New Lower Bounds on the Number of Vehicles for the Vehicle Routing Problem with Time Windows

Sohaib Afifi, Rym Nesrine Guibadj, Aziz Moukrim

New Lower Bounds for the Vehicle Routing Problem with Time Windows

Sohaib AFIFI, Rym Nesrine GUIBADJ and Aziz MOUKRIM

Université de Technologie Compiègne
Laboratoire Heudiasyc, UMR 7253 CNRS, 60205 Compiègne, France
{sohaib.afifi,rym-nesrine.guibadj,aziz.moukrim}@hds.utc.fr

Abstract. The Vehicle Routing Problem with Time Windows (VRPTW) consists in determining the routing plan of vehicles with identical capacity in order to supply the demands of a set of customers with predefined time windows. This complex multi-constrained problem has been widely studied due to its industrial, economic and environmental implications. In this work, we are interested in defining the number of vehicles needed to visit all the customers. This objective is very important to evaluate the fixed costs for operating the fleet. In this paper, we provide an analysis of several lower bounds based on incompatibility between customers and vehicle capacity constraints. Then we develop an adaptation of Energetic Reasoning algorithm for VRPTW with a limited fleet. The proposed approach focuses on some time-intervals and exploits time constraints, incompatibility graph and bin packing models in order to obtain new valid lower bounds for the fleet size. Experiments conducted on the standard benchmarks show that our algorithms outperform the classical lower bound techniques and prove the optimality for 339 out of 468 instances.

Keywords: vehicle routing, time windows, lower bounds, energetic reasoning.

1 Introduction

In today's business world, transportation costs become a major share of the total logistic expenses of companies. That is why many companies try to improve their transportation by using rational manners and effective tools. The objective of these problems is to make a vehicle scheduling strategy in order to minimize the number of routes and the corresponding total travel distance or cost. In the literature such problems are referred to as *routing problems*.

The vehicle routing problem with time windows (VRPTW) [8] is among the most studied variants of routing problems due its wide range of applications. Common examples are newspaper delivery, beverage and food delivery, commercial and industrial waste collection [11]. In VRPTW, a set of customers must be served by a fleet of vehicles located in a single depot. A quantity of goods should be delivered to each customer whose service takes an amount of time.

Every customer is associated with a time window that represents the interval of time when the customer is available to receive the vehicle service. This means that if the vehicle arrives too soon, it should wait until the opening of the time window to serve the customer while too late arrival is not allowed. Since deliveries cannot be split, a customer is always served by a single vehicle. All vehicles are identical and have a maximum capacity Q. The aim is to plan the minimal number of routes starting and ending in a unique depot in order to serve all the customers while respecting all the time windows and capacity constraints.

VRPTW was first introduced by [23]. Both exact and heuristic algorithms have been proposed to solve VRPTW. Most of the exact methods focus on the variant of the problem where the number of available vehicles is not fixed. A review on the exact methods up to 2002 is reported in [5]. Kallehauge in [15] gave a detailed analysis of existing formulations. More recently, Baldacci et al. [3] reviewed mathematical formulations, relaxations and recent exact methods. They reported the computational comparison between the methods proposed in [13], [6] and [2] that are considered as the most effective exact methods in the literature. These approaches have significantly improved the quality of the lower bounds for instances with up to 100 customers. The key factor of their success is the effective combination between the set partitioning formulation and the column generation based algorithms.

Since, VRPTW is an NP-Hard problem [19], the computational times for exact methods can be very high, even for instances with a moderate size. This has been the motivation for some researches to develop approximate methods. It is worth pointing out that the literature concerning VRPTW is split according to the objective considered. While exact methods usually minimize the total traveled distance, most heuristics consider a hierarchical objective which first minimizes the number of vehicles used and then the total distance. Thus, a solution that employs fewer vehicles is always better than a one using more, even if the total traveled distance of the first solution is worse. The best performing heuristics are the hybrid genetic algorithm of [14], the column generation heuristic of [1] and the memetic algorithm of [18]. A new optimization framework was later developed by [25] for the distance minimization objective only. This framework is an iterative procedure between optimization and deterioration phases and uses a genetic algorithm as an optimization methodology. A third stream of research focuses on solving VRPTW as a multi-objective problem in which both vehicles and cost are considered depending on the needs of the user [24] [22].

The goal of this paper is to use scheduling methods via Energetic Reasoning in order to develop new lower bounding procedures for VRPTW. This is mainly based on constraint propagation concept. The objective is to reduce the computational effort by removing some values from the variables of the problem because a given subset of the constraints cannot be satisfied. The remained of the paper is organized as follows. Section 2 briefly describes the problem. In Sections 3 and 4, the detailed description of the proposed lower bound methods is given and in Section 5 the results of a computational study are reported. Finally, Section 6 provides some concluding remarks.

2 Problem formulation

In the following, we present a mixed integer formulation for VRPTW. The problem is modeled by an oriented graph $G = (V^+, E)$, where $V^+ = \{0, 1, 2, ..., n\}$ is the vertex set representing the set of customers $V = \{1, 2, ..., n\}$ and the depot 0. $E = \{(i, j) : i \neq j, i, j \in V^+\}$ is the edge set. The capacities of all vehicles are equal and are denoted by Q. A demand q_i, a service time s_i and a time window $[e_i, l_i]$ are associated to each vertex $i \in V$. If the vehicle v arrives earlier than e_i, it must wait before the service can start. Each edge $(i, j) \in E$ is associated with a travel cost $\delta_{i,j}$ which satisfies the triangle inequality. The vehicle must start and finish its tour at the depot. Each customer must be served within a predefined time window and assigned to exactly one vehicle. The total size of deliveries for customers assigned to the same vehicle must not exceed the vehicle capacity Q and the travel cost/time $C(R)$ of each tour R must not exceed l_0 which is the latest possible arrival time to the depot.

