Constraint violation in vehicle routing

Matteo Salani **Lorenzo Ruinelli**

Luca Maria Gambardella

IDSIA / USI-SUPSI, Dalle Molle Institute for Artificial Intelligence

Manno-Lugano, Switzerland

Email: matteo.salani@idsia.ch

Abstract

Nowadays, automated route planning models and algorithms are supporting decision makers in designing efficient delivery plans. Anyway, in practical applications the management of constraints is not as rigorous as in mathematical models. Human planners tend to evaluate solutions with a potential violation of some constraints if they estimate to obtain some clear benefit from that. In this paper we propose an alternative model to deal with constraint violation inspired by practice. It overcomes strict management of constraints as done in classical mathematical models and does not over-burden decision makers with an excessive number of parameters. The proposed model is formed of two-levels: the first level solves the nominal problem, i.e., without constraint violations. The second level minimizes the violation of constraints and imposes a minimal improvement on the primary objective with respect to the optimal solution of the nominal problem. To obtain an optimal solution of the presented model, we propose a non-trivial implementation of a branch-and-cut-and-price procedure. Furthermore, we introduce and discuss a novel embedded relaxation procedure which is efficiently used to identify unfeasible nodes of the search tree. A computational experience over a set of 120 instances is presented and optimal results on all instances are reported.

1 Introduction

In the logistic industry, route planning is a relevant decision process for shipping companies. Indeed, optimized decisions allow for substantial savings in transportation costs. The academic counterpart of the route planning process has been introduced with the Vehicle Routing

Problem (VRP) more than fifty years ago (Dantzig and Ramser, 1959). Since then, the gap between practical applications and academic models has been continuously reduced. Nowadays, models and solution algorithms are abundant in the arsenal of available quantitative methods to solve practical variants of the VRP (Toth and Vigo, 2002; Golden et al., 2008; Vidal et al., 2012).

When real-world routing is concerned, the number of practical features to be modeled is impressive. Opposed to the VRP, the fleet of vehicles of real-world routing problems is generally heterogeneous, meaning that different vehicle types are considered. Vehicles may posses specific characteristics such as refrigerated units, hydraulic load lifters and so on, which are required to perform deliveries. Additionally, vehicle's capacity is multidimensional, i.e., lower and upper limits are associated with volume, weight and autonomy. For recent references on heterogeneous VRP, consider Baldacci et al. (2009) and Bettinelli et al. (2011). Another real-world feature that attracted a remarkable research effort is the definition of time windows constraints (Schrage, 1981; Gendreau and Tarantilis, 2010). Real-world routing often considers multiple time windows associated with morning and afternoon shifts (Christiansen and Fagerholt, 2002; Ceselli et al., 2009). For what concerns the modeling of loading operations, several modeling advances need to be mentioned: pick-up and delivery (Savelsbergh and Sol, 1995), simultaneous delivery and collection (Bianchessi and Righini, 2007; Dell'Amico et al., 2006) and vehicle routing with back-hauls (Goetschalckx and Jacobs-Blecha, 1989; Toth and Vigo, 1997). Finally, the option of continuous or discrete delivery splitting has received increasing attention in the last years (Archetti et al., 2008; Salani and Vacca, 2011).

Our past experiences in real-world routing and a recent collaboration with a logistic company (Ruinelli et al., 2012), made us reflect upon the differences between the management of constraints in real applications and in mathematical models. In the context of the mentioned collaboration, we had access to handmade routing plans. We observed that most of the constraints defined in our mathematical model were violated by handmade solutions. The large majority of the violations were minor ones but some of them were substantial as if some of the constraints were intentionally violated. For example, time windows constraints were violated with major violations of up to three hours. The experience of human planners and their sensibility on every particular case, let them decide for some violations, even major ones because they evaluated the benefit associated with each violation. Our opinion is that it is very difficult for mathematical models to embrace such experience and sensibility in a compact and manageable form. In this work we intend to fill part of this gap by designing a mathematical model for explicit constraint violation and providing an exact optimization algorithm based

on branch-and-cut-and-price. Furthermore, we introduce and discuss an embedded model relaxation which is efficiently used to identify unfeasible nodes of the search tree.

In the reminder of the paper, we focus on the violation of time windows constraints even if the proposed method is rather general. The reason is that time windows is the most studied type of constraints. Time windows are defined as a time interval in which the delivery service is allowed to start. Furthermore, time windows constraints often model preferences rather than physical limitations and therefore are more subject to violations.

Commonly, the violation of time windows is discouraged by some fixed or proportional penalties. We refer to this set of problems as VRP with soft time windows (VRPSTW). Heuristics (Koskosidis et al., 1992) as well as exact approaches based on column generation and branch-and-price (Liberatore et al., 2011) have been proposed.

A modeling restriction to VRPSTW has been addressed in Qureshi et al. (2010) where authors consider the option of late arrivals only. Both an exact approach based on column generation and a heuristic method based on two genetic algorithms are presented. Instances with up to 75 nodes can be solved to optimality.

Recently, Tas et al. (2013) proposed a tabu search algorithm to solve the VRP with soft time windows and stochastic travel times. Lu and Yu (2012) extended the model of soft time windows to Pick-Up and Delivery VRP and proposed a genetic algorithm and presented a computational experience conducted on randomly generated instances derived from the Solomon's data set for VRPTW. Finally, Ibaraki et al. (2008) solves VRPSTW with convex and piece-wise linear penalty functions using local search. Optimal visiting time, given an order of customers, is computed using dynamic programming. The computational campaign is quite comprehensive as instances from the datasets by Solomon (1983) and Gehring and Homberger (2002) have been considered. Results are compared with other 8 different methods obtaining the best performance on large instances.

