Pheromone-based Heuristic Column Generation for Vehicle Routing Problems with Black Box Feasibility

Florence Massen[1], Yves Deville[1], and Pascal Van Hentenryck[2]

[1] ICTEAM, Université catholique de Louvain, Belgium
{Florence.Massen, Yves.Deville}@uclouvain.be
[2] Optimization Research Group, NICTA, University of Melbourne, Australia
pvh@nicta.com.au

Keywords: Column Generation, Set Partitioning, Vehicle Routing, Black Box Feasibility

Abstract. This paper proposes an abstraction of emerging vehicle routing problems, the Vehicle Routing Problem with Black Box Feasibility. In this problem the routes of a basic VRP need to satisfy an unknown set of constraints. A black box function to test the feasibility of a route is provided. This function is considered of non-linear complexity (in the length of the route). Practical examples of such problems are combinations of VRP with Loading problems or VRP with Scheduling problems. The difficulty in addressing the VRP with Black Box Feasibility lies in the unknown problem structure and the costly feasibility check. We propose a column generation-based approach to locally optimize this problem. Columns are heuristically generated by so-called Collector ants, executing a construction heuristic while guided by pheromones. To find an integer solution we solve an integer Set Partitioning Problem defined on the set of columns generated by the ants. We test the proposed approach on two applications from the literature, the Three-Dimensional Loading Capacitated Vehicle Routing Problem and the Multi-Pile Vehicle Routing Problem, showing the applicability of our approach and its good behavior compared to dedicated approaches.

1 Introduction

Vehicle Routing Problems (VRPs) have received a great deal of attention since as early as the 1960's. While initially only basic variants have been considered, during the following decades research has focused on more complex variants, such as problems with time windows or with pick-up and delivery. Recent years have seen the emergence of rich vehicle routing problems, which strive to give a more realistic representation of problems encountered in the real world. Rich problems often require handling the combination of different complicating constraints which are typically considered individually in the literature.

In this context a new type of routing problems has emerged, problems combining routing with different combinatorial problems (e.g. combination of routing with loading (3L-CVRP [6]) or routing with scheduling (VRPTW with Driver Regulations [10])). Such problems are often tackled using very dedicated approaches. The aim of this paper

is to propose a generalized reformulation for this type of problem as well as an optimization procedure for the generic problem resulting from this reformulation. To do this, we introduce the VRP with Black Box Feasibility (VRPBB). This problem is an extension of basic VRPs. Besides respecting the VRP constraints (capacity, time windows, ...) each route needs to verify an unknown set of constraints F. The feasibility of a route with respect to F is verified using a black box algorithm.

We propose to reformulate the VRPBB as a Set Partitioning Problem which we address using a (non-exact) column generation-based approach. Collector ants heuristically generate and collect columns while guided by pheromone deposits stemming from an external oracle. This oracle computes pheromone deposits in function of the current relaxed solution. The approach allows us to iteratively improve the lower bound of the considered problem. An integer solution is found by solving the Set Partitioning Problem using a MIP solver.

The contribution of this paper is three-fold. First we propose a new generic problem, the Vehicle Routing Problem with Black Box Feasibility, which allows to represent Vehicle Routing Problems demanding the feasibility of a combinatorial problem per route. Second, we propose an algorithm to solve this generic problem. Our method works thus independently of the combinatorial problem to be solved. It can easily be applied to a new problem by plugging an appropriate black box function. Finally we demonstrate the applicability of the proposed method on two problems, the Three-dimensional Loading CVRP and the Multi-Pile VRP and compare our results with those of existing dedicated approaches, to show that our approach is highly competitive.

Related Work Optimization under the presence of black box functions is an active research area. Classical metaheuristics such as Genetic Algorithms or Simulated Annealing were designed as black box optimizers [9]. Expensive black box functions arise in structural optimization problems where simulations are necessary to judge the quality of a solution such as in ship-hull design or design of compression zones of cars. The evaluation of black box functions considered in this domain can take up to twenty hours [8]. Note that this is contrary to the VRPBB, where we assume the feasibility evaluation to be relatively expensive in comparison with typical VRP variants.

Literature focusing explicitly on Vehicle Routing Problems with expensive feasibility checks is scarce. Only one paper is known to the authors ([13]); it evaluates the efficiency of different index structures to be used to store feasibility in the case of a combined VRP and two-dimensional loading problem. A specific focus on VRPs with sparse feasibility graphs was given in [1].

Overview The remainder of this paper is structured as follows. Section 2 introduces the Vehicle Routing Problem with Black Box Feasibility. Section 3 describes the proposed column generation-based approach and its different building blocks. Sections 4.1 and 4.2 describe the 3L-CVRP and the MP-VRP, two applications on which we tested our approach. Existing work on both problems is described as well. Section 5 presents the Experimental Results obtained on benchmark instances for the 3L-CVRP and the MP-VRP. Those results are followed by a discussion. Finally, in Section 6 conclusions and perspectives are given.

