# A Multi-Stage Very Large-Scale Neighborhood Search for the Vehicle Routing Problem with Soft Time-Windows

Sébastien Mouthuy, Florence Massen, and Yves Deville

Université catholique de Louvain
Department of Computing Science and Engineering
Place Sainte-Barbe 2, 1348 Louvain-la-Neuve, Belgium
{Sebastien.Mouthuy, Florence.Massen, Yves.Deville}@uclouvain.be

Pascal Van Hentenryck

NICTA and the Australian National University
Tower A,
7 London Circuit
Canberra City ACT 2601
Australia
pvh@nicta.com.au

November 8, 2013

**Abstract**

This paper considers the Vehicle Routing Problem with Soft Time Windows, a challenging routing problem, where customer's time windows may be violated at a certain cost. The Vehicle Routing Problem with Soft Time Windows has a lexicographic objective function, aiming at minimizing first the number of routes, then the number of violated time windows and finally the total routing distance. We present a multi-stage Very Large-Scale Neighborhood search for this problem. Each stage corresponds to a Variable Neighborhood Descent over a parametrizable Very Large-Scale Neighborhood. These neighborhoods contain an exponential number of neighbors, as opposed to classical local search neighborhoods. Often, searching Very Large-Scale Neighborhoods can produce local optima of a higher quality than polynomial-sized neighborhoods. Furthermore we use a sophisticated heuristic to determine service start times allowing to minimize the number of violated time windows. We test our approach on number of different problem types, and compare the results to the relevant state-of-the-art. The experimental results show that our algorithm improves best-known solutions on 53% of the most studied instances. Many of these improvements stem from a reduction of the number of vehicles, a critical objective in Vehicle Routing Problems.

## 1 Introduction

Vehicle Routing Problems with Hard Time Windows (VRPTHW), where the service at each customer must begin in a fixed time window, have received a lot of attention in the literature. However the case

where the time windows of the customers may be violated at a certain penalty is treated considerably less frequently. This latter problem is known as the Vehicle Routing Problem with Soft Time Windows (VRPSTW). The VRPSTW aims at scheduling several customer visits among a set of vehicles. For each customer, hard and soft time windows are given. Each customer has to be visited within the associated hard time window, whereas there is a penalty cost incurred for each customer not visited within the associated soft time window. The problem then asks for the solution that has the lowest number of routes, the lowest number of violated time windows and the lowest total distance. This paper studies the application of Very Large-Scale Neighborhoods to the VRPSTW. The objective function for the VRPSTW is hierarchical, which is why we chose a multi-stage approach. Such approaches have been shown to work well for this type of problems, see for example (Bent and Van Hentenryck, 2004) or (Homberger and Gehring, 2005). Our algorithm uses the same very large-scale neighborhood for each stage, but with different objective functions: Vehicle reduction, minimization of the soft time window violations, and travel distance minimization. Moreover, each stage uses a variable neighborhood descent (VND) (Mladenović and Hansen, 1997). Experimental results indicate that our multi-stage VLSN algorithm improves best-known solutions on 37%, 57% and 100% of the Type 1,2 and 3 instances respectively, which are standard benchmarks for the VRPSTW (see (Fu et al., 2007)). Equally interesting is the fact that the multi-stage algorithm decreases the number of routes in 27% of the instances (up to 4 less routes) and the soft time window violations in 35% (up to 27% less violated time windows) of the remaining instances.

This paper describes extended work from a previous paper presented at the 9th Metaheuristics International Conference in Udine, Italy (Mouthuy et al., 2011). Our contributions are multifold. First we present a Very Large-Scale Neighborhood, parametrizable with the objective to minimize, for the Vehicle Routing Problem with Soft Time Windows. We show how this VLSN can be embedded in a Variable Neighborhood Descent, upon which we construct a multi-stage approach. Next, as opposed to the majority of the state-of-the-art for the VRPSTW we consider the problem of setting service start times such as to minimize the soft time window violations, and propose a sophisticated but simple algorithm to do this. We also extensively test our proposed multi-stage approach on standard benchmark instances. Our method is able to provide a new best solution in 50% of the considered Type 1, 2 and 3 instances. For the Type 3 instances with $p_{max} = 5\%$ or $p_{max} = 10\%$ and $w_{max} = 10\%$ we are even able to improve all of the best known solutions.

The remainder of this paper is organized as follows. In section 1.1 we overview existing work on the Vehicle Routing Problem with Soft Time Windows and briefly review Very Large-Scale Neighborhoods and related concepts for Vehicle Routing Problems. In section 2 we introduce the Vehicle Routing Problem with Hard Time Windows and show how the Soft Time Problem can be derived from it. Next, we explain how we can decide whether for a given route we can find service start times for each customer such that the hard time windows are respected. This is done in section 3. We continue by introducing Very Large-Scale Neighborhoods and related concepts in section 4. Section 5 explains how we modelize the VRPSTW as a Combinatorial Optimization Problem. We introduce our parametrizable Very Large-Scale Neighborhood for the VRPSTW in section 6. The multi-stage VLSN search is then presented in section 7. Finally we provide experimental results and compare them to the state-of-the-art in section 8.

## 1.1 Related Work

**Vehicle Routing Problems with Soft Time Windows** Different variations of the VRPSTW have been considered throughout the state-of-the-art. The authors in (Fu et al., 2007) classified the possible types of problems with linear penalty functions into 6 types. So far Types 1–5 have been considered in the literature.
In (Taillard et al., 1997), an adaptive memory tabu search is presented for Type 1 problems. In this prob-

lem vehicles may arrive early at a customer and wait, however late arrival incurs a penalty per late time unit. The proposed approach is tested on the (adapted) Solomon benchmark instances for the VRPHTW ((Solomon, 1987)) and is able to find feasible solutions (i.e. not violating any of the original hard time windows and respecting the number of vehicles) on all but one instance.
A problem of Type 2 is considered in (Koskosidis et al., 1992). In this problem a penalty is incurred per early and late time unit. An iterative cluster-first route-second heuristic is used to find high-quality solutions. The authors test their approach on the (adapted) Solomon benchmark instances on which they decreased the vehicle capacities to favor the uniform distribution of customers between clusters. When compared to the results presented in (Solomon, 1987) on the VRPHTW, they improve the results in terms of total distance on a majority of instances. Only on a handful of instances does this improvement come at the cost of time window violations.
The Type 3 problem was first considered in (Balakrishnan, 1993). Here a maximum allowed violation of the soft time windows and a maximum waiting time are given. A penalty is incurred per early or late time unit. The authors propose three solution construction heuristics for this problem. They test their heuristics on the Solomon benchmark instances and compare their results to the results on the VRPHTW. They conclude that the number of routes as well as the total distance can be decreased by allowing controlled violations of the time windows. In (Chiang and Russell, 2004) a Tabu Search is proposed for the Type 3 problem. The presented Tabu Search is able to solve all problem instances, which wasn't the case for the heuristics proposed in (Balakrishnan, 1993). In terms of number of routes and total distances the Tabu Search outperforms the heuristics on all instances, this at the cost of a higher percentage of violated time windows. The unified tabu search from (Fu et al., 2007) is tested on problems of Type 1, 2 & 3. The obtained results are compared to (Taillard et al., 1997; Koskosidis et al., 1992; Balakrishnan, 1993; Chiang and Russell, 2004) in terms of (by order of priority) number of routes, percentage of non-violated time windows and total distance. The proposed method behaves well on all considered problem types, improving at least half of the results presented in the considered papers. An iterative route construction and improvement algorithm is presented in (Figliozzi, 2010). The authors propose a post-optimization on the service start times in order to reduce the time window violations. An evaluation is done on the Type 3 problems. The authors are able to improve the number of vehicles of (Fu et al., 2007) in more than half of the instances. A problem generator-solver heuristic for VRPSTWs is proposed in (Ioannou et al., 2003). Based on a given problem instance, their generator produces further problem instances, in which a varying number of customers have soft time windows. Originally only a fraction of customers have soft time windows, this fraction is then increased steadily. Each of the generated problems is solved using an adapted nearest neighbor heuristic. The approach also tries to fix small time window violations, as long as this does not increase the number of routes. In their experimental results the authors consider Type 3 problems, and evaluate their approach on the (adapted) Solomon benchmarks. They compare their results to those of (Balakrishnan, 1993) and (Koskosidis et al., 1992) in terms of number of routes and percentage of non-violated time windows, finding they are able to improve the number of non-violated time windows.
The authors in (Qureshi et al., 2009) propose a Column Generation algorithm for Type 4 VRPSTWs. They compute the allowed violation of the time window individually for each customer, based on its distance from the depot and a penalty per late time unit. Their approach is tested on the (adapted) Solomon benchmarks and on a real-life instance.
A problem with general time windows is considered in (Ibaraki et al., 2005). The penalty function associated with early or late arrivals can be non-convex and discontinuous. They propose a local search based on a cyclic exchange neighborhood, a concept similar to VLSNs. Given the nature of the penalty function their algorithm also needs to decide the optimal service start times for the customers, this is done using a dynamic programming method. They test their method on the VRPHTW using the Solomon instances, as well as on two different scheduling problems. On the VRPHTW their results are slightly outperformed by different state-of-the-art approaches, which are however tailored specifically to the VRPHTW.

A VRPSTW with heterogeneous fleet is considered in (Calvete et al., 2007). The authors propose a goal programming model for their problem. The objective is not only to minimize the total travel distance and the soft time window violation, but also vehicle capacity and labor underutilization. Their proposed approach first enumerates feasible routes and then solves a set partitioning problem over the set of generated routes. The number of feasible routes produced is evaluated by applying their approach on custom instances as well as on adapted Solomon benchmark instances, corresponding to Type 3 problem.
A shortest path problem with time windows is considered in (Ioachim et al., 1998). A cost on the nodes, depending on the service start time is considered. The authors propose a dynamic programming algorithm to optimally solve this problem. They test their approach on networks constructed from Solomon's benchmark instances.

**Very Large-Scale Neighborhoods** Very Large-Scale Neighborhood (VLSN) search, is a class of local-search algorithms whose neighborhoods contain an exponential number of neighbors. By considering neighborhoods of exponential size, VLSN search can produce local optima of higher quality than polynomial-sized neighborhoods. These exponential neighborhoods are obtained by considering, as neighbors, configurations that can be reached by a set or sequence of atomic moves.
VLSN algorithms have been successfully applied to a variety of NP-Hard problems such as Vehicle Routing Problems (Ergun et al., 2002; Thompson and Psaraftis, 1993), Capacitated Minimum Spanning Tree (Ahuja et al., 2001, 2003), Exam Timetabling (Abdullah et al., 2007b, 2004, 2007a), the Quadratic Assignment Problem (Deineko and Woeginger, 2000) and Block-to-train Assignment (Jha et al., 2008). VLSN search algorithms behave very well on these applications because there are many constraints that are hard to satisfy. The ability to perform many moves at once enables the search algorithm to improve solutions even though the constraints are tight. The reader is referred to (Ahuja et al., 2002) for a survey of the main applications of VLSN approaches.
Several neighborhoods such as ejection chains (Glover and Rego, 2006), chain-exchanges (Fahrion and Wrede, 1990), cyclic exchanges (Ibaraki et al., 2005) and cyclic transfers (Thompson and Psaraftis, 1993) that have been proposed for the Vehicle Routing Problem and its variants correspond to VLSNs. Note that Large Neighborhood Search, such as used in (Shaw, 1998) can also be considered a VLNS search.
A framework for constraint-based very large neighborhood search has been presented in (Mouthuy et al., 2012) and (Mouthuy, 2011).

# 2 Problem description

In this section we first introduce the Vehicle Routing Problem with Hard Time Windows and then explain how the VRP with Soft Time Windows can be derived from it. Then we provide the notations that will be used in the remainder of this paper.

## 2.1 Vehicle Routing Problem with Hard Time Windows

Let a complete and weighted graph $G = (V, E)$, with $V = \{0, \ldots, n\}$ the set of vertexes. Vertexes $1, \ldots, n$ correspond to customers while vertex 0 represents a depot. With each edge $(i, j) \in E$ is associated a cost $c_{ij}$ representing the distance to travel from vertex $i$ to vertex $j$. Throughout the paper we consider that travel time and distance are equal. Let furthermore a fleet of $K$ vehicles, each of associated capacity $Q$. With each customer $i \in \{1, \ldots, n\}$ are associated the following:

1. a demand $q_i$ that needs to be delivered at the customer's location
2. a time window $[e_i; l_i]$ ($e_i < l_i$ and $l_i \geq c_{0i}$)
3. a duration $d_i$ which corresponds to the time a vehicle spends at the customer's location

Furthermore a time window $[e_0; l_0]$ is associated with the depot.
A solution to the VRPHTW associates a (possibly empty) route with each vehicle in the fleet and a service start time $s_i$ with each customer such that:

- each route starts and ends at the depot
- the sum of the demands $q_i$ on each route does not exceed the vehicle capacity $Q$
- each customer is visited by exactly one route
- the vehicles do not start from the depot earlier than $e_0$
- the vehicles return to the depot by $l_0$
- the service start time $s_i$ lies in $[e_i; l_i]$ for each customer $i$
- no vehicle waits for more than $w_{max}$ time units at a customer location
- vehicles leave as soon as they have finished servicing a customer

The objective for the VRPHTW is lexicographical, the goal is to minimize in the following order: 1) the number of non-empty routes 2) the total traveled distance.
If $w_{max} > 0$, it can be of interest to delay the start of service to ensure that the maximal waiting time at the following customer is not exceeded. Of course delaying service at the current customer incurs a waiting time as well. When $w_{max} = \infty$ it makes sense, from a feasibility point of view, to start service as early as possible at each customer.

## 2.2 Vehicle Routing Problem with Soft Time Windows

In the Vehicle Routing Problem with Soft Time Windows, two types of time windows are associated with each customer $i$: hard time windows $[e_i; l_i]$ as in the VRPHTW and soft time windows $[e_i^*; l_i^*]$ such that $[e_i^*; l_i^*] \subseteq [e_i; l_i]$. The service start time of each customer $s_i$ must be contained in the hard time window. It may however fall outside the soft time window, at the cost of a penalty. Usually this penalty depends on how much the time window is violated (i.e. it increases with $max(0, e_i^* - s_i, s_i - l_i^*)$). The depot has a hard time window which can not be violated.
The VRPSTW has a lexicographic objective function. It aims at minimizing in the following order: 1) the number of non-empty routes 2) the number of violated soft time windows 3) the total traveled distance. Note that in the literature on the VRPSTW the second objective is also sometimes stated as minimizing the amount of penalties paid, which may not directly correspond to the minimization of the number of violated soft time windows.

