The VRP with Time Windows

J.-F. Cordeau
École des Hautes Études Commerciales, Montréal
cordeau@crt.umontreal.ca

Guy Desaulniers
Département de mathématiques et génie industriel
École Polytechnique de Montréal, Montréal
guyd@crt.umontreal.ca

Jacques Desrosiers
Service de l'enseignement des méthodes quantitatives de gestion
École des Hautes Études Commerciales, Montréal
Jacques.Desrosiers@hec.ca

Marius M. Solomon
Management Science Group
Northeastern University, Boston
msolomon@cba.neu.edu

François Soumis
Département de mathématiques et génie industriel
École Polytechnique de Montréal, Montréal
soumi@crt.umontreal.ca

and GERAD

February, 1999
Revised: June, 2000

Les Cahiers du GERAD

G–99–13

Abstract

This paper presents a survey of the research on the Vehicle Routing Problem with Time Windows (VRPTW), an extension of the Capacitated Vehicle Routing Problem. In the VRPTW, the service at each customer must start within an associated time window and the vehicle must remain at the customer location during service. Soft time windows can be violated at a cost while hard time windows do not allow for a vehicle to arrive at a customer after the latest time to begin service. We first present a multi-commodity network flow formulation with time and capacity constraints for the VRPTW. Approximation methods proposed in the literature to derive upper bounds are then reviewed. Then we explain how lower bounds can be obtained using optimal approaches, namely, Lagrangean relaxation and column generation. Next, we provide branching and cutting strategies that can be embedded within these optimal approaches to produce integer solutions. Special cases and extensions to the VRPTW follow as well as our conclusions.

Résumé

Cet article synthèse porte sur les récents développements concernant le problème du routage de véhicules sous des contraintes de fenêtres de temps. Dans ce problème, le service à un client doit débuter à l'intérieur d'un intervalle de temps. Celui-ci peut être, soit relaché au prix d'une certaine pénalité, soit rigide, auquel cas, il n'est pas permis de dépasser la limite supérieure. Nous présentons un modèle de réseau multi-flots avec des contraintes de temps et de capacité. Les méthodes heuristiques permettant de calculer des bornes supérieures sont d'abord présentées. Suivent les modèles d'optimisation basés sur la relaxation lagrangienne et la génération de colonnes pour évaluer des bornes inférieures. Enfin, on présente les stratégies de coupes et de branchements liées à ces méthodes afin de déterminer des solutions entières. L'article se termine par l'étude de cas particuliers et d'extensions ainsi que nos conclusions.

À venir dans/*Forthcoming in*:

The Vehicle Routing Problem, Chapter 7, Paolo Toth and Daniele Vigo (eds), SIAM Monographs on Discrete Mathematics and Applications, SIAM, Philadelpia, Pa.

The Vehicle Routing Problem with Time Windows (VRPTW) is the extension of the CVRP where the service at each customer must start within an associated time window and the vehicle must remain at the customer location during service. Soft time windows can be violated at a cost while hard time windows do not allow for a vehicle to arrive at a customer after the latest time to begin service. In the latter case, if it arrives before the customer is ready to begin service, it waits. We will concentrate on hard time window scenarios where research has flourished over the last two decades.

As mentioned in Chapter 1, the VRPTW is NP-hard. Indeed, even finding a feasible solution to the VRPTW with a fixed fleet size is itself a NP-complete problem (Savelsbergh [77]). Hence, the early work on the VRPTW has been case study oriented (Pullen and Webb [73]; Knight and Hofer [56]; Madsen [65]). Later research shifted focus to the design of heuristics capable of solving realistic size problems and the development of effective optimal approaches.

This chapter is organized as follows. The first section presents a multi-commodity network flow formulation with time and capacity constraints for the VRPTW. Approximation methods proposed in the literature to derive upper bounds are then reviewed in Section 2. The following section explains how lower bounds can be obtained using optimal approaches, namely, Lagrangean relaxation and column generation. Next, Section 4 provides branching and cutting strategies that can be embedded within these optimal approaches to produce integer solutions through a branch-and-bound scheme. Then, Sections 5 and 6 present special cases and extensions to the VRPTW, respectively. We review computational experience with leading algorithms in Section 7. Finally, conclusions are drawn in Section 8.

1 Problem Formulation

Starting from the notation given in Chapter 1, the VRPTW is defined on the network $G = (V, A)$ where the depot is represented by the two nodes 0 and $n+1$. All feasible vehicle routes correspond to paths in G that start from node 0 and end at node $n+1$. A time window is also associated with nodes 0 and $n+1$, i.e., $[a_0, b_0] = [a_{n+1}, b_{n+1}] = [E, L]$, where E and L represent the earliest possible departure from the depot and the latest possible arrival at the depot, respectively. Moreover, zero demands and service times are defined for these two nodes, that is, $d_0 = d_{n+1} = s_0 = s_{n+1} = 0$. Feasible solutions exist only if $a_0 = E \leq \min\limits_{i \in V\setminus\{0\}} b_i - t_{0i}$ and $b_{n+1} = L \geq \min\limits_{i \in V\setminus\{0\}} a_i + s_i + t_{i0}$. Note also that an arc $(i, j) \in A$ can be eliminated due to temporal considerations, if $a_i + s_i + t_{ij} > b_j$, or capacity limitations, if $d_i + d_j > C$, or by other factors. Finally, let us mention that when vehicles are allowed to remain at the depot, especially in the case where the primary objective consists of minimizing the number of vehicles used, the arc $(0, n+1)$ with $c_{0,n+1} = t_{0,n+1} = 0$ must be added to the arc set A.

We next present a mathematical programming formulation for the VRPTW involving two types of variables: flow variables x_{ijk}, $(i, j) \in A, k \in K$, equal to 1 if arc (i, j) is used by vehicle k, and 0 otherwise; and time variables w_{ik}, $i \in V, k \in K$, specifying the start of service at node i when serviced by vehicle k.

Formulation. The VRPTW can then be formally described as the following multi-commodity network flow model with time window and capacity constraints:

$$\text{(VRPTW)} \qquad \min \sum_{k \in K} \sum_{(i,j) \in A} c_{ij} x_{ijk} \tag{1}$$

subject to

$$\sum_{k \in K} \sum_{j \in \Delta^+(i)} x_{ijk} = 1, \qquad \forall i \in N \tag{2}$$

$$\sum_{j \in \Delta^+(0)} x_{0jk} = 1, \qquad \forall k \in K \tag{3}$$

$$\sum_{i \in \Delta^-(j)} x_{ijk} - \sum_{i \in \Delta^+(j)} x_{jik} = 0, \qquad \forall k \in K,\ \forall j \in N \tag{4}$$

$$\sum_{i \in \Delta^-(n+1)} x_{i,n+1,k} = 1, \qquad \forall k \in K \tag{5}$$

$$x_{ijk}(w_{ik} + s_i + t_{ij} - w_{jk}) \leq 0, \qquad \forall k \in K,\ \forall (i,j) \in A \tag{6}$$

$$a_i\Big(\sum_{j \in \Delta^+(i)} x_{ijk}\Big) \leq w_{ik} \leq b_i\Big(\sum_{j \in \Delta^+(i)} x_{ijk}\Big), \qquad \forall k \in K,\ \forall i \in N \tag{7}$$

$$E \leq w_{ik} \leq L, \qquad \forall k \in K,\ \forall i \in \{0, n+1\} \tag{8}$$

$$\sum_{i \in N} d_i \sum_{j \in \Delta^+(i)} x_{ijk} \leq C, \qquad \forall k \in K \tag{9}$$

$$x_{ijk} \geq 0, \qquad \forall k \in K,\ \forall (i,j) \in A \tag{10}$$

$$x_{ijk} \text{ binary}, \qquad \forall k \in K,\ \forall (i,j) \in A. \tag{11}$$

The objective function (1) of this nonlinear formulation expresses the total cost. Given that $N = V \setminus \{0, n+1\}$ represents the set of customers, constraints (2) restrict the assignment of each customer to exactly one vehicle route. Next, constraints (3)–(5) characterize the flow on the path to be followed by vehicle k. Additionally, constraints (6)–(8) and (9) guarantee schedule feasibility with respect to time considerations and capacity aspects, respectively. Note that for a given k, constraints (7) force $w_{ik} = 0$ whenever customer i is not visited by vehicle k. Finally, (11) impose binary conditions on the flow variables.