2 Vehicle Routing Problem with Black Box Feasibility

Capacitated Vehicle Routing Problem The Capacitated Vehicle Routing Problem ([15]) is the most basic vehicle routing problem and underlying to most VRP variants. It is defined on a complete and weighted graph $G = (V, E)$ where $V = \{0, 1, ..., n\}$ is a set of $n+1$ vertices and E the set of weighted edges connecting every pair of vertices. Vertex 0 represents the depot while vertices $1, ..., n$ are the n customers to be served. The non-negative weight c_{ij} $(i, j = 0, ..., n : i \neq j)$ of an edge (i, j) corresponds to the cost of traveling from vertex i to vertex j. The homogeneous fleet is limited to K vehicles, each associated with a maximum capacity D. With each customer i $(i = 1, ..., n)$ is associated a demand d_i. A route r is defined by the set of visited non-depot vertices S and the sequence σ in which those vertices are visited. Each route, starts from and ends at the depot. The route r can be denoted as (S, σ).
Finally the goal is to devise a solution composed by at most K routes such that:

- each customer is visited on exactly one route
- the sum of demands of the customers on a route does not exceed the maximum capacity D
- the total traveling cost, equal to the sum of the weights of traversed edges, is minimized

VRP with Black Box Feasibility In the VRPBB, each feasible route, besides verifying the constraints associated with the underlying VRP variant, must verify an unknown set of fixed constraints called F. Let $feas(r, c) = true$ indicate that route r satisfies constraint $c \in F$. A tentative route r is considered feasible with respect to F if and only if $\bigwedge_{c \in F} feas(r, c)$. The black box provides a deterministic function returning a boolean indicating the feasibility of route r with respect to F. This function is considered computationally expensive (compared to feasibility functions commonly encountered in the VRP) and more precisely of non-linear complexity in the length of the route. In the following a VRP-feasible route is a route feasible with respect to the CVRP constraints, a black box feasible route is a route feasible with respect to F and a feasible route is a route both VRP-feasible and black box feasible.

3 Approach

3.1 Addressing the VRPBB as a Set Partitioning Problem

We are not looking to globally optimize the VRPBB. Local Search in combination with good neighborhood functions has shown its efficiency in local optimization in number of problems. However a local search approach does not seem appropriate for the VRPBB. Given the set of unknown constraints F we cannot be certain that a feasible neighborhood for a given VRPBB is connected. Furthermore we cannot extract a measure of violation from the black box. This makes it difficult to apply a metaheuristic approach, and we thus decided to reformulate the VRPBB as a Set Partitioning Problem (SPP). Let $\mathscr{R}$ the set of all possible routes, c_r the cost of route r and x_r a variable indicating whether route r is to be used in the optimal solution. The SPP analogous to the Vehicle Routing Problem is defined as follows:

$$
\begin{aligned}
Min \quad & \sum_{r \in \mathscr{R}} c_r x_r \\
s.t. \quad & \sum_{r \in \mathscr{R}} v_{ir} x_r = 1 \quad \forall i \in V \backslash 0 \\
& \sum_{r \in \mathscr{R}} x_r \leq K \\
& x_r \in \{0,1\} \quad \forall r \in \mathscr{R}
\end{aligned}
$$

$$
v_{ir} = \begin{cases} 1 & \text{if customer } i \text{ is visited in route } r \\ 0 & \text{otherwise} \end{cases} \qquad \forall i \in V \backslash 0, \quad \forall r \in \mathscr{R}
$$

The SPP is defined over the set of all possible routes $\mathscr{R}$. Computing $\mathscr{R}$ is obviously intractable for all but the very smallest Vehicle Routing Problems. A well-known approach for such huge problems is Delayed Column Generation. The idea is to work with the linear relaxation of the original problem and iteratively add new columns to this relaxed problem formulation, thus working on a restricted problem, called the Restricted Master Problem (RMP). New columns are discovered by solving a subproblem. New interesting columns are identified using their reduced cost, which is obtained using the dual costs stemming from the dual of the current RMP. Typically only columns with negative reduced cost (in the case of a minimization problem) are added to the problem formulation. In our case, generating columns amounts to generating feasible routes.

3.2 Heuristic Column Generation

Since we are not looking for an exact solution to our problem we will use a heuristic column generation approach (see Algorithm 1). At each iteration we generate a number of feasible routes which will be added to the problem formulation (to the column pool $\mathscr{R}^*$, lines 4-6 in our algorithm). Those routes are generated using so-called Collector ants, explained in the next section. It is the Collector ants that will ensure the VRP- and the black box feasibility of the generated routes. The current RMP is then solved optimally (line 7), providing the dual costs Π allowing to compute the reduced cost of new columns. The *initial* flag sets the Collector ants in a special mode until the current RMP becomes feasible. Note that in practice we execute lines 7–11 only if the I Collector ants were able to find at least 20 feasible routes since the last time those lines were executed. Finally we update the pheromone matrix τ, based on the current relaxed solution (line 10). An integer solution is found (line 13) by solving the SPP on the pool of accumulated columns (using a MIP solver). To decrease the number of calls to the costly black box feasibility check we store the feasibility information of routes previously checked in a feasibility pool (Ψ in Algorithm 1).

Collector Ants We propose heuristic executions called Collector ants to generate feasible routes (see Algorithm 2). Collector ants are based on the savings-based ants from [12], which themselves are based on the Savings heuristic [3]. An ant-based approach is a natural choice in our context due to the ease with which the guidance from the current

Algorithm 1: Main algorithm for solving the VRPBB

```
1  initialize τ, Π, Ψ;                    // pheromones, dual costs, feasibility pool
2  initial ← true; R* ← ∅; Sol ← ⊥;
3  while ¬ stopping criterion do
4      repeat I times
5          R* ← R* ∪ CollectorAnt(initial, τ, Π, Ψ);
6      end
7      Sol, Π ← solveRMP(R*);
8      if Sol ≠ ⊥ then
9          initial ← false;
10         τ ← updatePheromones(Sol, τ);
11     end
12 end
13 return solveSPP(R*, Ψ);
```