## 2.3 Notations

For ease of notation we assume one dummy copy of the depot exists per vehicle. We thus have the set of customers defined as $Customers = \{1, \ldots, n\}$, the set of depots defined as $Depots = \{n+1, \ldots, n+K\}$ and the set of sites that can appear in vehicle routes as $Sites = Customers \cup Depots$.

The **successor** of a site $i$, denoted by $i^+$ corresponds to the site visited immediately after site $i$ in its route.

The **predecessor** of a site $i$, denoted by $i^+$ corresponds to the site visited immediately before site $i$ in its route.

A **route** corresponds to a sequence of vertexes $r = \langle r_0, \ldots, r_m \rangle$, with $r_0 \in Depots$ and $r_1, \ldots, r_m \in Customers$ and $r_1 \neq r_2 \neq \ldots r_{m-1} \neq r_m$. The set of customers visited in route $r$ is given by $custs(r)$. We use the following shortcut $\#r = \#custs(r)$ to denote the number of customers visited in route $r$. A route $r$ is called empty if $\#r = 0$. The successor of site $r_i$ in route $r$ is given by $r_{i+1}$, while its predecessor is given by $r_{i-1}$. The successor of the the last customer $r_m$ is defined as the depot $r_{m+1} = r_0$. The position of the site $i \in sites(r)$ in route $r$ is denoted by $rank(i,r)$. The subroute of length $p$ beginning with site $i$ is denoted by $r[i;p]$. Note that in order for $r[i;p]$ to be *consistent*, $i \in sites(r)$ and $rank(i,r) + p - 1 \leq |r|$ need to be satisfied.

A **schedule** associates with each site $i \in Sites$ a service start time $s_i$. The service start time $s_j$ at depot $j$ ($j \in Depots$) corresponds to the time a vehicle *arrives* at the depot after executing its route. A schedule for route $r$ associates a service start time with each $i \in sites(r)$.

# 3 Feasible service start times

In most VRPHTW variants considered in the literature, the waiting time is not restricted, thus it is not necessary to compute optimal service start times, as service can simply start as early as possible at each customer. This is different however for the VRPSTW we consider. First, we have a (possibly limited) waiting time. Situations can occur where this limit can only be respected at some customer $j$ by delaying service at some customer $i$. Second, delaying service, or starting service early, can possibly help to decrease the number of violated soft time windows. Thus we need a mechanism to determine the service start times for customers on a route, such that the resulting schedule helps us to guarantee feasibility w.r.t. the hard time windows and the limited waiting time, while minimizing the number of violated soft time windows.

In this section we explain how, given a route, we can compute, for each customer in the route, a restricted time interval. When the service start time of each customer is chosen from this interval (assuming it is non-empty), the route is guaranteed to be feasible w.r.t. the hard time window constraints and the limited waiting time. Using this interval we can then derive an algorithm allowing us to find a feasible schedule for the route. We also show how the same reasoning can be used for soft time windows.

The idea of defining a restricted interval in which service can start is not new, it has been used for example in (Azi et al., 2007). While most of the literature on the VRPSTW does not consider the problem of determining appropriate service start times, this has been done in (Figliozzi, 2010) and (Ibaraki et al., 2005). In (Figliozzi, 2010) the authors let service start as early as possible at customers. On each of the optimized routes they then apply a service start improvement procedure. For each route, they delay, starting with the last customer of the route, the service start time of all customers as much as necessary to fall into the soft time window, without violating the following customer's time window. In (Ibaraki et al., 2005), where the authors consider a Vehicle Routing Problem with general time windows, a dynamic programming approach is used to determine the service start times allwoing to minimize the total earliness and lateness penalties.

In the remainder of this paper we assume the following conditions on the time windows hold for all $i \in Customers$

- $e_i \geq e_0$
- $l_i \leq l_0$
- $e_i \geq e_0 + c_{0,i}$

These conditions can be easily enforced by adapting the customer's time windows. Obviously this does not restrict the set of feasible solutions, as a customer cannot be feasibly served before the start or after the end of the depot's time window. Furthermore, service at a customer can not start before the vehicle has had time to travel from the depot to the customer.

## 3.1 Time-related notions

We now introduce some necessary notions, needed to reason about the service start times.

**Departure times** With each site $i \in Sites$ is associated a time $\delta_i$ corresponding to the time the vehicle leaves site $i$. We have:

$$\delta_i \geq e_0 \qquad \forall i \in Depots \tag{1}$$
$$\delta_i = s_i + d_i \qquad \forall i \in Customers \tag{2}$$

Thus vehicles can depart from the depot only when the depot's time window has started. They depart from a customer after the service, which started at $s_i$ and took $d_i$ time units, has finished. Vehicles are not allowed to stay at a customer's location after completing service.

**Consistent service start times** In order to be consistent with the time needed to travel between sites the start of service times must respect the following constraints:

$$s_{i^+} = \delta_i + c_{i,i^+} \qquad \forall i \in Depots \tag{3}$$
$$s_{i^+} \geq \delta_i + c_{i,i^+} \qquad \forall i \in Customers \tag{4}$$

Constraint 3 states that service starts immediately upon arrival at the first customer in a route (i.e. no waiting is allowed at the first customer). Note that Constraint 1 allows to delay departure from the depot when convenient. Constraint 4 ensures that there is enough time for the vehicle to travel from one site to the next before service starts at this latter site. Service start times $\{s_i | i \in Sites\}$ are said consistent if they respect constraints (1)-(4).

**Feasible schedule** In order to have a feasible schedule service start time have to be consistent and respect the following constraints:

$$e_i \leq s_i \leq l_i \qquad \forall i \in Sites \tag{5}$$
$$s_i + d_i + c_{i,i^+} \geq s_{i^+} - w_{max} \qquad \forall i \in Customers \tag{6}$$

Constraint 5 states that the time windows of customers and depots must be respected. Constraint 6 enforces the maximum waiting time.

**Earliest and latest possible service start times** Given a route $r$, it is possible to define earliest and latest possible service start times for each site $i$ in the route. The earliest service start time $y_i$ is defined as follows:

(7)

$$y_i = \begin{cases} e_i & \forall i \in Depots \\ \max(e_i, y_{i^+} - c_{i,i^+} - d_i - w_{max}) & \forall i \in Customers \end{cases} \tag{8}$$

If the vehicle serves a site $i$ at a time $t < y_i$ this means that it will have to wait more than $w_{max}$ time units at one of the subsequent customers in the route.

The latest service start time $z_i$ is defined as follows:

$$z_i = \begin{cases} l_i & \forall i \in Depots \\ \min(l_i, z_{i^+} - c_{i,i^+} - d_i) & \forall i \in Customers \end{cases} \tag{9}$$

If the vehicle serves a site $i$ at a time $t > z_i$, this means that it will not be able to respect the end of the time window of some subsequent customer.

By definition, we have $e_i \leq y_i$ and $z_i \leq l_i$. Clearly if $y_i > z_i$ for some $i \in Sites$, no feasible schedule can exist for route $r$. If $y_i \leq z_i$, we have $e_i \leq y_i \leq z_i \leq l_i$. In the following we will show that if $y_i \leq z_i \forall i \in sites(r)$ a feasible schedule for route $r$ exists.

**Proposition 1.** *Given a route $r = \langle r_0, \ldots, r_m \rangle$ with $y_i \leq z_i$, $\forall i \in sites(r)$, serving a site $r_k$ $(1 \leq k \leq m)$ in the interval $[y_{r_k}, z_{r_k}]$ allows to serve the subsequent sites i of $r_k$ within their hard time-windows.*

Formally:

$$\forall k = 1, \ldots, m \forall s_{r_k} \in [y_{r_k}; z_{r_k}], \exists \{s_{r_{k'}} \in [y_{r_{k'}}; z_{r_{k'}}] | k' = k+1, \ldots, m+1\} \text{ such that}$$
$$\{s_{r_k}, s_{r_{k+1}}, \ldots, s_{r_{m+1}}\} \text{ respect constraints (4) and (6)}$$

The proof of proposition 1 can be found in the appendix (section A).

Based on the proof of proposition 1 we can state the following corollary:

**Corollary 1.** *Given a route $r = \langle r_0, \ldots, r_m \rangle$ the following method determines consistent service times $\{s_i | i \in cust(r)\}$ wrt the hard time-windows:*

*1) select a service start time $s_{r_1} \in [y_{r_1}; z_{r_1}]$*

*2) select service start times $s_{k+1}$ successively for $k = 1, \ldots, m-2$ in the interval*

$$s_{r_{k+1}} \in [\max(y_{k+1}, s_{r_k} + d_{r_k} + c_{r_k, r_{k+1}}); \min(z_{r_{k+1}}, s_{r_k} + d_{r_k} + c_{r_k, r_{k+1}} + w_{max})].$$

## 3.2 Earliest and latest preferred service start times

We can also define the latest preferred delivery time $z_i^*$ as the latest time a vehicle can serve customer $i$ in order to be able to serve the subsequent visits of $i$ within their soft time windows. We also define the earliest preferred delivery time $y_i^*$. The definitions of $z_i^*$ and $y_i^*$ are similar to the definitions (9) and (8). Notice that setting a service time $s_i \notin [y_i^*; z_i^*]$ for some customer $i$ will induce at least one unit of penalty for subsequent visits of $i$.
We will explain in section 7 how use corollary 1 and $y_i^*, z_i^*$ to derive a service start times for the customers on each route.

# 4 Very Large Scale Neighborhoods

In this section we introduce Very Large-Scale Neighborhoods and related concepts.

Very Large-Scale Neighborhood search is a Local Search technique used to solve **Combinatorial Optimization Problems**. Whereas in classical Local Search each neighbor is reached by performing a **Local Move** on a current solution, VLSN search uses sequences of such moves to compute a neighboring solution. The result is that VLSNs have an exponential size, contrary to traditional Local Search neighborhoods which are typically polynomial in size. The advantage of using exponential-sized neighborhoods is that they often allow to produce local optima of higher quality than polynomial-sized neighborhoods. VLSNs usually have a structure such that the best neighbor w.r.t the objective function can be computed in polynomial time. A VLSN is encoded in an **Improvement graph**. Identifying cycles (respecting a given condition) in the improvement graph corresponds to identifying a neighbor respecting the problem constraints.
In the following we give formal definitions for the concepts of Combinatorial Optimization Problem, Local Move, Very Large-scale Neighborhood and Improvement Graph.

**Combinatorial Optimization Problems** Let $\mathscr{X} = [X_1, X_2, \ldots, X_p]$ be a set of $p$ variables which take their values in their domain $D$. An assignment is a function $\sigma : \mathscr{X} \rightarrow D$ that assigns a value to each variable. We denote the set of all possible assignments as $\Lambda$. A constraint is a function $\mathscr{C} : \Lambda \rightarrow \mathbb{N}$ evaluating the infeasibility of the given assignment w.r.t a specific requirement. A solution w.r.t. the constraint $\mathscr{C}$ is an assignment $\sigma$ such that $\mathscr{C}(\sigma) = 0$. An objective is a function $f : \Lambda \rightarrow \mathbb{Z}$ giving the evaluation of the quality of a given assignment. Finally, we define a combinatorial optimization problem (COP) as the tuple $\mathscr{P} = \langle f, \mathscr{C}, \mathscr{X}, D \rangle$. Solving a COP requires finding a solution w.r.t constraint $\mathscr{C}$ minimizing $f$. Note that $\mathscr{C}$ can be a conjunction of different simpler constraints.

**Local Moves** A Local Search algorithm uses the concept of local moves to transit from a given assignment to a neighboring assignment. A move is thus a function $m : \Lambda \rightarrow \Lambda$ modifying a given assignment to produce an assignment. The set of possible moves is problem-dependent and is denoted $\mathscr{M}$. Local search typically considers small conjunctions of local moves, such as to respect structural constraints (i.e. removing and reinserting a customer in the VRP). Typically moves modifying the current assignment only slightly are considered. Computing the differentiation (i.e. the impact) of a given move on the constraint and objective function is a central operation in local search. In this paper, we use $\Delta_{\mathscr{C}}(m, \sigma)$ and $\Delta_f(m, \sigma)$ to denote the changes induced in the violation of constraint $\mathscr{C}$ and in the value of objective $f$ by performing move $m$ on assignment $\sigma$. The values are defined as follows: $\Delta_{\mathscr{C}}(m, \sigma) = \mathscr{C}(m(\sigma)) - \mathscr{C}(\sigma)$ and $\Delta_f(m, \sigma) = f(m(\sigma)) - f(\sigma)$.

**Very Large-Scale Neighborhood** A Very Large-Scale Neighborhood (VLSN) considers a constant set $\mathscr{M}$ of local moves and, at each iteration, selects and applies a sequence $M$ of these moves : $M = [m_1, m_2, \ldots, m_g]$ with $m_1 \neq m_2 \neq \ldots m_g$ and $\{m_1, m_2, \ldots, m_g\} \subseteq \mathscr{M}$. As there is an exponential number of such sequences, there are exponentially many neighbors. A sequence of moves $M$ must respect constraint $\mathscr{C}$. However we consider $\mathscr{C}$ is a conjunction of two constraints $\mathscr{C} = \mathscr{C}_{struct} + \mathscr{C}_{op}$ with $\mathscr{C}_{op}$ possibly empty. Constraint $\mathscr{C}_{struct}$ is typically a structural constraint, such as a partitioning or a permutation constraint. Constraint $\mathscr{C}_{op}$ corresponds to the conjunction of all other side-constraints of the problem. Local moves, when applied individually, typically violate constraint $\mathscr{C}_{struct}$. VLSN ensures that the selected sequence $M$ preserves the respect of $\mathscr{C}_{struct}$. Thus $\mathscr{C}_{struct}$ is respected before and after applying sequence $M$ but may be violated by intermediate assignments. On the other hand $\mathscr{C}_{op}$ may not be violated at any moment.
Thus, the VLSN algorithms consider the set of moves $M' = \{m \in \mathscr{M} | \mathscr{C}_{op}(m(\sigma)) = 0\}$, and selects a sequence $M$ from $M'$ whose impact on the current assignment $\sigma$ is such that (1) the structural constraint

$\mathscr{C}_{struct}$ is respected, and (2) the objective function $f$ is minimized. We denote $M(\sigma) = m_1 \circ \cdots \circ m_g(\sigma)$ the application of a sequence of local moves $M = [m_1, \ldots, m_g]$ ($M \subseteq \mathscr{M}$) on an assignment $\sigma$. We define the differentiation of a sequence of moves $M$: $\Delta_{\mathscr{C}}(M, \sigma) = \mathscr{C}(M(\sigma)) - \mathscr{C}(\sigma)$ and $\Delta_f(M, \sigma) = f(M(\sigma)) - f(\sigma)$. Note that only sequences $M$ of independent and compositional moves may be considered. Detailed explanations on this requirement and on how it can be enforced can be found in (Mouthuy et al., 2012).