In all mentioned papers, the proposed approach to deal with time windows violation is based on the concept of soft constraints. Indeed, the violation of time windows is penalized by some sort of penalty function. While the idea of defining soft constraints for constraint violation is quite flexible, for example it is widely used in scheduling of workforce (see, e.g., Berrada et al. (1996)), it may baffle human planners when asked to set appropriate penalties for every unit of violation. Conceptually, planners should provide a cost value for every minute of early or late time windows violation. Additionally, as we will show in the computational experience of this paper, behavior of optimal solutions is undesirably fluctuating with respect to each of the objectives (time windows violation and cost) for different instances and

the same settings of the parameters. Therefore, with a soft constraint approach, the concept optimality of the primary objective is lost.

An alternative way of modeling the concept of constraint violation has been introduced in chance constrained programming (CCP) (Charnes and Cooper, 1959). In CCP, the presence of stochastic data may render the optimal solution of the deterministic version of the problem solved considering average coefficients unfeasible for some realization of the uncertain parameters. In CCP, the odds of not violating a set of constraints is bounded to be above a requested threshold. The solution of CCP problems is commonly more complex than the original problem as all realizations of uncertain parameters should be considered. To mitigate this drawback, the sample average approximation method (SAA)has been successfully applied to several problems modeled with CCP (Pagnoncelli et al., 2009). Anyway, to the best of our knowledge no attempt to model complex VRP with time window as a CCP and solve it with SAA exists. Additionally, in this paper we address the deliberated choice of violating the time windows constraint by a certain amount and we do not consider any uncertainty in the data such as stochastic travel times. Therefore, an approach based on CCP is not appropriate in our context.

Our first attempt to address constraint violation was to extract some knowledge from handmade plans. We analized constraint violation and observed some regular patterns. We observed that working time and driving time were systematically violated. Also, maximum distances between stops were often violated. We extracted some distribution for constraint violation and relaxed constraints by the mode of these distributions. In particular we relaxed the working time and driving time constraints by 3% and maximum distances between stops by 15%. Solutions were not completely satisfactory for our industrial partner as in some cases constraint violations were permitted in order to save few euros.

In this paper we propose an alternative model which is expected to be more acceptable for decision makers. Our goal is to design a method which: requires a limited number of parameters; preserves the concept of optimality by keeping a clear target (e.g., cost); can be solved as efficiently as other constraint violation methods, such as soft constraints.

In section 2 we provide a mathematical formalization of our approach. We propose an exact algorithm based on branch-and-price-and-cut in section 3. Section 4 presents computational experiments performed on a test set derived from the well known Solomon's data set for the VRPTW (Desaulniers et al., 1997). Some final remarks are reported in section 5.

2 A mathematical model for minimal time windows violation

We recall the definition of the VRPTW. A graph $G(V,A)$ is given, where the set of vertices $V = N \cup \{0\}$ is composed of a set of N vertices representing the customers and a special vertex $\{0\}$ representing the depot. Non-negative weights t_{ij} and c_{ij} are associated with each arc $(i,j) \in A$; representing the traveling time and the transportation cost, respectively; traveling times satisfy the triangle inequality. A positive integer demand d_i is associated with each vertex $i \in N$ and a capacity Q is associated with each vehicle of a set K of available vehicles. A non-negative integer service time s_i and a time window $[a_i, b_i]$, defined by two non-negative integers, are also associated with each vertex $i \in N$. The problem asks to find a set of up to $|K|$ routes with overall minimum cost visiting all customers at most once and respecting time windows and vehicles' capacity.

In classical VRPTW, in case of early arrival at customer's location, the vehicle has to wait until the opening of the time window, a_i. In VRP with Soft Time Windows (VRPSTW) (Liberatore et al., 2011), constant or proportional penalties are incurred for early of late service. However, in both cases vehicles are allowed to wait at no cost at customer's location. In other words, the instant at which the delivery service starts can be chosen to lie between the arrival and the opening of the time window.

On the contrary in our model, we propose the opportunity for time windows violation suggested by practice: in case of early arrival, drivers have the option to request an immediate early service or to wait for the opening of the time window in case of early arrival. Late services are also allowed to take place up to a constant delay.

We recall a valid formulation for the VRPTW without time windows violation as a set covering problem:

$$z_\Omega = minimize \sum_{r \in \Omega} c^r x^r \tag{1}$$

$$s.t. \sum_{r \in \Omega} f_i^r x^r \geq 1 \qquad \forall i \in N \tag{2}$$

$$\sum_{r \in \Omega} x^r <= |K| \tag{3}$$

$$x^r \in \{0,1\} \qquad \forall r \in \Omega \tag{4}$$

where Ω is the set of feasible vehicle routes in which customers are visited within their

time windows and the vehicle's capacity is not exceeded. c^r is the cost of route r that is obtained as the sum of the arcs in the route. f_i^r is the number of times route r visits customer i.

Model (1)-(4) is also suitable for VRPSTW by extending the set Ω to routes violating time windows constraints but respecting vehicles' capacity. Indeed, when time windows are violated penalties for early and late services are directly accounted in the cost of the route c^r.

Instead, for our variant of VRPTW we propose a two-level model. The model for the first level is (1)-(4) in which no violation of time windows is permitted. Therefore it equals the model presented for VRPTW. The second level assumes that the optimal value z_Ω^* is known and strictly positive. The model for the second level reads as follows:

$$g_\Theta = minimize \sum_{r\in\Theta} v^r x^r \tag{5}$$

$$s.t. \sum_{r\in\Theta} f_i^r x^r \geq 1 \qquad \forall i \in N \tag{6}$$

$$\sum_{r\in\Theta} c^r x^r \leq (1-\beta)\cdot z_\Omega^* \tag{7}$$

$$\sum_{r\in\Theta} x^r <= |K| \tag{8}$$

$$x^r \in \{0,1\} \qquad \forall r \in \Theta \tag{9}$$

where Θ is the set of feasible vehicle routes in which the vehicle's capacity is not exceeded. v^r is the overall time window violation of route r. The cost of route r is accounted in constraint (7) which states that the overall cost of the solution must be strictly lower than the cost of the optimal solution without time windows violation by at least a certain amount.