relaxed solution can be implemented using pheromones. Each ant starts from an initial state where each customer is visited in a route of its own (line 1). At each iteration the ant will build a set Ω of potential route merges (lines 4 – 14). One of those merges is then selected and executed (line 15). Merging two routes $0-i_1-i_2-\cdots-i_m-0$ and $0-j_1-j_2-\cdots-j_q-0$ corresponds to removing edges $(i_m,0)$ and $(0,j_1)$ from the current state while adding edge (i_m,j_1). The gain in distance resulting from these modifications is known as the savings value and defined as $s_{i_m,j_1}=c_{i_m,0}+c_{0,j_1}-c_{i_m,j_1}$. The ant merges routes until no further merge is possible (lines 3 – 18). Note that in Algorithm 2 we consider that a merge m contains the following information: the edge (i,j) to be added to the current state and the associated savings value s_{ij}. To build the set Ω of potential merges, the ant will first gather the set of all possible merges M given the current state S (lines 2 and 16). The set of all possible merges M will contain all merges that are VRP-feasible. The ant will also compute the attractiveness value for each of these merges as follows: $attractiveness(m,\tau)=s_{ij}^{\beta}+\tau_{ij}^{\alpha}$ where (i,j) is the edge to be introduced in merge m. It then considers all the merges in M by non-increasing attractiveness value (line 4). An important decision for the ant is which merges to accept into Ω and which ones to refuse. The decision depends on the *initial* flag, the reduced cost of the route resulting from the tentative merge, the fact whether the resulting route is known to the feasibility pool Ψ and finally the feasibility of the route (lines 6 – 13). The rationale behind this approach is to keep a balance between diversification (between ant executions), extending the feasibility pool and intensification towards feasible routes. Note that, depending on the black box constraints, an infeasible route might contribute to a feasible merge. Finally the merge from Ω to be executed is selected using roulette wheel selection based on the attractiveness of all the merges in Ω (line 15). Collector ants will collect all the feasible routes they encounter while executing the Savings heuristic (line 10). This means that not only routes that were part of the current state S at some point are collected by the ants, but also routes resulting from merges that were in Ω but never selected for execution. Finally the routes collected by the ants are individually post-processed using a limited number of reinsertion and 2-opt moves (line 19).

Algorithm 2: Collector ant algorithm

Input: $initial, \tau, \Pi, \Psi$ `// flag, pheromones, dual costs, feasibility pool`
Output: $collected$ `// set of collected routes`

initialize S to state using shuffle routes;
$i \leftarrow 0$; $collected \leftarrow \emptyset$; $\Omega \leftarrow \emptyset$; $M \leftarrow$ `getPossibleMerges(`S`)`;
while $M \neq \emptyset$ **do**
  foreach $m \in M$ **by** $-attractiveness(m, \tau)$ **do**
    route $r \leftarrow$ `getRouteFromMerge(`m`)`;
    if $initial \vee$ `getReducedCost(`r, Π`)` $< 0 \vee r \notin \Psi$ **then**
      if $r \notin \Psi$ **then** $\Psi[r] \leftarrow$ `checkBlackBoxFeasibility(`r`)`;
      if $\Psi[r]$ **then**
        $\Omega \leftarrow \Omega \cup m$; $i \leftarrow i + 1$;
        $collected \leftarrow collected \cup r$;
      end
      if $i \geq \nu$ **then** break;
    else $\Omega \leftarrow \Omega \cup m$;
  end
  select $m \in \Omega$; $S \leftarrow$ `executeMerge(`m, S`)`;
  $M \leftarrow$ `getPossibleMerges(`S`)`;
  $i \leftarrow 0$; $\Omega \leftarrow \emptyset$;
end
$collected \leftarrow$ `postprocess(`$collected, \Psi$`)`;
return $collected$

Pheromone Update Each ant disposes of the pheromone matrix τ, indicating the amount of pheromones on every edge (i, j). The pheromones influence the attractiveness of an edge, and thus the probability of an edge to enter Ω and to be executed by an ant. The idea behind the pheromones is to guide the ants towards routes of good quality and with higher probability of being black box feasible. Since routes in the current relaxed solution are known to be optimal for the current RMP as well as black box feasible, we want the ants to produce routes similar to those in the relaxed solution, in the hope of producing routes of similar qualities. Once the current relaxed solution has been computed the pheromone matrix is updated as follows:

$$\tau_{ij} = \rho\,\tau_{ij} + \sigma_{ij}\,\varepsilon \quad \forall (i,j)(i = 0..n,\ j = 0..n,\ i \neq j)$$

where ρ ($0 \leq \rho \leq 1$) is the trail persistence, ε is a small constant and σ_{ij} is the number of times (i, j) appears in the current relaxed solution. The first term thus corresponds to pheromone evaporation, while the second is the deposit.

4 Applications of the VRPBB

4.1 The Three-Dimensional Loading Capacitated Vehicle Routing Problem

The Three-dimensional Loading Capacitated Vehicle Routing Problem (3L-CVRP) was first introduced in [6]. It combines vehicle routing with three-dimensional loading. In

this variant of the VRP, each customer demands a set of three-dimensional boxes. To be feasible a route must not only respect the capacity constraint, but a feasible loading must exist for all the items to be delivered in that route. A loading is considered feasible if it respects typical bin-packing as well as some real-world constraints. The loading feasibility check can easily be considered as a black box with the properties described in Section 2.