The binary conditions (11) allow constraints (6) to be linearized as:

$$(7.6a) \qquad w_{ik} + s_i + t_{ij} - w_{jk} \leq \Big(1 - x_{ijk}\Big) M_{ij}, \qquad \forall k \in K,\ \forall (i,j) \in A,$$

where M_{ij} are large constants. Furthermore, M_{ij} can be replaced by $\max\{b_i+s_i+t_{ij}-a_j, 0\}$, $(i,j) \in A$, and constraints (6) or (7.6a) need only be enforced for arcs $(i,j) \in A$, such that $M_{ij} > 0$; otherwise, when $\max\{b_i + s_i + t_{ij} - a_j, 0\}$=0, these constraints are satisfied for all values of w_{ik}, w_{jk} and x_{ijk}.

Network Lower Bound. This can be derived by relaxing the time and capacity constraints (6)–(9). Generally, the bound deteriorates as time window width increases and/or capacity constraints become tighter. It is often of poor quality as there is usually an integrality gap with respect to the number of vehicles. Note however that if the latter constraints are not binding and if $a_i = b_i$, for all $i \in N$, we obtain the fixed schedule problem for which the network lower bound is optimal.

Linear Programming Lower Bound. This bound is obtained as the solution to the linear program using constraints (7.6a) in place of (6) and with the binary requirements (11) omitted. This is the above network flow problem with the additional time and capacity constraints. Nevertheless, in many cases, this bound is no better than the network relaxation bound. This is because it is relatively easy to obtain a fractional near optimal linear programming solution to problem (1)–(10) for which the time constraints are inactive. To see this, set the time variables at the center of their respective time window, i.e., $w_{ik} = (a_i + b_i)/2$, $i \in N, k \in K$. Then, constraints (7.6a) are satisfied if, for all $(i, j) \in A$ and $k \in K$:

$$\left(\frac{a_i + b_i}{2}\right) + s_i + t_{ij} - \left(\frac{a_j + b_j}{2}\right) \leq (1 - x_{ijk})(b_i + s_i + t_{ij} - a_j).$$

Since any existing arc $(i, j) \in A$ satisfies $a_i + s_i + t_{ij} - b_j \leq 0$, and constraints (7.6a) are only defined for arcs (i, j) such that $b_i + s_i + t_{ij} - a_j > 0$, the previous constraint set can be rewritten as:

$$x_{ijk} \leq \frac{1}{2}\left(1 + \frac{b_j - a_i - s_i - t_{ij}}{b_i + s_i + t_{ij} - a_j}\right), \qquad \forall k \in K,\ \forall (i, j) \in A.$$

In the above inequality, the right hand side is greater than or equal to $\frac{1}{2}$. Therefore, (7.6a) is always satisfied if $x_{ijk} \leq \frac{1}{2}$, for all $(i, j) \in A$, $k \in K$, such that $b_i + s_i + t_{ij} - a_j > 0$. A similar argument can be used for the capacity constraints.

Algorithms. Much stronger lower bounds can be derived by decomposing the VRPTW model into intelligently selected blocks and using these in the solution process. This requires an extensive effort and is the subject of Section 3. In the next section, we focus on the derivation of upper bounds through approximate methods. Virtually all methods to be described in these two sections conduct some form of preprocessing. This involves reducing time window width and eliminating infeasible arcs. These processes are described at length in Desrosiers *et al.* [35].

2 Upper Bounds: Heuristic Approaches

Given the inherent computational difficulty of the VRPTW, a variety of heuristics have been reported in the literature, mostly for the hard time window version. In this section, we review some of these approximation methods.

Route Construction. These algorithms build a feasible solution by inserting at every iteration an unrouted customer into a current partial route. This process is performed either sequentially, one route at a time, or in parallel, where several routes are considered simultaneously. Two key questions are posed in the design of such methods: Which customer to select next for insertion? And, where will it be inserted? To address them, researchers have considered criteria involving the minimum additional distance and time, maximum savings, and others.

Sequential insertion heuristics were proposed by Solomon [83]. His extensive computational results highlighted a two-phase insertion algorithm. In the first phase, each unrouted customer is assigned its best feasible insertion position based on the minimum additional distance and time required. In the second, the method selects the customer to insert using a maximum savings concept. Solomon [82] also showed that the worst-case ratio of this and many other approximation methods on n-customer problems is at least of order n. A parallel variant of the above insertion procedure was suggested by Potvin and Rousseau [71].

Route Improvement. These methods iteratively modify the current solution by performing local searches for better neighboring solutions. Generally, a neighborhood comprises the set of solutions that can be reached from the present one by swapping a subset of r arcs between solutions. An r-exchange is implemented only if it leads to an improved feasible solution. It can be performed within and/or between routes. The process terminates when a r-optimal solution is found, that is, one that cannot be improved by further r-interchanges.

Early route improvement procedures were proposed by Russell [75], Cook and Russell [21], and Baker and Schaffer [5]. Even though these authors kept r small, r=2 or 3, the neighborhoods generated still proved very large. This led to effective, but severely time consuming methods. To alleviate this problem, later procedures relied on OR-opt exchanges (Or [67]) which only consider currently adjacent customers for 2- and 3-opt interchanges. Solomon, Baker, and Schaffer [84] extended this method to the VRPTW by also accounting for the time orientation of a route. That is, at each iteration, up to three adjacent customers are shifted to a later position on the same route, between two currently consecutive customers. Schedule shifts are also used to speed up the screening of infeasible solutions. The efficient implementation of this process and of the objective function evaluation has also been addressed by Savelsbergh ([77], [78] and [79]). Recently, further suggestions have been offered by Kindervater and Savelsbergh [55], and Cordone and Wolfler Calvo [26].

Another OR-opt based algorithm has been suggested by Thompson and Psaraftis [91]. They define the neighborhood of the current solution in terms of feasible transfers of sets of demands belonging to adjacent customers. The exchanges are attempted among a subset of routes that form a cyclic permutation. The authors implemented a method based on 2- and 3-cyclic 1-transfers.

Composite Heuristics. These methods blend route construction and improvement algorithms. Kontoravdis and Bard [61] have devised a heuristic that combines a greedy heuristic and randomization to produce initial routes in parallel. These are then improved through local search. As part of this phase, certain routes may be eliminated. The authors also proposed three lower bounds for the fleet size. Two are based on Bin Packing structures generated by the capacity or time window constraints. Another is derived from the associated graph created by pairs of customers who have incompatible demands or time windows.

Russell [76] developed a procedure that embeds route improvement within the tour construction process. The rationale is that this may alleviate some of the difficulties that tour improvement algorithms have to subsequently improve initial solutions. He proposes to switch customers between routes as well as the elimination of routes during the construction process.

Cordone and Wolfler Calvo [27] use similar ideas in the design of a composite heuristic, where local search is performed hierarchically. First, within a classical 2- and 3-opt exchange framework, they attempt to decrease the number of routes by moving a route into others, one customer at a time. Second, another heuristic is used to try to step away from a local optimum. This procedure resolves the problem with a partly modified objective function since the current solution may not be a local optimum for the related objective.

Metaheuristics. Metaheuristics are the core of recent work on approximation methods for the VRPTW, and mainly include simulated annealing, tabu search and evolutionary algorithms such as genetic search. Unlike local search heuristics that terminate once a local optimum has been reached, these methods are aimed at exploring a larger subset of the solution space in the hope of finding a near-optimal solution. Whereas simulated annealing depends mostly on random steps to escape local optima, tabu search uses short and long term memory to avoid cycling and orient the search toward unexplored regions of the solution space. Evolutionary algorithms are derived from an analogy with the natural evolution process and consist of iteratively selecting, recombining and mutating encoded solutions in order to obtain superior individuals.