The basic idea of the two-level model is that given the cost of the optimal solution when all constraints are fulfilled, planners are more facilitated in imposing a minimal objective improvement for accepting and evaluating a constraint violation. As requested, the objective function improvement β is the only additional parameter that is required. Furthermore the role of the primary objective is preserved meaning that there is a clear distinction between the primary objective (cost) and the secondary one (time windows violation). The model can be seen as a slight modification of the lexicographic approach for bi-objective optimization (Steuer, 1986). The difference is that in the lexicographic approach no improvement is requested in the primary objective and the optimization is carried over the same set of feasible solutions.

3 Branch-and-price-and-cut with embedded relaxation

Models (1)-(4) and (5)-(9) may contain a number of variables which grows exponentially with the size of the instance and cannot be dealt with explicitly. Therefore, to compute a valid lower bounds, we solve the linear relaxation of the two models recurring to a column generation procedure. Column generation is an implicit enumeration scheme that recurs iteratively to a pricing problem to prove optimality or to price out new profitable variables. In order to obtain feasible integer solutions we embed the column generation bounding procedure into an enumeration tree (Desaulniers et al., 2005; Vanderbeck and Wolsey, 1996).

In particular, we relax integrality conditions (4) and (9) and consider a subset of variables $\tilde{\Omega}$ and $\tilde{\Theta}$. We remark that $\Omega \subseteq \Theta$ but not necessarily $\tilde{\Omega} \subseteq \tilde{\Theta}$.

We focus our attention to the solution of model (5)-(9) which is more interesting for our application and we refer to it as the Restricted Master Problem (RMP). The solution procedure for model (1)-(4) is quite similar. In order to guarantee the feasibility of the linear relaxation of RMP we add to the model a set of artificial variables with a sufficiently large coefficient M in the objective function and positive and negative unitary coefficients to constraints (6) and (7), respectively. At each column generation iteration the linear relaxation of the RMP is solved, and we search for new columns with negative reduced cost. The reduced cost of each column $r \in \Theta$ is:

$$\bar{v}^r = v^r - \sum_{i \in N} f_i^r \pi_i - c^r \rho - \gamma \tag{10}$$

where π_i is the nonnegative dual variable associated to the ith constraint of the set (6), ρ is the nonpositive dual variable associated with the threshold constraint (7) and γ is the nonpositive dual variable associated with constraint (8). If columns with negative reduced cost are found, they are inserted into the RMP and the process is iterated; otherwise, the optimal solution of the linear relaxation of the RMP is also an optimal solution of the linear relaxation of the MP and represents a valid lower bound to (5)-(9).

The pricing problem can be modeled as a resource constrained elementary shortest path problem (RCESPP). The RCESPP is the problem of finding the minimum cost elementary path and is strongly NP-hard Dror (1994), since the underlying graph may have negative cost cycles. The underlying graph $G(V,A)$ is made of a set V of vertices and a set of A arcs. Let N be the set of items. Then $V = N \cup \{s,t\}$, where s and t are special vertices representing

the depot. A non-negative prize π_i is associated with each vertex $i \in N$. A vehicle must go from s to t, visiting a subset of the other vertices; no cycles are allowed. The objective is to minimize the cost, given by the sum of the costs of the arcs traversed minus the sum of the prizes collected at the vertices visited. The basic dynamic programming approach to the RCESPP is based on the algorithm devised by Desrosiers et al. (1981) for the RCSPP (an extension of the Bellman-Ford algorithm with the addition of resource constraints). The algorithm assigns states to each vertex: each state of vertex i represents a path from s to i . Each state has an associated resource consumption vector R and each component of R represents the consumption of a different resource along the path. R encodes the set of visited items as a binary vector which ensures path's elementarity. Each state has an associated cost C computed according to (10) and the optimal solution is given by the minimum cost state associated with t.

In our implementation we extended the algorithms presented in Righini and Salani (2006) and Righini and Salani (2008) in order to model early visits or waiting decisions: when a vehicle arrives at the location of customer i before time windows opening, i.e., at time $T_i < a_i$, two new states are generated. The first state models early visit decision. Therefore, the associated cost C is increased by $a_i - T_i$. The second state models a waiting decision at no additional cost and $T_i := a_i$. We remark that the two new states do not dominate each other. In time or late arrivals do not generate two new states as there is no advantage of postponing the service at any further instant in time.

The presented pricing algorithm has an exponential worse case time complexity. As commonly done in this cases, we devised 3 pricing algorithms of increasing complexity: a greedy pricing (GP), a randomized greedy pricing (RGP) and a heuristic dynamic programming pricing (DPH). When the lower complexity pricing fails in finding new columns, the next is executed. If no column is produced, the exact pricing is solved and eventually the optimality is proven.

Additional accelerating techniques have been implemented. Namely, dual space stabilization inspired by Addis et al. (2012) and valid k-path inequalities by Kohl et al. (1999).

In the search tree, branching is required when the optimal solution of RMP master problem is fractional. Anyway, it is not a good practice to branch directly on route variables as this results in an unbalanced search tree. Therefore, we obtain the corresponding non integral arc-flow formulation and devise a branching scheme according to it. In the corresponding three index arc-flow formulation, binary variables y_{ij}^k are defined for each vehicle k and arc $(i, j) \in A$. y_{ij}^k assumes value 1 if vehicle k travels directly from node i to node j, 0 otherwise.