Existing work on the 3L-CVRP Most of the existing work on the 3L-CVRP has been extended from the two-dimensional to the three-dimensional case. In [6] the authors propose a Tabu Search (TS) allowing visits to capacity- and loading-infeasible solutions. The excess length of the loading of a route with respect to the real length of the vehicle is used as a measure of violation of the loading constraints. The loading subproblem is addressed using a Tabu Search and two greedy heuristics adapted from well-known loading heuristics. Finally the TS algorithm is tested on benchmark instances generated from known CVRP benchmark instances. Using their Guided Tabu Search (GTS) the authors in [14] are able to improve the costs over 21 instances out of 27 compared to [6]. The loading-feasibility of routes is verified using a bundle of packing heuristics. In [5] (ACO) an adapted version of the Savings-based ants introduced in [12,11] is proposed. The probability of a merge is adapted to the 3L-CVRP by introducing a notion of the loading compactness of the route resulting from that merge. The loading-feasibility of a route is checked using a Local Search and the same heuristics as in [6]. The ACO is able to outperform the TS in 26 out of 27 instances and to beat the GTS in 23 out of 27 instances. A Hybrid Tabu Search (HTS) is presented in [2]. The HTS is split in two phases, one aiming at reducing the number of vehicles in the initial solution, and the other aiming at minimizing the total traveling cost. The author proposes a tree traversal algorithm with a limited number of backtracks. The HTS is able to improve 18 out of the 27 benchmark instances when compared to [5]. For an extensive overview of the combination of routing with loading problems see [7]. Note that some of these methods could be more easily generalized than others. The approaches from [14] and [2], for example, use loading-specific knowledge (free volume and customer demands) only in the effort to construct an initial feasible solution.

Three-dimensional Loading Capacitated Vehicle Routing Problem The 3L-CVRP is defined on top of the CVRP. The homogeneous fleet is limited to K vehicles, each associated with a maximum capacity D and a rectangular loading space of width W, height H and length L. The loading space is accessible only from the rear of the vehicle. Each customer $i\,(i = 1,...,n)$ demands a set I_i of m_i items of total weight d_i. Each item $I_{ik}(k = 1,...,m_i\,,\, i = 1,...,n)$ is a box of width w_{ik}, height h_{ik} and length l_{ik}, and is either fragile or non-fragile. The goal is to find a set of at most K routes each starting and ending at the depot, visiting every customer exactly once such that the total cost is minimized and the following conditions hold for every route $r = (S, \sigma)$:

- $\sum_{i \in S} d_i \leq D$ (capacity constraint)
- a feasible orthogonal loading exists for $\bigcup_{i \in S} \bigcup_{k=1}^{m_i} \{I_{ik}\}$

Feasible Loading for the 3L-CVRP A loading of a route $r = (S, \sigma)$ is the assignment of coordinates to the lower left corner of each item $I_{ik}(k = 1, ..., m_i), i \in S$. The origin of the coordinate system is assumed in the lower left back corner of the vehicle. A loading of route $r = (S, \sigma)$ is considered feasible if the typical 3-dimensional Bin Packing constraints, as well as the *Orientation*, *Fragility* and *LIFO Policy* constraints are respected. For a more detailed description of those constraints refer to [7].

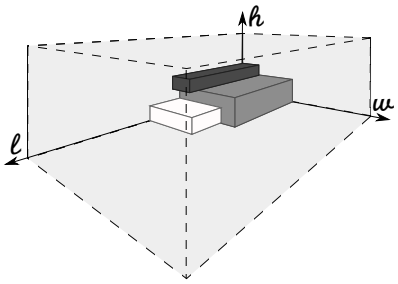

Fig. 1: Example of a loading violating the LIFO constraint. The loading container corresponds to the dashed lines. The unloading side is on the far side of the origin. Unloading of the white item (current visit) is hindered by the dark item (visited next).

Loading Feasibility Black Box The feasibility of a route in the 3L-CVRP, aside from respect of the CVRP constraints, depends on the existence of a feasible orthogonal loading for that route. We decided to reimplement the algorithm described in [2]. The author proposes a tree traversal method to solve the loading problem. Items are placed as far left, back and low as possible. Since this loading algorithm is of high complexity ($O(m^3)$, m being the number of items per route), the tree search is limited in the number of backtracks. Furthermore branches with low potential of resulting in a feasible loading are cut. Note that this approach is not complete and may consider as infeasible a route for which a feasible loading exists.

4.2 The Multi-Pile Vehicle Routing Problem

The Multi-Pile Vehicle Routing Problem (MP-VRP) was first introduced in [4] and is derived from a real-world problem encountered at an Austrian wood-products retailer. Customers request a set of chipboards of different height and length which need to be loaded onto pallets (the combination of chipboards and pallets will be considered an item) and then onto p piles inside the vehicle while respecting a set of constraints. Empty spaces in the loading are filled with bulk material to ensure stability.

Existing work on the MP-VRP Doerner et al. introduced the problem in [4] and in the same paper propose a Tabu Search (TS) algorithm as well as an Ant Colony Optimization (ACO) approach, which both are conceptually similar to [6] and [5]. In the case of the ACO the authors introduce a second pheromone matrix for the loading part of

the problem. For both TS and ACO they propose a heuristic and a basic dynamic programming approach to solve the loading problem. The heuristic is based on a branch and bound method for the $P||C_{max}$ scheduling problem, of which the loading problem encountered in the MP-VRP is a generalization. Both algorithms are tested on a set of randomly generated benchmark instances as well as on real-world instances. The ACO algorithm beats the TS in 17 out of the 21 random instances and 4 out of the 5 real-world instances, both while taking (significantly in the case of the real world instances) less time to converge. In [16] the authors propose a Variable Neighborhood Search (VNS) for the MP-VRP. During the exploration of the Cross-Exchange neighborhoods the authors use their implementation of the loading heuristic used in [4], allowing solutions however to be slightly infeasible. Before moving to a (potentially infeasible) solution, this solution is checked using an exact dynamic programming approach. The authors obtain good results when compared to ACO and TS, improving 19 out of the 21 random instances and 4 out of the 5 real-world instances. The ACO algorithm uses very problem specific knowledge while the TS uses a measure of violation of the loading constraint in its objective function. The VNS also uses the violation of the loading constraint as well as the knowledge about the speed and the efficiency of the two feasibility checks it uses.