**Improvement Graphs** An Improvement Graph is a problem-specific encoding of a VLSN given a current assignment $\sigma$. The semantics of the nodes in the Improvement Graph depend on the problem. Each edge $e$ in the graph is associated with a local move $m$. The weight of this edge corresponds to $\Delta_f(m, \sigma)$ (the impact on the objective $f$ of executing move $m$ on assignment $\sigma$). With each node in the improvement graph is associated a color, this, in such a way that any color-disjoint cycle represents a sequence of moves which, after application to the current assignment $\sigma$, results in an assignment satisfying the structural constraint $\mathscr{C}_{struct}$ .
An improvement graph thus is a directed, weighted, colored, labeled graph $IG(\sigma) = (V, E, w, \eta, \varphi)$, where

- $V$ is the set of nodes
- $E$ is the set of directed edges
- $w_{ij}$ is the weight of the edge $(i, j)$ corresponding to an impact on the objective $f$
- $\eta : E \to \mathscr{M}$ is a function that assigns a move to each edge
- $\varphi : V \to \mathbb{N}$ is a function that assigns a color to each node

The VLSN corresponding to graph $IG(\sigma)$ is then defined as

$$VLSN(IG(\sigma)) = \left\{ \eta(C)(\sigma) \middle| C \text{ is a color-disjoint cycle in } IG(\sigma) \right\}$$

where $\eta(C) = \{\eta((i, j)) | (i, j) \in C\}$. Thus the VLNS is a set of sets of moves whose corresponding edges in the Improvement Graph form a color-disjoint cycle. This neighborhood, was introduced in (Thompson and Orlin, 1989).

Given a constraint $\mathscr{C}_{struct}$, an improvement graph $G(\sigma)$ is *cycle-consistent* w.r.t. $\mathscr{C}_{struct}$ if the violations of $\mathscr{C}_{struct}$ are not affected by applying the moves of any color-disjoint cycle, i.e., $\forall C \in G(\sigma)$ : $\mathscr{C}_{struct}(\eta(C)(\sigma)) = \mathscr{C}_{struct}(\sigma)$. Note that, the moves appearing in a color-disjoint cycle are guaranteed to be independent and compositional (see (Mouthuy, 2011) for more information).

The identification of neighbors in the VLSN is thus done by identifying color-disjoint cycles in the Improvement Graph. An algorithm to compute color-disjoint cycles including a node given as input is presented in (Thompson and Orlin, 1989). It builds a shortest-path tree rooted at the input node, and ensures as good as possible, that all the nodes of any path in this tree are color-disjoint.

# 5 Modeling the VRPSTW as a COP

In this section we explain how we model the VRPSTW as Combinatorial Optimization Problem such as defined in section 4.

The VRPSTW is modeled as the problem $\mathscr{P}_{VRP} = \langle f, \mathscr{C}, \mathscr{X}, D \rangle$, where $f$ is the objective function, $\mathscr{C}$ are the constraints, $\mathscr{X}$ the variables and $D$ the variable domains.

**Variables and domains** There are $K$ variables, each representing a route. The domain $D$ of these variables is the set of all possible sequences on the elements 1 to $n+K$ (with $1 \ldots, n$ the customers and $n+1, \ldots, n+K$ the dummy depots). An assignment $\sigma$ assigns thus to each of the $K$ variables $\mathscr{X} = [S_1, \ldots, S_K]$ a sequence of sites. The $k$th route in assignment $\sigma$ is denoted by $\sigma(S_k)$. For the sake of clarity we abuse the notations and will use the term route to design the variables as well as their value. The values $y_i$, $z_i$, $z_i^*$ are derived from the routing and are not decision variables. We will show in section 7 that service start times $s_i$ can also be derived from the routing automatically.

**Constraints** In order for an assignment $\sigma$ to represent a feasible solution, structural constraints and operational constraints need to be enforced.
The structural constraint $\mathscr{C}_{struct} = \mathscr{C}_{VRP}$ requires that (a) the first visit of any vehicle $k$ is the dummy visit $n+k$ representing the depot, and (b) the sequences represent a partition of the visits 1 to $n+K$ . Thus for an assignment $\sigma$, $\mathscr{C}_{VRP}(\sigma) = 0$ if and only if $\sigma$ respects conditions (a) and (b). The operational constraints $\mathscr{C}_{op}$ consider time windows and vehicle capacities: $\mathscr{C}_{op} = \mathscr{C}_{capa} + \mathscr{C}_{Htw}$ where $\mathscr{C}_{Htw}$ are the hard time window constraints and $\mathscr{C}_{capa}$ the capacity constraints. The violations of these constraints are defined as: $\mathscr{C}_{Htw}(\sigma) = \sum_{i=1}^{n+K} \max(0, y_i - z_i)$ and $\mathscr{C}_{capa}(\sigma) = \sum_{k=1}^{K} \max(0, \sum_{i \in S_k} q_i - Q)$.

**Objectives** The objective function to be minimized is $f_{VRP}(\sigma) = \langle card(\sigma), \mathscr{C}_{Stw}(\sigma), dist(\sigma) \rangle$ where *card* is the function counting the number of vehicles used, $\mathscr{C}_{Stw}$ is the violation of the soft time-windows and *dist* is the function giving the total distance of a routing.
We define $card(\sigma) = \#\{S_k \in \mathscr{X} \mid \#\sigma(S_k) > 1\}|$ and $dist(\sigma) = \sum_{i=1}^{n+k} c_{i,i^+}$. The violation of the soft time-windows is computed as $\mathscr{C}_{Stw}(\sigma) = \sum_{i=1}^{n} \Big( \max(0, s_i - l_i^*, e_i^* - s_i) > 0 \Big)$. The minimizing the objective function corresponds thus to minimizing in the following order: 1) the number of non-empty routes 2) the number of violated soft time windows and the total traveled distance.

# 6 Very Large Scale Neighborhoods for the VRPSTW

In this section we present the VLSN used in our Multi-stage approach. Our VLSN is parametrized by $f$ and $L$, where $f$ represents the objective function and $L$ corresponds to the size of the neighborhood. Throughout our VND we use the same VLSN, parametrized with varying $f$ and $L$.

## 6.1 Local Moves

We consider three types of local moves: insertion, removal and replace. Let $r_1 \neq r_2$ be two routes, $i \in custs(r_1), j \in custs(r_2)$ and $r_1[i;m], r_2[j;n]$ be two consistent subroutes. The local moves are then defined as follows:

- $insert(r_1, i, m, r_2, j)$ inserts the subroute $r_1[i;m]$ in $r_2$, right after $j$ (route $r_1$ is not modified);
- $remove(r_2, j, n)$ removes the subroute $r_2[j;n]$ from $r_2$;

- *replace*$(r_1,i,m,r_2,j,n,p)$ inserts subroute $r_1[i;m]$ in $r_2$ at position $p$ and then removes $r_2[j;n]$ from $r_2$ (with $p \leq rank(j,r_2)$ or $p \geq rank(j,r_2)+n$).

In our VLSN we do not consider all *insert* and *replace* moves but only those minimizing the impact on the objective $f$. The *insert* move minimizing the impact on $f$ is denoted by $insert(r_1,i,m,r_2,f)$ and corresponds to $insert(r_1,i,m,r_2,j)$ where $j$ is the position minimizing $\Delta_f(insert(r_1,i,m,r_2,j),\sigma)$. The *replace* move minimizing the impact on $f$ is denoted by $replace(r_1,i,m,r_2,j,n,f)$ and corresponds to $replace(r_1,i,m,r_2,j,n,p)$ where $p$ is the position minimizing $\Delta_f(replace(r_1,i,m,r_2,j,n,p),\sigma)$.

## 6.2 Improvement Graph

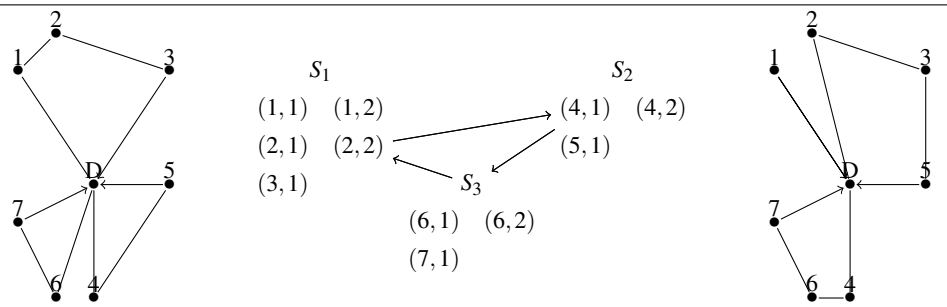
Based on the local moves we can define an improvement graph $IG(f,L)(\sigma)$ for the VRPSTW that is cycle-consistent with the structural constraint $\mathscr{C}_{VRP}$. The improvement graph, like the VLSN it encodes, is parametrized by an objective $f$ and a parameter $L$. Parameter $L$ corresponds to the length of the subroutes considered by the local moves.

**Definition 1.** *Given an assignment $\sigma$, the improvement graph $IG(f,L)(\sigma) = (V = V_1 \cup V_2, E, w, \eta, \varphi)$ is defined as follows:*

- $V_1 = \{(i,m) | 1 \leq i \leq n, 1 \leq m \leq L, i \in Customers, \sigma(S)[i;m]$ *is consistent with* $i \in \sigma(S), S \in \mathscr{X}\}$
- $V_2 = \mathscr{X} = \{S_1,\ldots,S_K\}$
- $E = \{(a,b) \in V \times V | a \in V_1 \vee b \in V_1 \wedge a,b$ *correspond to different routes*$\}$,
- $\eta(a,b) = \begin{cases} replace(S_k,i,m,S_{k'},j,n,f) & \text{if } a=(i,m), b=(j,n) \in V_1, i \in \sigma(S_k), j \in \sigma(S_{k'}), k \neq k' \\ insert(S_k,i,m,S_{k'},f) & \text{if } a=(i,m) \in V_1, b=S_{k'} \in V_2, i \in \sigma(S_k), k \neq k' \\ remove(S,i,m) & \text{if } a=S_k \in V_2, b=(i,m) \in V_1, i \in \sigma(S) \end{cases}$
- $w_{ab} = \begin{cases} \Delta_f(\eta_{ab},\sigma) & \text{if } \Delta_{\mathscr{C}_2}(\eta(a,b),\sigma) = 0 \\ +\infty & \text{otherwise} \end{cases}$
- $\varphi(a) = \begin{cases} k & \text{if } a \in V_1 \text{ with } a=(i,m) \text{ and } i \in \sigma(S_k) \\ k' & \text{if } a \in V_2 \text{ with } a = S_{k'} \end{cases}$

The improvement graph has two types of vertexes. Vertexes in set $V_1$ represent consistent subroutes (starting with customer $i$ and of length $m$). The vertexes in set $V_2$ simply correspond to routes. Edges must be incident to at least one $V_1$ vertex. Furthermore edges may not link vertexes concerning the same route. This means that if one of the nodes is a node representing a subroute of route $r$, then it may not be linked to a node representing route $r$ and neither to a node representing a subroute of $r$.
The end-nodes of the edges define the type of move associated with the edge. Edges where both end-nodes are in set $V_1$ represent a *replace* move, edges from a node in $V_1$ to a node in $V_2$ are associated with an *insert* move and edges from a node in $V_2$ to a node in $V_1$ represent a *remove* move.
The weight associated with each edge corresponds to the variation in the objective function $f$ incurred by applying the move corresponding to the edge. If the move results in a violation of the operational constraints, the variation is set to $\infty$ in order to forbid such moves.
Finally a color is associated with each node in the improvement graph. With each route will be associated a color. A node corresponding to a subroute of route $r$ takes the color corresponding to $r$, and a node corresponding to route $r$ takes the color corresponding to $r$ as well.

**Example 1** Example of Improvement Graph.




Let $n = 7$, $\sigma$ such that $\sigma(S_1) = [8,1,2,3], \sigma(S_2) = [9,4,5], \sigma(S_3) = [10,6,7]$. The routing described by $\sigma$ is illustrated on the left. The improvement graph $IG(dist,2)(\sigma) = \langle V,E,\eta,w,\varphi\rangle$ is illustrated in the middle, where *dist* is the function of the overall distance of a routing. For sake of clarity, only the set of nodes $V$ and three edges are represented. The edge $((2,2),(4,1))$ corresponds to the move $replace(S_1,2,2,S_2,4,1,dist)$. Similarly $\eta(((4,1),S_3)) = insert(S_2,4,1,S_3,dist)$ and $\eta((S_3,(2,2))) = remove(S_1,2,2)$. The three nodes of this cycle are color-disjoint and the routing obtained (illustrated on the right) respects the structural constraint $\mathscr{C}_{VRP}$. Notice that visit 4 is inserted right before visit 6 because this position minimizes *dist*.

Any color-disjoint cycle in this graph corresponds to moves whose application respects the structural vehicle routing constraint $\mathscr{C}_{VRP}$. As the coloring of the nodes does not depend on neither parameter $L$ nor on the objective function $f$ we can state the following:

**Proposition 2.** *The improvement graph $IG(f,L)(\sigma)$ is cycle-consistent wrt the Vehicle Routing structure for any objective function $f$ and any value of $L$.*

Since the improvement graph is parametrized by $f$ and $L$, it is possible to define a variety of neighborhoods of the form $VLSN(IG(f,L)(\sigma))$, by varying the objective function $f$ and the length of the subroutes cosnidered by the local moves $L$. This feature is critical to our multi-stage algorithm for the VRPSTW.

An example of an improvement graph for the VRPSTW with a color-disjoint cycle are given in Example 1. The nodes represent either consistent subroutes $r_k[i;m]$ of the current assignment or entire routes. The moves *insert*, *remove* and *replace* are represented by edges. Only the edges corresponding to the cycle are represented.