Our branching scheme is standard:

- if the number of vehicles corresponding to the left hand side of constraint (8) is fractional, ($\sum_{r\in\tilde{\Theta}} x^r = \tilde{K}$), branching is performed by enforcing $\sum_{r\in\tilde{\Theta}} x^r \leq \lfloor \tilde{K} \rfloor$ on the first child node and $\sum_{r\in\tilde{\Theta}} x^r \geq \lceil \tilde{K} \rceil$ on the second child node.
- if there is an arc $(i,j) \in A$ visited a fractional number of times ($\sum_{k\in K} y_{ij}^k = \tilde{y}_{ij}$), branching requires additional constraints in the master problem: $y_{ij} \leq \lfloor \tilde{y}_{ij} \rfloor$ on the first child node and $y_{ij} \geq \lceil \tilde{y}_{ij} \rceil$ on the second child node. This branching requires the dual values associated with the additional constraints to be collected and accounted in the pricing; however, the structure of the pricing problem is not affected.

After some preliminary computational experiments, we observed that the solution of model (5)-(9) required a substantially greater amount of computational time w.r.t the time required to solve the original counterpart without time windows violation. In some instances the time required was one or more orders of magnitude higher. The reasons for this increased computational effort are the following:

- The set of feasible columns Θ has bigger size than Ω as it contains also routes violating time windows constraints.
- The exact solution of the pricing problem is harder because two new non-dominated states are generated at each label extension. Therefore the overall number of generated labels is greater.
- For all routes $r \in \Theta \cap \Omega$, $v_r = 0$. This increases both the complexity of the column generation procedure and the pricing algorithm. In the dynamic programming algorithm, much more labels result non dominated.
- Proving the infeasibility of a node in the search tree, in particular with respect to constraint (7) is hard and requires additional effort. Indeed, a linear relaxation of the RMP may satisfy constraint (7) but no integral solution does.

In order to improve the overall performance of the branch-and-price-and-cut algorithm we propose to integrate a model relaxation into the procedure. We obtain the relaxation by exploiting model (1)-(4) and the following properties:

Property 1. Let us define z_Θ as the model (1)-(4) in which the set Ω is substituted with Θ. We observe that z_Θ is a valid relaxation of z_Ω. Therefore, $z_\Theta^* \leq z_\Omega^*$. The proof is trivial as $\Omega \subseteq \Theta$.

Property 2. A subtree rooted at any node of the search tree for model (5)-(9) can be pruned if $z^*_\Theta > (1-\beta)\cdot z^*_\Omega$. The proof is also trivial as z^*_Θ represents the best value of any integer solution in the subtree related to the primary objective $\sum_{r\in\Theta} c^r x^r$ and it is a valid relaxation of z^*_Ω for the same subtree.

The above properties lead to an implementation of an embedded relaxation for the branch-and-price-and-cut procedure. The procedure is described in algorithm 1. The outer loop applies the branch-and-price-and-cut to solve model (5)-(9). At every iteration one node of the search tree is analyzed to compute a valid lower bound LB_{g_Θ}. Before doing so, the model is relaxed following properties 1 and 2 and an embedded search tree is initialized. The coefficients in the objective function of columns already present in the RMP are changed from v^r to c^r and constraint (7) is relaxed. The column generation procedure computes a lower bound LB_{z_Θ} by adding both columns and valid cuts in a standard way. When the embedded search tree has been explored, the optimal value z^*_Θ (or any valid lower bound $LB_{EmbTree(z_\Theta)}$) is used to validate the feasibility of the node under examination. When feasible, the node is analysed to compute a lower bound LB_{g_Θ} via column generation, otherwise it can be pruned according to property 2. At the end of both column generation procedures, in case of fractional optimal solutions for the linear relaxation of the RMP, the branching scheme introduced above is applied.

We remark that columns and cutting planes are added to the same RMP by both column generation procedures. Columns and cuts are kept throughout the entire procedure. Indeed, columns belong to the set Θ and cutting planes are valid for both models (1)-(4) and (5)-(9). This way of dealing with columns and cuts related to the same "convexified" polyhedron but obtained using diverse objective functions and corresponding dual vectors can be put in relation with the way the box method for stabilized column generation is obtained: the objective function is perturbed to obtain different more promising dual vectors for pricing more profitable columns (du Merle et al., 1999). In our case our attention is more on proving the feasibility of the nodes of the search tree in presence of some sort of knapsack constraints. We believe that the proposed embedded relaxation approach can be extended and used for many problems solved via branch-and-price where the unfeasibility of some nodes of the search tree may slow down the solution for standard implementations.

Algorithm 1 Embedded relaxation

Require: $\tilde{\Theta}$, z^*_Ω
  $Tree_\Theta$:= Initialize root node; $g^*_\Theta := +\infty$; $LB_{g_\Theta} := 0$;
  while $Tree_\Theta \neq \emptyset$ **do**
    $Node_\Theta$:= extract node($Tree_\Theta$);
    $EmbTree_\Theta$:= Initialize root node ($Node_\Theta$); $z^*_\Theta := +\infty$; $LB_{z_\Theta} := 0$;
    Adjust coefficients in RMP from v^r to c^r;
    Relax constraint (7);
    while $EmbTree_\Theta \neq \emptyset$ **do**
      $EmbTreeNode_\Theta$:= extract node($EmbTree_\Theta$);
      LB_{z_Θ} := Solve Column Generation (c^r, $EmbTreeNode_\Theta$);
      if Integral(RMP) **and** $LB_{z_\Theta} < z^*_\Theta$ **then**
        $z^*_\Theta := LB_{z_\Theta}$;
      else if $LB_{z_\Theta} < z^*_\Theta$ **then**
        Branch($EmbTree_\Theta$);
      else
        Prune($EmbTreeNode_\Theta$);
      end if
    end while
    if $z^*_\Theta \leq (1-\beta) \cdot z^*_\Omega$ **then**
      LB_{g_Θ} := Solve Column Generation (v^r, $Node_\Theta$);
      if Integral(RMP) **and** $LB_{g_\Theta} < g^*_\Theta$ **then**
        $g^*_\Theta := LB_{g_\Theta}$;
      else if $LB_{g_\Theta} < g^*_\Theta$ **then**
        Branch($Tree_\Theta$);
      else
        Prune($Node_\Theta$);
      end if
    else
      Prune($Node_\Theta$);
    end if
  end while