In recent years, several efficient tabu search approaches have been proposed. Taillard *et al.* [87] described a metaheuristic based on tabu search for the VRP with soft time windows. By strongly penalizing any lateness, the same approach can also be used to address the problem with hard time windows. The metaheuristic relies on the concept of adaptive memory first introduced by Rochat and Taillard [74] and on the decomposition and reconstruction procedure previously proposed by Taillard [88] for the VRP. The adaptive memory is in fact a pool of routes taken from the best solutions visited during the search. This memory is first partially filled with routes produced by a randomized insertion procedure based on Solomon's I1 heuristic (Solomon [83]). At each iteration of the metaheuristic, a solution is constructed, through a randomized selection process, from the routes in the adaptive memory. This solution is then improved through repeated calls to the tabu search heuristic. The routes of the resulting solution are then stored in the adaptive memory (provided that the memory is not full or that the solution is better than the worst solution stored in memory) and the process continues until some stopping criterion is met.

The calls to the tabu search heuristic are driven by a decomposition and reconstruction mechanism that partitions (through a sweep procedure) the current solution into a number of disjoint subsets of routes. Each subset is then processed by a different tabu search and the best routes found for every subset are merged to form the new solution for the next decomposition and reconstruction step. These steps are repeated for a certain number of iterations, and the decomposition changes from one iteration to the next by choosing a different starting angle for creating partitions through the sweep procedure. The tabu search is quite standard and consists of choosing at each iteration the best non-tabu solution in the neighborhood of the current solution. This neighborhood is created through an exchange procedure, called CROSS exchange, that swaps sequences of consecutive customers between two routes. This operator generalizes both the 2-opt* (Potvin and Rousseau [72]) and Or-opt (Or [67]) exchanges, but is a special case of the λ-interchanges (Osman [68]) since it restricts the subsets of customers chosen in each route to be consecutive. To optimize individual routes, the neighborhood is enlarged by including CROSS exchanges that apply to a single route: two edges are removed from a route, and the segment between the two edges is moved at another location within the same route.

Whereas most tabu search heuristics are based on a two-phase approach in which an improvement procedure is invoked after an initial solution has been completely constructed, a metaheuristic embedding reactive tabu search in the parallel construction approach of Russell [76] was developed by Chiang and Russell [18]. Reactive tabu search was first proposed by Battiti and Tecchiolli [6] and consists of dynamically varying the size of the tabu list during the search process: the list size is increased if identical solutions occur too frequently, and is decreased if no feasible solution can be found because all feasible moves are tabu. Using various customer ordering rules and criteria for measuring the best insertion points, the procedure first constructs six different solutions by gradually inserting customers and calling the tabu search heuristic on the partial solutions. The best of these solutions is then used as a starting point for the final call to the heuristic. In all steps, the λ-interchange mechanism (Osman [68]) is used to generate the neighborhood and two types of exchanges are allowed: switch one customer from one route to another and exchange two customers that belong to different routes. A very similar approach, embedding a tabu list enhanced simulated annealing algorithm within a parallel construction procedure, was also proposed by the same authors (Chiang and Russell [17]).

Other tabu search heuristics for the VRPTW were also developed by Carlton [16], Potvin *et al.* [70] and Brandão [15]. Recently, Cordeau, Laporte, and Mercier [25] introduced a tabu search heuristic that generates a single initial solution and applies a very simple exchange procedure for a predetermined number of iterations. An exchange removes a chosen customer from its current route and inserts it into the route of a different vehicle by using a least-cost insertion criterion. When the search terminates, exchanges within the routes of the best identified solution are performed by a post-optimizer that uses a specialized TSPTW heuristic (Gendreau *et al.* [43]). A diversification mechanism based on solution attributes is used to ensure a broad exploration of the solution space. The heuristic was also enhanced to deal with different extensions of the VRPTW. Specifically, using the algorithmic framework previously proposed by Cordeau, Gendreau and

Laporte [23] for the Periodic VRP and the Multi-Depot VRP, the authors derived a unified tabu search procedure capable of handling these generalizations of the VRPTW. The heuristic was adapted to these environments by introducing a new type of exchange that modifies the combination of visit days or the depot assigned to a customer. In addition, Cordeau and Laporte [24] showed that the Site Dependent VRPTW can also be solved using the same methodology. In the latter problem, several types of vehicle are available and compatibility constraints restrict the choice of vehicle that can visit each customer.

Another alternative to the two-phase construction and improvement approach used in most metaheuristics is a guided local search method described by Kilby, Prosser, and Shaw [54]. The guided local search paradigm is a memory-based approach that shares similarities with tabu search, but operates by augmenting the cost function with a penalty term based on how near the search moves to previously visited local minima, thus encouraging diversification. The method is used to drive a local search heuristic that modifies the current solution by performing one of four moves: 2-opt exchanges within a route, switching a customer from one route to another, exchanging customers that belong to two different routes, and swapping the ends of two routes. Instead of building an initial solution with a complex procedure, all customers are first assigned to a virtual vehicle whereas the routes for the actual vehicles are left empty. Because a penalty is associated with not performing a visit, a feasible solution will be constructed in the process of minimizing cost. The guided local search algorithm starts from this solution and performs a series of moves until a local minimum is reached. The objective function is then changed by adding a term that penalizes the presence of the arcs used in the solution. The search simply iterates by finding new local minima and accumulating more penalties until a stopping criterion is met.

Metaheuristics combining genetic algorithms, simulated annealing and tabu search were proposed by Thangiah, Osman, and Sun [90]. Initial solutions for the metaheuristics are obtained by either the Push-Forward insertion method (Solomon [83]) or a sectoring heuristic based on genetic algorithms. This heuristic first clusters the customers using the genetic algorithm and then routes the customers within each sector using a cheapest insertion method. At each iteration, the crossover operator exchanges a randomly selected portion of the sector divisions between selected individuals to produce offsprings for the next generation. The simulated annealing algorithm starts from an initial solution produced by either of these methods and tries to identify an improved solution at each iteration using the λ-interchange mechanism of Osman [68]. In order to diversify the search process and avoid moves that result in cycles, the simulated annealing algorithm is in fact combined with tabu search, and moves are thus selected at each iteration from a list of non-tabu candidates. The search process allows for intermediate infeasible solutions by using an objective function that imposes penalties on capacity and time window constraint violations. The authors also compared these metaheuristics with a less sophisticated local search descent method with moves selected from the set of λ-interchanges.

Homberger and Gehring [48] proposed two evolution strategies for the VRPTW. Like genetic algorithms, evolution strategies belong to the class of evolutionary algorithms, and both methods manipulate populations of individuals that represent solutions to an optimization problem. However, evolution strategies do not encode individuals. Instead, the evolution process is simulated on problem solutions and the search operators manipulate these solutions directly. The two solution methods described by the authors are based on the popular (μ, λ) evolution strategy. Starting from a population $P(t)$ with μ individuals, subsets of individuals are randomly selected and recombined to yield a total of $\lambda > \mu$ offsprings. Each offspring is then subjected to a mutation operator and the μ most fitted offsprings are finally chosen to form the new population $P(t+1)$. The fitness of an individual is normally proportional to the objective function value of the corresponding solution. Since the parents are not involved in the selection process, deterioration may occur during the evolution and the search may thus escape from a local optimum.

In the first method, new individuals are generated directly through mutations and no recombinations take place. Mutations are obtained by performing one or several moves from the families of Or-opt (Or [67]), 2-opt* (Potvin and Rousseau [72]) and λ-interchanges (Osman [68]). In the second method, offsprings are generated through a two-step recombination procedure in which three individuals are involved. To initialize both methods, the individuals of a starting population are generated by means of a stochastic approach based on the savings algorithm of Clarke and Wright [20]. Throughout the evolution, the fitness criterion first discriminates individuals by the number of vehicles used, and then by total distance traveled. One important drawback of this approach is that the two methods tend to produce solutions of inconsistent quality from one test instance to another. As a result, choosing between the two strategies is very difficult and both methods should be used to ensure that a good quality solution is obtained for any given instance. Related work on genetic algorithms has been conducted by Potvin and Bengio [69], Blanton and Wainwright [9] and Thangiah and Petrovic [89].

