactly once. Removing these arcs leaves the sole possibility that v_i is directly followed by v_j. Columns using these arcs are removed exactly as explained above.

This branching rule relies on the property that a solution with $f_{ij} \in \{0, 1\}$ for every arc (v_i, v_j) represents a feasible set of routes. Indeed, the solution is supported by a subgraph where each customer has a degree two, which defines a set of independent routes. With this property, branching is only useful when an arc with $0 < f_{ij} < 1$ exists. Note that, unfortunately, this rule cannot be easily transposed when costumers are not necessarily visited exactly once in optimal solutions (see, e.g., Boussier et al. 2007).

An additional branching rule commonly used when the number of vehicles used in the solution is fractional, is to change the fleet size constraint. Two branches are derived, where the upper limit of the first branch (resp., lower limit of the second branch) is the current value rounded down (resp., rounded up). This rule only implies to update constraint (12). This rule is generally given a higher priority compared to the previous one, as its impact on the solution is expected to be stronger.

Note that branching might make the problem unfeasible. This situation is specially prone to happen when limiting the number of vehicles. Feasibility issues as discussed at the end of Sect. 3.2 are thus still relevant during the branching phase.

3.5 Illustration

In this subsection we give a quick illustration of the column generation algorithm. For this purpose we define a small instance of the VRPTW as follows. We consider the graph with 4 vertices depicted on Fig. 3, which also gives time windows and costs. Service times are set to 0. Demand is set to 1 for every customer, with a vehicle capacity sufficiently high (say, 10) so that the capacity constraint is irrelevant. Equivalently, the fleet is assumed oversized (say, 10 vehicles).

Let us apply column generation, starting with an initial set Ω_1 composed of 3 routes: routes r_1, r_2 and r_3 respectively visiting customer v_1, v_2 and v_3. The Restricted Master and Dual programs are then respectively:

$$
\begin{array}{lll}
\text{minimize } 2\theta_1 + 2.8\theta_2 + 2\theta_3 & & \text{maximize } \lambda_1 + \lambda_2 + \lambda_3 \\
\text{subject to} & & \text{subject to} \\
\theta_1 \geq 1 & & \lambda_1 \leq 2 \\
\theta_2 \geq 1 & & \lambda_2 \leq 2.8 \\
\theta_3 \geq 1 & & \lambda_3 \leq 2 \\
\theta_1, \theta_2, \theta_3 \geq 0 & & \lambda_1, \lambda_2, \lambda_3 \geq 0
\end{array}
$$

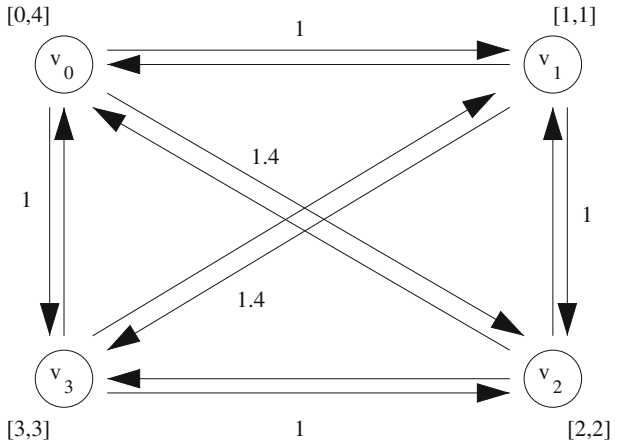

Fig. 3 Graph, costs and time windows for the illustrative example

Note that we do not include the fleet size constraint in these models, for the purpose of the example and since it is always valid here. Trivially, optimal solutions for these programs are $\theta = (1, 1, 1)$ and $\lambda = (2, 2.8, 2)$, with a value 6.8. The subproblem will then be able to find that, for example, route $v_0 \rightarrow v_1 \rightarrow v_2 \rightarrow v_0$ has a negative reduced cost equal to $3.4 - 2 - 2.8 = -1.4$. This route, called r_4, is added to Ω_1, yielding the following models:

$$\begin{array}{l} \text{minimize } 2\theta_1 + 2.8\theta_2 + 2\theta_3 + 3.4\theta_4 \\ \text{subject to} \\ \begin{array}{rl} \theta_1 \qquad + \theta_4 & \geq 1 \\ \theta_2 \quad + \theta_4 & \geq 1 \\ \theta_3 \quad & \geq 1 \\ \theta_1, \theta_2, \theta_3, \theta_4 & \geq 0 \end{array} \end{array} \qquad \begin{array}{l} \text{maximize } \lambda_1 + \lambda_2 + \lambda_3 \\ \text{subject to} \\ \begin{array}{rl} \lambda_1 \qquad & \leq 2 \\ \lambda_2 \quad & \leq 2.8 \\ \lambda_3 & \leq 2 \\ \lambda_1 + \lambda_2 & \leq 3.4 \\ \lambda_1, \lambda_2, \lambda_3 & \geq 0 \end{array} \end{array}$$

Again, one can easily see that solutions $\theta = (0, 0, 1, 1)$ and $\lambda = (2, 1.4, 2)$ are optimal for these programs, with a value 5.4. Note that, here, λ is one among many optimal dual solutions. The subproblem might then propose route $v_0 \rightarrow v_1 \rightarrow v_2 \rightarrow v_3 \rightarrow v_0$ whose reduced cost is $4 - 2 - 1.4 - 2 = -1.4$. This route, r_5, is added to Ω_1, yielding models:

$$\begin{array}{l} \text{minimize } 2\theta_1 + 2.8\theta_2 + 2\theta_3 + 3.4\theta_4 + 4\theta_5 \\ \text{subject to} \\ \begin{array}{rl} \theta_1 \qquad + \theta_4 + \theta_5 & \geq 1 \\ \theta_2 \quad + \theta_4 + \theta_5 & \geq 1 \\ \theta_3 \quad + \theta_5 & \geq 1 \\ \theta_1, \theta_2, \theta_3, \theta_4, \theta_5 & \geq 0 \end{array} \end{array} \qquad \begin{array}{l} \text{maximize } \lambda_1 + \lambda_2 + \lambda_3 \\ \text{subject to} \\ \begin{array}{rl} \lambda_1 \qquad & \leq 2 \\ \lambda_2 \quad & \leq 2.8 \\ \lambda_3 & \leq 2 \\ \lambda_1 + \lambda_2 & \leq 3.4 \\ \lambda_1 + \lambda_2 + \lambda_3 & \leq 4 \\ \lambda_1, \lambda_2, \lambda_3 & \geq 0 \end{array} \end{array}$$

Solutions $\theta = (0, 0, 0, 0, 1)$ and $\lambda = (2, 1.4, 0.6)$ are optimal, with a value 4. It can easily be checked that only two routes are missing in Ω_1 and that the reduced cost of these two routes is positive. Hence, every route in Ω has a nonnegative reduced cost and the algorithm stops. Note that in this case the solution of the linear relaxation directly provides the integer optimal solution.

4 Additional topics

4.1 From ESPPRC to SPPRC…back to ESPPRC

In the first implementation of branch-and-price for the VRPTW, Desrochers et al. (1992) transformed (10–13) to obtain a more tractable subproblem. This transformation takes advantage on the fact that the Elementary Shortest Path Problem with Resource Constraints, though NP-hard in the strong sense, admits a pseudo-polynomial algorithm when the elementary path condition is removed. Thereby, Desrochers et al. (1992) propose the two following modifications:

- Ω is enlarged to include non-elementary paths,
- a_{ik} becomes the number of times vertex v_i is visited in route r_k.

Despite these two changes, the model still provides the optimal solution of the VRPTW (non-elementary paths do not permit to improve the value of the solution). The branch-and-price scheme described above can also still be applied except that condition (23), which is still valid, now sets the subproblem as a (non-elementary) shortest path problem with resource constraints (SPPRC). The subproblem can be solved with a dynamic programming recursion similar to the one mentioned above. Even if the size of the subproblem state space increases with this modification, dominance rules are far more efficient and the global efficiency is improved. However, with this new formulation, the linear relaxation provides a weaker lower bound, which complicates the pruning of nodes during the tree search.