4 Computational results

We performed our experiments using the well-known Solomon's data set (Solomon, 1983). For all instances of classes R1 and RC1 we considered the first $n = 25$ customers. Therefore, our initial data set is made of 20 instances. We obtained 20 additional instances by restricting the vehicle's capacity to 100 units while in the original instances vehicle's capacity equals 200 units.

All tests were performed on a PC equipped with an Intel Core i7 2.67 GHz 2 Cores processor with 3 GB RAM. The branch-and-price-and-cut is coded in ANSI-C and the linear relaxation solver is IBM-Cplex 12.0

Optimal solutions for z_Ω have been computed using the code by Salani and Vacca (2011) with an average computational time of 10 seconds per instance. Anyway, optimal solutions for original instances are also available on the internet (see, e.g., `http://w.cba.neu.edu/~msolomon/problems.htm`). In our experiments we require a minimal improvement in the primary objective of 1%, 5% and 10%. The budget allowed for maximal time windows violation equals 15 minutes per customer. Even if this limit seems restrictive it results in a more

interesting set of instances as some of them are unfeasible with respect to constraint (7) especially those with requested improvement of 10%. Finally, the overall number of instances amounts to 120. A time limit of ten hours has been imposed on all runs.

Tables 1 and 2 report on the exact solution of model (5)-(9) with the branch-and-price-and-cut procedure designed in section 3. Tables report on the minimal time windows violation necessary to achieve the required primary objective improvements and compare the improvement in performances achieved by using the embedded relaxation method.

Both tables 1 and 2 are organized as follows: the first column reports on the instance name followed by the optimal value without time windows violation z^*_Ω; the following three sets of columns are related to primary object improvement 1%, 5% and 10%, respectively. For 1% requested improvement we report the minimal time windows violation, g^*_Θ, the time required to prove its optimality or that no optimal solution exists by a regular implementation of the branch-and-price-and-cut procedure, $T_{g^*}(s)$. Finally, we report the time required by the implementation with embedded relaxation, $T^E_{g^*}(s)$. For 5% and 10% requested improvement we do not report the time required by the regular implementation as it is always dominated by the embedded relaxation procedure and most of the time did not converge within the time limit.

When column g^*_Θ contains an asterisk $(*)$ it means that the corresponding instance is unfeasible with respect to constraint (7). A dash line $(-)$ in the cell related to the computational time means that the corresponding procedure did not converge within the time limit. Whenever both procedures did not converge within the time limit, the column g^*_Θ reports on the best available upper bound if any exists. With a set of additional tests conducted without time limits we obtained the proof of optimality of all reported results for g^*_Θ.

Observing table 1 we remark that the branch-and-price-and cut with embedded relaxation was able to find an optimal solution or that no one exists for 59 out of 72 instances. The introduction of the embedded relaxation helped to speed up the procedure. This procedure managed to solve 11 instances which were not solved by the standard implementation within the time limit. Despite of that the required computational time remains substantial: several hours are required to prove optimality. We recall that the optimal solution for the problem without time windows violation requires a matter of seconds to be computed.

Similar observations can be drawn from table 2. In this set of instances the presence of unfeasible instances is much more evident. All RC instances with capacity 100 with requested improvement of 5% and 10% are unfeasible. The embedded relaxation method is able to detect unfeasibility in a matter of fraction of secods. We observe that the standard imple-

Instance	z^*_Ω	Improvement 1%			Improvement 5%		Improvement 10%	
		g^*_Θ	$T_{g^*}(s)$	$T^E_{g^*}(s)$	g^*_Θ	$T^E_{g^*}(s)$	g^*_Θ	$T^E_{g^*}(s)$
r101_200	617.10	7.00	3.38	2.67	21.80	5.91	41.40	2.56
r102_200	547.10	2.40	65.12	38.25	2.40	68.23	29.40	1648.68
r103_200	454.60	7.00	-	16756.00	23.60	35905.99	*	0.22
r104_200	416.90	6.70	-	-	43.30	-	*	6.47
r105_200	530.50	2.20	23.65	12.20	19.60	30.03	48.90	41.98
r106_200	465.40	11.00	-	17185.00	21.00	11821.03	57.10	16205.24
r107_200	424.30	4.60	-	-	47.30	-	*	0.92
r108_200	397.30	2.30	-	-	27.60	-	*	2.4
r109_200	441.30	20.90	-	3079.91	*	2.09	*	2.11
r110_200	444.10	0.10	-	15964.30	10.20	22045.98	*	8.67
r111_200	428.80	3.90	-	28277.10	22.30	-	*	3.21
r112_200	393.00	21.00	-	-	59.00	-	*	0.89
r101_100	617.10	7.00	3.11	2.37	21.80	5.47	41.40	3.08
r102_100	547.10	2.40	87.14	19.94	2.40	48.87	29.40	1325.86
r103_100	454.60	7.00	-	11908.91	23.60	19055.03	*	0.2
r104_100	416.90	6.70	-	-	*	0.69	*	0.69
r105_100	530.50	2.20	28.88	10.8	19.60	30.4	48.90	41.18
r106_100	465.40	11.00	-	12798.32	25.80	7356.04	57.10	14056.04
r107_100	428.40	4.20	-	-	*	0.67	*	0.75
r108_100	403.20	2.30	-	-	*	1.79	*	1.88
r109_100	441.30	20.90	-	1408.5	*	0.86	*	0.89
r110_100	444.10	0.10	-	10415.33	10.20	11185.59	*	1.33
r111_100	428.80	3.90	-	26795.29	35.50	-	*	0.68
r112_100	401.70	*	-	95.99	*	97.31	*	8.66

Table 1: R Instances. Optimal results for objective improvement of 1%, 5% and 10%.