# 7 A Multi-Stage Very Large-Scale Neighborhood Search for the VRPSTW

In this section we present our multi-stage Very Large-Scale Neighborhood Search for the VRPSTW. The idea is that there is a specialized phase, aimed at reducing each of the single objectives (this has been done, for instance, in (Bent and Van Hentenryck, 2004; Homberger and Gehring, 2005) for the VRP with hard time windows). Our algorithm contains three phases: first we try to minimize the number of vehicles, then we try to minimize the number of violated soft time windows and finally we try to minimize the total distance. In each of the phases a VND is performed over a VLNS, and only improving moves (i.e. negative color-disjoint cycles) are executed.
We start our multi-stage approach from an initial solution where each customer is visited in a route of its own.

We first describe the Variable Neighborhood Descent over the VLSNs. Then we present each of the phases of our multi-stage approach. Finally we explain how the service start times are computed.

## 7.1 Variable Neighborhood Descent over VLSNs

Our Variable Neighborhood Descent (VND) employs a given VLSN, parametrized by $f$ the function to minimize and $L_{max}$ the maximum length of the subroutes considered in the neighborhood. The different neighborhoods considered in the VND correspond to the VLSN whose size is increased using parameter $L$. Thus the VND considers $L_{max}$ neighborhoods, instantiated from a same VLSN using values $L = 1, \ldots, L_{max}$. The Improvement Graphs encoding our VLSNs are searched for improving color-disjoint cycles using the algorithm from (Thompson and Orlin, 1989). The VND moves to the next neighborhood once no improving color-disjoint cycle can be found in the current improvement graph. Throughout the VND, we locally optimize each of the routes in the current assignment before searching for the next move in the current VLSN. This is done by selecting the best 2opt move. The procedure stops once no further improving 2opt move can be found.
The high-level algorithm of a VND over a sequence of increasing VLNSs is given in Algorithm 1. It takes as input: $\sigma$ the current assignment, $L_{max}$, the maximum neighborhood size, $f$ the objective function to minimize, $\Xi$ the type of search (first-improvement or best-improvement) to perform over the VLSNs and a stopping criterion.

## 7.2 The different stages of our approach

**Stage 1: Minimizing the number of vehicles**

To reduce the number of vehicles, the algorithm selects a route and tries to move its customers to another route, one at a time. This process is repeated until no further route can be emptied.
First, a non-empty route $r_k$ is selected. To perform a first optimization step, we then try to minimize the number of vehicles and total distance (in that order) by applying the first negative cycle found over the corresponding VLSN ($\hat{l}$ is an adaptive parameter). Note that this cycle may not add any further customers to route $r_k$. Then the customer $i$ with the tightest hard time windows is chosen from $r_k$. The other customers in route $r_k$ are frozen, this means that the only modification allowed on route $r_k$ is the removal of customer $i$. Moves that result in the insertion of new customers in $r_k$ are forbidden by this as well. Then a best-improvement search is performed over the VLSN. The objective function for this VLSN is adapted to the objective of reducing the number of vehicles. This search stops as soon as visit $i$ has been moved to another vehicle or after a time limit has been reached. If the search does not allow to move visit $i$ to

**Algorithm 1:** `VND(`$\sigma, L_{max}, f, \Xi$`, stopping criterion)`, a Variable Neighborhood Descent over a sequence of $L_{max}$ VLSNs minimizing $f$

```
   Input: σ, L_max, f, Ξ, stopping criterion
 1 ℓ = 1;
 2 repeat
 3 |   perform local optimization on each route, updating σ;
 4 |   perform Ξ-search over VLSN(IG(f,ℓ)(σ));
 5 |   if color-disjoint cycle found then
 6 |   |   execute move corresponding to cycle, updating σ;
 7 |   |   ℓ = 1;
 8 |   else
 9 |   |   if ℓ = L_max then
10 |   |   |   break;
11 |   |   else
12 |   |   |   ℓ = ℓ + 1;
13 |   |   end
14 |   end
15 until stopping criterion;
16 return σ;
```

another route, the search selects a different route to empty. The first stage stops as soon as a time limit is reached.

The objective function used in this stage is a lexicographic objective function with three elements as in (Bent and Van Hentenryck, 2004). Assume we selected customer $i$ to be removed from route $r_{k'}$. Then we define the **minimal delay** of visit $i \in custs(r_{k'})$ and route $r_k$ ($r_k \neq r_{k'}$) as the minimal violation of the hard time-windows constraint incurred by the insertion of visit $i$ into route $r_k$.

$$mdl_{k,i}(\sigma) = \min_{j \in custs(r_k)} \Delta_{\mathscr{C}_{Htw}}(insert(r_{k'}, i, 1, r_k, j), \sigma).$$

The total minimum delay w.r.t. visit $i$ is then given by $mdl_i(\sigma) = \sum_{k=1}^{K} h(mdl_{k,i})$ where $h$ is a function favoring small values[1]. Thus minimizing $mdl_i$ favors insertion of $i$ in routes where the resulting hard time window is minimal. To favor solutions that actually do remove customer $i$ from its current route $r_{k'}$ we add a binary component $rm_i(\sigma)$ that takes value 1 if $i$ is visited in route $r_{k'}$ in assignment $\sigma$, and value 0 in the other case.
The resulting objective function for the minimization of the number of vehicles becomes $\langle card, rm_i, mdl_i \rangle$. The pseudo-code for this first stage is given in Algorithm 2. The stopping criterion `stopCrit1` is reached as soon as visit $i$ has been removed from its route or the overall time limit is reached.

**Stage 2: Minimizing the soft time windows violation**

To reduce the soft time windows violations, the algorithms executes the following VND:

`VND(`$\sigma, L_{max}, \langle card, \mathscr{C}_{Stw} \rangle$`, first-improvement,stopCrit2)`

The stopping criterion `stopCrit2` corresponds to a limit on the number of iterations.

[1] Our experiments use $h(x) = \frac{C}{x} - \frac{x^2}{C'}$, where $C, C'$ are two constants.

**Algorithm 2:** Stage 1: Minimizing the number of routes

**Input**: $\sigma$, $L_{max}$

**repeat**
  **select** *a non-empty route k in* $\sigma$ **do**
    perform `first-improvement search` over $VLSN(IG(\langle card, dist\rangle, \hat{\ell})(\sigma))$;
    **while** *the route k is not empty (*$\#S_k > 0$*)* **do**
      **select** *visit* $i \in S_k$ *with the smallest hard time windows* **do**
        $\sigma = freeze(S_k, i, \sigma)$;
        $\sigma$ = `VND(`$\sigma, L_{max}, \langle card, rm_i, mdl_i\rangle$`,best-improvement, stopCrit1)`;
        $\sigma = unFreeze(S_k)$;
      **end**
      **if** *the attempt of removing visit i from* $S_k$ *failed (*$i \in S_k$*)* **then** break;
    **end**
  **end**
**until** *until time limit reached*;

**Stage 3: Minimizing the total distance**

The overall distance is minimized using:

$$\texttt{VND(}\sigma, L_{max}, \langle card, \mathscr{C}_{Stw} + dist\rangle\texttt{,best-improvement,stopCrit3)}$$

Experimental results indicated that it was best to relax the soft time windows and penalize them in the objective function, instead of enforcing their best value found in the second step.

### 7.3 Scheduling start of service

To define the violations of the soft time windows, we must define the service start time $s_i$ of customers. Customers are served as early as possible while trying to minimize the violations of the soft time windows constraint. Corollary 1 specifies that the soonest a customer $i$ can be served is at time $\max(y_i, \hat{s_{i^-}} + d_{i^-} + c_{i^-,i})$, with

$$\hat{s_i} = \begin{cases} e_0 & \text{if } i \in Depots \\ s_i & \text{if } i \in Customers \end{cases}$$

Preferably, customer $i$ should be served at time $y_i^*$. However, if $z_i^* \leq y_i^*$, at least one subsequent customer of $i$ would be served outside his preferred time-window (it is impossible to achieve a zero-violation for the soft time windows constraint). So, it is desirable to serve $i$ at time $z_i^*$; the preferred time window of customer $i$ will be violated (as $z_i^*$ will lie outside the soft time window), but maybe the preferred time windows of all subsequent customers of $i$ will be satisfied. So the preferred service start time would be $\min(y_i^*, z_i^*)$. This preferred service time and the hard time windows lead to the following definition of service start times

$$s_i = \max(y_i, \hat{s}_{i^-} + d_{i^-} + c_{i^-,i}, \min(y_i^*, z_i^*)),\ \forall i \in Customers$$

## 8 Experimental Results

In this section we compare the performance of our multi-stage VLSN search to the state-of-the-art. First we explain the problem Types we consider and the benchmark instances we use. Then, we compare our results to the best known solutions from the literature. Finally a short summary is provided.

## 8.1 Problem Types and Benchmark instances

In (Fu et al., 2007) a classification of VRPSTWs into 6 problem Types has been proposed. We consider all Types in our experimental section. In each of the Types, each customer is associated with a hard time window $[e_i;l_i]$ and a soft time window $[e_i^*;l_i^*]$ giving the preferred earliest and latest service time.

**Type 1** has been investigated in (Taillard et al., 1997; Fu et al., 2007; Figliozzi, 2010). A vehicle can arrive early at a customer $i$ but cannot serve him before his earliest service time ($e_i = e_i^*$). However, the vehicle can serve the customer late ($l_i^* \leq l_i$), with a penalty linear to the delay. There is no hard latest service start time for the customers ($l_i = l_0$) but, for the depot, the time-window $[e_0, l_0]$ is hard.

**Type 2** has been considered in (Koskosidis et al., 1992; Fu et al., 2007). A vehicle can serve a customer before his preferred earliest service time ($e_i^* \geq e_i$) and after his preferred latest service time ($l_i^* \leq l_i$). Serving a customer outside his preferred time window incurs a linear penalty cost. There is no hard earliest or latest service time for the customers ($e_i = e_0, l_i = l_0$) but, for the depot, the time-window $[e_0, l_0]$ is hard.

**Type3** has been investigated in (Balakrishnan, 1993; Chiang and Russell, 2004; Fu et al., 2007) and (Figliozzi, 2010). Each customer has a soft and a hard time window. The hard time window requires for each customer $i$, that it be served within a certain percentage $p_{\max}$ of the total route duration $D = l_0 - e_0$: $e_i = e_i^* - D\frac{p_{\max}}{100}$ and $l_i = l_i^* + D\frac{p_{\max}}{100}$. Moreover, a vehicle is allowed to arrive earlier than $e_i$ at a customer $i$ but cannot wait more than $w_{max}$ before serving the customer. The parameter $w_{max}$ is also expressed as a percentage of the total route duration $D$: $w_{max} = D\frac{w_{max}}{100}$. Typical values of $p_{\max}$ and $w_{max}$ are $0, 5$ or $10$.

**Type 4** has been considered in (Qureshi et al., 2009). It allows a vehicle to arrive early at a customer $i$. However the customer cannot be served before his earliest service time ($e_i = e_i^*$). The vehicle can serve the customer late, but within a certain percentage $p_{\max}$ of the total route duration $D = l_0 - e_0$: $l_i = l_i^* + D\frac{p_{\max}}{100}$.

**Type 5** has been considered in (Fagerholt, 2001). It allows vehicles to serve a customer early and late, but only up to a certain percentage of its soft time window. The hard time window requires that each customer $i$has to be served within a certain percentage $p_{\max}$ of the total route duration $D = l_0 - e_0$: $e_i = e_i^* - D\frac{p_{\max}}{100}$ and $l_i = l_i^* + D\frac{p_{\max}}{100}$.

**Type 6** corresponds to Type 3, where the limitation on the maximum allowable waiting time $w_{max}$ is dropped.

To perform our experiments we adapted the Solomon benchmark instances ((Solomon, 1987)), as described in (Fu et al., 2007). The Solomon instances are 100 customer instances separated into different problem sets: R1, C1, RC1, R2, C2 and RC2. The problems in classes R1 and R2 have their customers uniformly distributed at random, while they are clustered in C1 and C2 and semi-clustered in RC1 and RC2. Then problems in R1, C1 and RC1 have a short scheduling horizon (the depot's time window is short) and small vehicle capacities, typically resulting in a high number of short routes. On the other hand problems in R2, C2 and RC2 have a long scheduling horizon and high vehicle capacities, typically resulting in a low number of long routes.

## 8.2 Experimental Setup

The multi-stage VLSN search was implemented in Comet (Van Hentenryck and Michel, 2005), based on the Constraint-Based Very Large-Scale Neighborhood Search framework presented in (Mouthuy et al., 2012). All experiments were run on an Intel Q6600 Quadcore 2,4GHz CPU.

Parameter $L_{max}$ was set to 2 as our preliminary experiments revealed this value achieved the best results. The time limit for the vehicle reduction stage corresponds to 250 CPU seconds. For the second stage we used an iteration limit of 200 iterations. Finally the third stage is stopped after 1000 iterations. The lexicographic objective functions of each stage are treated as weighted sums. In the first stage we use (in order of the priority of the individual objectives) the weight coefficients $10^7$ and 1 for the first optimization step done for each route, and then $10^7$, $10^6$ and 1. In the second stage we use coefficients $10^7$ and 1. In the third stage we use for the first individual objective $10^7$ and in the second individual objective (the sum of $\mathscr{C}_{stw}$ and $dist$) we use $10^3$ and 1. Finally constants $C$ and $C'$ used in the computation of the total minimum delay in stage 1 are set to $C = 10^5$ and $C' = 10^6$.

## 8.3 Comparison with state-of-the-art

In the following we compare the best solution we obtained over 10 runs (same methodology as in (Fu et al., 2007)) for problem Types 1, 2 and 3 with the best known solutions from the literature. A solution $sol_A$ is considered better than $sol_B$ if it has a lower number of vehicles, or same number of vehicles and a higher percentage of non-violated soft time windows or a tie on the first two and a lower total distance.
We also provide results for problems of Types 4-6 (see Appendix for these additional results). To the best of our knowledge, this is the first time results on adapted Solomon benchmark instances for problems Types 5 and 6 are published. Results for Type 4 problems are presented in (Qureshi et al., 2009), but the computation of the soft time windows is done differently than we did here. Also a Type 5 problem is considered as ship scheduling problem in (Fagerholt, 2001), the experiments are performed using real-life data.
In the comparative tables we indicate the following information for the best known result as well as for our results: $K/Dist$. the number of vehicles and total distance (rounded in our case) and *Non-viol. TW* the percentage of customers whose soft time windows are not violated. For our results we furthermore indicate the total execution time (rounded), averaged over the 10 runs. Finally we denote with '*' instances in which we achieve a tie, and with '**' instances in which we improve the best known solution.