Actually, Desrochers et al. (1992) were more clever and excluded paths with 2-cycles, *i.e.*, paths with cycles composed of two arcs. This condition does not have any impact on the Master Problem except changing again the definition of Ω, and can be handled by the SPPRC dynamic programming solution algorithm with a low computing cost. This implementation of column generation became a standard in the papers following Desrochers et al. (1992).

Feillet et al. (2004) evaluate the tractability of maintaining the elementary path condition (Chabrier 2006 develop similar ideas, with similar results). This can be done rather easily from Desrochers et al.'s algorithm by including new resources and new dominance relations. Computationaly, it can be observed that the gain in terms of quality of the bound is sometimes decisive, but does not always counterbalance the loss in terms of computing time. Following the idea of forbidding 2-cycles, Irnich and Villeneuve (2006) rather study the removing of k-cycles. They evaluate it for $k = 3$ and $k = 4$ and conclude that the gap with the ESPPRC-based bound can significantly be tightened, with a reasonable computing time.

This collection of ideas has recently converged toward an efficient compromise solution simultaneously developed by Boland et al. (2006) and Righini and Salani (2008).

These authors propose to solve dynamically the ESPPRC by progressively adding the elementary path constraint. The SPPRC is first solved. If the set of routes of negative cost found is not empty and only contains non-elementary routes, a single-visit constraint is added to a vertex and the solution algorithm is repeated. The subproblem solution stops when an elementary route of negative cost is found or when no route (elementary or not) with negative reduced cost exists. Another recent proposition, by Desaulniers et al. (2008), is to compute a lower bound where only a subset of customers (dynamically computed) is subject to the single-visit condition.

4.2 Cutting planes

An important approach for offsetting the relative weakness of the bounds obtained when the elementary path condition is relaxed, is the dynamic generation of cuts. Kohl et al. (1999) introduced the so-called k-path cuts:

$$\sum_{v_i \in V \setminus S} \sum_{v_j \in S} f_{ij} \geq k, \tag{24}$$

where f_{ij} is the flow on arc (v_i, v_j) as defined above and S is a vertex set whose demand cannot be satisfied with $k-1$ vehicles. These cuts can easily be integrated into the Master Problem and handled by the subproblem by integrating the new dual variables into cost definitions on appropriate arcs. However, they are quite complicated to separate. Kohl et al. (1999) limit their study to k-path cuts with $k \leq 2$. The case $k = 1$ is very simple. When $k = 2$, a heuristic algorithm is proposed to find maximal sets S such that $\sum_{v_i \in V \setminus S} \sum_{v_j \in S} f_{ij} < 2$ and a Traveling Salesman Problem with Time Windows (TSPTW) solution algorithm is used to determine whether these sets can be visited by a single vehicle or not. The case $k = 3$ was addressed later by Cook and Rich (1999). These cuts were recently generalized in Desaulniers et al. (2008).

More recently, a strong interest was raised about the possibility of returning back to a set partitioning formulation for the master problem (that is, replacing constraints (11) with set partitioning constraints) and exploiting families of valid inequalities for the set partitioning polytope. While this tendancy was mainly investigated for the solution of the Vehicle Routing Problem (VRP), leading to very efficient algorithms (see, *e.g.*, Baldacci et al. 2007 for a detailed survey on these issues for the VRP), a few work also exists for the VRPTW. An important contribution to this field is the phD thesis of Spoorendonk (2008) and related publications (Jepsen et al. 2008; Spoorendonk and Desaulniers 2008; Desaulniers et al. 2009). Several families of cuts are investigated as subset row inequalities (based on the Chvatal-Gomory cutting scheme) or clique inequalities. The reader is referred to the above references for detailed discussions on these important issues.