Instance	z^*_Ω	Improvement 1%			Improvement 5%		Improvement 10%	
		g^*_Θ	$T_{g^*}(s)$	$T^E_{g^*}(s)$	g^*_Θ	$T^E_{g^*}(s)$	g^*_Θ	$T^E_{g^*}(s)$
rc101_200	461.10	3.40	5054.65	559.09	4.30	313.28	4.30	41.76
rc102_200	351.80	0.30	1800.77	449.14	16.30	4422.00	*	0.30
rc103_200	332.80	2.00	-	-	38.40	-	*	0.97
rc104_200	306.60	14.30	-	-	*	1.06	*	1.20
rc105_200	411.30	3.40	3660.23	312.04	7.90	157.84	7.90	8.58
rc106_200	345.50	3.00	1176.41	192.00	11.10	473.00	75.90	2500.73
rc107_200	298.30	*	-	0.51	*	0.51	*	0.51
rc108_200	294.50	*	-	1.73	*	1.73	*	1.73
rc101_100	534.30	15.30	57.19	12.72	*	0.09	*	0.09
rc102_100	523.70	18.00	711.33	132.76	*	0.15	*	0.15
rc103_100	514.70	*	435.90	0.12	*	0.12	*	0.12
rc104_100	506.70	*	1610.69	0.14	*	0.14	*	0.14
rc105_100	527.50	16.10	258.04	83.23	*	0.15	*	0.15
rc106_100	515.60	46.00	1099.15	47.20	*	0.13	*	0.13
rc107_100	505.70	*	843.70	0.09	*	0.09	*	0.09
rc108_100	505.70	*	2173.81	0.01	*	0.01	*	0.01

Table 2: RC Instances. Optimal results for objective improvement of 1%, 5% and 10%.

mentation of the branch-and-price-and-cut procedure requires much more time to converge. It fails to converge on 4 out of 16 instances. We recalll that only those with 1% requested improvement are tested against the standard implementation. For the remaining instances the embedded relaxation method outperforms the standard procedure by one to several orders of magnitude.

In order to compare our results with an exact method for the VRPSTW, we executed the exact branch-and-price procedure by Liberatore et al. (2011) over the same set of instances. In particular, we intend to compare the values of time windows violation and primary objective when a method based on soft-constraints, i.e. a linear combination of primary objective and time windows violation, is used. To perform our comparison, we set the penalty for time windows violation equal to 1 as done in the tests reported by Liberatore et al. (2011).

Results are reported in tables 3 and 4. Tables are organized as follows: the first column reports on the instance name followed by the optimal value without time windows violation z^*_Ω; the following two columns are dedicated to the optimal solution of VRPSTW and they report the value of the violation of time windows, g_Θ and the primary objective value, z_Θ. We recall that their sum is optimal for the model with soft time windows. Subsequent columns report on the minimal time windows violation, g^*_Θ and the value of the primary objective corresponding to the optimal solution, $z(g^*_\Theta)$ for the two level approach with requested improvement of 1%, 5% and 10%. As above, when column g^*_Θ contains an asterisk $(*)$ it means that the corresponding instance is unfeasible with respect to constraint (7) and requested objective improvement.

We observe in table 3 that the soft constraints method always produces a solution. For 3 out of 24 instances it reproduces the optimal solution with no time windows violation. For some other instances (e.g., r101 and r102) the time windows violation is larger than that necessary to obtain a 10% improvement over the optimal solution. We recall that the reported solutions have been obtained by setting the same penalty to all customers. Different penalties would indeed lead to different solutions. This fluctuating behavior of the algorithm is undesirable for planners. They cannot rely on these results as the same settings for the parameters lead to substantially different solutions in different but structurally similar instances. The observation of table 4 leads to similar conclusions. The approach based on soft time windows leads to fluctuating results. Finally, for instances r112_100, rc101_100 and rc107_200, we observe the undesired behavior mentioned in the introduction: violations are permitted to save small fractions of the primary objective.