To date, these metaheuristics have produced excellent quality solutions but also have to contend with two main difficulties. First, they are very time consuming in comparison with local search heuristics. Second, finding appropriate transformations that change a current feasible solution into another is a challenge. This is relatively simple for the classical vehicle routing problem (see the survey paper by Golden *et al.* [45]) as well as for the VRPTW, but it becomes extremely difficult for most extensions encountered in real-world applications, such as multiple depots, heterogeneous fleet of vehicles, driver work rule restrictions, and others. An interesting application of tabu search to a real-world problem is described in Semet and Taillard [81].

Parallel Implementations. This line of research has been followed to explore whether tabu search methods retain solution quality when computing time is truncated. Parallelization consists of partitioning the neighborhood among several processors. The results of their searches are fedback to a master processor which, in turn, supplies them with fresh information. Schulze and Fahle [80] report encouraging results. Badeau *et al.* [4] examine a parallel implementation of the heuristic by Taillard *et al.* [87]. The authors conclude that parallelization of the sequential algorithm maintains solution quality, for equal computing efforts. This implies a substantial speed increase in practice.

Optimization-Based Heuristics. Koskosidis, Powell, and Solomon [62] exploit a mixed integer programming model to generalize the Fisher and Jaikumar [40] heuristic for problems with soft time windows. At each iteration, customers are assigned to vehicles by solving a Capacitated Clustering Problem. The route and schedule of each vehicle is then derived by solving the corresponding Traveling Salesman Problem (TSP) with soft time windows. The TSP solutions also generate the improved approximate clustering costs to be used at the next iteration.

Approximation methods can also be derived directly from optimization algorithms, by heuristically solving different phases of the process. More specifically, this includes partial exploration of a branch-and-bound tree. For example, one can obtain an integer solution by using a depth-first strategy and then explore the tree for the remaining available CPU time. Alternatively, elimination of branches on heuristic ground rules accelerate the decision process and may provide quite good solutions.

Asymptotically Optimal Heuristics. Such a method, called the Location Based Heuristic (LBH), is proposed by Bramel and Simchi-Levi [12] and represents another generalization of the Fisher and Jaikumar [40] approach. That is, while Koskosidis, Powell, and Solomon [62] assign customers to vehicles by solving a capacitated clustering problem, Bramel and Simchi-Levi [12] transform the VRPTW into a Capacitated Location Problem with Time Windows (CLPTW). This problem consists of determining where vehicles should be housed given a set of possible depot locations and which customers they should serve.

The constraints forcing each customer to be served by exactly one vehicle are relaxed and the resulting problem is separable by site and solved through Lagrangean relaxation. For a given set of multipliers, the solution to the Lagrangean problem provides information used to construct feasible solutions to the CLPTW and the VRPTW. By identifying each possible depot location with a customer site, the reduced costs of the N problems are used to determine seed customers and the set customers that can feasibly be associated with each seed. The cost of this solution is then compared to the cost of the best known solution and the multipliers are updated to start a new iteration. The heuristic terminates when the step size reaches a preset value. The authors use probabilistic analysis to prove that the heuristic is asymptotically optimal. Note that the LBH variant for the VRP was shown earlier to be asymptotically optimal by Bramel and Simchi-Levi [11]. Furthermore, since the LBH is an extension of the Generalized Assignment Heuristic of Fisher and Jaikumar [40], this also exhibits asymptotically optimal behavior (see [13]).

3 Lower Bounds from Decomposition Approaches

In this section, we present two decomposition approaches that derive lower bounds for the VRPTW. Other work on optimization methods includes the early papers by Christofides, Mingozzi and Toth [19] and Kolen, Rinnooy Kan, and Trienekens [60]. Their methods are based on dynamic programming and state space relaxation.

3.1 Lagrangean Relaxation

Lagrangean relaxation is a popular decomposition approach that can be used for different VRPTW formulations and variants. The usual tradeoff between ease of solving the Lagrangean subproblem and the quality of the bound obtained is straightforward for the VRPTW. If the difficult time and capacity related constraints are relaxed, the resulting Lagrangean subproblem is a pure network flow problem, for which the Integrality Property holds (see Geoffrion [44]). The Lagrangean bound will then be no better than the linear programming lower bound. As discussed above, the integrality gap will generally be too large to be explored by branch-and-bound. To improve the Lagrangean bound, one should then retain the complicating constraints in the Lagrangean subproblem and relax part of the network flow constraints. Choosing these appropriately preserves a constrained shortest path structure for the Lagrangean problem. At present, this type of structure constitutes the basis of the most successful decomposition approaches for the VRPTW (Lagrangean relaxations, bundle methods, and column generation).

Specifically, given the set of multipliers $\boldsymbol{\alpha} = (\alpha_i,\ i \in N)$ associated with constraints (2) requiring that each customer be visited once, the Lagrangean subproblem $L(\boldsymbol{\alpha})$ obtained by relaxing these constraints in the objective function is defined as:

$$\begin{array}{ll} \min & \displaystyle\sum_{k\in K}\sum_{(i,j)\in A} c_{ij}x_{ijk} + \sum_{k\in K}\sum_{i\in N}\alpha_i(1-\sum_{j\in\Delta^+(i)} x_{ijk}) \\ \text{subject to} & (3)\text{–}(11). \end{array}$$

This subproblem involves a modified objective function and constraint sets (3)–(11). That is, the path constraints (3)–(5); constraint set (6) and the time window constraints (7)–(8), which together ensure the feasibility of the time schedule; also constraints (9), which guarantee capacity availability; and the binary requirements (11) on the flow variables. An appealing property of this structure is that it can be decomposed into $|K|$ disjoint subproblems, one for each vehicle. Furthermore, each subproblem represents an elementary shortest path problem with capacity and time window constraints, whose solution can be obtained on a bounded polyhedron.

For any multiplier vector $\boldsymbol{\alpha}$, the optimal objective function value of the Lagrangean subproblem $L(\boldsymbol{\alpha})$ is a (dual) lower bound for the solution of the respective VRPTW problem. In addition, when all vehicles are identical, only one subproblem needs to be solved to compute this bound. The problem of finding the Lagrangean bound L defined as:

$$L = \max_{\alpha} L(\boldsymbol{\alpha}),$$

is a concave non-differentiable maximization problem. Subgradient and bundle methods (Kohl and Madsen [59]) have been applied to determine optimal multiplier values. Due to the time window and capacity constraints, the subproblems do not possess the Integrality Property. Consequently, solving them as integer programs narrows the integrality gap between the optimal solution of the linearized version of formulation (1)–(10) and the optimal integer VRPTW solution to (1)–(11).

The Capacity and Time Constrained Shortest Path Problem. The elementary version of this problem mentioned above is NP-hard and no polynomial or pseudo-polynomial algorithms are known for its solution. However, when non-elementary path solutions are allowed, i.e., solutions where paths may involve cycles of finite duration and/or load due to the time window and capacity restrictions present in the subproblem, pseudo-polynomial algorithms have been developed for its solution (see Desrosiers *et al.* [35]).

The inclusion of non elementary paths is a computational necessity that potentially weakens the lower bound obtained. However, some strength in the bound can be regained by using a 2–cycle elimination procedure (Houck *et al.* [49]; Kolen, Rinnooy Kan, and Trienekens [60]) within the solution process for the constrained shortest path problem. Note that a 2-cycle is a cycle where a customer is visited twice with only one customer in between. Yet, paths containing cycles cannot appear in any solution to the VRPTW since the covering constraints (2) enforce that each customer must be visited exactly once. Hence, they have to be eliminated during the search for integer solutions.