### Type 1 instances

We compare our results to those published in (Taillard et al., 1997; Fu et al., 2007; Figliozzi, 2010). Note that throughout the literature, different objectives are considered for this problem. The objective considered in (Taillard et al., 1997) is the minimization of the total distance and the summed lateness penalties. In (Fu et al., 2007) the objective is to minimize first the number of routes, then the total deviation of time window to start service and then the total distance. Finally (Figliozzi, 2010) minimizes first the number of routes, then the number of violated time windows and finally the total distance.
The comparative results on Type 1 instances is given in Table 1, we use the shorthands IRCI, UTS and TSH to refer to the algorithms presented in (Figliozzi, 2010), (Fu et al., 2007) and (Taillard et al., 1997). For the instances with a short planning horizon (R1,C1 and RC1), our multi-stage approach is able to improve the number of vehicles in 9 out of 29 instances, and ties with the best known value in the remaining instances. Furthermore in 7 of the instances where we improve the number of vehicles we also improve the percentage of non-violated time windows. The same is achieved in 7 instances where we tie with the best known number of vehicles. The improvement in number of vehicles or percentage of violated time windows usually comes at the cost of a higher total distance. An exception is instance R103 where our method is able to improve over the best known solution in the 3 objectives.
For the instances with a long planning horizon (R2,C2 and RC2), our multi-stage approach improves over the state-of-the-art in 3 instances, each time by tying the number of vehicles and improving upon the percentage of non-violated time windows. Three observations can be made. First, it is merely on

4 out of the 56 instances that our multi-stage approach is not able to find the best known number of vehicles. Next, our method clearly performs better on the random instances (R instances, customers are distributed randomly) than on the clustered instances (C instances, customers are distributed in clusters). Here, our multi-stage approach appears to get trapped in local optima. In these instances, once routes have been assigned to clusters, it becomes difficult to find improving moves (i.e. negative cycles) in the VLSNs. Our multi-stage approach only performs improving moves, which works at our disadvantage on these instances. In contrast, the algorithms presented in (Taillard et al., 1997) and (Fu et al., 2007), allowing the search also to move solutions worse than the current one, perform particularly well on this type of instances. A third observation is that our multi-stage approach performs better on instances with a short planning horizon and low capacities, that is instances with many short routes. This might possibly explained by the low $L_{max}$ value we use. Moves on short subroutes have a higher impact in shorter routes than in very long routes.
When we compare our computation times for R1 and RC1 to those indicated in (Figliozzi, 2010) and (Fu et al., 2007), we reach comparable values (in terms of processing power, our Q6600 CPU should be more efficient than the 600MHz and 1.16GHz CPUs used in these works). Our computation times for C1 are higher than those from (Fu et al., 2007) . Finally on problem sets R2, C2, and RC2 our computation times are substantially higher. As the scheduling horizon is long for these instances, the number of local moves respecting the hard time windows is higher than for the instances with a short scheduling horizon, where only few feasible moves inserting new customers in a route exist. The execution time will increase with the number of moves respecting the hard time windows (corresponding to the number of edges in our improvement graphs), especially since in the first and last stage of our approach a best-improvement search is employed.

**Type 2 instances**

For the Type 2 instances, the state-of-the-art is given by (Fu et al., 2007; Koskosidis et al., 1992). The authors in (Koskosidis et al., 1992) want to minimize the total distance and the total penalties to be paid for early or late arrivals. In (Fu et al., 2007) the objective is to minimize first the number of routes, then the total deviation of time window to start service and then the total distance. The comparative results on Type 2 instances are given in Table 2, we use the shorthands OBH, and UTS to refer to the algorithms presented in (Koskosidis et al., 1992) and (Fu et al., 2007).
Our multi-stage approach is able to improve upon the best known results on all instances, except the clustered instances (C) where our method is outperformed by the state-of-the-art. While our method reaches the lowest number of vehicles, it is unable to respect all of the time windows, as in the best known solutions. Furthermore our total traveled distance is also consistently higher than that of the best known solution in that problem set. The improvements over the state-of-the-art are all in terms of number of vehicles, which we reduced by up to 5 vehicles. This reduction comes at the cost of a higher percentage of violated time windows. Our algorithm also achieves a lower total distance on all but one of the R and RC instances. A look at the total execution time reveals that this execution time is significantly higher for the R and RC instances than for the C instances. On Type 2 instances, our computation times are consistently higher than those reported in (Fu et al., 2007). Again, this can be explained by the number of edges in the improvements graphs. As in the Type 2 instances none of the customers have hard time windows, most local moves will respect the hard time window constraints (which are thus only applicable to the depot). This results in bigger improvement graphs, which will of course impact the time to search for negative color-disjoint cycles.

Table 1: Results on Type 1 instances

| Instance | Best known | | | This paper | | | |
|---|---|---|---|---|---|---|---|
| | *K/Dist.* | *Non-viol. TW (%)* | *Ref.* | *K/Dist.* | *Non-viol. TW (%)* | $sec_{tt}$ *(sec)* | |
| R101 | 12/1128.7 | 44 | IRCI | 11/1214 | 61 | 499 | ** |
| R102 | 11/1058.7 | 54 | IRCI | 10/1172 | 68 | 523 | ** |
| R103 | 10/1027.4 | 66 | IRCI | 9/1008 | 76 | 599 | ** |
| R104 | 9/983.5 | 99 | UTS | 9/977 | 93 | 624 | |
| R105 | 11/1073.5 | 58 | IRCI | 10/1186 | 69 | 573 | ** |
| R106 | 10/1047.4 | 67 | IRCI | 10/1091 | 81 | 640 | ** |
| R107 | 10/1126.7 | 100 | TSH | 9/1013 | 83 | 555 | ** |
| R108 | 9/968.6 | 100 | TSH | 9/1004 | 96 | 696 | |
| R109 | 10/1001.4 | 72 | IRCI | 10/1132 | 88 | 629 | ** |
| R110 | 9/1013.4 | 71 | IRCI | 9/1010 | 82 | 527 | ** |
| R111 | 10/1104.8 | 100 | TSH | 9/1016 | 82 | 635 | ** |
| R112 | 9/940.9 | 83 | IRCI | 9/1000 | 95 | 644 | ** |
| C101 | 10/828.94 | 100 | TSH, UTS | 10/970 | 96 | 879 | |
| C102 | 10/828.94 | 100 | TSH, UTS | 10/921 | 98 | 1085 | |
| C103 | 10/828.06 | 100 | TSH | 10/955 | 100 | 1251 | |
| C104 | 10/828.94 | 100 | TSH, UTS | 10/942 | 100 | 1522 | |
| C105 | 10/824.78 | 100 | TSH, UTS | 10/865 | 99 | 933 | |
| C106 | 10/828.94 | 100 | TSH, UTS | 10/866 | 98 | 1111 | |
| C107 | 10/828.94 | 100 | TSH, UTS | 10/832 | 99 | 960 | |
| C108 | 10/828.94 | 100 | TSH, UTS | 10/870 | 98 | 1172 | |
| C109 | 10/828.94 | 100 | TSH, UTS | 10/863 | 100 | 1356 | |
| RC101 | 11/1255.3 | 56 | IRCI | 11/1373 | 78 | 496 | ** |
| RC102 | 10/1030.1 | 68 | IRCI | 10/1261 | 80 | 473 | ** |
| RC103 | 10/1154.6 | 75 | IRCI | 10/1214 | 93 | 567 | ** |
| RC104 | 10/1135.8 | 100 | TSH | 9/1129 | 88 | 638 | ** |
| RC105 | 11/1219.7 | 62 | IRCI | 10/1333 | 69 | 501 | ** |
| RC106 | 10/11150.3 | 73 | IRCI | 10/1300 | 87 | 493 | ** |
| RC107 | 10/1123 | 72 | IRCI | 10/1259 | 92 | 609 | ** |
| RC108 | 10/1139.8 | 100 | TSH | 9/1126 | 86 | 620 | ** |
| R201 | 3/1500.4 | 89 | UTS | 3/1265 | 82 | 6775 | |
| R202 | 3/1205.8 | 100 | UTS | 3/1100 | 89 | 8895 | |
| R203 | 2/901.8 | 70 | IRCI | 2/938 | 82 | 8609 | ** |
| R204 | 2/854.3 | 100 | UTS | 2/876 | 94 | 7392 | |
| R205 | 3/1001.8 | 100 | UTS | 3/1127 | 94 | 9466 | |
| R206 | 2/956.9 | 75 | IRCI | 2/959 | 85 | 6671 | ** |
| R207 | 2/903 | 100 | UTS | 2/878 | 94 | 5790 | |
| R208 | 2/738.3 | 100 | UTS | 2/824 | 99 | 8278 | |
| R209 | 2/950.5 | 74 | IRCI | 3/1010 | 96 | 11260 | |
| R210 | 2/963.8 | 86 | IRCI | 3/1041 | 96 | 12401 | |
| R211 | 2/953.2 | 100 | UTS | 2/902 | 94 | 5082 | |
| C201 | 3/591.56 | 100 | TSH, UTS | 3/591 | 100 | 2969 | |
| C202 | 3/591.56 | 100 | TSH, UTS | 3/637 | 98 | 2949 | |
| C203 | 3/591.17 | 100 | TSH, UTS | 3/761 | 100 | 3738 | |
| C204 | 3/590.6 | 100 | TSH, UTS | 3/709 | 100 | 5156 | |
| C205 | 3/588.88 | 100 | TSH, UTS | 3/594 | 100 | 3429 | |
| C206 | 3/588.49 | 100 | TSH, UTS | 3/593 | 100 | 3753 | |
| C207 | 3/588.29 | 100 | TSH, UTS | 3/647 | 100 | 4175 | |
| C208 | 3/588.32 | 100 | TSH, UTS | 3/631 | 100 | 5180 | |
| RC201 | 3/1147.4 | 52 | IRCI | 4/1276 | 81 | 5590 | |
| RC202 | 3/1435.6 | 100 | UTS | 3/1264 | 87 | 5535 | |
| RC203 | 3/1062.4 | 100 | UTS | 3/1041 | 92 | 7464 | |
| RC204 | 2/850.7 | 86 | IRCI | 2/797 | 91 | 3538 | ** |
| RC205 | 3/1656.8 | 93 | UTS | 4/1165 | 87 | 6352 | |
| RC206 | 3/1158.8 | 100 | TSH | 3/1201 | 94 | 5845 | |
| RC207 | 3/1082.3 | 100 | TSH | 3/1115 | 91 | 6672 | |
| RC208 | 2/885.5 | 79 | IRCI | 3/993 | 99 | 8369 | |

Table 2: Results on Type 2 instances.

| Instance | Best known | | | This paper | | | |
|---|---|---|---|---|---|---|---|
| | *K/Dist.* | *Non-viol. TW (%)* | *Ref.* | *K/Dist.* | *Non-viol. TW (%)* | $sec_{tt}$ *(sec)* | |
| R101 | 14/1872.94 | 56 | UTS | 9/1002 | 31 | 9070 | ** |
| R102 | 13/1732.54 | 71 | UTS | 9/1018 | 52 | 9049 | ** |
| R103 | 12/1542.79 | 91 | UTS | 9/961 | 74 | 9027 | ** |
| R104 | 10/1107.18 | 100 | UTS | 9/1003 | 93 | 9008 | ** |
| R108 | 10/968.32 | 100 | UTS | 9/970 | 95 | 9006 | ** |
| R109 | 11/1379.87 | 96 | UTS | 9/1041 | 68 | 9033 | ** |
| C101 | 10/828.94 | 100 | UTS | 10/931 | 95 | 1587 | |
| C102 | 10/828.94 | 100 | UTS | 10/854 | 97 | 1634 | |
| C103 | 10/829 | 100 | OBH | 10/930 | 99 | 1622 | |
| C104 | 10/829 | 100 | OBH | 10/904 | 98 | 1640 | |
| C105 | 10/828.94 | 100 | UTS | 10/953 | 96 | 1642 | |
| C106 | 10/828.94 | 100 | UTS | 10/894 | 95 | 1588 | |
| C107 | 10/828.94 | 100 | UTS | 10/886 | 96 | 1489 | |
| C108 | 10/828.94 | 100 | UTS | 10/899 | 100 | 1547 | |
| C109 | 10/828.94 | 100 | UTS | 10/931 | 99 | 1420 | |
| RC101 | 13/1851.22 | 74 | UTS | 9/1145 | 36 | 9065 | ** |
| RC102 | 13/1772.42 | 99 | UTS | 9/1109 | 54 | 9047 | ** |
| RC103 | 11/1416.81 | 100 | UTS | 10/1196 | 85 | 10016 | ** |
| RC104 | 10/1262.55 | 100 | UTS | 9/1080 | 88 | 9013 | ** |
| RC106 | 12/1531.57 | 99 | UTS | 9/1121 | 57 | 9044 | ** |
| RC108 | 11/1224.72 | 100 | UTS | 9/1103 | 82 | 9019 | ** |

**Type 3 instances**

In Type 3 instances a limit is given on the maximum waiting time ($w_{max}$) and on the maximum violation of the hard time windows ($p_{max}$). Both are expressed as percentages of the total allowed route duration. To be consistent with the literature we use values 5% and 10% for $p_{max}$, whereas $w_{max} = 10\%$.
We compare our multi-stage approach to the algorithms presented in(Balakrishnan, 1993; Chiang and Russell, 2004; Fu et al., 2007; Figliozzi, 2010). In (Chiang and Russell, 2004) the authors want to minimize (in order of priority) the number of routes, the total distance and duration, and then the penalties incurred by time window violations. In (Fu et al., 2007) the objective is to minimize first the number of routes, then the total deviation of time window to start service and then the total distance. Finally (Figliozzi, 2010) minimizes first the number of routes, then the number of violated time windows and finally the total distance.
The comparative results on Type 3 instances are given in Table 3, we use the shorthands SH, MH, UTS and IRCI to refer to the algorithms presented in (Balakrishnan, 1993), (Chiang and Russell, 2004), (Fu et al., 2007) and (Figliozzi, 2010).
Our approach is able to improve over the state-of-the-art in all instances. The improvement stems from a lower number of vehicles in 5 out of 16 instances and a higher percentage of non-violated time windows (and tie in number of vehicles) in 11 out of 16 instances. Finally notice the average total execution time is similar to the one for the Type 1 instances. In (Fu et al., 2007) and (Figliozzi, 2010) only ranges of computation times for all Type 3 instances (all values for $p_{max}$ and $w_{max}$) combined are indicated. However, if we compare those indicated ranges to our computation times on the considered instances we interestingly never even reach half of the highest computation time reported in (Fu et al., 2007). Our times lie in the range indicated in (Figliozzi, 2010), although we improve their best solution on all instances.