Multi-Pile Vehicle Routing Problem The MP-VRP is defined on top of the basic CVRP. The capacity of the vehicles is infinite, so capacity constraints are not considered. The homogeneous fleet is not limited, i.e. $K = n$. Each vehicle corresponds to a container of width W, height H and length L. The length of the vehicle is divided in p piles, each of width W. The height of those piles is limited by the height H of the vehicle. Each customer $i\,(i = 1,...,n)$ demands a set I_i of m_i items each of length $l_{ik} \in \{L/p, L\}$, width W and height $h_{ik}(k = 1,...,m_i)$. That is, all items take up the entire width of the vehicle, and take up either the length of one pile (short items) or the length of the vehicle (long items). The goal is to find a set of routes each starting and ending at the depot, visiting every customer exactly once such that the total cost is minimized and the following condition holds for every route $r = (S, \sigma)$: a feasible loading exists for $\bigcup_{i \in S} \bigcup_{k=1}^{m_i} \{I_{ik}\}$ on the p piles.

Feasible Loading for the MP-VRP A loading of a route $r = (S, \sigma)$ is the assignment of a pile and a coordinate (height) to the lower left corner of each item $I_{ik}(k = 1,...,m_i), i \in S$. A loading of route $r = (S, \sigma)$ is considered feasible if the typical two-dimensional bin packing conditions, as well as the following *Sequential Loading* condition is verified. When a customer $i \in S$ is visited it must be possible to unload all its items without moving items belonging to another customer $j \neq i$. This implies that no objects belonging to a customer $j \in S\ (j \neq i)$ s.t. j is visited after i may hinder the unloading of the items of i. An item I_{jm} hinders the unloading of an item I_{in} if I_{jm} is placed between I_{in} and the top of the pile(s) in which I_{in} is placed.

Loading Feasibility Black Box The authors from [16] kindly provided us with their implementation for the Loading Feasibility Check. They reimplemented the loading heuristic from [4] (using a different preprocessing method) as an upper bound. They also provided the exact dynamic programming method, which establishes a dominance

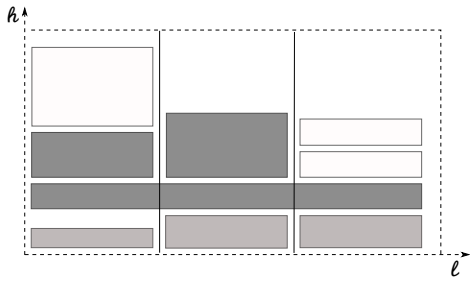

Fig. 2: Example of a loaded vehicle (dashed lines) with $p = 3$ seen from the side.

criterion based on the heights of the different piles, which they sort in non-decreasing order by height. Dominance between two partial loadings is then established by comparing the height of the piles. The authors furthermore introduce a pruning mechanism using a set of lower bounds stemming from the $P||C_{max}$. The upper bound heuristic is of temporal complexity linear in the number of items to be loaded. The exact algorithm runs in a time exponential in the size of the route. This is why we enforce a time limit θ (in seconds) on the execution of the exact algorithm. The method as we use it is thus not complete.

5 Experimental Results

The proposed approach (with the exception of the feasibility black boxes) has been implemented in Comet. We used CLP as LP solver and SCIP as IP solver. All tests were executed in 10 runs on an Intel Xeon 2.53GHz Processor, on a machine disposing of 23GB of RAM. Our parameter values were partially inspired by those used in [5]. We set $\alpha = 5, \beta = 5$. The pheromone matrix was initialized to 2, the evaporation rate $\rho = 0.95$ and $\varepsilon = 0.15$. The parameter ν giving the size of the merge set Ω was set to $max\{n, 50\}$. The parameter I corresponding to the number of collector ants per iteration was set to 5 for the 3L-CVRP and 1 for the MP-VRP in order to take into account the difference in time efficiency of the respective black box functions . We indicate the best and average solution costs found over all runs by z_{min} and z_{avg}. The total execution times (in CPU seconds) are given by sec_{tt}. We also indicate the percentage gap (in %) between the best (average) solutions found using our approach and the ones found in previous work by g_{min} (g_{avg}).
The gap is computed according to : $g = \frac{this\,work - other\,work}{other\,work} * 100$.

5.1 Results on the 3L-CVRP

The benchmark instances presented in [6] have been used to test our approach. They can be downloaded from `http://www.or.deis.unibo.it/research.html`. A detailed description on how the instances were generated is given in [6]. We impose the same time limits as [6] for the column generation part of our approach. After this time limit, we stop generating columns and solve the integer SPP on the column pool. Small instances ($n \leq 25$) are given a time limit of 1800 CPU seconds, medium instances

($25 < n < 50$) are given 3600 CPU seconds and large instances ($n \geq 50$) are allowed 7200 CPU seconds. When reaching this time limit, we finish the current column generation iteration and move on to solving the SPP. If a threshold of stable iterations ($K * 20$), that is, iterations without improvement of the lower bound, is reached, the column generation process is stopped as well. On average 16 % of the tested routes were found to be black box feasible. In the following we compare our results with those described in [2] (HTS), [5] (ACO), [14] (GTS) and [6] (TS) (c.f. Table 1). Note that the comparison with the latter 3 is not completely fair since we are not using the same loading algorithm they did. For HTS and ACO the best and average solution values over 10 runs are available. For GTS only the best solution value over 100 runs has been published. The TS is deterministic, so only one set of solution values is available. For ACO and TS the time at which the incumbent solution is found is provided. We did not measure this since it would have implied solving the integer SPP once per iteration, which would have been too time-consuming. Note that the results for HTS, ACO, GTS and TS were all executed on faster processors than ours.