The model involves three types of variables: the binary routing variables $x_{ij} \in \{0, 1\}\,(i, j \in V^+)$, the scheduling variables $w_i \geq 0\,(i \in V)$ and the vehicle load variables $y_i\,(i \in V)$. The routing variables x_{ij} is one if a vehicle traverses the arc $(i, j) \in E$. The scheduling variable w_i denotes the time the vehicle arrives at customer $i \in V$. y_i denotes the vehicle load at departure from customer i. The formulation is as follows:

$$\min \sum_{i \in V} x_{0i} \tag{1}$$

subject to:

$$\sum_{j \in V^+} x_{ij} = 1 \qquad \forall i \in V \tag{2}$$

$$\sum_{j \in V^+} x_{ij} - \sum_{j \in V^+} x_{ji} = 0 \qquad \forall i \in V^+ \tag{3}$$

$$w_j \geq w_i + x_{ij}(\max(\delta_{i,j} + s_i, e_j - l_i)) - (1 - x_{ij})(l_i - e_j) \qquad \forall i, j \in V \tag{4}$$

$$e_i \leq w_i \leq l_i \qquad \forall i \in V \tag{5}$$

$$y_j \geq y_i + q_j - (1 - x_{ij})(Q - q_j) \qquad \forall i, j \in V \tag{6}$$

$$q_i \leq y_i \leq Q \qquad \forall i \in V \tag{7}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i, j \in V^+ \tag{8}$$

The objective (1) is to minimize the total number of vehicles used to serve the customers. Constraints (2) and (3) define the routing network and the constraints (4) and (5) guarantee the connectivity of each tour and ensure that the time windows are respected. We assume that the time windows are ajusted such that $e_i = \max(e_i, \delta_{0,i})$ and $l_i = \min(l_i, l_0 - (\delta_{i,0} + s_i))\ \forall i \in V$. Constraints (6) and (7) ensure that the vehicle's capacity is not exceeded. Also constraints (6) eliminate subtours in a manner similar to (4). Finally, (8) are integral constraints.

3 Classical lower bounding techniques

There were only few attempts to propose lower bounds for VRPTW when the objective is to minimize the number of vehicles. To the best of our knowledge, the most competitive results are currently offered by Kontoravdis and Bard [17]. In this section, we briefly review the main features of their lower bounding heuristics.

3.1 A lower bound based on incompatibilities between customers

The first lower bound is deduced from the incompatibility constraints. Let i and j be two customers. If there is no feasible route containing i and j then they define an incompatible pair denoted by $i||j$. Such a situation occurs if one of the following conditions is verified:

1. Customers i and j cannot be in the same route due to their time window constraints:
   $(e_i + s_i + \delta_{i,j} > l_j) \wedge (e_j + s_j + \delta_{j,i} > l_i) \Rightarrow i||j$.
2. The travel cost of any tour with i and j exceeds the cost limit l_0:
   $(C(R_1) > l_0) \wedge (C(R_2) > l_0)$ where $R_1 = (0, i, j, 0) \wedge R_2 = (0, j, i, 0) \Rightarrow i||j$.
3. The sum of the demands is greater than the vehicle capacity:
   $q_i + q_j > Q \Rightarrow i||j$.

Using these conditions, we build the graph of incompatibilities between customers defined as: $G^V_{inc} = (V, E_V)$ where $E_V = \{(i, j) \in V \times V : i||j\}$. Based on this graph the minimum number of routes to be used, denoted LB_{Clique}, is equal to the size of the maximum clique extracted from G^V_{inc}.

3.2 A lower bound based on vehicle capacity constraints

The second bound is based on a relaxation of time window constraints. When considering only the capacity constraints, VRPTW can be reduced to a Bin Packing Problem (BPP). Each vehicle is considered as a bin with fixed size Q and each customer demand as an item with size q_i that should be put in a bin. Any lower bound $LB_{Capacity}$ on the number of bins required to pack all the items is considered as a valid lower bound for VRPTW.

3.3 A lower bound based on the amount of needed travel time

This lower bound consists of calculating the minimum number of bins LB_{BP} of capacity l_0 to pack $n + m$ items. The size θ_i of an item $i, 1 \leq i \leq n$, represents the necessary amount of time that a vehicle needs to serve customer i and to travel to its closest neighbor. This time is defined by:

$$\theta_i \leftarrow \min_{j \in V^+} \{\max(\delta_{i,j} + s_i, e_j - l_i)\} \tag{9}$$

The sizes of the other m items correspond to the m least travel times from the depot to the first served customers where $m = \max(LB_{Clique}, LB_{Capacity})$.

4 New lower bounds inspired from Energetic Reasoning

In this section, we first present a brief overview of Energetic Reasoning, then we discuss its adaptation to VRPTW.

4.1 Energetic Reasoning

Energetic Reasoning (ER) is one of the most powerful propagation algorithms. It has been originally developed by Erschler et al. [7] for Cumulative Scheduling Problems (CuSP). The idea is to propose a smart way to simultaneously consider time and resource constraints in a unique reasoning. In this context, the energy is generally defined by multiplying the time duration by the resource quantity of a given time interval. Considering the quantities of energy supplied by the resources and consumed by the tasks within given intervals, the energetic approach aims to develop satisfiability tests to ensure that a given schedule is feasible. Since its inception, Energetic Reasoning has gained popularity and has been used for solving more complex scheduling problems [21].