In addition to the above schemes, Fisher, Jörnsten and Madsen [41] also used a Lagrangean relaxation based on a K–tree structure, where K is the set of available vehicles. This is an extension of the classical 1-tree approach for the traveling salesman problem to the case of capacity constrained vehicles (Fisher [39]). In their approach it is assumed that each route contains at least two customers. The authors relax the flow conservation as well as the capacity and time constraints. Vehicle capacity is handled by introducing constraints requiring that some nonempty subsets of customers $S, S \subset N$, $|S| \geq 2$, must be serviced by at least $\kappa(S)$ vehicles, that is:

$$\sum_{k \in K} \sum_{i \in \overline{S}} \sum_{i \in S} x_{ijk} \geq \kappa(S),$$

where $\overline{S} = V \setminus S$. Time windows are treated similarly: if the path (not necessarily from node 0 to $n+1$) represented by the set of arcs $A' \subset A$ violates the time window restrictions, the constraint:

$$\sum_{k \in K} \sum_{(i,j) \in A'} x_{ijk} \leq |A'| - 1$$

is generated and Lagrangean relaxed. New capacity and time constraints are generated as they are violated.

Variable Splitting. Generally, this leads to various Lagrangean relaxation schemes, each one exploiting different solvable structures. In this dual approach, the variables in some of the constraints are renamed. New constraints, coupling the original and the new variables, are introduced and Lagrangean relaxed. This decomposes the problem into two or more independent problems. In the VRPTW, the sums:

$$\sum_{j \in \Delta^+(i)} x_{ijk}, \quad \forall i \in N, \quad \forall k \in K,$$

are replaced by the integer variables y_{ik} in some constraints. One may think of each such variable as the number of times customer i is serviced by vehicle k. The new constraints:

$$\sum_{j \in \Delta^+(i)} x_{ijk} \;=\; y_{ik}, \quad \forall i \in N, \quad \forall k \in K,$$

are introduced and Lagrangean relaxed. The resulting Lagrangean subproblem now decomposes into two problems, one in the y_{ik} variables and one in the flow, time and capacity variables.

For the VRPTW, it is natural to decompose the problem into a semi-assignment type problem, defined using the new variables in constraint set (2) and solved by inspection, and a set of shortest path problems with capacity and time constraints, one for each available vehicle. In this case, variable splitting does not allow for any improvement of the Lagrangean lower bound since the semi-assignment problem possesses the Integrality Property. The capacity constraints can alternatively be considered in the semi-assignment problem, yielding a generalized assignment problem. In conjunction with the time window constrained shortest path problem, this may result in a theoretical improvement of the Lagrangean bound, unfortunately unobserved in practice. Halse [46] has implemented this latter decomposition approach. Finally, the reader can find an analysis of the quality of the bounds obtainable from Variable Splitting for the VRPTW in Kohl [57].

3.2 Column Generation

The column generation approach to be described in this section represents a generalization of the Dantzig-Wolfe decomposition (Dantzig and Wolfe [28]). It has successfully been applied to the VRPTW by Desrochers, Desrosiers, and Solomon [33] and Kohl *et al.* [58]. We would like to emphasize the importance of presenting the decomposition process in its entirety starting from the multi-commodity network flow formulation rather than directly formulating the problem as a Set Partitioning problem on which column generation is applied. Indeed, this clearly illustrates how to exploit the multi-commodity network flow model to devise efficient branching and cutting strategies compatible with the column generation approach in order to obtain integer solutions as discussed in Section 4.

The decomposition is based on two structures: a master problem and a subproblem. The master problem retains constraint sets (1)–(2), and (11), i.e., the objective function, the covering of each customer exactly once, and the binary requirements on the flow variables. The subproblem involves a modified objective function, to be detailed later, and constraint sets (3)–(11). Again, it decomposes into $|K|$ independent subproblems, each being an elementary shortest path problem with capacity and time window constraints.

The Set Partitioning Formulation. The master problem can be reformulated to highlight a Set Partitioning structure. To see this, consider the process of solving the relaxed subproblem that generates elementary paths and possibly paths containing finite cycles. Each such path p can be described using *integer* flow values $\hat{x}_{ijkp}$, $(i,j) \in A$. Let Ω be the path set. Then, for a given $k \in K$, any solution x_{ijk} to the master problem can be expressed as a non-negative convex combination of paths:

$$x_{ijk} = \sum_{p\in\Omega} \hat{x}_{ijkp}\theta_{kp}, \ \forall (i,j) \in A, \qquad \sum_{p\in\Omega} \theta_{kp} = 1, \quad \theta_{kp} \geq 0, \ \forall p \in \Omega.$$

Define now parameter c_{kp} as the cost of path p for vehicle k. Let also the non-negative integer constant a_{ikp} indicate the number of times customer i is visited by vehicle k on path p. Formally:

$$c_{kp} = \sum_{(i,j)\in A} c_{ij}\hat{x}_{ijkp}, \qquad \forall k \in K, \ \forall p \in \Omega,$$

$$a_{ikp} = \sum_{j\in\Delta^+(i)} \hat{x}_{ijkp}, \qquad \forall i \in N, \ \forall k \in K, \ \forall p \in \Omega.$$

Substituting these expressions into (1)–(2) and (11), and rearranging the summation order expresses the master problem as a Set Partitioning structure:

$$\min \sum_{k\in K} \sum_{p\in\Omega} c_{kp}\theta_{kp} \tag{12}$$

subject to

$$\sum_{k\in K} \sum_{p\in\Omega} a_{ikp}\theta_{kp} = 1, \qquad \forall i \in N \tag{13}$$

$$\sum_{p\in\Omega} \theta_{kp} = 1, \qquad \forall k \in K \tag{14}$$

$$\theta_{kp} \geq 0, \qquad \forall k \in K, \ \forall p \in \Omega \tag{15}$$

$$x_{ijk} = \sum_{p\in\Omega} \hat{x}_{ijkp}\, \theta_{kp}, \qquad \forall k \in K, \ \forall (i,j) \in A \tag{16}$$

$$x_{ijk} \text{ binary}, \qquad \forall k \in K, \ \forall (i,j) \in A. \tag{17}$$

In (14), the coefficient of θ_{kp} is equal to 1, for all $k \in K$ and $p \in \Omega$. Indeed, this constraint corresponds to (3) or to (5) in the original formulation, i.e.,

$$\sum_{j\in\Delta^+(0)} x_{0jk} = \sum_{i\in\Delta^-(n+1)} x_{i,n+1,k} = 1, \qquad \forall k \in K.$$

A Lower Bound. A (primal) lower bound on the optimal integer solution of the VRPTW model can be derived from the following bi-level solution process. At the top level, the relaxed master problem is optimized over the current subset of columns as a linear program defined by (12)–(15). At the bottom level, the subproblem looks for minimum marginal cost columns given the available cost information. If the minimum is negative, the corresponding column is sent above to be appended to (12)–(15) and this coordinating problem is solved again. Otherwise, the lower bound has been found as the current linear programming optimal solution. This bound has proven very effective in practice for the VRPTW and many other vehicle routing and crew scheduling environments (Desrosiers *et al.* [35]). Recently, it was shown to be asymptotically optimal by Bramel and Simchi-Levi [14], which explains in part its performance.

This bound is equal to the previously defined Lagrangean bound L. To see this, let α_i, $i \in N$, and γ_k, $k \in K$, be the dual variables associated with constraint sets (13) and (14), respectively. These are obtained by solving (12)–(15) over the current subset of columns with the simplex method. They can be used to define the marginal cost $\overline{c}_{kp}$ of path p for subproblem k as:

$$\begin{aligned}
\overline{c}_{kp} &= c_{kp} - \sum_{i \in N} \alpha_i a_{ikp} - \gamma_k \\
&= \sum_{(i,j) \in A} c_{ij} \hat{x}_{ijkp} - \sum_{i \in N} \alpha_i \Big(\sum_{j \in \Delta^+(i)} \hat{x}_{ijkp} \Big) - \gamma_k \Big(\sum_{j \in \Delta^+(0)} \hat{x}_{0jk} \Big) \\
&= \sum_{(i,j) \in A \,:\, i \in N} (c_{ij} - \alpha_i) \hat{x}_{ijkp} + \sum_{(0,j) \in A} (c_{0j} - \gamma_k) \hat{x}_{0jkp}.
\end{aligned}$$

In turn, the marginal cost $\overline{c}_{ij}$, $(i,j) \in A$, of an arc can then be expressed as:

$$\overline{c}_{ij} = \begin{cases} c_{ij} - \alpha_i & \text{if} \quad i \in N, \\ c_{ij} - \gamma_k & \text{otherwise.} \end{cases}$$

The marginal cost column minimization problem over the set Ω can now be formulated as:

$$\begin{aligned}
\min \quad & \sum_{k \in K} \sum_{(i,j) \in A} \overline{c}_{ij} x_{ijk} \\
\text{subject to} \quad & (3)\text{–}(11).
\end{aligned}$$

This optimization problem is equivalent to solving the Lagrangean subproblem $\mathrm{L}(\boldsymbol{\alpha})$ defined in section 3.1.