Type 3 C instances haven't been considered in the literature so far, we provide our results for these instances in the appendix in Table 4.

## 8.4 Summary

In summary, the results indicate that our multi-stage algorithm provides significant improvements over the state-of-the-art, both in vehicle reduction and in minimizing the violations of the soft-constraints. Among the three first Types of problems over the R1,C1 and RC2 instances, it improved 74% of the best known solutions, of which 39% in terms of number of vehicles used, which is the most significant criteria for practical applications. On instances where customers are uniformly distributed, the multi-stage VLSN algorithms is dominated by the tabu-search algorithms from (Fu et al., 2007) and (Taillard et al., 1997).

All the remaining tables (Tables 4 through 10 in the appendix) present the results obtained by our algorithm for instances with no (comparable) experimental results found in the literature. We report them to allow comparisons in future work since these instances are also of practical interest.

# 9 Conclusions and Perspectives

This paper considers the Vehicle Routing Problem with Soft Time Windows (VRPSTW), a challenging routing problem due to its combination of hard time windows and a lexicographic objective function minimizing the number of vehicles, the violations of soft time windows, and the total travel distance. We present a multi-stage Very Large-Scale Neighborhood (VLSN) search algorithm for the VRPSTW. Each of the three stages performs a Variable Neighborhood Descent over a parametrizable VLSN. Each stage features a (lexicographic) objective function adapted to the goal of the specific stage: Vehicle reduction, minimization of the soft time window constraint violations, and minimization of the travel distance. VRPSTWs are classified in 6 types in the literature. We present results for each of these types, and are the

Table 3: Results on Type 3 instances with $w_{max} = 10\%$.

| | Instance | Best known | | | This paper | | | |
|---|---|---|---|---|---|---|---|---|
| | | *K/Dist.* | *Non-viol. TW (%)* | *Ref.* | *K/Dist.* | *Non-viol. TW (%)* | $sec_{tt}$ *(sec)* | |
| $p_{max} = 5\%$ | R101 | 14/1633 | 68 | IRCI | 13/1582 | 42 | 526 | ** |
| | R102 | 12/1404 | 63 | IRCI | 12/1364 | 80 | 610 | ** |
| | R103 | 11/1374 | 93 | IRCI | 10/1184 | 81 | 529 | ** |
| | R109 | 11/1393 | 93 | IRCI | 10/1211 | 78 | 521 | ** |
| | RC101 | 13/1778 | 93 | IRCI | 12/1586 | 67 | 425 | ** |
| | RC102 | 12/1635 | 98 | IRCI | 11/1556 | 81 | 451 | ** |
| | RC103 | 10/1256 | 83 | IRCI | 10/1236 | 91 | 511 | ** |
| | RC106 | 11/1336 | 81 | UTS | 11/1451 | 93 | 424 | ** |
| $p_{max} = 10\%$ | R101 | 12/1376 | 31 | UTS | 12/1349 | 47 | 495 | ** |
| | R102 | 10/1173 | 33 | MH | 10/1259 | 38 | 554 | ** |
| | R103 | 10/1185 | 76 | UTS | 10/1170 | 84 | 559 | ** |
| | R109 | 10/1116 | 53 | IRCI | 10/1175 | 80 | 522 | ** |
| | RC101 | 11/1322 | 43 | IRCI | 11/1512 | 64 | 466 | ** |
| | RC102 | 11/1367 | 74 | UTS | 11/1443 | 87 | 489 | ** |
| | RC103 | 10/1194 | 79 | IRCI | 10/1221 | 91 | 464 | ** |
| | RC106 | 10/1160 | 49 | MH | 10/1331 | 70 | 540 | ** |

first to present results on Type 6. Our approach is tested on standard benchmark instances and compared to the state-of-the-art approaches for each of the considered types. Experimental results indicate that our multi-stage VLSN search improves best-known solutions on 53% of the Type 1, 2 and 3 instances. For Type 3 it is even able to improve over 100% of the considered instances. Many of these improvements stem from a reduced number of routes, typically a very critical objective in Vehicle Routing Problems.

The experimental results also clearly show that the proposed algorithm performs better on problem instances where customers are randomly distributed and where the routes are shorter. Future work will focus on extending the current approach in order to achieve as good performance on the clustered instances as we already have for the random ones.

**Acknowledgements** The first author has been supported by the Belgian FNRS ('Aspirant'). This research was partially supported by the Interuniversity Attraction Poles Programme (Belgian State, Belgian Science Policy), the FRFC project 2.4504.10 of the Belgian FNRS (National Fund for Scientific Research). This work was also conducted in part at NICTA and is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

# References

Abdullah, S., Ahmadi, S., Burke, E., and Dror, M. (2004). Applying Ahuja-Orlin's large neighborhood for constructing examination timetabling solution. In *Proceedings of the Fifth International Conference*

*on the Practice and Theory of Automated Timetabling*, number 3616 in Lecture Notes in Computer Science, pages 413–420. Springer.

Abdullah, S., Ahmadi, S., Burke, E., and Dror, M. (2007a). Investigating Ahuja-Orlin's large neighbourhood search approach for examination timetabling. *OR Spectrum*, 29:351–372.

Abdullah, S., Ahmadi, S., Burke, E., Dror, M., and McCollum, B. (2007b). A tabu-based large neighbourhood search methodology for the capacitated examination timetabling problem. *Journal of the Operational Research Society*, 58:1494–1502.

Ahuja, R. K., Orlin, J. B., and Sharma, D. (2001). Multi-exchange neighborhood structures for the capacitated minimum spanning tree problem. *Mathematical Programming*, 91:71–97.

Ahuja, R. K., Orlin, J. B., and Sharma, D. (2003). A composite very large-scale neighborhood structure for the capacitated minimum spanning tree problem. *Operations Research Letters*, 31:185–194.

Ahuja, R. K., Özlem Ergun, Orlin, J. B., and Punnen, A. P. (2002). A survey of very large-scale neighborhood search techniques. *Discrete Appl. Math.*, 123(1-3):75–102.

Azi, N., Gendreau, M., and Potvin, J.-Y. (2007). An exact algorithm for a single-vehicle routing problem with time windows and multiple routes. *European Journal of Operational Research*, 178(3):755–766.

Balakrishnan, N. (1993). Simple heuristics for the vehicle routeing problem with soft time windows. *The Journal of the Operational Research Society*, 44(3):279–287. ArticleType: research-article / Full publication date: Mar., 1993 / Copyright 1993 Operational Research Society.

Bent, R. and Van Hentenryck, P. (2004). A two-stage hybrid local search for the vehicle routing problem with time windows. *Transportation Science*, 38:515–530.

Calvete, H. I., Galé, C., Oliveros, M.-J., and Sánchez-Valverde, B. (2007). A goal programming approach to vehicle routing problems with soft time windows. *European Journal of Operational Research*, 177(3):1720–1733.

Chiang, W. C. and Russell, R. A. (2004). A metaheuristic for the vehicle-routeing problem with soft time windows. *The Journal of the Operational Research Society*, 55(12):1298–1310. ArticleType: research-article / Full publication date: Dec., 2004 / Copyright 2004 Operational Research Society.

Deineko, V. G. and Woeginger, G. J. (2000). A study of exponential neighborhoods for the travelling salesman problem and for the quadratic assignment problem. *Mathematical Programming*, 87:519–542.

Ergun, O., Orlin, J. B., and Steele-Feldman, A. (2002). Creating very large scale neighborhoods out of smaller ones by compounding moves: A study on the vehicle routing problem. Technical Report 4393-02, MIT Sloan School of Management.

Fagerholt, K. (2001). Ship scheduling with soft time windows: An optimisation based approach. *European Journal of Operational Research*, 131(3):559–571.

Fahrion, R. and Wrede, M. (1990). On a principle of chain-exchange for vehicle-routeing problems (1-VRP). *The Journal of the Operational Research Society*, 41(9):821–827.

Figliozzi, M. A. (2010). An iterative route construction and improvement algorithm for the vehicle routing problem with soft time windows. *Transportation Research Part C: Emerging Technologies*, 18(5):668679.

Fu, Z., Eglese, R., and Li, L. Y. (2007). A unified tabu search algorithm for vehicle routing problems with soft time windows. *Journal of the Operational Research Society*, 59(5):663673.

Glover, F. and Rego, C. (2006). Ejection chain and filter-and-fan methods in combinatorial optimization. *4OR*, 4(4):263296.

Homberger, J. and Gehring, H. (2005). A two-phase hybrid metaheuristic for the vehicle routing problem with time windows. *European Journal of Operational Research*, 162(1):220–238.

Ibaraki, T., Imahori, S., Kubo, M., Masuda, T., Uno, T., and Yagiura, M. (2005). Effective local search algorithms for routing and scheduling problems with general time-window constraints. *Transportation Science*, 39(2):206232.

Ioachim, I., Glinas, S., Soumis, F., and Desrosiers, J. (1998). A dynamic programming algorithm for the shortest path problem with time windows and linear node costs. *Networks*, 31(3):193204.

Ioannou, G., Kritikos, M., and Prastacos, G. (2003). A problem generator-solver heuristic for vehicle routing with soft time windows. *Omega*, 31(1):41–53.

Jha, K. C., Ahuja, R. K., and Şahin, G. (2008). New approaches for solving the block-to-train assignment problem. *Networks*, 51(1):48–62.

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):6985.

Mladenović, N. and Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research*, 24(11):10971100.

Mouthuy, S. (2011). Constraint-based very large-scale neighborhood search. Ph.D. Thesis.

Mouthuy, S., Deville, Y., and Van Hentenryck, P. (2011). A multi-stage very large-scale neighborhood search for the vehicle routing problem with soft time-windows. In *Proceedings of the 9th Metaheuristics International Conference (MIC2011)*.

Mouthuy, S., Van Hentenryck, P., and Deville, Y. (2012). Constraint-based very large-scale neighborhood search. *Constraints*, 17(2):87–122.

Qureshi, A., Taniguchi, E., and Yamada, T. (2009). An exact solution approach for vehicle routing and scheduling problems with soft time windows. *Transportation Research Part E: Logistics and Transportation Review*, 45(6):960–977.

Shaw, P. (1998). Using constraint programming and local search methods to solve vehicle routing problems. In *Principles and Practice of Constraint Programming CP98*, volume 1520, pages 417–431.

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35(2):254–265.

Taillard, E., Badeau, P., Gendreau, M., Guertin, F., and Potvin, J.-Y. (1997). A tabu search heuristic for the vehicle routing problem with soft time windows. *Transportation science*, 31(2):170186.

Thompson, P. M. and Orlin, J. B. (1989). The theory of cyclic transfers. Technical Report OR 200-89, Massachusetts Institute of Technology, Operations Research Center.

Thompson, P. M. and Psaraftis, H. N. (1993). Cyclic transfer algorithm for multivehicle routing and scheduling problems. *Operations research*, 41(5):935946.

Van Hentenryck, P. and Michel, L. (2005). *Constraint-Based Local Search*. MIT Press.

# A Proof of Proposition 1

**Proof of Proposition 1**
Proposition 1 can be proven by induction using inductive Hypothesis 1:

**Hypothesis 1** ($P(n)$)**.** *If $y_l \leq z_l, \forall l \in sites(r)$, then for any service time $s_{r_k}$ such that $k+n=m$ and $y_{r_k} \leq s_{r_k} \leq z_{r_k}$, there is a start of service time $s_j$ for each $j = r_{k+1}, \ldots, r_{m+1}$ such that $y_j \leq s_j \leq z_j$ and such that $[s_{r_k}, s_{r_{k+1}}, \ldots, s_{r_{m+1}}]$ respects constraints* (4) *to* (6) *with $i = r_k, i = r_{k+1}, \ldots, i = r_m$ and constraint* (5) *with $i = r_{m+1}$.*

**Base case: n=0**
This is the case where $k+0=m$, thus customer $r_k$ is the last customer in the route. After serving customer $r_k$, the vehicle will return to the depot $r_0$ (remember we consider the successor of the last customer $r_m$ the depot, thus $r_{m+1} = r_0$). Hypothesis $P(0)$ becomes:
If $y_l \leq z_l, \forall l \in sites(r)$, then for any service time $s_{r_k}$ such that $k=m$ and $y_{r_k} \leq s_{r_k} \leq z_{r_k}$, there is a start of service time $s_{r_0}$ such that $y_{r_0} \leq s_{r_0} \leq z_{r_0}$ and such that $\{s_{r_k}, s_{r_{m+1}}\}$ respects constraints (4) to (6) with $i = r_k = r_m$ and constraint (5) with $i = r_{m+1} = r_0$.

Suppose $s_{r_0} = s_{r_k} + c_{r_k,r_0} + d_{r_k}$. First we need to verify whether $s_{r_0}$ respects $y_{r_0} \leq s_{r_0} \leq z_{r_k}$. It is known that $y_{r_0} = e_{r_0}$ and $z_{r_0} = l_{r_0}$ from (8) and (9), thus it is sufficient to verify $e_{r_0} \leq s_{r_0} \leq l_{r_0}$. Since $s_{r_0} \geq s_{r_k}$ and $s_{r_k} \geq y_{r_k}$ (by hypothesis P(n)) and $y_{r_k} \geq e_{r_k}$ (by definition of $y_{r_k}$) and $e_{r_k} \geq e_{r_0}$ (by problem definition) we have $s_{r_0} \geq e_{r_0}$. By (9) we know that

$$
\begin{array}{rll}
z_{r_k} & \leq z_{r_0} - c_{r_k,r_0} - d_{r_k} & \\
\Leftrightarrow z_{r_k} & \leq l_{r_0} - c_{r_k,r_0} - d_{r_k} & \text{from (9)} \\
\Leftrightarrow s_{r_k} & \leq l_{r_0} - c_{r_k,r_0} - d_{r_k} & \text{since } s_{r_k} \leq z_{r_k} \text{ by hypothesis P(n)} \\
\Leftrightarrow s_{r_k} + c_{r_k,r_0} + d_{r_k} & \leq l_{r_0} & \\
\Leftrightarrow s_{r_0} & \leq l_{r_0} &
\end{array}
$$

Thus we have $e_{r_0} \leq s_{r_0} \leq l_{r_0}$.