In order to keep the same notation used for vehicle routing problem, we describe the CuSP as follows. We consider a set V of n activities to be scheduled on a resource of quantity m. Each activity i has a release time e_i, a latest start time l_i and a processing time s_i. Moreover, the activity i requires a constant amount b_i of resource throughout its processing. We will deal here only with the case where $b_i = 1, \forall i \in V$. This is equivalent to the problem of scheduling n activities on m identical parallel machines. For ease of presentation, we denote this problems as PMSP.

Given a time interval $[t_1, t_2]$, with $t_1 < t_2$, the part of an activity i that must be processed between t_1 and t_2 is called *work* of i in the time interval $[t_1, t_2]$. To compute this *work*, the activities are either *left-shifted* or *right-shifted* on their time window, which means that, they can start either at their release date e_i, or at their latest start time l_i. Thus, the work of an activity i over $[t_1, t_2]$ is equal to the minimum between its *left work* and its *right work*. For convenience, the left work, the right work and the work of an activity i over $[t_1, t_2]$ are denoted respectively $W_{left}(i, t_1, t_2)$, $W_{right}(i, t_1, t_2)$ and $W(i, t_1, t_2)$. They are formally defined as follows:

$$W_{left}(i, t_1, t_2) = \min\{\ t_2 - t_1, s_i, \max(0, e_i + s_i - t_1)\} \tag{10}$$

$$W_{right}(i, t_1, t_2) = \min\{\ t_2 - t_1, s_i, \max(0, t_2 - l_i)\} \tag{11}$$

$$W(i, t_1, t_2) = \min(\ W_{left}(i, t_1, t_2), W_{right}(i, t_1, t_2)) \tag{12}$$

Finally, we define the total work over $[t1, t2]$ as the sum of the works of all the activities $W(t_1, t_2) = \sum_{i=1}^{i=n} W(i, t_1, t_2)$ and the available energy in the considered interval as $E(t_1, t_2) = m * (t_2 - t_1)$. If the total work is greater than the available energy then no feasible solution exists.

Proposition 1 ***satisfiability test***
if $\exists [t_1, t_2]$, $W(t_1, t_2) > E(t_1, t_2)$ *then the instance is infeasible.*

Note that one crucial point to apply efficiently Energetic Reasoning is to determine the relevant time-intervals on which it may be useful to check feasibility conditions. Baptiste et al. [4] have proved that the only relevant time intervals $[t_1, t_2]$ that need to be considered are those where $t_1 \in T_1$ and $t_2 \in T_2$ such as $t_1 < t_2$, $T_1 = \{e_i, i \in V\} \cup \{l_i, i \in V\} \cup \{e_i + s_i, i \in V\}$ and $T_2 = \{l_i + s_i, i \in V\} \cup \{e_i + s_i, i \in V\} \cup \{l_i, i \in V\}$. Therefore, the satisfiability test algorithm runs in $O(n^3)$. The detailed steps are summarized in Algorithm 1.

Algorithm 1: satisfiability test of Energetic Reasoning

```
   Data: I : PMSP instance;
1  begin
2      initialization;
3      T1 = {e_i, i ∈ V} ∪ {l_i, i ∈ V} ∪ {e_i + s_i, i ∈ V};
4      T2 = {l_i + s_i, i ∈ V} ∪ {e_i + s_i, i ∈ V} ∪ {l_i, i ∈ V};
5      foreach t1 ∈ T1 do
6          foreach t2 ∈ T2 such as t1 < t2 do
7              W ← 0;
8              foreach i ∈ V do
9                  W ← W + W(i, t1, t2);
10             if W > m * (t2 − t1) then
11                 Infeasible instance ;
```

4.2 From VRPTW to PMSP

Our approach is to relax a VRPTW instance, where a limited number of vehicles is given, in order to obtain a PMSP instance. Once the transformation is performed, we apply the same satisfiability test on the relaxed m-VRPTW instance, using Algorithm 1. Starting from a trivial value $m = \max(LB_{Clique}, LB_{Capacity})$, feasibility tests are carried out to detect an infeasibility (that is, the vehicle number cannot be less than or equal to m). If an infeasibility is detected, then $m+1$ is a valid lower bound. The process is iterated until no infeasibility is detected.

A trivial relaxation of an m-VRPTW instance can be done by ignoring travel times, customer demands and vehicle capacities. We obtain a PMSP where the vehicles are considered as m identical parallel machines, the number of activities is equal to the number of customers n and each activity i has to be processed for s_i units of time by only one machine. The processing of activity i cannot be started before its release date e_i, and each activity i has a delivery time $l_i + s_i$.

In vehicle routing problems, travel times are not negligible compared to the service times. Ignoring the travel time would undervalue the energy consumed. Therefore, few adjustments could be performed and Energetic Reasoning becomes inefficient. Better results are obtained by considering the time that a

vehicle needs to travel in order to visit each customer. First, the travel time $\delta_{i,j}$ between the customers i and j is updated to eliminate the waiting time at the customer j.

$$\delta_{i,j} \leftarrow \max(\delta_{i,j}, e_j - (l_i + s_i)) \quad \forall i \in V \ \forall \ j \in V^+ \tag{13}$$

Then, the number of potential successors of customer i is reduced. This is performed by eliminating the transition $\delta_{i,j}$ if j cannot be served after i due to its time window:

$$if(e_i + s_i + \delta_{i,j} > l_j) \quad then \quad \delta_{i,j} \leftarrow \infty \qquad \forall i \in V^+ \ \forall \ j \in V^+ \setminus \{i\} \tag{14}$$

Before giving the detail of our travel evaluation procedure, we note by I' the instance derived from the m-VRPTW instance I. We associate I' with a graph $G' = (V', E')$ which is built by performing the following transformations:

1. We introduce m artificial departure vertices V_d and m artificial arrival vertices V_a. Then, we define the set $V' = V \cup V_d \cup V_a$ with $V = \{1, ..., n\}$, $V_d = \{n+1, ..., n+m\}$ and $V_a = \{n+m+1, ..., n+2 \cdot m\}$.
2. The set of arcs is defined by $E' = E \cup \{(i,j) : i \neq j, i \in V \cup V_d, j \in V \cup V_a\}$.
3. The distance matrix $\Delta = (\delta_{i,j})$ is extended to $\Delta' = (\delta'_{i,j})$ which is associated to E' such as:

$$\delta'_{i,j} = \begin{cases} \delta_{i,j} \ (i,j \in V), \\ \delta_{0,j} \ (i \in V_d, j \in V), \\ \delta_{i,0} \ (i \in V, j \in V_a), \\ \infty \ \ (i \in V', j \in V_d) \ or \ (i \in V_a, j \in V') \end{cases} \tag{15}$$

The set of vertices V' in G' denotes the $n + 2 \cdot m$ activities assigned to I'. The artificial departure activities corresponding to V_d have a time window equal to $[0, 0]$ and durations equal to the m smallest travel time from the depot to customers $\{\delta^{min}_{0,1}, ..., \delta^{min}_{0,m}\}$. This supposes that the vehicles must leave the depot immediately in order to visit the m first customers. The artificial arrival activities corresponding to V_a have the largest possible time window $[0, l_0]$ and no processing time. For the remaining activities, the range of the possible start dates is equal to the customer's time window $[e_i, l_i]$ and the processing time is equal to the sum of the service time s_i and the minimal travel time that the vehicle will necessary perform to reach the next customer.

$$[e'_i, l'_i] = \begin{cases} [0,0] & i \in V_d, \\ [0, l_0] & i \in V_a, \\ [e_i, l_i] & i \in V \end{cases} \tag{16}$$

$$s'_i = \begin{cases} \delta^{min}_{0,i-n} & i \in V_d, \\ 0 & i \in V_a, \\ s_i + min_{j \in V'}\{\delta'_{i,j}\} & i \in V \end{cases} \tag{17}$$

Each row i , $i \in V'$ of the extended matrix Δ' is updated by subtracting the smallest element from the remaining ones (18). This means that the minimal travel time after serving customer i is subtracted from the total distance of any solution since every solution must include only one customer from this row. This process is called *reducing the rows*. It was introduced by [20] in order to solve the well known Traveling Salesman Problem.

$$\delta'_{i,j} = \begin{cases} \delta'_{i,j} - min_{j \in V'}\{\delta'_{i,j}\} & \forall i \in V j \in V', \\ \max(0, \delta'_{i,j} - \delta^{min}_{0,i-n}) & \forall i \in V_d j \in V' \end{cases} \tag{18}$$

Next, we apply the same argument to the resulting matrix, by considering the minimal travel time to arrive from any customer j to customer i (19). This time is added at the beginning of activity i. For this reason, the bounds of the corresponding time window are shifted (20) (21). After these reducing operations, the matrix Δ' contains at least one zero in each row and each column. The Figure 1 illustrates the relaxation of an m-VRPTW instance with 4 customers and 2 vehicles.

$$s'_i \leftarrow s'_i + min_{j \in V'}\{\delta'_{j,i}\} \qquad \forall i \in V' \tag{19}$$

$$e'_i \leftarrow \max(0, e'_i - min_{j \in V'}\{\delta'_{j,i}\}) \qquad \forall i \in V' \tag{20}$$

$$l'_i \leftarrow \max(0, l'_i - min_{j \in V'}\{\delta'_{j,i}\}) \qquad \forall i \in V' \tag{21}$$

According to the evaluation procedure of travel times, we distinguish two possible lower bounds $LB_{ER_{eval1}}$ and $LB_{ER_{eval2}}$. The former is obtained if the travel times to the successors are considered before the remaining travel times from the predecessors whereas the latter is obtained by reversing the order of the considered travels. LB_{ER} denotes the maximum between $LB_{ER_{eval1}}$ and $LB_{ER_{eval2}}$.

4.3 Bin-packing lower bounds and Energetic Reasoning

We extend Energetic Reasoning, using the Bin-Packing Problem with Conflicts (BPPC), to get tighter lower bounds for VRPTW. In each time-interval $[t_1, t_2]$, we compute the mandatory parts of activities and then we deduce an associated bin-packing instance. The decision version of BPPC that we use can be formulated as follows: given a set of items with different weights and a graph where the vertices represent the items and the edges represent the conflicts between the pairs of items; is there a packing of these items in less than m bins with a capacity T ?

We now state the link between a necessary condition for the existence of m-VRPTW solution and the existence of BPPC solution. Let $I'(V', G_{inc}, m)$ denote a relaxed instance of m-VRPTW where V' is the set of activities, m

the number of available vehicles and G_{inc} the graph of incompatibilities between activities. Let $[t_1, t_2]$ be a time-interval, we assume that the corresponding mandatory parts of activities have been computed: $W(i, t_1, t_2), \forall i \in V'$. Then, $BPPC(I', G_{inc}, t_1, t_2)$ denotes the packing instance which is associated to the scheduling instance I' in the time-interval $[t_1, t_2]$. $BPPC(I', G_{inc}, t_1, t_2)$ is made of n' items of size $W_i = W(i, t_1, t_2), \forall i \in \{1, ..., n\}$ and m bins. The size of the bin is equal to the length of the time-interval $T = t_2 - t_1$. Then, deciding whether all mandatory parts of the activities can be scheduled within $[t_1, t_2]$ in I' is equivalent to determine for $BPPC(I', G_{inc}, t_1, t_2)$ if all items can be packed into the available bins.

Property 1. If there exists a time-interval $[t_1, t_2]$, such that $BPPC(I', G_{inc}, t_1, t_2)$ has no solution, then there is no solution to the initial problem $I'(V, G_{inc}, m)$.