A set of negative marginal cost paths is generated every time the subproblem is solved by dynamic programming. At every iteration but the last, this set generally has a fairly high cardinality. This observation forms the basis for accelerating the solution of the linear relaxation of the master problem, i.e., the linear program (12)–(15), by selecting several columns simultaneously. Moreover, node disjoint paths can be selected by using a greedy algorithm. Such choices replicate the structure of integer solutions and often prove beneficial downstream in the branching phase. The current best dual lower bound can also be used at branching nodes to stop the iterative process before reaching the optimality criteria. This diminishes the tailing-off effect experienced by column generation methods for linear programming settings.

Commodity Aggregation. When all vehicles are identical, as is the case for the generic VRPTW, the linear relaxation of the master problem admits a commodity independent formulation. This commodity aggregation results in a single subproblem and allows the master problem to be formulated with fewer variables and constraints. The commodity independent formulation is equivalent to the classical linear relaxation of the set partitioning formulation with an additional limit placed on the number of vehicles used. Indeed, index k can be removed from parameters c_{kp} and a_{ikp}. We then aggregate the convex combination constraints (14) by letting:

$$\theta_p = \sum_{k \in K} \theta_{kp}, \quad \forall p \in \Omega.$$

This results in $\sum_{p \in \Omega} \theta_p = |K|$, making index k unnecessary for (12)–(13) and resulting in the formulation:

$$\min \sum_{p \in \Omega} c_p \theta_p \tag{18}$$

subject to

$$\sum_{p \in \Omega} a_{ip} \theta_p = 1, \qquad \forall i \in N \tag{19}$$

$$\sum_{p \in \Omega} \theta_p = |K|, \tag{20}$$

$$\theta_p \geq 0, \qquad \forall p \in \Omega \tag{21}$$

$$\theta_p = \sum_{k \in K} \theta_{kp}, \qquad \forall p \in \Omega \tag{22}$$

$$\sum_{p \in \Omega} \theta_{kp} = 1, \qquad \forall k \in K \tag{23}$$

$$\theta_{kp} \geq 0, \qquad \forall k \in K,\ \forall p \in \Omega \tag{24}$$

$$x_{ijk} = \sum_{p \in \Omega} \hat{x}_{ijkp}\, \theta_{kp}, \qquad \forall k \in K,\ \forall (i,j) \in A \tag{25}$$

$$x_{ijk} \text{ binary}, \qquad \forall k \in K,\ \forall (i,j) \in A. \tag{26}$$

Relaxing the binary requirements, also eliminates constraints (25) which become irrelevant. For any fractional θ_p-solution to (18)–(21), there exists a solution in θ_{kp} that satisfies (22)–(24). Setting

$$\theta_{kp} = \frac{\theta_p}{|K|}, \quad \forall k \in K,\ \forall p \in \Omega,$$

provides such a solution. Consequently, since any solution consisting of the aggregated variables θ_p, $p \in \Omega_p$, can be converted into a solution in terms of the disaggregated variables

θ_{kp}, $k \in K$, $p \in \Omega$, problem (18)–(21) can be used as the linear relaxation of the master problem.

In the case where the solution of the aggregated linear relaxation of the master problem is integer, it is easy to convert it to a binary solution in terms of the variables θ_{kp}. One has to simply assign the first solution path to the first vehicle, the second path to the next vehicle, and so on. Finally, if the aggregated solution is mixed integer, both of the above conversion processes need to be applied accordingly.

A Hybrid Approach. Recently Kallehauge [51] and Kallehauge, Larsen, and Madsen [52] implemented a hybrid approach combining the Lagrangean relaxation approach used by Kohl and Madsen [59] with the generalized Dantzig-Wolfe decomposition of Desrochers, Desrosiers, and Solomon [33] and Kohl *et al.* [58]. In the first phase, Lagrangean relaxation is used in order to take advantage of the faster multiplier convergence and the easier subproblems. Then, in phase two, the authors switch to Dantzig-Wolfe decomposition and use the columns found in phase one to initiate the procedure.

4 Integer Solutions

Usually, to solve the original multi-commodity flow formulation (1)–(11) optimally, one has to make branching and cutting decisions on the binary flow variables, and on time variables when their integrality is required. The decomposition process involving Lagrangean relaxation or column generation is then repeated at each branching node. Since the solutions obtained from the Lagrangean subproblem $\mathrm{L}(\boldsymbol{\alpha})$ of Section 3.1 define paths that are usually not feasible for the whole problem, it is rather difficult to design good branching and cutting strategies. Alternatively, column generation offers much more flexibility since the values of the original variables of the multi-commodity flow model can be easily derived.

Specifically, these can be divided into path exclusive and path shared decisions. The former concern only a single path, such as fixing a flow variable at 0 or at 1, or dividing the time window of a time variable. These local decisions are made directly on the subproblem network without altering the shortest path solution approach. The columns that no longer satisfy a branching decision are removed from the current master problem. The latter decisions concern several paths, such as when an integer cut on the total cost is imposed. These global decisions are kept at the master problem level. We now present several examples of branching and cutting decisions on arc flow and time variables for the VRPTW. Additionally, we discuss the possibility of making binary decisions on path flow variables.

Binary Decisions on Arc Flow Variables. Since any customer i must be covered exactly once, the linear combinations of flow variables:

$$x_{iJK'} = \sum_{k \in K'} \sum_{j \in J} x_{ijk}, \quad \forall i \in N,\ \forall J \subseteq \Delta^+(i),\ \forall K' \subseteq K,$$

are good candidates for binary branching decisions. When $x_{iJK'}$ is fractional at the current branching node, it can be set to 1 on one branch and to 0 on the other. In the former case, $x_{iJK'} = 1$ only requires that some customer (or the depot) in subset J be visited immediately after customer i by some vehicle. However, $x_{ijk} = 1$ forces customer i and j to be consecutively serviced by vehicle k. Similarly, if $|J| = 1$, then this customer must immediately succeed customer i. Finally, when $J = \Delta^+(i)$, the decision $x_{iJK'=1}$ assigns customer i to a vehicle in subset K'. In particular, if $|K'| = 1$, then this vehicle must service customer i. All the branching decisions discussed above do not affect the mathematical structure of the constrained shortest path subproblem. As an example, setting the variable x_{ijK}, with $j \in N$, to 1, eliminates the arcs $\{(i,j') \in A \, : \, j' \neq j\}$ and $\{(i',j) \in A \, : \, i' \neq i\}$ from network G. Or, fixing variable x_{ijK} at 0, allows arc $(i,j) \in A$ to be deleted from network G.