The proposed generic approach allows to significantly improve the results found by [6] (TS) and [14] (GTS). In comparison with [5] (ACO) and [2] (HTS) it is highly competitive. The results indicate that, while for instances 1–9 execution stops well ahead of the time limits, for most of the other instances the time allocated to the column generation part of our approach is completely consumed. Our total execution times are thus comparable to those from TS (we used the same time limits) and are slightly worse than the times used by GTS and ACO. The GTS was stopped after 6000 stable iterations, while the ACO was halted after a number of iterations depending on the problem size, or after 3 CPU hours. The results reported in [2] (HTS) were obtained under a time limit and a limit on the number of iterations, both values depending on the instance size. The execution times reported for the HTS indicate that the proposed approach is very time-efficient. It greatly improves the execution times from TS, GTS and ACO and consequently ours as well, especially on large instances. The somewhat higher execution times of our algorithm can be explained by the genericity of our approach. Since we do not exploit, as the existing approaches do, explicit knowledge on the problem structure to guide the search, it seems self-evident the procedure demand a somewhat higher execution time. Of course the development time of our approach is significantly shorter, as to apply our approach it is sufficient to provide a function testing feasibility. In summary, our generic approach thus allows to find results better or comparable to those found using dedicated approaches, this at the cost of a higher execution time, but demanding a significantly lower development time.

5.2 Results on the MP-VRP

The random benchmark instances presented in [4] have been used. They can be downloaded from `http://prolog.univie.ac.at/research/VRPandBPP/`. The interested reader is invited to refer to [4] for a detailed description of the instances. Note that the underlying CVRP instance for the instances in *CMT07* of the MP-VRP and instance 26 of the 3L-CVRP is the same. As for the 3L-CVRP we again impose a time limit, set here to 1800 CPU seconds for our column generation phase. The limit on stable iterations is set to $K * 10$. On average 27.8 % of the tested routes were found to be black

box feasible. In the following we compare our results with those described in [16,4] (Table 2). Note, again, that the comparison is not completely fair. The results presented in [4] were obtained using a heuristic feasibility check. While we use the same exact feasibility check as [16], we imposed a different time limit (θ) on the exact algorithm. The results for the ACO ([4]) and the VNS ([16]) are given over 10 runs. The TS ([4]) is again deterministic, thus the solution values over one run are reported. Since execution times reported for the VNS do not include the preprocessing time needed by their upper bound heuristic, we did not include it either. Unfortunately we could not determine whether the execution times in [4] do include their preprocessing times. We did not measure the time the incumbent solution was found, as it would have implied solving the integer SPP once per iteration. During preliminary experiments we found that $\theta = 5\,sec$ is an appropriate time limit to use with our approach. This value gives a good balance between the time spent in the feasibility black box and the performance of the feasibility check. All presented results are obtained using $\theta = 5\,sec$. Note that the results for the VNS were obtained using a faster processor than ours. The tests for the ACO and the TS were executed on a processor of comparable speed to ours.

In terms of solution quality our generic approach does not allow to reach the same standard as the existing (dedicated) approaches. In terms of execution time, our results indicate that the 30 CPU minutes we allocated the column generation part of our algorithm were always almost completely used up. The VNS algorithm in [16] used the same time limit, but was also stopped after 10^7 stable iterations. The authors in [4] limit the time of the ACO and the TS to 2 CPU hours. The ACO algorithm was furthermore stopped after 100000 iterations, the TS after 250000 iterations. The bigger instances, on which our approach performs less good than ACO and TS are also those where these two algorithms have a higher total execution time. The same does however not hold for the VNS, who reaches better quality solutions in less time. The speed of the VNS can be explained through the intelligent use of the different feasibility checks, an efficient but problem-specific method. In summary our generic approach performs less well than the existing approaches for instances $n > 50$, but is also allocated a smaller time limit than ACO and TS. The reasonable loss of solution quality comes with the gain in development time resulting from the genericity of our approach.

6 Conclusion

This paper introduces the novel generic problem of Vehicle Routing with Black Box Feasibility (VRPBB). In the VRPBB routes need to verify an unknown set of hard intra-route constraints. The feasibility of a route can only be verified using a computationally expensive black box algorithm. The difficulty of this problem stems mainly from the unknown problem structure, but also from the costly feasibility check. To tackle the VRPBB we propose a column generation-based approach. Collector ants generate and collect columns with the potential to increase the quality of the current lower bound. Those ants are guided by pheromone deposits derived from the current relaxed solution. An integer solution is finally found by solving a Set Partitioning Problem on the collected routes. We showed the applicability of the proposed method on two practical applications, the Three-dimensional Loading Capacitated Vehicle Routing Problem and

the Multi-Pile Vehicle Routing Problem. We show that our generic approach is competitive with dedicated approaches, this at the cost of somewhat higher execution times. The development effort necessary to use our approach is significantly lower than the effort necessary to build a dedicated approach. Future work will consist in including our method as pricing step in a heuristic Branch-and-Price framework. We will also test the proposed approach on other applications such as the VRPTW with Driver Regulations.

Acknowledgment The authors would like to thank Fabien Tricoire for making available the feasibility checks for the MP-VRP as well as for his explanations. The first author is supported by the National Research Fund, Luxembourg. This research is also partially supported by the Interuniversity Attraction Poles Programme (Belgian State, Belgian Science Policy) and the FRFC project 2.4504.10 of the Belgian FNRS.