As stated in Section 4.2, Energetic Reasoning uses two procedures to determine the processing time of activities. The new obtained lower bound LB_{ERBPPC} represents the maximum between LB_{ERBPPC_eval1} and LB_{ERBPPC_eval2}. In the same way and by ignoring the conflict constraints, we can obtain a quicker lower bound LB_{ERBPP}.

The example in Figure 2 illustrates the contribution of bin packing lower bounds in the improvement of Energetic Reasoning results. We consider a VRPTW instance with 8 customers defined by their time windows and service times. We suppose that the vehicle capacity is large enough to satisfy all customers demands and that customer 8 cannot be served with any other customer. The results obtained by $LB_{Capacity}$ and LB_{Clique} are equal to 1 and 2 respectively. When analyzing the interval $[t_1, t_2]$, the Energetic Reasoning LB_{ER} gives 3. This result is improved by applying Bin Packing lower bound and taking into account the conflicts between customers ($LB_{ERBPP} = 4$ and $LB_{ERBPPC} = 5$).

5 Numerical results

We tested our algorithms on the well known instances of Solomon [23], Gehring and Homberger [9]. The benchmark comprises 6 sets ($R1$, $C1$, $RC1$, $R2$, $C2$, $RC2$). Each data set contains $25, 50, 100, 200, 400, 600, 800$ and 1000 customers who have specific euclidean coordinates. Customers' locations are determined using a random uniform distribution for the problem sets $R1$ and $R2$, but are restricted to be within clusters for the sets $C1$ and $C2$. Sets $RC1$ and $RC2$ have a combination of clustered and randomly placed customers. Sets $R1$, $C1$ and $RC1$ have a short scheduling horizon with tight time windows, while $R2$, $C2$ and $RC2$ are based on wide time windows. Our algorithms are coded in C++ and all experiments were conducted on an Intel(R) Core(TM) 2 Duo 2.93GHz.

Finding a clique with the greatest cardinality involves the use of an exact method with exponential worst case performance. Nevertheless, our experiments on the standard benchmarks show that the maximum clique can be identified in a fraction of a second using the exact method described in [16]. For the Bin

Packing Problem, we use the heuristic algorithm developed by [12] to get good lower bounds in a reasonable computational times. When conflicts are considered in solving Bin Packing, we apply the approach proposed in [10]. For performance purpose, we launch this algorithm only on intervals with big conflict density (80%) and with a time out of 3 hours.

Table 1 and Table 2 compare the performance of our Energetic Reasoning bounds: LB_{ER}, LB_{ERBPP} and LB_{ERBPPC} to the elementary bounds of litterature: LB_{Clique}, $LB_{Capacity}$ and LB_{BP}. The column $BestUB$ represents the overall best-published upper bounds. The maximum of the lower bounds is reported in column $BestLB$. In $AvgGAP$, we present the average gap between $BestUB$ and $BestLB$.

In general, the proposed techniques prove the optimality of 339 instances among the 468 instances tested and give near optimal solution for the rest. The average performance of $LB_{Capacity}$ is consistently better than LB_{Clique}, but LB_{Clique} outperforms $LB_{Capcaity}$ in 5 instances in $C1$, 25 instances in $R1$, 4 instances in $RC1$ and 1 instance in $RC2$ by a margin of 128. This is due to the structure of the data sets which does not favor time and capacity incompatible pairs. On the other hand, the three new lower bounds: LB_{ER}, LB_{ERBPP} and LB_{ERBPPC} produced consistent results across all data sets. Compared to the classical lower bound techniques: LB_{Clique}, $LB_{Capacity}$ and LB_{BP}, they give better bounds for 23 instances.

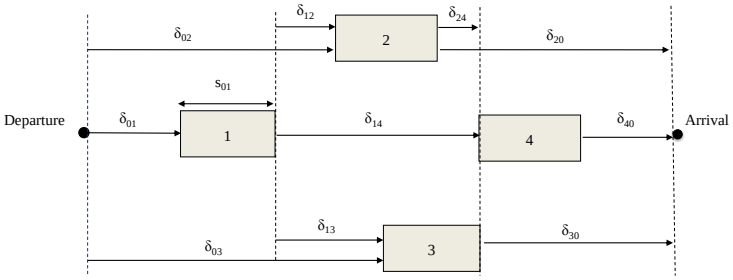
When Energetic Reasoning is combined to BPP (LB_{ERBPP}) and BPPC (LB_{ERBPPC}), the results outperform the bounds produced by LB_{ER} in 3 instances. This is due to the fact that the incompatibilities are considered at each examined time interval. These results confirm that the association of ER and BPPC is very efficient for VRPTW. To conclude, the overall performance of the new lower bounding procedures has been encouraging. The use of Energetic Reasoning improves many lower bounds and gives good results for both capacity constrained problems and time constrained problems.

6 Conclusion

In this paper, we introduced several combinatorial optimization methods which can be used to get lower bounds for the Vehicle Routing Problem with Time Windows (VRPTW). Investigating the concept of Energetic Reasoning, we were able to propose new lower bounding techniques based on the transformation of m-VRPTW instance to PMSP. The numerical results confirm the contribution brought by the new proposed techniques. With a very fast computing time, we were able to prove the optimality of several solutions and provide a reasonable approximation of the optimal number of vehicles required to visit all the customers. This suggests that our lower bounds techniques can quickly produce a good estimation of the fleet size. A challenging area for future research is to develop an exact method using the proposed lower bound procedures.