Integer Decisions on Arc Flow Variables. While a number of other viable flow variable mixes to be used for branching and cutting can easily be accommodated at the subproblem level, others cannot. As an illustration of linear combinations that need to be addressed at the master problem level, consider the one that calculates the number of vehicles routed in subset K':

$$x_{0JK'} = \sum_{k \in K'} \sum_{j \in J} x_{0jk}, \quad K' \subseteq K, \; J = \Delta^+(i) \setminus \{n+1\}.$$

When the value of this variable $\hat{x}_{0JK'}$ is fractional, branching forces it to values either less than or equal to $\lfloor \hat{x}_{0JK'} \rfloor$, or greater than or equal to $\lceil \hat{x}_{0JK'} \rceil$, respectively. As another example, in a problem where the minimum number of vehicles is sought, a cut on the variable x_{0JK}, $J = \Delta^+(i) \setminus \{n+1\}$, can be introduced when this is fractional. Yet another instance occurs when the objective function has integer cost coefficients but its current value is fractional. Then:

$$\sum_{k \in K} \sum_{(i,j) \in A} c_{ij} x_{ijk} \geq \lceil \sum_{k \in K} \sum_{(i,j) \in A} c_{ij} \hat{x}_{ijk} \rceil,$$

is a valid cut. This is a specific case of the family of cuts that can be described as a weighted sum of the flow variables:

$$\sum_{k \in K} \sum_{(i,j) \in A} b_{ij} x_{ijk} \geq b, \tag{27}$$

where b_{ij}, $(i,j) \in A$, and b are unrestricted parameters. Applying the decomposition process to the above constraint results in an equivalent constraint in the master problem, written as:

$$\sum_{p \in \Omega} b_p \theta_p \geq b,$$

where $b_p = \sum_{(i,j) \in A} b_{ij} \hat{x}_{ijkp}, \forall k \in K, \forall p \in \Omega$, is the contribution of path p to constraint (27). Denoting by β its associated dual variable, it is fairly easy to show that the marginal cost $\bar{c}_{ij}$, $(i,j) \in A$, of an arc becomes:

$$\overline{c}_{ij} = \begin{cases} c_{ij} - \alpha_i - \beta b_{ij} & \text{if} \quad i \in N, \\ c_{ij} - \gamma_k - \beta b_{ij} & \text{otherwise.} \end{cases}$$

Binary Decisions on Path Flow Variables. It is easy to show that replacing in the aggregated set partitioning formulation (18)–(26) the binary requirements (26) by:

$$\theta_p \text{ binary}, \quad \forall p \in \Omega \tag{28}$$

yields a formulation equivalent to the multi-commodity network flow formulation (1)–(11). Using the simplifications presented in the last paragraphs of Section 3.2, this new formulation can be restricted to (18)–(21) and (28). This transformation opens up the possibility of defining branching decisions on the binary path flow variables θ_p, $p \in \Omega$.

On the one hand, when such a variable takes a fractional value, it is very simple to set it to 1 by adjusting the right-hand side of constraint (20), removing the covering constraints (19) associated with the customers covered by the corresponding path, and removing the nodes associated with these customers in the subproblem network. Such a decision simplifies the problem without altering its structure. On the other hand, as mentioned in several papers, it is much more difficult to impose the alternate decision, that is, to set to 0 a fractional path flow variable. Indeed, in this case, one must ensure that the corresponding path will not be generated again by the subproblem. Forbidding the generation of specific paths modifies the nature of the subproblem and requires the use of a different dynamic programming algorithm for solving it.

One possibility would be to use a k-shortest path algorithm, where k is set to the number of forbidden paths plus one. However, such an algorithm has not yet been proposed in the literature when time window constraints are considered. Another possibility consists of using a dynamic programming algorithm for time constrained shortest path problems (for example, that of Soumis and Desrochers [34]) coupled with a pre-labeling procedure, such as the one proposed by Arunapuram, Mathur, and Solow [3]. A pre-label is defined for each node of each forbidden path except for the last node. This pre-label represents the part of the forbidden path from the source node to the node associated with the pre-label, and contains additional information that forbids the extension of this label to the next node on the forbidden path.

To our knowledge, this branching strategy has yet to be tested on VRPTW instances. We conjecture that it should be useful for fixing path variables with a fractional value close to 1 so as to rapidly reduce the size of the problem without losing the exactness of the algorithm.

Subtour Elimination and 2-Path Cuts. For each nonempty subset of customers $S \subseteq N$, define the following variable to represent the flow into S:

$$x(S) = \sum_{k \in K} \sum_{i \in \overline{S}} \sum_{j \in S} x_{ijk},$$

where $\overline{S} = V \setminus S$. The usual subtour elimination constraints can be formulated as $x(S) \geq 1,\ \forall S \subseteq N, |S| \geq 2$. These can be generalized by replacing their right hand side with $\kappa(S)$, the smallest number of vehicles needed to service all customers in S. Constraint:

$$x(S) \geq \kappa(S)$$

is called a κ-path inequality since it requires that at least κ paths enter subset S in any feasible integer solution. For the VRPTW, the lower bounds obtained by considering capacity alone are unlikely to be very strong, especially when the time windows are relatively binding. Since the time constraints must be taken into account as well, it is difficult to calculate $\kappa(S)$. For this reason, Kohl *et al.* [58] have restricted their attention to subsets S satisfying:

$$1 < \hat{x}(S) < 2 \quad \text{and} \quad \kappa(S) \geq 2,$$

where $\hat{x}(S)$ denotes the value of $x(S)$ in a given solution. In other words, the authors try to identify subsets of customers S requiring at least two vehicles, but presently serviced by less than two. To determine whether $\kappa(S) = 1$ for a particular S, one needs to check if the available capacity on a single vehicle is sufficient - which can be done in linear time - and the feasibility of the corresponding TSPTW. This latter problem is NP-hard in the strong sense, but when $|S|$ is rather small, the problem is relatively easy to solve by dynamic programming (Dumas *et al.* [37]; Mingozzi, Bianco, and Ricciardelli [66]). Therefore, for such problem sizes, there is a fast, though not polynomial, algorithm to determine whether $\kappa(S) \geq 2$.

Larsen [63] devised a parallel branch-and-bound implementation of the approach used by Kohl *et al.* [58]. Further improvements proposed by the author include a forced early stop and column deletion. The forced early stop terminates the route generation process as soon as one route with negative reduced cost is returned. The idea behind this stopping criterion is that the routes generated in the initial phase are often of low quality and therefore it is profitable to cut down the execution time at this stage. The column deletion procedure deletes from the master problem any column that has not been part of a basis at a given number of branch-and-bound nodes. This reduces the time spent solving the linear relaxation of the master problem, although some routes might have to be recomputed later on. Experimental results indicated that in order to avoid having to regenerate deleted routes, column deletion should not be performed too often. Larsen [63] suggests applying it after every 20 branch-and-bound nodes. This approach was later used by Kallehauge, Larsen, and Madsen [52].

κ-Path Cuts and Parallelism. Recently, Cook and Rich [22] enhanced the above method by improving the search for κ-path inequalities, and allowing values of κ up to 6. Specifically, the authors used Karger's [53] randomized minimum-cut algorithm to generate cutting planes. Moreover, they parallelized the cutting plane generator and also the branch-and-bound, using the TreadMarks [8] distributed shared memory system. The value of $\kappa(S), S \subset V$, is derived by finding the minimum number of vehicles required in a smaller VRPTW instance. If this number is greater than $\hat{x}(S)$, then a valid κ-path inequality is generated. We discuss their computational results in Section 7.

Integer Decisions on (Fractional and Integer) Time Variables. These variables also constitute a meaningful branching tool for problems with fairly narrow time windows. To describe their handling, we first compute the start of service at customer $i \in N$ as:

$$w_i = \sum_{k \in K} \sum_{p \in \Omega} \hat{w}_{ikp} \theta_{kp}, \qquad i \in N,$$

where $\hat{w}_{ikp}$ represents the unique start of service at customer i on path p of vehicle k. If a customer i is visited more than once on path p, i.e., on a cycle, the start of service $\hat{w}_{ikp}$ is taken as the sum of all the times when service begins. Then, w_i above represents the weighted average of these times. If variable w_i is required to be integer, but presently takes the *fractional* value, $\hat{w}_i$, then two branches are created:

$$w_i \leq \lfloor \hat{w}_i \rfloor \quad \text{and} \quad w_i \geq \lceil \hat{w}_i \rceil.$$

These decisions are imposed on the subproblem network G by redefining the time window at node i. Note that this type of decision is also applicable for an *integer* value $\hat{w}_i$ obtained as a convex combination of different service times on several paths. The two branches are then given by:

$$w_i \leq \hat{w}_i - 1 \quad \text{and} \quad w_i \geq \hat{w}_i.$$

On each branch, the columns that do not satisfy the corresponding decision are removed from the current subset of master problem columns.