Next we need to show that $s_{r_k}$ respects Eq. (4). This is verified by definition of $s_{r_0}$:

$$
\begin{array}{rll}
\delta_{r_k} + c_{r_k,r_0} & \leq s_{r_0} & \\
\Leftrightarrow s_{r_k} + d_{r_k} + c_{r_k,r_0} & \leq s_{r_0} & \text{(by Eq.(2))} \\
\Leftrightarrow s_{r_k} + d_{r_k} + c_{r_k,r_0} & \leq s_{r_k} + d_{r_k} + c_{r_k,r_0} &
\end{array}
$$

Both $s_{r_k}$ and $s_{r_0}$ respect Eq. (5) since $y_{r_k} \leq s_{r_k} \leq z_{r_k}$ and $e_{r_k} \leq y_{r_k} \leq z_{r_k} \leq l_{r_k}$ and $y_{r_0} \leq s_{r_0} \leq z_{r_0}$ and $e_{r_0} \leq y_{r_0} \leq z_{r_0} \leq l_{r_0}$.

Finally $s_{r_k}$ respects Eq. (6) since:

$$
\begin{array}{rl}
s_{r_k} + d_{r_k} + c_{r_k,r_0} & \geq s_{r_0} - w_{max} \\
\Leftrightarrow s_{r_k} + d_{r_k} + c_{r_k,r_0} + w_{max} & \geq s_{r_k} + d_{r_k} + c_{r_k,r_0}
\end{array}
$$

Thus for any $s_{r_k}$ s.t. $y_{r_k} \leq s_{r_k} \leq z_{r_k}$ with $k+0=m$ there is a $s_{r_0}$ such that $\{s_{r_k}, s_{r_0}\}$ respects constraints (4) to (6) with $i = r_k = r_m$ and constraint (5) with $i = r_{m+1} = r_0$ and we have proven $P(0)$.

**Inductive step: if P(n) holds then P(n+1) holds as well**
Assume that the following holds: If $y_l \leq z_l, \forall l \in sites(r)$, then for any service time $s_{r_k}$ such that $k+n=m$ and $y_{r_k} \leq s_{r_k} \leq z_{r_k}$, there is a start of service time $s_j$ for each $j = r_{k+1}, \ldots, r_{m+1}$ such that $y_j \leq s_j \leq z_j$ and such that $\{s_{r_k}, s_{r_{k+1}}, \ldots, s_{r_{m+1}}\}$ respects constraints (4) to (6) with $i = r_k, i = r_{k+1}, \ldots, i = r_m$ and constraint (5) with $i = r_{m+1} = r_0$.

We need to show that if $y_l \leq z_l, \forall l \in sites(r)$, then for any service time $s_{r_{k-1}}$ such that $k-1+n+1=m$ and $y_{r_{k-1}} \leq s_{r_{k-1}} \leq z_{r_{k-1}}$, there is a start of service time $s_j$ for each $j = r_k, \ldots, r_{m+1}$ such that $y_j \leq s_j \leq z_j$ and such that $\{s_{r_{k-1}}, s_{r_k}, \ldots, s_{r_{m+1}}\}$ respects constraints (4) to (6) with $i = r_{k-1}, i = r_k, \ldots, i = r_m$ and constraint (5) with $i = r_{m+1} = r_0$..

Given $P(n)$ it is sufficient to show that for any $s_{r_{k-1}}$ such that $y_{r_{k-1}} \leq s_{r_{k-1}} \leq z_{r_{k-1}}$ there is a $s_{r_k}$ such that $y_{r_k} \leq s_{r_k} \leq z_{r_k}$ and such that $s_{r_{k-1}}$ and $s_{r_k}$ respect Constraints (4) to (6) with $i = k-1$ and $i = k$.

Assume some $s_{r_k-1}$ such that $y_{r_{k-1}} \leq s_{r_{k-1}} \leq z_{r_{k-1}}$ and some $s_{r_k}$ such that $y_{r_k} \leq s_{r_k} \leq z_{r_k}$. Then the following hold:

$$\begin{aligned}
& y_{r_{k-1}} && \geq y_{r_k} - c_{r_{k-1},r_k} - d_{r_{k-1}} - w_{max} && \text{by Eq. (8)} \\
\Leftrightarrow\ & y_{r_k} && \leq y_{r_{k-1}} + c_{r_{k-1},r_k} + d_{r_{k-1}} + w_{max} && \\
\Leftrightarrow\ & y_{r_k} && \leq s_{r_{k-1}} + c_{r_{k-1},r_k} + d_{r_{k-1}} + w_{max} && \text{since } s_{r_{k-1}} \geq y_{r_{k-1}}
\end{aligned}$$

and

$$\begin{aligned}
& z_{r_{k-1}} && \leq z_{r_k} - c_{r_{k-1},r_k} - d_{r_{k-1}} && \text{by Eq. (9)} \\
\Leftrightarrow\ & z_{r_k} && \geq z_{r_{k-1}} + c_{r_{k-1},r_k} + d_{r_{k-1}} && \\
\Leftrightarrow\ & z_{r_k} && \geq s_{r_{k-1}} + c_{r_{k-1},r_k} + d_{r_{k-1}} && \text{since } s_{r_{k-1}} \leq z_{r_{k-1}}
\end{aligned}$$

Also, in order for $s_{r_{k-1}}$ to respect Eqs. (4) and (6) the following must hold:

$$s_{r_{k-1}} + d_{r_{k-1}} + c_{r_{k-1},r_k} \leq s_{r_k} \leq s_{r_{k-1}} + d_{r_{k-1}} + c_{r_{k-1},r_k} + w_{max}$$

Therefore a start of service time $s_{r_k}$ such that $y_{r_k} \leq s_{r_k} \leq z_{r_k}$ and such that Eqs. (4) and (6) are respected must lie in the interval:

$$[max(y_{r_k}, s_{r_{k-1}} + d_{r_{k-1}} + c_{r_{k-1},r_k}); min(z_{r_k}, s_{r_{k-1}} + d_{r_{k-1}} + c_{r_{k-1},r_k} + w_{max})]$$

Since $y_{r_k} \leq s_{r_{k-1}} + c_{r_{k-1},r_k} + d_{r_{k-1}} + w_{max}$ and $z_{r_k} \geq s_{r_{k-1}} + c_{r_{k-1},r_k} + d_{r_{k-1}}$ as shown previously and since $y_{r_k} \leq z_{r_k}$ this interval is non-empty. Since $e_{r_k} \leq y_{r_k} \leq z_{r_k} \leq l_{r_k}$ a start of service time $s_{r_k}$ in this interval also respects Eq. (5).

Finally, if $s_{r_k}$ lies in this interval then $s_{r_{k-1}}$ respects Eqs. (4) to (6) since:

- $s_{r_{k-1}} + d_{r_{k-1}} + c_{r_{k-1},r_k} \leq max(y_{r_k}, s_{r_{k-1}} + d_{r_{k-1}} + c_{r_{k-1},r_k})$
  (Eq. (4) holds with $i = r_{k-1}$)

- $s_{r_{k-1}} + d_{r_{k-1}} + c_{r_{k-1},r_k} + w_{max} \geq min(z_{r_k}, s_{r_{k-1}} + d_{r_{k-1}} + c_{r_{k-1},r_k} + w_{max})$
  (Eq. (6) holds with $i = r_{k-1}$)

- $e_{r_{k-1}} \leq y_{r_{k-1}} \leq s_{r_{k-1}} \leq z_{r_{k-1}} \leq l_{r_{k-1}}$
  (Eq. (5) holds with $i = r_{k-1}$)

This shows that if $y_l \leq z_l, \forall l \in sites(r)$, then for any service time $s_{r_k}$ such that $y_{r_k} \leq s_{r_k} \leq z_{r_k}$, there is a start of service time $s_j$ for each $j = r_{k+1}, \ldots, r_m$ such that $y_j \leq s_j \leq z_j$ and such that $\{s_{r_k}, s_{r_{k+1}}, \ldots, s_{r_{m+1}}\}$ respects constraints (4) to (6) with $i = r_k, i = r_{k+1}, \ldots, i = r_m$ and constraint (5) with $i = r_{m+1} = r_0$. $\square$

# B Additional Results

| $p_{\max}$ | $w_{\max}$ | Instance | *K/Dist.* | *Non-viol. TW (%)* | $sec_{tt}$ *(sec)* |
|---|---|---|---|---|---|
| 5% | 5% | C101 | 11/867 | 99 | 487 |
| | | C102 | 10/1076 | 100 | 776 |
| | | C103 | 10/1092 | 100 | 909 |
| | | C104 | 10/1009 | 100 | 1103 |
| | | C105 | 10/871 | 100 | 577 |
| | | C106 | 10/880 | 100 | 568 |
| | | C107 | 10/865 | 100 | 632 |
| | | C108 | 11/957 | 100 | 719 |
| | | C109 | 10/897 | 100 | 897 |
| 5% | 10% | C101 | 11/901 | 100 | 520 |
| | | C102 | 10/922 | 100 | 732 |
| | | C103 | 10/1036 | 100 | 944 |
| | | C104 | 10/971 | 100 | 1089 |
| | | C105 | 10/829 | 100 | 568 |
| | | C106 | 10/851 | 100 | 572 |
| | | C107 | 10/869 | 100 | 638 |
| | | C108 | 11/895 | 100 | 735 |
| | | C109 | 10/868 | 100 | 890 |
| 10% | 5% | C101 | 10/831 | 98 | 591 |
| | | C102 | 10/1131 | 100 | 862 |
| | | C103 | 10/1010 | 100 | 1082 |
| | | C104 | 10/985 | 100 | 1235 |
| | | C105 | 10/871 | 100 | 714 |
| | | C106 | 11/972 | 100 | 733 |
| | | C107 | 10/869 | 98 | 803 |
| | | C108 | 11/902 | 100 | 850 |
| | | C109 | 10/888 | 100 | 1041 |
| 10% | 10% | C101 | 10/935 | 99 | 631 |
| | | C102 | 10/1019 | 100 | 871 |
| | | C103 | 10/865 | 100 | 1159 |
| | | C104 | 10/972 | 100 | 1270 |
| | | C105 | 11/907 | 100 | 704 |
| | | C106 | 11/926 | 100 | 747 |
| | | C107 | 10/842 | 97 | 785 |
| | | C108 | 11/917 | 100 | 888 |
| | | C109 | 10/936 | 100 | 1079 |

Table 4: Type 3, problem set C1

| Instance | *K/Dist.* | *Non-viol. TW (%)* | $sec_{tt}$ *(sec)* |
|---|---|---|---|
| R101 | 15/1599 | 84 | 346 |
| R102 | 13/1530 | 86 | 406 |
| R103 | 11/1246 | 95 | 445 |
| R104 | 10/1097 | 100 | 484 |
| R105 | 12/1434 | 83 | 358 |
| R106 | 11/1323 | 94 | 392 |
| R107 | 10/1191 | 97 | 415 |
| R108 | 9/1037 | 98 | 490 |
| R109 | 11/1248 | 95 | 378 |
| R110 | 10/1203 | 91 | 404 |
| R111 | 10/1206 | 97 | 406 |
| R112 | 10/1036 | 100 | 461 |
| C101 | 10/829 | 100 | 467 |
| C102 | 10/842 | 100 | 702 |
| C103 | 11/882 | 100 | 866 |
| C104 | 10/917 | 100 | 954 |
| C105 | 10/869 | 100 | 522 |
| C106 | 10/831 | 100 | 536 |
| C107 | 10/879 | 100 | 574 |
| C108 | 10/868 | 100 | 680 |
| C109 | 10/904 | 100 | 789 |
| RC101 | 13/1668 | 86 | 323 |
| RC102 | 12/1593 | 96 | 380 |
| RC103 | 11/1344 | 98 | 393 |
| RC104 | 10/1229 | 100 | 428 |
| RC105 | 12/1611 | 91 | 344 |
| RC106 | 11/1466 | 96 | 351 |
| RC107 | 11/1373 | 100 | 386 |
| RC108 | 10/1267 | 99 | 396 |
| R201 | 4/1390 | 100 | 2702 |
| R202 | 3/1296 | 98 | 4729 |
| R203 | 3/1079 | 100 | 7889 |
| R204 | 3/841 | 100 | 9218 |
| R205 | 3/1170 | 100 | 4427 |
| R206 | 3/1033 | 100 | 5745 |
| R207 | 3/980 | 100 | 7445 |
| R208 | 2/928 | 100 | 8392 |
| R209 | 3/1078 | 100 | 4880 |
| R210 | 3/1038 | 99 | 5792 |
| R211 | 3/884 | 100 | 6306 |
| C201 | 3/591 | 100 | 1957 |
| C202 | 3/591 | 100 | 3976 |
| C203 | 4/673 | 100 | 5528 |
| C204 | 4/731 | 100 | 6854 |
| C205 | 3/589 | 100 | 1789 |
| C206 | 3/589 | 100 | 1934 |
| C207 | 3/589 | 100 | 2116 |
| C208 | 3/591 | 100 | 2441 |
| RC201 | 4/1600 | 97 | 2527 |
| RC202 | 4/1384 | 99 | 3083 |
| RC203 | 3/1269 | 100 | 5794 |
| RC204 | 3/951 | 100 | 6867 |
| RC205 | 4/1391 | 97 | 2913 |
| RC206 | 4/1303 | 100 | 3229 |
| RC207 | 4/1233 | 100 | 3478 |
| RC208 | 3/1014 | 100 | 4255 |