Acknowledgements

This work is partially supported by the Regional Council of Picardie and the European Regional Development Fund (ERDF), under PRIMA project. It is also partially supported by the National Agency for Research (project ATHENA, Reference ANR-13-BS02-0006-01).



a) Illustration of an m-VRPTW instance with 4 customers and 2 vehicles

i	$[e_i,l_i]$	s_i
1	[2,3]	2
2	[6,6]	2
3	[7,7]	2
4	[9,10]	2
0	[0,14]	0

b) The customers data

from \ to	1	2	3	4	0
1	∞	1	2	4	3
2	∞	∞	∞	1	5
3	∞	∞	∞	∞	6
4	∞	∞	∞	∞	2
0	3	5	6	2	∞

c) The distance matrix Δ

departure vertices / arrival vertices

	∞	1	2	4	∞	∞	3	3
	∞	∞	∞	1	∞	∞	5	5
	∞	∞	∞	∞	∞	∞	6	6
	∞	∞	∞	∞	∞	∞	2	2
departure vertices	3	5	6	2	∞	∞	∞	∞
	3	5	6	2	∞	∞	∞	∞
arrival vertices	∞	∞	∞	∞	∞	∞	∞	∞
	∞	∞	∞	∞	∞	∞	∞	∞

d) The extended distance matrix Δ'

	∞	**0**	1	3	∞	∞	2	2
	∞	∞	∞	0	∞	∞	4	4
	∞	∞	∞	∞	∞	∞	**0**	**0**
	∞	∞	∞	∞	∞	∞	**0**	**0**
	1	3	4	**0**	∞	∞	∞	∞
	0	2	3	**0**	∞	∞	∞	∞
	∞	∞	∞	∞	∞	∞	∞	∞
	∞	∞	∞	∞	∞	∞	∞	∞

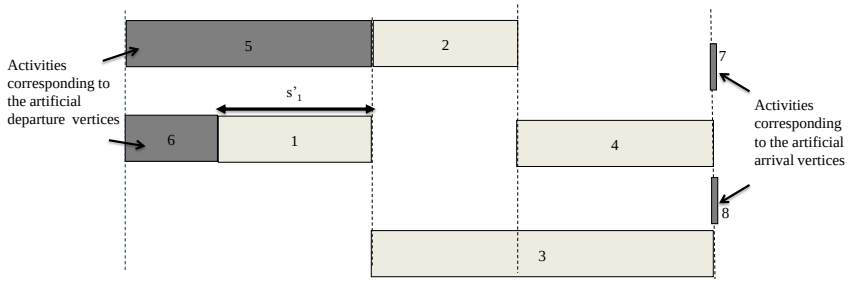
e) Reducing the rows

	∞	**0**	**0**	3	∞	∞	2	2
	∞	∞	∞	0	∞	∞	4	4
	∞	∞	∞	∞	∞	∞	**0**	**0**
	∞	∞	∞	∞	∞	∞	**0**	**0**
	1	3	3	**0**	∞	∞	∞	∞
	0	2	2	**0**	∞	∞	∞	∞
	∞	∞	∞	∞	∞	∞	∞	∞
	∞	∞	∞	∞	∞	∞	∞	∞

f) Reducing the columns

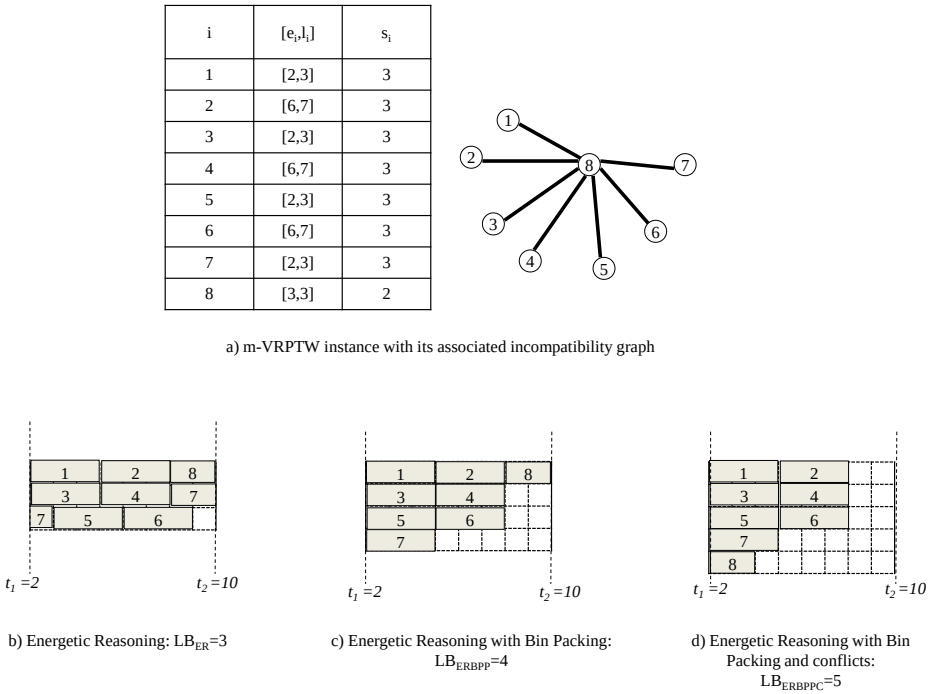
i'	$[e'_i,l'_i]$	s'_i
1	[2,3]	3
2	[6,6]	3
3	[6,6]	9
4	[9,10]	4
5	[0,0]	2
6	[0,0]	3
7	[0,14]	0
8	[0,14]	0

g) The activities data of the resulting PMSP



h) Illustration of the resulting PMSP

Fig. 1. An example of m-VRPTW instance relaxed to PMSP instance.



a) m-VRPTW instance with its associated incompatibility graph

b) Energetic Reasoning: LB_{ER}=3

c) Energetic Reasoning with Bin Packing: LB_{ERBPP}=4

d) Energetic Reasoning with Bin Packing and conflicts: LB_{ERBPPC}=5

Fig. 2. Illustration of Energetic Reasoning lower bounds.