		Soft TW		Improvement 1%		Improvement 5%		Improvement 10%	
Instance	z^*_Ω	g_Θ	z_Θ	g^*_Θ	$z(g^*_\Theta)$	g^*_Θ	$z(g^*_\Theta)$	g^*_Θ	$z(g^*_\Theta)$
r101_200	617.10	46.70	538.00	7.00	605.40	21.80	583.40	41.40	553.40
r102_200	547.10	35.20	475.80	2.40	515.50	2.40	515.50	29.40	483.30
r103_200	454.60	9.90	439.80	7.00	448.10	23.60	430.80	*	*
r104_200	416.90	6.70	410.10	6.70	410.10	43.30	389.20	*	*
r105_200	530.50	40.20	481.30	2.20	523.00	19.60	503.70	48.90	474.40
r106_200	465.40	21.00	440.80	11.00	453.70	21.00	440.80	57.10	415.00
r107_200	424.30	6.70	413.60	4.60	418.70	47.30	402.00	*	*
r108_200	397.30	2.30	393.00	2.30	389.20	27.60	373.90	*	*
r109_200	441.30	0.00	441.30	20.90	435.40	*	*	*	*
r110_200	444.10	11.20	407.60	0.10	429.80	10.20	420.10	*	*
r111_200	428.80	3.90	416.80	3.90	416.80	22.30	407.20	*	*
r112_200	393.00	0.00	393.00	21.00	385.30	59.00	370.70	*	*
r101_100	617.10	46.70	538.00	7.00	605.40	21.80	583.40	41.40	553.40
r102_100	547.10	35.20	475.80	2.40	515.50	2.40	515.50	29.40	483.80
r103_100	454.60	9.90	439.80	7.00	448.10	23.60	430.80	*	*
r104_100	416.90	6.70	410.10	6.70	410.10	*	*	*	*
r105_100	530.50	40.20	481.30	2.20	523.00	19.60	503.70	48.90	474.40
r106_100	465.40	35.80	427.60	11.00	453.70	25.80	440.50	57.10	415.00
r107_100	428.40	6.90	418.90	4.20	422.10	*	*	*	*
r108_100	403.20	2.30	398.90	2.30	398.90	*	*	*	*
r109_100	441.30	0.00	441.30	20.90	435.40	*	*	*	*
r110_100	444.10	14.80	409.50	0.10	432.80	10.20	420.10	*	*
r111_100	428.80	3.90	417.50	3.90	417.50	35.50	403.30	*	*
r112_100	401.70	3.10	398.20	*	*	*	*	*	*

Table 3: R Instances. Comparison of soft time windows with penalty 1 with the two level model with objective improvement of 1%, 5% and 10%.

Instance	z^*_Ω	Soft TW		Improvement 1%		Improvement 5%		Improvement 10%	
		g_Θ	z_Θ	g^*_Θ	$z(g^*_\Theta)$	g^*_Θ	$z(g^*_\Theta)$	g^*_Θ	$z(g^*_\Theta)$
rc101_200	461.10	12.10	359.60	3.40	455.90	4.30	413.00	4.30	413.00
rc102_200	351.80	3.30	338.80	0.30	346.00	16.30	333.50	*	*
rc103_200	332.80	2.00	329.40	2.00	329.40	38.40	316.00	*	*
rc104_200	306.60	0.00	306.60	14.30	299.70	*	*	*	*
rc105_200	411.30	14.80	346.60	3.40	405.00	7.90	358.00	7.90	358.00
rc106_200	345.50	11.10	327.60	3.00	341.00	11.10	327.60	75.90	310.50
rc107_200	298.30	1.00	296.30	*	*	*	*	*	*
rc108_200	294.50	0.00	294.50	*	*	*	*	*	*
rc101_100	534.30	2.00	531.50	15.30	527.50	*	*	*	*
rc102_100	523.70	0.00	523.70	18.00	517.80	*	*	*	*
rc103_100	514.70	0.00	514.70	*	*	*	*	*	*
rc104_100	506.70	0.00	506.70	*	*	*	*	*	*
rc105_100	527.50	0.00	527.50	16.10	521.20	*	*	*	*
rc106_100	515.60	0.00	515.60	46.00	509.50	*	*	*	*
rc107_100	505.70	0.00	505.70	*	*	*	*	*	*
rc108_100	505.70	0.00	505.70	*	*	*	*	*	*

Table 4: RC Instances. Comparison of soft time windows with penalty 1 with the two level model with objective improvement of 1%, 5% and 10%.

5 Conclusions

In this paper we propose an alternative model to deal with constraint violation. It has been inspired by practical application and direct interaction with planners of different shipping companies. The model proposed in this papers aims at overcoming strict management of constraints as done in classical mathematical models for routing applications and at the same time to not over-burden decision makers with an excessive number of parameters. It ultimately aims at being more acceptable for decision makers.

The proposed method is based on a two-level approach. The first level is used to solve the nominal problem, i.e., without any constraint violation. The second level accounts for constraint violation in the objective function and constraints the primary objective to be improved at least by a certain percentage with respect to the optimal solution of the nominal problem.

We present a non-trivial implementation of a branch-and-price-and-cut procedure in which an embedded relaxation is used to speed-up to fathoming of unpromising nodes of the search tree. We believe that this embedded relaxation procedure can be exploited in all situations in which additional constraints render standard implementation of column generation unacceptably slow.

A computational experience over a set of 120 instances is presented. We provide the

optimal solution of all the instances whenever it exists. The current method is able to solve instances with up to 25 nodes. Computational time to prove unfeasibility is very low thanks to the embedded relaxation procedure. The computational time to prove optimality for some instances is very high when compared with the time required to solve the nominal counterpart. One possible direction to overcome this drawback is to improve further the pricing procedure embedded in column generation. This will be addressed in a future study.

References

Addis, B., Carello, G., and Ceselli, A. (2012). Exactly solving a two-level location problem with modular node capacities. *Networks*, 59:161–180.

Archetti, C., Savelsbergh, M., and Speranza, M. (2008). To split or not to split: That is the question. *Transportation Research Part E*, 44:114–123.

Baldacci, R., Battarra, M., and Vigo, D. (2009). Valid inequalities for the fleet size and mix vehicle routing problem with fixed costs. *Networks*, 54(4):178–189.

Berrada, I., Ferland, J. A., and Michelon, P. (1996). A multi-objective approach to nurse scheduling with both hard and soft constraints. *Socio-Economic Planning Sciences*, 30(3):183 – 193.

Bettinelli, A., Ceselli, A., and Righini, G. (2011). A branch-and-cut-and-price algorithm for the multi-depot heterogeneous vehicle routing problem with time windows. *Transportation Research Part C: Emerging Technologies*, 19(5):723 – 740.

Bianchessi, N. and Righini, G. (2007). Heuristic algorithms for the vehicle routing problem with simultaneous pick-up and delivery. *Computers & Operations Research*, 34(2):578–594.