Table 5: Type 4, $p_{max} = 5\%$

| Instance | *K/Dist.* | *Non-viol. TW (%)* | $sec_{tt}$ *(sec)* |
|---|---|---|---|
| R101 | 13/1485 | 62 | 342 |
| R102 | 12/1349 | 81 | 393 |
| R103 | 10/1184 | 84 | 421 |
| R104 | 10/1078 | 100 | 476 |
| R105 | 12/1377 | 83 | 366 |
| R106 | 11/1262 | 92 | 405 |
| R107 | 10/1132 | 93 | 427 |
| R108 | 9/1018 | 99 | 434 |
| R109 | 11/1254 | 97 | 392 |
| R110 | 10/1187 | 94 | 397 |
| R111 | 10/1154 | 95 | 416 |
| R112 | 10/1031 | 100 | 457 |
| C101 | 10/910 | 97 | 525 |
| C102 | 10/1133 | 100 | 706 |
| C103 | 10/1065 | 100 | 925 |
| C104 | 10/939 | 100 | 1150 |
| C105 | 10/869 | 100 | 607 |
| C106 | 10/878 | 100 | 621 |
| C107 | 10/913 | 99 | 646 |
| C108 | 10/871 | 100 | 750 |
| C109 | 10/870 | 100 | 899 |
| RC101 | 12/1635 | 75 | 337 |
| RC102 | 11/1473 | 85 | 363 |
| RC103 | 10/1316 | 89 | 385 |
| RC104 | 10/1269 | 100 | 443 |
| RC105 | 12/1585 | 89 | 355 |
| RC106 | 11/1441 | 91 | 347 |
| RC107 | 10/1322 | 85 | 411 |
| RC108 | 10/1245 | 100 | 384 |
| R201 | 4/1333 | 96 | 3328 |
| R202 | 3/1244 | 95 | 4745 |
| R203 | 3/1055 | 98 | 7582 |
| R204 | 3/916 | 100 | 10398 |
| R205 | 3/1132 | 99 | 5407 |
| R206 | 3/1035 | 100 | 6842 |
| R207 | 3/959 | 100 | 8711 |
| R208 | 2/832 | 100 | 9950 |
| R209 | 3/1001 | 99 | 5945 |
| R210 | 3/1076 | 100 | 6825 |
| R211 | 3/865 | 100 | 7937 |
| C201 | 3/591 | 100 | 2116 |
| C202 | 4/687 | 99 | 5421 |
| C203 | 4/752 | 100 | 6674 |
| C204 | 4/725 | 100 | 7241 |
| C205 | 3/590 | 100 | 2563 |
| C206 | 3/589 | 100 | 2156 |
| C207 | 3/592 | 100 | 2029 |
| C208 | 3/592 | 100 | 2313 |
| RC201 | 4/1536 | 94 | 2523 |
| RC202 | 3/1375 | 94 | 3467 |
| RC203 | 3/1165 | 98 | 5097 |
| RC204 | 3/979 | 100 | 7146 |
| RC205 | 4/1507 | 96 | 3192 |
| RC206 | 3/1291 | 98 | 3338 |
| RC207 | 4/1201 | 99 | 4043 |
| RC208 | 3/1069 | 100 | 5469 |

Table 6: Type 4, $p_{max} = 10\%$

| Instance | *K/Dist.* | *Non-viol. TW (%)* | $sec_{tt}$ *(sec)* |
|---|---|---|---|
| R101 | 14/1574 | 64 | 353 |
| R102 | 12/1432 | 76 | 439 |
| R103 | 10/1241 | 68 | 429 |
| R104 | 9/1025 | 90 | 476 |
| R105 | 12/1411 | 74 | 364 |
| R106 | 11/1312 | 93 | 409 |
| R107 | 10/1166 | 97 | 430 |
| R108 | 9/1043 | 96 | 456 |
| R109 | 11/1261 | 92 | 404 |
| R110 | 10/1197 | 93 | 399 |
| R111 | 10/1179 | 94 | 414 |
| R112 | 10/1047 | 100 | 472 |
| C101 | 10/829 | 100 | 498 |
| C102 | 10/940 | 100 | 758 |
| C103 | 10/954 | 100 | 923 |
| C104 | 10/931 | 100 | 1125 |
| C105 | 10/875 | 100 | 544 |
| C106 | 10/839 | 100 | 595 |
| C107 | 10/868 | 100 | 630 |
| C108 | 10/842 | 100 | 727 |
| C109 | 10/896 | 100 | 883 |
| RC101 | 12/1578 | 69 | 332 |
| RC102 | 11/1507 | 76 | 361 |
| RC103 | 10/1298 | 88 | 398 |
| RC104 | 10/1227 | 99 | 523 |
| RC105 | 11/1483 | 70 | 349 |
| RC106 | 11/1463 | 86 | 363 |
| RC107 | 11/1384 | 98 | 394 |
| RC108 | 10/1229 | 98 | 395 |
| R201 | 4/1376 | 98 | 3442 |
| R202 | 3/1394 | 94 | 5059 |
| R203 | 3/1125 | 100 | 8760 |
| R204 | 3/826 | 100 | 11747 |
| R205 | 3/1140 | 99 | 5165 |
| R206 | 3/1050 | 100 | 7053 |
| R207 | 3/961 | 100 | 8660 |
| R208 | 3/790 | 100 | 10771 |
| R209 | 3/1030 | 100 | 6974 |
| R210 | 3/1212 | 100 | 7407 |
| R211 | 3/928 | 100 | 7913 |
| C201 | 3/591 | 100 | 1814 |
| C202 | 4/626 | 100 | 4462 |
| C203 | 4/667 | 100 | 6213 |
| C204 | 3/683 | 100 | 7255 |
| C205 | 3/591 | 100 | 1755 |
| C206 | 3/594 | 100 | 1985 |
| C207 | 3/598 | 100 | 2100 |
| C208 | 3/605 | 100 | 2250 |
| RC201 | 4/1594 | 95 | 2440 |
| RC202 | 4/1340 | 98 | 4510 |
| RC203 | 4/1162 | 100 | 5764 |
| RC204 | 3/921 | 100 | 7018 |
| RC205 | 4/1374 | 96 | 3248 |
| RC206 | 4/1340 | 100 | 3868 |
| RC207 | 3/1214 | 99 | 4099 |
| RC208 | 3/1006 | 100 | 5571 |

Table 7: Type 5, $p_{max} = 5\%$

| Instance | *K/Dist.* | *Non-viol. TW (%)* | $sec_{tt}$ *(sec)* |
|---|---|---|---|
| R101 | 12/1445 | 45 | 377 |
| R102 | 11/1333 | 68 | 413 |
| R103 | 10/1168 | 79 | 409 |
| R104 | 9/1018 | 88 | 443 |
| R105 | 11/1262 | 62 | 374 |
| R106 | 10/1245 | 73 | 392 |
| R107 | 10/1107 | 91 | 428 |
| R108 | 9/996 | 98 | 440 |
| R109 | 10/1185 | 78 | 396 |
| R110 | 10/1187 | 91 | 401 |
| R111 | 10/1061 | 93 | 460 |
| R112 | 9/990 | 93 | 433 |
| C101 | 10/863 | 98 | 626 |
| C102 | 10/865 | 100 | 891 |
| C103 | 10/912 | 100 | 1126 |
| C104 | 10/938 | 100 | 1305 |
| C105 | 10/829 | 100 | 723 |
| C106 | 10/876 | 100 | 786 |
| C107 | 10/862 | 100 | 811 |
| C108 | 11/876 | 100 | 921 |
| C109 | 10/863 | 100 | 1044 |
| RC101 | 11/1467 | 52 | 363 |
| RC102 | 11/1486 | 84 | 367 |
| RC103 | 10/1293 | 88 | 397 |
| RC104 | 10/1221 | 97 | 436 |
| RC105 | 11/1420 | 77 | 354 |
| RC106 | 11/1385 | 86 | 395 |
| RC107 | 10/1333 | 88 | 396 |
| RC108 | 10/1244 | 96 | 407 |
| R201 | 3/1402 | 83 | 3639 |
| R202 | 3/1301 | 94 | 6586 |
| R203 | 3/1061 | 97 | 9683 |
| R204 | 3/844 | 100 | 11660 |
| R205 | 3/1152 | 100 | 6315 |
| R206 | 3/1040 | 100 | 9322 |
| R207 | 3/918 | 99 | 11647 |
| R208 | 3/771 | 100 | 9994 |
| R209 | 3/1082 | 100 | 8240 |
| R210 | 3/1062 | 99 | 10096 |
| R211 | 3/868 | 100 | 11241 |
| C201 | 3/591 | 100 | 2087 |
| C202 | 4/676 | 100 | 6693 |
| C203 | 4/715 | 100 | 6688 |
| C204 | 3/805 | 99 | 6469 |
| C205 | 3/589 | 100 | 2012 |
| C206 | 3/591 | 100 | 2521 |
| C207 | 3/622 | 100 | 3275 |
| C208 | 3/632 | 100 | 4848 |
| RC201 | 4/1602 | 91 | 3286 |
| RC202 | 3/1357 | 95 | 4400 |
| RC203 | 3/1191 | 96 | 6493 |
| RC204 | 3/1001 | 99 | 9705 |
| RC205 | 3/1379 | 86 | 3599 |
| RC206 | 3/1273 | 95 | 4003 |
| RC207 | 3/1311 | 95 | 5299 |
| RC208 | 3/1091 | 100 | 7959 |

Table 8: Type 5, $p_{max} = 10\%$

| Instance | *K/Dist.* | *Non-viol. TW (%)* | $sec_{tt}$ *(sec)* |
|---|---|---|---|
| R101 | 14/1555 | 64 | 343 |
| R102 | 12/1399 | 78 | 393 |
| R103 | 10/1252 | 79 | 417 |
| R104 | 10/1069 | 98 | 473 |
| R105 | 12/1412 | 78 | 354 |
| R106 | 11/1324 | 91 | 415 |
| R107 | 10/1202 | 95 | 419 |
| R108 | 9/1027 | 97 | 483 |
| R109 | 11/1231 | 92 | 448 |
| R110 | 10/1206 | 91 | 405 |
| R111 | 10/1179 | 94 | 428 |
| R112 | 10/1047 | 100 | 509 |
| C101 | 10/829 | 100 | 506 |
| C102 | 10/951 | 100 | 764 |
| C103 | 10/999 | 100 | 890 |
| C104 | 10/964 | 100 | 1078 |
| C105 | 10/829 | 100 | 563 |
| C106 | 10/841 | 100 | 555 |
| C107 | 10/898 | 100 | 628 |
| C108 | 10/837 | 100 | 766 |
| C109 | 11/891 | 100 | 899 |
| RC101 | 12/1549 | 66 | 333 |
| RC102 | 11/1509 | 79 | 367 |
| RC103 | 10/1267 | 90 | 396 |
| RC104 | 10/1220 | 100 | 425 |
| RC105 | 11/1508 | 73 | 351 |
| RC106 | 11/1451 | 93 | 361 |
| RC107 | 10/1277 | 87 | 385 |
| RC108 | 10/1256 | 98 | 389 |
| R201 | 4/1391 | 98 | 3205 |
| R202 | 3/1262 | 93 | 5346 |
| R203 | 3/1046 | 99 | 8978 |
| R204 | 3/858 | 100 | 10569 |
| R205 | 4/1081 | 100 | 4933 |
| R206 | 3/1121 | 100 | 6960 |
| R207 | 3/940 | 100 | 8901 |
| R208 | 3/808 | 100 | 8506 |
| R209 | 3/1023 | 100 | 6753 |
| R210 | 3/1092 | 100 | 7459 |
| R211 | 3/883 | 100 | 8504 |
| C201 | 3/591 | 100 | 1859 |
| C202 | 3/618 | 100 | 4015 |
| C203 | 4/679 | 100 | 6846 |
| C204 | 4/769 | 100 | 6417 |
| C205 | 3/598 | 100 | 1941 |
| C206 | 3/600 | 100 | 1866 |
| C207 | 3/594 | 100 | 2063 |
| C208 | 3/600 | 100 | 2272 |
| RC201 | 4/1539 | 97 | 2252 |
| RC202 | 4/1427 | 98 | 3838 |
| RC203 | 4/1162 | 100 | 5983 |
| RC204 | 3/979 | 99 | 7027 |
| RC205 | 4/1495 | 95 | 3254 |
| RC206 | 4/1357 | 100 | 3860 |
| RC207 | 3/1362 | 100 | 4230 |
| RC208 | 3/994 | 100 | 5598 |

Table 9: Type 6, $p_{max} = 5\%$

| Instance | *K/Dist.* | *Non-viol. TW (%)* | $sec_{tt}$ *(sec)* |
|---|---|---|---|
| R101 | 12/1446 | 47 | 359 |
| R102 | 11/1290 | 65 | 424 |
| R103 | 10/1158 | 82 | 414 |
| R104 | 9/1014 | 90 | 421 |
| R105 | 11/1328 | 67 | 379 |
| R106 | 10/1182 | 78 | 412 |
| R107 | 10/1126 | 92 | 453 |
| R108 | 9/1003 | 97 | 462 |
| R109 | 10/1188 | 78 | 396 |
| R110 | 10/1122 | 91 | 422 |
| R111 | 9/1008 | 70 | 435 |
| R112 | 9/1023 | 95 | 497 |
| C101 | 10/837 | 96 | 633 |
| C102 | 10/964 | 99 | 819 |
| C103 | 10/1003 | 100 | 1110 |
| C104 | 10/975 | 100 | 1229 |
| C105 | 10/830 | 98 | 714 |
| C106 | 10/834 | 100 | 717 |
| C107 | 10/883 | 98 | 771 |
| C108 | 10/871 | 99 | 897 |
| C109 | 10/917 | 100 | 1006 |
| RC101 | 11/1471 | 58 | 353 |
| RC102 | 11/1446 | 86 | 375 |
| RC103 | 10/1279 | 90 | 389 |
| RC104 | 10/1205 | 98 | 456 |
| RC105 | 11/1498 | 80 | 363 |
| RC106 | 10/1305 | 73 | 380 |
| RC107 | 10/1309 | 85 | 380 |
| RC108 | 10/1239 | 97 | 445 |
| R201 | 3/1439 | 86 | 3864 |
| R202 | 3/1172 | 92 | 6476 |
| R203 | 3/1020 | 97 | 10234 |
| R204 | 3/859 | 100 | 11787 |
| R205 | 3/1151 | 100 | 7167 |
| R206 | 3/1041 | 98 | 8291 |
| R207 | 3/968 | 99 | 10296 |
| R208 | 2/854 | 97 | 10898 |
| R209 | 3/1076 | 99 | 7631 |
| R210 | 3/1060 | 98 | 9677 |
| R211 | 3/866 | 100 | 10049 |
| C201 | 3/591 | 100 | 2276 |
| C202 | 4/681 | 100 | 5774 |
| C203 | 4/758 | 100 | 6830 |
| C204 | 3/805 | 100 | 6046 |
| C205 | 3/589 | 100 | 2400 |
| C206 | 3/612 | 100 | 2353 |
| C207 | 3/667 | 100 | 3044 |
| C208 | 3/721 | 100 | 5183 |
| RC201 | 4/1504 | 90 | 3470 |
| RC202 | 3/1387 | 89 | 4174 |
| RC203 | 3/1251 | 96 | 6613 |
| RC204 | 3/934 | 100 | 9229 |
| RC205 | 3/1362 | 87 | 3651 |
| RC206 | 3/1409 | 99 | 4348 |
| RC207 | 3/1285 | 98 | 5690 |
| RC208 | 3/1047 | 100 | 7529 |

Table 10: Type 6, $p_{max} = 10\%$