Data Set	n	*Classical*						*New*						*BestLB*	*BestUB*	*AvgGAP*
		Clique		*Capacity*		*BP*		*ER*		*ERBPP*		*ERBPPC*				
		LB	CPU	LB	CPU	LB	CPU	LB	CPU	LB	CPU	LB	CPU			
C1	25	1.89	0	3	0	2	0	3	0	3	0	3	0	3	3	0
C2	25	1	0	1	0	1	0	1.13	0	1.13	0	1.13	0	1.13	1.13	0
R1	25	3.58	0	2	0	3.25	0	3.92	0	3.92	0.01	4	0.26	4	4.75	0.75
R2	25	1	0	1	0	1	0	1.09	0	1.09	0	1.09	0.04	1.09	1.27	0.18
RC1	25	2.75	0	3	0	2.25	0	3.13	0	3.13	0	3.13	0.03	3.13	3.25	0.13
RC2	25	1	0	1	0	1	0	1	0	1	0.01	1	0.08	1	1.5	0.5
C1	50	3	0	5	0	4	0	5	0	5	0	5	0	5	5	0
C2	50	1	0	2	0	2	0	2	0	2	0	2	0	2	2	0
R1	50	5.25	0	4	0	5.17	0	6.33	0.02	6.33	0.09	6.42	76.41	6.42	7.42	1
R2	50	1	0	1	0	1.18	0	1.27	0.01	1.27	0.06	1.27	0.23	1.27	2	0.73
RC1	50	4.25	0	5	0	4.13	0	5.25	0.01	5.25	0.05	5.38	89.88	5.38	6.5	1.13
RC2	50	1.13	0	1	0	1	0	1.13	0.01	1.13	0.05	1.13	0.28	1.13	2	0.88
C1	100	4.89	0	10	0	8	0	10	0	10	0	10	0	10	10	0
C2	100	1.38	0	3	0	3	0	3	0	3	0	3	0	3	3	0
R1	100	7.92	0	8	0	8.33	0	10.42	0.11	10.42	0.45	10.42	364	10.42	11.92	1.5
R2	100	1.18	0	2	0	2	0	2.09	0.07	2.09	0.31	2.09	27.2	2.09	2.73	0.64
RC1	100	6.38	0	9	0	7.63	0	9.63	0.1	9.63	0.45	9.63	99.52	9.63	11.5	1.88
RC2	100	1.13	0	2	0	2	0	2.13	0.11	2.13	0.44	2.13	33.16	2.13	3.25	1.13

Table 1. Average lower bound results and CPU times for Solomon instances

Data Set	n	*Classical*						*New*						*BestLB*	*BestUB*	*AvgGAP*
		Clique		*Capacity*		*BP*		*ER*		*ERBPP*		*ERBPPC*				
		LB	CPU	LB	CPU	LB	CPU	LB	CPU	LB	CPU	LB	CPU			
C1	200	8.8	0	18	0	15	0	18.3	0.41	18.3	1.82	18.3	429.41	18.3	18.9	0.6
C2	200	2.1	0	6	0	6	0	6	0	6	0	6	0	6	6	0
R1	200	10.4	0	18	0	7.3	0	18.2	0	18.2	0	18.2	0	18.2	18.2	0
R2	200	1.6	0	4	0	2	0	4	0	4	0	4	0	4	4	0
RC1	200	6.8	0	18	0	6.5	0	18	0	18	0	18	0	18	18	0
RC2	200	1.9	0	4	0	2	0	4	0.15	4	0.75	4	140.56	4	4.3	0.3
C1	400	16.9	0.01	36	0.01	26.5	0.01	36.5	2.8	36.5	12.47	36.5	2880.12	36.5	37.6	1.1
C2	400	2.5	0.01	11	0.01	11	0.01	11.2	2.6	11.2	12.31	11.2	2880.09	11.2	11.6	0.4
R1	400	17.9	0.01	36	0.01	11.7	0.01	36.4	0	36.4	0	36.4	0	36.4	36.4	0
R2	400	2.4	0.01	8	0.01	2.7	0.01	8	0	8	0	8	0	8	8	0
RC1	400	12.8	0.01	36	0.01	10.8	0.01	36	0	36	0	36	0	36	36	0
RC2	400	3.1	0.01	8	0.01	2.8	0.01	8	1.11	8	5.55	8	1440.04	8	8.4	0.4
C1	600	24.2	0.03	56	0.02	40.4	0.02	56.4	5.35	56.4	31.25	56.4	2160.32	56.4	57.2	0.8
C2	600	4.3	0.02	17	0.01	15.9	0.02	17.1	5.92	17.1	28.72	17.1	2160.31	17.1	17.4	0.3
R1	600	28.1	0.03	54	0.01	13.9	0.02	54.5	0	54.5	0	54.5	0	54.5	54.5	0
R2	600	4.3	0.02	11	0.01	3.4	0.01	11	0	11	0	11	0	11	11	0
RC1	600	19.7	0.04	55	0.02	12.2	0.02	55	0	55	0	55	0	55	55	0
RC2	600	4.7	0.02	11	0.01	2.9	0.02	11	3.63	11	18.38	11	1440.16	11	11.4	0.4
C1	800	34.4	0.08	72	0.05	49.3	0.05	72.8	18.36	72.8	97.45	72.8	4322.5	72.8	75	2.2
C2	800	5.7	0.05	22	0.04	21	0.04	22.2	48.55	22.2	228.25	22.2	9722.18	22.2	23.3	1.1
R1	800	35.3	0.06	72	0.05	15.8	0.05	72.8	0	72.8	0	72.8	0	72.8	72.8	0
R2	800	5.3	0.05	15	0.04	3.5	0.04	15	0	15	0	15	0	15	15	0
RC1	800	27.5	0.41	72	0.05	14.9	0.04	72	0	72	0	72	0	72	72	0
RC2	800	6.6	0.05	15	0.04	3.5	0.04	15	8.96	15	45.7	15	2160.52	15	15.4	0.4
C1	1000	44.6	0.34	90	0.09	58	0.09	91	35.22	91	185.64	91	4324.8	91	93.9	2.9
C2	1000	7.9	0.11	28	0.09	25.1	0.07	28.1	67.81	28.1	284.21	28.1	6484.59	28.1	28.8	0.7
R1	1000	44.5	0.13	91	0.09	19.5	0.08	91.9	0	91.9	0	91.9	0	91.9	91.9	0
R2	1000	6.5	0.09	19	0.08	4	0.07	19	0	19	0	19	0	19	19	0
RC1	1000	31.5	0.79	90	0.08	17.7	0.08	90	0	90	0	90	0	90	90	0
RC2	1000	7.8	0.1	18	0.08	4.1	0.07	18	8.21	18	40.32	18	1080.57	18	18.2	0.2