4.3 Dantzig-Wolfe decomposition

In this subsection, we apply Dantzig-Wolfe decomposition to model (1–9) and show how model (10–13) is obtained. Set Ω, previously introduced, can be defined by:

$$\sum_{\{v_j \in V \mid (v_i, v_j) \in A\}} x_{ij} - \sum_{\{v_j \in V \mid (v_j, v_i) \in A\}} x_{ji} = 0 \quad (v_i \in V), \tag{25}$$

$$\sum_{\{v_i \in V \mid (v_0, v_i) \in A\}} x_{0i} \leq 1, \tag{26}$$

$$\sum_{(v_i, v_j) \in A} d_i x_{ij} \leq Q, \tag{27}$$

$$s_i + serv_i + c_{ij} - s_j + M x_{ij} \leq M \quad ((v_i, v_j) \in A, v_j \neq v_0), \tag{28}$$

$$s_i + serv_i + c_{i0} - b_0 + M x_{i0} \leq M \quad ((v_i, v_0) \in A), \tag{29}$$

$$a_i \leq s_i \leq b_i \quad (v_i \in V), \tag{30}$$

$$x_{ij} \in \{0, 1\} \quad ((v_i, v_j) \in A). \tag{31}$$

Seeing that (3–9) is Ω^U, (1–9) can be rewritten as follows:

$$\text{minimize} \sum_{1 \leq u \leq U} \sum_{(v_i, v_j) \in A} c_{ij} \left(\sum_{r_k \in \Omega} b_{ijk} \theta_k^u \right) \tag{32}$$

subject to

$$\sum_{1 \leq u \leq U} \sum_{\{v_j \in V \mid (v_i, v_j) \in A\}} \left(\sum_{r_k \in \Omega} b_{ijk} \theta_k^u \right) \geq 1 \quad (v_i \in V \backslash \{v_0\}), \tag{33}$$

$$\sum_{r_k \in \Omega} \theta_k^u = 1 \quad (1 \leq u \leq U), \tag{34}$$

$$\theta_k^u \in \{0, 1\} \quad (r_k \in \Omega, 1 \leq u \leq U). \tag{35}$$

In this model, constraints (34) and (35) enforce that a path r_k is selected in Ω for each vehicle u (with the corresponding variable $\theta_k^u = 1$); note that Ω includes the *null* route in which the vehicle stays at the depot. Constraint (33) enforce to visit each customer at least once. The model can be reformulated as:

$$\text{minimize} \sum_{1 \leq u \leq U} \sum_{r_k \in \Omega} c_k \theta_k^u \tag{36}$$

subject to

$$\sum_{1 \leq u \leq U} \sum_{r_k \in \Omega} a_{ik} \theta_k^u \geq 1 \quad (v_i \in V \backslash \{v_0\}), \tag{37}$$

$$\sum_{r_k \in \Omega} \theta_k^u = 1 \quad (1 \leq u \leq U), \tag{38}$$

$$\theta_k^u \in \{0, 1\} \quad (r_k \in \Omega, 1 \leq u \leq U). \tag{39}$$

Index u can be removed from the model with the following modifications:

$$\text{minimize} \sum_{r_k \in \Omega} c_k \theta_k \tag{40}$$

subject to

$$\sum_{r_k \in \Omega} a_{ik} \theta_k \geq 1 \quad (v_i \in V \backslash \{v_0\}), \tag{41}$$

$$\sum_{r_k \in \Omega} \theta_k \leq U, \tag{42}$$

$$\theta_k \in \mathbb{N} \quad (r_k \in \Omega). \tag{43}$$

Here, $\theta_k = \sum_{1 \leq u \leq U} \theta_k^u$. Constraints (42) could have been more naturally written $\sum_{r_k \in \Omega} \theta_k = U$. However, it is possible and more convenient to formulate it as proposed, since the *null* route does not interfere with constraints (41) and has a cost 0. Hence, one can easily use the variable devoted to the *null* route as a slack variable. Note again that customers are visited exactly once in optimal solutions and that, consequently, θ_k will not exceed 1 (except for the *null* route). Actually, the *null* route could equivalently be removed from Ω in this model.

More details and a more generic view on Dantzig-Wolfe decomposition can be found in Vanderbeck (2000) and Vanderbeck and Savelsbergh (2006), for example.