Ceselli, A., Righini, G., and Salani, M. (2009). A column generation algorithm for a vehicle routing problem with economies of scale and additional constraints. *Transportation Science*, 43(1):56–69.

Charnes, A. and Cooper, W. (1959). Chance-constrained programming. *Management Science*, 6(1):73–79.

Christiansen, M. and Fagerholt, K. (2002). Robust ship scheduling with multiple time windows. *Naval Research Logistics (NRL)*, 49(6):611–625.

Dantzig, G. B. and Ramser, J. H. (1959). The Truck Dispatching Problem. *Management Science*, 6(1):80–91.

Dell'Amico, M., Righini, G., and Salani, M. (2006). A branch-and-price approach to the vehicle routing problem with simultaneous distribution and collection. *Transportation Science*, 40(2):235–247.

Desaulniers, G., Desrosiers, J., Dumas, Y., Solomon, M., and Soumis, F. (1997). Daily aircraft routing and scheduling. *Management Science*, 43(6):841–855.

Desaulniers, G., Desrosiers, J., and Solomon, M., editors (2005). *Column Generation*. GERAD 25th Anniversary Series. Springer.

Desrosiers, J., Pelletier, P., and Soumis, F. (1981). *Plus court chemin avec contraintes d'horaires*. Montréal: Université de Montréal, Centre de recherche sur les transports.

Dror, M. (1994). Note on the complexity of the shortest path models for column generation in vrptw. *Operations Research*, 42(5):977–978.

du Merle, O., Villeneuve, D., and and, J. D. (1999). Stabilized column generation. *Discrete Mathematics*, 194:229–237.

Gehring, H. and Homberger, J. (2002). Parallelization of a two-phase metaheuristic for routing problems with time windows. *Journal of Heuristics*, 8(3):251–276.

Gendreau, M. and Tarantilis, C. T. (2010). Solving large-scale vehicle routing problems with time windows: state-of-the-art. Technical Report 2010-04, CIRRELT.

Goetschalckx, M. and Jacobs-Blecha, C. (1989). The vehicle routing problem with backhauls. *European Journal of Operational Research*, 42(1):39 – 51.

Golden, B., Raghavan, S., and Wasil, E. A. (2008). *The vehicle routing problem : latest advances and new challenges*. Operations research/Computer science interfaces series, 43. Springer.

Ibaraki, T., Imahori, S., Nonobe, K., Sobue, K., Uno, T., and Yagiura, M. (2008). An iterated local search algorithm for the vehicle routing problem with convex time penalty functions. *Discrete Applied Mathematics*, 156(11):2050 – 2069.

Kohl, N., Desrosiers, J., Madsen, O., Solomon, M., and Soumis, F. (1999). 2-path cuts for the vehicle routing problem with time windows. *Transportation Science*, 33(1):101–116.

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

Liberatore, F., Righini, G., and Salani, M. (2011). A column generation algorithm for the vehicle routing problem with soft time windows. *4OR: A Quarterly Journal of Operations Research*, 9(1):49–82.

Lu, C.-C. and Yu, V. F. (2012). Data envelopment analysis for evaluating the efficiency of genetic algorithms on solving the vehicle routing problem with soft time windows. *Computers & Industrial Engineering*, 63(2):520 – 529.

Pagnoncelli, B., Ahmed, S., and Shapiro, A. (2009). Sample average approximation method for chance constrained programming: Theory and applications. *Journal of Optimization Theory and Applications*, 142:399–416.

Qureshi, A. G., Taniguchi, E., and Yamada, T. (2010). Exact solution for the vehicle routing problem with semi soft time windows and its application. *Procedia - Social and Behavioral Sciences*, 2(3):5931 – 5943.

Righini, G. and Salani, M. (2006). Symmetry helps: Bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. *Discrete Optimization*, 3(3):255–273.

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

Ruinelli, L., Salani, M., and Gambardella, L. M. (2012). Hybrid column generation-based approach for vrp with simultaneous distribution, collection, pickup-and-delivery and real-world side constraints. In *Proceedings of ICORES 2012*, pages 247–255.

Salani, M. and Vacca, I. (2011). Branch and price for the vehicle routing problem with discrete split deliveries and time windows. *European Journal of Operational Research*, 213(3):470–477.

Savelsbergh, M. W. P. and Sol, M. (1995). The general pickup and delivery problem. *Transportation Science*, 29(1):17–29.

Schrage, L. (1981). Formulation and structure of more complex/realistic routing and scheduling problems. *Networks*, 11(2):229–232.

Solomon, M. (1983). *Vehicle Routing and Scheduling with Time Windows Constraints: Models and Algorithms*. PhD thesis, University of Pennsylvania.

Steuer, R. E. (1986). *Multiple Criteria Optimization: Theory, Computation and Application*. John Wiley, New York, 546 pp.

Tas, D., Dellaert, N., van Woensel, T., and de Kok, T. (2013). Vehicle routing problem with stochastic travel times including soft time windows and service costs. *Computers & Operations Research*, 40(1):214 – 224.

Toth, P. and Vigo, D. (1997). An exact algorithm for the vehicle routing problem with backhauls. *Transportation Science*, 31(4):372–385.

Toth, P. and Vigo, D. (2002). *The vehicle routing problem*. SIAM monographs on discrete mathematics and applications. Society for Industrial and Applied Mathematics.

Vanderbeck, F. and Wolsey, L. (1996). An exact algorithm for ip column generation. *Operations Research Letters*, 19:151–159.

Vidal, T., Crainic, T., M., G., and Prins, C. (2012). Heuristics for multi-attribute vehicle routing problems: A survey and synthesis. Technical Report 2012-05, CIRRELT.