References

1. Beasley, J., Christofides, N.: Vehicle routing with a sparse feasibility graph. Eur. J. Oper. Res. 98 (1997)
2. Bortfeldt, A.: A hybrid algorithm for the capacitated vehicle routing problem with three-dimensional loading constraints. Tech. rep., FernUniversität in Hagen (2010)
3. Clarke, G., Wright, J.: Scheduling of vehicles from a central depot to a number of delivery points. Oper. Res. 12 (1964)
4. Doerner, K., Fuellerer, G., Hartl, R., Gronalt, M., Iori, M.: Metaheuristics for the vehicle routing problem with loading constraints. Networks 49 (2007)
5. Fuellerer, G., Doerner, K.F., Hartl, R., Iori, M.: Metaheuristics for vehicle routing problems with three-dimensional loading constraints. Eur. J. Oper. Res. 201 (2010)
6. Gendreau, M., Iori, M., Laporte, G., Martello, S.: A tabu search algorithm for a routing and container loading problem. Transport. Sci. 40 (2006)
7. Iori, M., Martello, S.: Routing problems with loading constraints. TOP 18 (2010)
8. Jones, D., Schonlau, M., Welch, W.: Efficient global optimization of expensive black-box functions. J. Global Opt. 13 (1998)
9. Laguna, M., Molina, J., Pérez, F., Caballero, R., G-Hernández-Díaz, A.: The challenge of optimizing expensive black boxes: a scatter search/rough set theory approach. J. Oper. Res. Soc. 61 (2010)
10. Prescott-Gagnon, E., Desaulniers, G., Drexl, M., Rousseau, L.M.: European driver rules in vehicle routing with time windows. Transport. Sci. 44 (2010)
11. Reimann, M., Doerner, K., Hartl, R.: D-ants : Savings based ants divide and conquer the vehicle routing problem. Comput. Oper. Res. 31 (2004)
12. Reimann, M., Stummer, M., Doerner, K.: A savings based ant system for the vehicle routing problem. In: Proceedings of the Genetic and Evolutionary Computation Conference (2002)
13. Strodl, J., Doerner, K., Tricoire, F., Hartl, R.: On index structures in hybrid metaheuristics for routing problems with hard feasibility checks: An application to the 2-dimensional loading vehicle routing problem. In: Hybrid Metaheuristics (2010)
14. Tarantilis, C., Zachariadis, E., Kiranoudis, C.: A hybrid metaheuristic algorithm for the integrated vehicle routing and three-dimensional container-loading problem. IEEE Trans. Intell. Transp. Syst. 10 (2009)
15. Toth, P., D.Vigo (eds.): The Vehicle Routing Problem. SIAM Monographs on Discrete Mathematics and Applications (2002)
16. Tricoire, F., Doerner, K.F., Hartl, R.F., Iori, M.: Heuristic and exact algorithms for the multi-pile vehicle routing problem. OR Spectrum (2009)

Table 1: Application 3L-CVRP : Pheromone-based Heuristic Column Generation approach compared to [2] (HTS), [5] (ACO), [14] (GTS) and [6] (TS). The gaps are given in %, the execution times in CPU seconds.

Instance			this work			HTS [2]			ACO [5]			GTS [14]		TS [6]	
Name	K	n	z_{min}	z_{avg}	sec_{tt}	$g_{min}\%$	$g_{avg}\%$	sec_{tt}	$g_{min}\%$	$g_{avg}\%$	sec_{tt}	$g_{min}\%$	sec_{tt}	$g_{avg}\%$	sec_{tt}
(1) E016-03m	4	15	302.02	303.19	282.5	0.00	0.39	41.6	-0.69	-0.71	12.0	-6.05	13.2	-4.15	1800
(2) E016-05m	5	15	334.96	334.96	33.7	0.00	0.00	0.3	0.00	0.00	0.6	0.00	11.5	-4.45	1800
(3) E021-04m	4	20	392.46	394.30	415.5	1.12	-2.48	159.1	-1.81	-3.78	121.8	-8.93	540.6	-11.93	1800
(4) E021-06m	6	20	437.19	437.19	385.9	0.00	-0.17	12.4	-0.79	-0.79	5.4	-4.55	323.5	-2.52	1800
(5) E022-04g	6	21	447.73	447.73	426.1	0.93	0.05	170.5	-0.71	-1.20	30.9	-3.88	99.6	-3.56	1800
(6) E022-06m	6	21	498.16	498.16	425.4	0.00	-0.66	15.3	-0.03	-0.66	18.4	-1.93	1212.4	-1.25	1800
(7) E023-03g	6	22	769.68	769.68	944.1	0.00	-2.70	62.8	-2.83	-3.49	67.4	-3.38	364.8	-7.45	1800
(8) E023-05s	6	22	845.50	849.27	659.0	4.27	3.07	98.9	3.03	3.48	78.6	-4.02	230.0	-2.58	1800
(9) E026-08m	8	25	630.13	630.13	200.0	0.00	-5.01	11.2	-0.85	-0.85	16.3	-1.88	982.2	-5.40	1800
(10) E030-03g	8	29	820.72	822.41	2950.7	0.05	-0.83	139.8	-2.38	-2.22	246.7	-7.24	1308.4	-9.74	3600
(11) E030-04s	8	29	778.10	778.42	2654.5	-2.80	-4.53	118.8	-4.98	-5.19	199.8	-10.91	522.5	-5.00	3600
(12) E031-09h	9	30	612.25	612.88	911.7	0.33	-3.65	12.8	-2.25	-2.57	48.2	-1.92	294.6	-5.94	3600
(13) E033-03n	8	32	2677.29	2677.72	3036.9	-0.10	-0.87	232.9	-2.28	-2.27	308.8	-4.37	2193.1	-8.56	3600
(14) E033-04g	9	32	1390.20	1417.89	3659.1	1.59	-0.14	312.2	-5.22	-3.69	642.8	-7.59	4581.3	-9.09	3600
(15) E033-05s	9	32	1342.31	1345.93	3697.1	0.83	-0.82	299.9	-1.85	-4.24	656.8	-5.17	2528.3	-7.33	3600
(16) E036-11h	11	35	698.61	698.61	1693.8	0.00	-0.80	2.4	-0.05	-0.05	14.8	0.00	4256.5	-1.31	3600
(17) E041-14h	14	40	866.40	866.84	2553.1	-1.12	-10.60	1.7	-0.25	-0.40	14.9	-0.73	2096.0	-5.87	3600
(18) E045-04f	11	44	1213.35	1230.72	3794.1	0.55	-0.27	315.1	-3.37	-2.41	2209.8	-6.42	2275.2	-12.12	3600
(19) E051-05e	12	50	757.35	758.93	7267.4	0.30	0.25	419.2	-2.55	-2.86	623.6	-7.49	2509.0	-12.90	7200
(20) E072-04f	18	71	586.92	593.68	8268.0	0.22	-0.78	432.1	-2.87	-2.88	3901.0	-8.52	1940.9	-18.91	7200
(21) E076-07s	17	75	1098.78	1109.35	8496.4	0.68	0.09	452.3	-1.02	-1.35	5180.6	-5.25	2823.4	-13.01	7200
(22) E076-08s	18	75	1159.07	1167.66	7980.5	-0.26	-0.69	428.6	-2.94	-2.49	2290.3	-6.93	2685.6	-8.63	7200
(23) E076-10e	17	75	1119.05	1132.02	8448.2	-1.68	-1.09	430.5	-3.41	-3.39	3727.6	-9.16	4659.1	-10.03	7200
(24) E076-14s	16	75	1119.57	1127.45	8358.5	0.45	-0.03	413.3	-1.52	-1.85	1791.5	-6.85	4854.1	-13.74	7200
(25) E101-08e	22	100	1434.25	1444.83	9713.9	2.56	0.57	463.4	0.32	0.59	8817.1	-1.59	5725.8	-8.02	7200
(26) E101-10c	26	100	1592.02	1606.69	9782.3	0.11	-1.47	436.7	-1.23	-0.64	6904.3	-7.00	6283.1	-13.06	7200
(27) E101-14s	23	100	1528.63	1537.37	9484.4	0.01	-0.30	441.6	-2.05	-2.30	10483.9	-7.16	9915.7	-12.03	7200
AVG			942.69	947.93	3945.3	0.30	-1.24	219.5	-1.65	-1.78	1793.1	-5.15	2415.9	-8.09	4200