4.4 Column generation and Lagrangian relaxation

Column generation and Lagrangian relaxation, although seemingly very different, share a lot of similarities. Lagrangian relaxation can be applied to model (1–9) by relaxing (in a Lagrangian fashion) constraints (2). For any multiplier vector $\lambda \geq 0$, the subsequent Lagrangian problem provides a lower bound:

$$z_{D(\lambda)} = \min \sum_{1 \leq u \leq U} \sum_{(v_i, v_j) \in A} c_{ij} x_{ij}^u - \sum_{v_i \in V \backslash \{v_0\}} \lambda_i \left(\sum_{1 \leq u \leq U} \sum_{\{v_j \in V | (v_i, v_j) \in A\}} x_{ij}^u - 1 \right), \tag{44}$$

subject to (3–9).

Or, equivalently:

$$z_{D(\lambda)} = \min \sum_{1 \leq u \leq U} \sum_{(v_i, v_j) \in A} c_{ij} x_{ij}^u - \sum_{v_i \in V \backslash \{v_0\}} \sum_{1 \leq u \leq U} \sum_{\{v_j \in V | (v_i, v_j) \in A\}} \lambda_i x_{ij}^u + \sum_{v_i \in V \backslash \{v_0\}} \lambda_i, \tag{45}$$

subject to (3–9).

Reminding that (3–9) is Ω^U, the Lagrangian problem can be stated as:

$$z_{D(\lambda)} = U \min_{r_k \in \Omega} \left(c_k - \sum_{v_i \in V \setminus \{v_0\}} a_{ik} \lambda_i \right) + \sum_{v_i \in V \setminus \{v_0\}} \lambda_i . \tag{46}$$

An optimized lower bound can be found by solving the Lagrangian dual Problem: $z_{LD} = \max_{\lambda \geq 0} z_D(\lambda)$. It can be linearized introducing a new variable λ_0:

$$z_{LD} = \max_{\lambda \geq 0, \lambda_0} \left(U\lambda_0 + \sum_{v_i \in V \setminus \{v_0\}} \lambda_i \right) \tag{47}$$

subject to

$$\lambda_0 \leq c_k - \sum_{v_i \in V \setminus \{v_0\}} a_{ik} \lambda_i \quad (r_k \in \Omega). \tag{48}$$

This formulation shows that z_{LD} is exactly the value of the optimal solution of $D(\Omega)$, *i.e.*, the bound provided by the column generation scheme (note that $\lambda_0 \leq 0$ due to the *null* route). Hence, column generation and Lagrangian relaxation are two decomposition schemes where identical Master Problems and subproblems communicate with the help of values λ. In the case of column generation, these values are the dual variables of MP. In Lagrangian relaxation, these values are the Lagrangian multipliers and are guided with a different strategy.

Kallehauge et al. (2001) propose to use Lagrangian relaxation at the root node of the search tree to improve the convergence of the method. The hybridization of Lagrangian relaxation and column generation is further explored in Kallehauge et al. (2006).

4.5 Stabilization

As guessed from the above, column generation methods depend heavily on dual variables to guide the search at the subproblem level. Most of the time, marginal costs associated with customers are not appropriately estimated by dual values until the very last iterations, at least for some customers. Thus, it is possible that in some routes (*i.e.*, for some dual constraints) some customers (*i.e.*, dual variables) pick up most of the total dual values. In this case, a path that visits each of those overweighted customers will be considered a good route in the subproblem, will be added to Ω_1 and will cause some iterations that could have been avoided.

This phenomenon can be illustrated with the example of Sect. 3.5. At the third, and last, iteration, optimal dual solution $\lambda = (2, 1.4, 0.6)$ was assumed to be returned when solving the Restricted Master Problem. Other dual optimal vectors, say $\lambda = (0, 2, 2)$ for example, could have been returned instead (see restricted dual program in Sect. 3.5 for checking feasability on this vector). Then, marginal cost of customers v_2 and v_3

would have been overestimated and route $v_0 \rightarrow v_2 \rightarrow v_3 \rightarrow v_0$ would have had a negative reduced cost of value -0.6, thus implying additional iterations. In this example, the unwanted behavior occurs because the Restricted Master Problem is degenerate and thus its dual has an infinite number of optimal solutions. Such situations are all the more detrimental given that dual values returned by standard functions in LP codes are extreme points of the dual polyhedron. Hence, not only do marginal costs tend to be very badly estimated, but also dual variables tend to switch between extremely different values.