Table 2. Average lower bound results and CPU times for Gehring and Homberger instances

Bibliography

[1] Alvarenga, G. B., Mateus, G. R., and de Tomi, G. (2007). A genetic and set partitioning two-phase approach for the vehicle routing problem with time windows. *Computers and Operations Research*, 34(6):1561 – 1584.

[2] Baldacci, R., Mingozzi, A., and Roberti, R. (2011). New route relaxation and pricing strategies for the vehicle routing problem. *Oper. Res.*, 59(5):1269–1283.

[3] Baldacci, R., Mingozzi, A., and Roberti, R. (2012). Recent exact algorithms for solving the vehicle routing problem under capacity and time window constraints. *European Journal of Operational Research*, 218(1):1–6.

[4] Baptiste, P., Pape, C. L., and Nuijten, W. (1999). Satisfiability tests and time-bound adjustments for cumulative scheduling problems. *Annals of Operations Research*, 92:305–333.

[5] Cordeau, J.-F., Desaulniers, G., Desrosiers, J., F., M. M. S., and Soumis (2001). The vehicle routing problem. chapter VRP with Time Windows, pages 157–193. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA.

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

[7] Erschler, J., Lopez, P., and Thuriot, C. (1991). Raisonnement temporel sous contraintes de ressources et problèmes d'ordonnancement. *Revue d'Intelligence Artificielle*, 5(3):7–36.

[8] Fisherand, M. L., Jörnsten, K. O., and Madsen, O. B. G. (1997). Vehicle routing with time windows: Two optimization algorithms. *Operations Research*, 45(3):488–492.

[9] Gehring, H. and Homberger, J. (1999). A parallel hybrid evolutionary metaheuristic for the vehicle routing problem with time windows. *Proceedings of EUROGEN99*, pages 57–64.

[10] Gendreau, M., Laporte, G., and Semet, F. (2004). Heuristics and lower bounds for the bin packing problem with conflicts. *Computers and Operations Research*, 31(3):347–358.

[11] Golden, B. L., Assad, A. A., and Wasil, E. A. (2001). The vehicle routing problem. chapter Routing vehicles in the real world: applications in the solid waste, beverage, food, dairy, and newspaper industries, pages 245–286. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA.

[12] Haouari, M. and Gharbi, A. (2005). Fast lifting procedures for the bin packing problem. *Discrete Optimization*, 2(3):201–218.

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

[14] Jung, S. and Moon, B. R. (2002). A hybrid genetic algorithm for the vehicle routing problem with time windows. In *GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference, New York, USA, 9-13 July 2002*, pages 1309–1316. Morgan Kaufmann.

[15] Kallehauge, B. (2008). Formulations and exact algorithms for the vehicle routing problem with time windows. *Computers and Operations Research*, 35(7):2307 – 2330.

[16] Konc, J. and Janezic, D. (2007). An improved branch and bound algorithm for the maximum clique problem. *proteins*, 58:569–590.

[17] Kontoravdis, G. and Bard, J. (1995). A GRASP for the vehicle routing problem with time windows. *ORSA journal on Computing*.

[18] Labadie, N., Prins, C., and Reghioui, M. (2008). A memetic algorithm for the vehicle routing problem with time windows. *Rairo-operations Research*, 42:415–431.
[19] Lenstra, J. K., Kan, A. H., and Rinnooy, G. (1981). Complexity of vehicle routing and scheduling problems. *Networks*, 11(2):221–227.
[20] Little, J. D. C., Murty, K. G., Sweeney, D. W., and Karel, C. (1963). An algorithm for the traveling salesman problem. 11(6):972–989.
[21] Néron, E., Baptiste, P., and Gupta, J. N. D. (2001). Solving hybrid flow shop problem using energetic reasoning and global operations. *Omega*, 29(6):501–511.
[22] Ombuki, B., Ross, B. J., and Hanshar, F. (2006). Multi-objective genetic algorithms for vehicle routing problem with time windows. *APPLIED INTELLIGENCE*, 24:17–30.
[23] Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35(2):254–265.
[24] Tan, K. C., Chew, Y. H., and Lee, L. H. (2006). A hybrid multiobjective evolutionary algorithm for solving vehicle routing problem with time windows. *Comput. Optim. Appl.*, 34(1):115–151.
[25] Ursani, Z., Essam, D., Cornforth, D., and Stocker, R. (2011). Localized genetic algorithm for vehicle routing problem with time windows. *Applied Soft Computing*, 11(8):5375–5390.