Table 2: Application MP-VRP : Pheromone-based Heuristic Column Generation approach compared to [16] (VNS) and [4] (ACO, TS). The gaps are given in %, the execution times in CPU seconds.

Instance			this work				VNS [16]		ACO [4]			TS [4]	
Name	n	Class	z_{min}	z_{avg}	sec_{tt}	$g_{min}\%$	$g_{avg}\%$	sec_{tt}	$g_{min}\%$	$g_{avg}\%$	sec_{tt}	$g_{avg}\%$	sec_{tt}
CMT01	50	1	587.29	599.63	1801.7	-0.55	1.32	<1800	-1.14	0.85	12.3	0.94	2966.9
		2	618.74	629.65	1803.9	0.59	-1.63	<1800	-0.63	1.10	11.5	1.41	2229.2
		3	629.68	637.41	1799.1	1.00	0.46	<1800	-1.21	-0.24	11.4	0.07	1809.0
CMT02	75	1	980.38	998.69	1888.1	-0.20	1.27	<1800	0.18	1.80	73.7	0.83	2599.8
		2	910.54	926.25	1872.6	1.17	2.80	<1800	-0.23	1.19	72.7	1.49	3594.9
		3	913.39	928.34	1891.6	2.71	4.06	<1800	-0.34	1.14	72.4	0.84	2694.8
CMT03	100	1	1246.66	1264.26	1888.2	4.60	5.58	<1800	4.35	4.59	364.3	4.53	4885.6
		2	1258.33	1274.75	2037.7	3.16	4.29	<1800	1.89	2.57	349.2	2.18	3585.3
		3	1221.36	1229.61	2111.0	5.20	5.77	<1800	3.01	3.55	366.8	2.80	3994.0
CMT04	150	1	1734.08	1769.78	1954.7	6.25	8.19	<1800	5.20	6.58	3978.3	5.80	7200.0
		2	1671.21	1706.22	1894.9	7.19	9.18	<1800	6.66	8.31	3998.5	6.43	7200.0
		3	1634.54	1692.81	1984.5	5.92	9.36	<1800	3.58	6.89	3940.5	6.29	7092.1
CMT05	199	1	2204.46	2214.07	1808.4	8.13	8.04	<1800	6.11	6.16	7200.0	5.06	7200.1
		2	1964.29	2011.34	1878.0	7.09	9.34	<1800	5.95	7.94	7200.0	7.04	7200.0
		3	2142.99	2168.45	1932.0	9.42	10.31	<1800	7.75	8.44	7200.0	6.18	7200.0
CMT06	120	1	2364.13	2417.32	2215.6	5.20	7.23	<1800	4.59	6.51	1466.8	5.47	6406.4
		2	2215.48	2262.40	2140.1	6.02	7.60	<1800	6.11	7.34	1327.1	6.60	7200.0
		3	2246.17	2326.61	2311.0	4.22	6.57	<1800	2.72	5.96	1368.7	3.97	4927.7
CMT07	100	1	1173.96	1188.15	2125.9	3.29	4.53	<1800	2.73	3.01	302.9	2.93	5918.0
		2	1240.68	1254.95	2209.1	1.31	2.14	<1800	0.07	0.49	303.5	1.42	4088.1
		3	1202.34	1219.89	2203.2	3.74	4.70	<1800	1.73	3.13	274.8	3.10	4052.3
AVG			1436.22	1462.88	1988.2	4.07	5.29	<1800	2.81	4.16	1899.8	3.59	4954.5