To prevent dual variables from taking extreme values, the general strategy proposed in the literature is to try to limit the distance traveled by the dual variables in the dual space from one iteration to another. A few techniques are available. A first technique consists in defining boxes around the previous values of dual variables and modifying the Restricted Master Problem so that the feasible dual space is limited to the area defined by these boxes. A second technique is to adapt the Restricted Master Problem so that the distance separating a dual solution from the previous optimal dual solution is linearly penalized. These two techniques were respectively proposed by Marsten et al. (1975) and by Kim et al. (1995) for improving the convergence of Kelley's cutting plane algorithm (1960). du Merle et al. (1999) have proposed to stabilize column generation with a method that combines these two concepts.

Although the approach of du Merle et al. (1999) is the most widely used for stabilizing column generation, several other stabilization approaches have been proposed in the literature. Neame (1999) proposes the following scheme. The dual prices used to generate new columns are obtained by taking a linear combination of the dual prices of the previous iteration and an extreme dual solution of the current Restricted Master Problem. Another approach developed in Rousseau et al. (2007) proposes to generate a set of random extreme optimal dual solutions and to provide the subproblem with a convex combination of these solutions.

More generally, the main principle of stabilization procedures is to penalize (in the Master Problem) moves from the best dual point among those visited so far, called stability center. For a general classification of these methods, one can distinguish bundle methods (Lemaréchal et al. 1995; Frangioni 2002) which use a quadratic stabilizing function penalizing dual solutions much away from the stability center, polyhedral stabilization methods which follow the same principle but rather use polyhedral stabilizing functions (du Merle et al. 1999) and center stabilization methods which consist in selecting a dual solution inside the Restricted Master Problem optimal solution space rather than an extreme point (Goffin and Vial 2002). For complete reviews and discussions on this important topic, see also Neame (1999); Lübbecke and Desrosiers (2005); Vanderbeck (2005) or Briant et al. (2008).

References

Baldacci R, Toth P, Vigo D (2007) Recent advances in vehicle routing exact algorithms. 4OR 5(4):269–298

Bard JF, Kontoravdis G, Yu G (2002) A branch-and-cut procedure for the vehicle routing problem with time windows. Transport Sci 36(2):250–269

Boland N, Dethridge J, Dumitrescu I (2006) Accelerated label setting algorithms for the elementary resource constrained shortest path problem. Oper Res Lett 34(1):58–68

Boussier S, Feillet D, Gendreau M (2007) An exact algorithm for team orienteering problems. 4OR 5:211–230

Braysy O, Gendreau M (2005a) Vehicle routing problem with time windows, part i: route construction and local search algorithms. Transport Sci 39(1):104–118

Braysy O, Gendreau M (2005b) Vehicle routing problem with time windows, part ii: metaheuristics. Transport Sci 39(1):119–139

Briant O, Lemaréchal C, Meurdesoif P, Michel S, Perrot N, Vanderbeck F (2008) Comparison of bundle and classical column generation. Math Program 113(2):299–344

Chabrier A (2006) Vehicle routing problem with elementary shortest path based column generation. Comput Oper Res 33:2972–2990

Cook W, Rich JL (1999) A parallel cutting-plane algorithm for the vehicle routing problem with time windows. Technical report TR99-04, Department of Computational and Applied Mathematics, Rice University

Cordeau J-F, Desaulniers G, Desrosiers J, Solomon MM, Soumis F (2001) The VRP with time windows. In: Toth P, Vigo D (eds) The vehicle routing problem, SIAM monographs on discrete mathematics and applications. pp 157–194

Desaulniers G, Desrosiers J, Solomon MM (2001) Accelerating strategies in column generation methods for vehicle routing and crew scheduling problems. In: Ribeiro CC, Hansen P (eds) Essays and surveys in metaheuristics. Kluwer, Boston pp 309–324

Desaulniers G, Desrosiers J, Solomon MM (eds) (2005) Column generation. GERAD 25th Anniversary Series. Springer

Desaulniers G, Desrosiers J, Spoorendonk S (2009) Cutting planes for branch-and-price algorithms. Technical report G-2009-52, GERAD, Canada

Desaulniers G, Lessard F, Hadjar A (2008) Tabu search, partial elementarity, and generalized k-path inequalities for the vehicle routing problem with time windows. Transport Sci 42(3):387–404

Desrochers M, Desrosiers J, Solomon MM (1992) A new optimization algorithm for the vehicle routing problem with time windows. Oper Res 40(2):342–354

Dror M (1994) Note on the complexity of the shortest path models for column generation in VRPTW. Oper Res 42(5):977–978

du Merle O, Villeneuve D, Desrosiers J, Hansen P (1999) Stabilized column generation. Discrete Math 194(1-3):229–237

Feillet D, Dejax P, Gendreau M, Gueguen C (2004) An exact algorithm for the elementary shortest path problem with resource constraints: application to some vehicle routing problems. Networks 44(3):216–229

Frangioni A (2002) Generalized bundle methods. SIAM J Optim 13(1):117–156

Goffin JL, Vial JP (2002) Convex nondifferentiable optimization: a survey focused on the analytic center cutting plane method. Optim Methods Softw 17(5):805–867

Irnich S, Desaulniers G (2005) Shortest path problems with resource constraints. In: Desaulniers G, Desrosiers J, Solomon MM (eds) Column generation, GERAD 25th anniversary series, chap 2. Springer, pp 33–65

Irnich S, Villeneuve D (2006) The shortest path problem with k-cycle elimination for k≥3. Transport Sci 18(3):391–406

Jepsen M, Petersen B, Spoorendonk S, Pisinger D (2008) Subset row inequalities applied to the vehicle routing problem with time windows. Oper Res 56(2):497–511

Kallehauge B, Larsen J, Madsen OBG (2001) Lagrangean duality applied on vehicle routing with time windows—experimental results. Technical report IMM-TR-2001-9, IMM, Technical University of Denmark

Kallehauge B, Larsen J, Madsen OBG (2006) Lagrangian duality applied to the vehicle routing problem with time windows. Comput Oper Res 33(5):1464–1487

Kelley JE (1960) The cutting plane method for solving convex programs. J SIAM 8:703–712

Kim S, Chang K-N, Lee J-Y (1995) A descent method with linear programming subproblems for nondifferentiable convex optimization. Math Program 71:17–28

Kohl N, Desrosiers J, Madsen OBG, Solomon MM, Soumis F (1999) 2-path cuts for the vehicle routing problem with time windows. Transport Sci

Lemaréchal C, Nemirovskii A, Nesterov Y (1995) New variants of bundle methods. Math Program 69: 111–149

Lübbecke ME, Desrosiers J (2005) Selected topics in column generation. Oper Res 53(6):1007–1023

Marsten RE, Hogan WW, Blankenship JW (1975) The BOXSTEP method for large-scale optimization. Oper Res 23:389–405

Neame PJ (1999) Nonsmooth dual methods in integer programming. PhD thesis, University of Melbourne

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

Rousseau L-M, Gendreau M, Feillet D (2007) Interior point stabilization for column generation. Oper Res Lett 35:660–668

Spoorendonk S (2008) *Cut and column generation*. PhD thesis, Technical University of Denmark

Spoorendonk S, Desaulniers G (2008) Clique inequalities applied to the vehicle routing problem with time windows. Technical report G-2008-72, GERAD, Canada. Forthcoming in INFOR

Vanderbeck F (2000) On Dantzig-Wolfe decomposition in integer programming and ways to perform branching in a branch-and-price algorithm. Oper Res 48(1):111–128

Vanderbeck F (2005) Implementing mixed integer column generation. In: Desaulniers G, Desrosiers J, Solomon MM (eds) Column generation, GERAD 25th anniversary series, chap 12. Springer, pp 331–358

Vanderbeck F, Savelsbergh MWP (2006) A generic view of Dantzig-Wolfe decomposition in mixed integer programming. Oper Res Lett 34:296–306