5 Special Cases

The following two special cases of the VRPTW have attracted attention in the literature. Both can be addressed using the exact methodology presented in the previous sections.

The Multiple Traveling Salesman Problem with Time Windows. This problem, an uncapacitated VRPTW, results from eliminating the capacity constraints (9) from formulation (1)–(11). It is also an immediate generalization of the fixed schedule problem where time windows are restricted to a single value. It has attested itself as a very rewarding model for applications in school and urban bus, ship, engine, and aircraft scheduling.

The early optimization-based heuristics of Appelgren ([1] and [2]) on ship scheduling, Levin [64] on aircraft fleet size, and Swersey and Ballard [86] on school-bus scheduling, all relied on discretizing the time windows. They contributed to the impetus for much more powerful approaches developed recently. Such exact algorithms for m-TSPTW have been developed in the context of urban bus scheduling by Bianco, Mingozzi, and Ricciardelli [7] and Desaulniers, Lavigne, and Soumis [32], and in the setting of daily aircraft scheduling by Desaulniers *et al.* [31]. The last two algorithms are variations of the column generation approach for the VRPTW presented earlier.

The Vehicle Routing Problem with Backhauls and Time Windows. We consider the variant of this problem where all deliveries must be made before any pickups take place. To show that this problem is a special case of the VRPTW, one must first define load variables l_{ik}, $i \in V$, $k \in K$, specifying the quantity already delivered by vehicle k just after servicing node i, and rewrite the capacity constraints (9) the same way as the time window constraints:

$$x_{ijk}(l_{ik} + d_j - l_{jk}) \leq 0, \qquad \forall k \in K,\ \forall (i,j) \in A \tag{29}$$

$$d_i(\sum_{j \in \Delta^+(i)} x_{ijk}) \leq l_{ik} \leq C(\sum_{j \in \Delta^+(i)} x_{ijk}), \qquad \forall k \in K,\ \forall i \in N \tag{30}$$

$$l_{0k} = 0, \qquad \forall k \in K \tag{31}$$

$$0 \leq l_{n+1,k} \leq C, \qquad \forall k \in K. \tag{32}$$

Next, one partitions N in two subsets of customers, N^D and N^P, that is, those requiring a delivery and those requiring a pickup, respectively. Then one removes from A all arcs linking a node in N^D to a node in N^P, and replaces constraint sets (30) and (32) by the following three sets of constraints:

$$\text{(7.30a)} \qquad d_i(\sum_{j \in \Delta^+(i)} x_{ijk}) \leq l_{ik} \leq C(\sum_{j \in \Delta^+(i)} x_{ijk}), \quad \forall k \in K, \forall i \in N^D$$

$$\text{(7.30b)} \qquad (C + d_i)(\sum_{j \in \Delta^+(i)} x_{ijk}) \leq l_{ik} \leq 2C(\sum_{j \in \Delta^+(i)} x_{ijk}), \quad \forall k \in K, \forall i \in N^P$$

$$\text{(7.32a)} \qquad C \leq l_{n+1,k} \leq 2C, \quad \forall k \in K,$$

where d_i denotes the quantity of load to be delivered or picked up at node i. Given these load intervals (7.30a), (7.30b) and (7.32a), as well as constraint set (29), one can see that when the delivery portion of a vehicle route is completed, C new units of loading capacity are restored to undertake pickups. Finally, note that (29) are always satisfied for cross arcs between N^D to N^P. Hence, they are not defined for them.

Given the above transformation, optimal VRPTW algorithms can then be employed for the Vehicle Routing Problem with Backhauls and Time Windows. Gélinas *et al.* [42] illustrate such an approach. More complex algorithms are however necessary when the pickup and delivery requests can be performed in any order. A real-world application for this problem structure has also been reported by Braca *et al.* [10]. Given the very large problem size, the authors used a Decision Support System based on a variation of the LBH heuristic (Bramel and Simchi-Levi [11]) to route school buses for the New York City Board of Education.

6 Extensions

In this section, we present several VRPTW extensions for which formulation (1)–(11) can be adapted or generalized. Most of the resulting models can be directly treated using Lagrangean relaxation or column generation embedded in a branch-and-bound search tree.

For the others, the same methodology applies but with more complex tools, namely specialized constrained shortest path algorithms.

Heterogeneous Fleet, Multiple-Depot, and Initial Conditions. The VRPTW model (1)–(11) can be generalized to account for vehicles of different size, for multiple depots, and even for situations requiring specific initial conditions for each vehicle. Indeed, in these settings, a specific network $G^k = (V^k, A^k)$, with its own origin and destination depot-nodes is defined for each vehicle $k \in K$, and all c_{ij} and t_{ij} parameters are indexed by k. To some extent, customer demands d_i and time windows $[a_i, b_i]$ can also depend on the servicing vehicle k.

In the presence of multiple depots and/or a heterogeneous fleet, vehicle aggregation can be performed as long as the conditions are identical for all vehicles in the same group. One constraint similar to (20) is retained for each group to describe the number of available vehicles within that group. The assignment of a route to a specific vehicle within a group can be done after the solution is obtained.

Fleet Size. Vehicle use can be taken into account by including a fixed charge c in the cost of all arcs $(0, j)$, $j \in N$. In this case, the number of vehicles utilized can be minimized by assigning a very large value to c. On the other hand, one may wish to set an upper limit κ on the number of vehicles that can be deployed. For the basic VRPTW, this can be easily imposed by defining K such that $|K| = \kappa$. However, when considering several depots or a heterogeneous fleet, the following constraint must be added to the multi-commodity network flow formulation with one network G^k per vehicle:

$$\sum_{k \in K} \sum_{j \in N^k} x_{0(k),jk} \leq \kappa,$$

where N^k denotes the set of customers compatible with vehicle k and $0(k)$ the origin depot-node of network G^k. Like the covering constraints (2), this constraint is relaxed in the objective function when using Lagrangean relaxation or remains at the master problem level in a column generation approach.

Multiple Time Windows. The definition of a single time window per customer can be extended to include multiple service options. This may necessitate changing the objective function to account for preferred service times. Multiple time windows have primarily been examined in the multi-period vehicle routing problem framework where they constitute full days. Each customer must be visited a specified number of times within the planning horizon. This problem is discussed further in the survey by Solomon and Desrosiers [85]. Note, however, that this generalization can also be treated by Lagrangean relaxation and column generation schemes which use time window constrained shortest paths as substructures.

Soft Time Windows. Recall that these constraints allow the vehicle to start service at the customer before or after its time window, respectively. As a result, the vehicle incurs additional costs. Formulation (1)–(11) can be extended to include soft time windows as in the following two scenarios. In the first, only deadlines can be violated at a cost and by a length of time limited by b'_i, $i \in N$. In this case, enlarged hard time windows $[a_i, b_i + b'_i]$, $i \in N$, are defined together with the following non-decreasing penalty functions that depend on the start of service time of vehicle k:

$$c_i(w_{ik}) = \begin{cases} 0 & \text{if } w_{ik} \in [a_i, b_i], \\ g_i(w_{ik}) & \text{if } w_{ik} \in (b_i, b_i + b'_i], \end{cases}$$

where $g_i(\cdot)$ is a positive non-decreasing function. This can be treated directly by Lagrangean relaxation or column generation with the sole modification of computing these additional costs upon the arrival at a customer node in the constrained shortest path dynamic programming algorithm (see Desaulniers *et al.* [29]).

Building on the previous instance, the second setting considers that the earliest start times can also be violated at a cost and by a length of time limited by a'_i, $i \in N$. Similarly, augmented time windows $[a_i - a'_i, b_i + b'_i]$, $i \in N$, are defined together with the following penalty functions:

$$c_i(w_{ik}) = \begin{cases} \lambda_i(a_i - w_{ik}) & \text{if } w_{ik} \in [a_i - a'_i, a_i), \\ 0 & \text{if } w_{ik} \in [a_i, b_i], \\ g_i(w_{ik}) & \text{if } w_{ik} \in (b_i, b_i + b'_i], \end{cases}$$

where λ_i is a positive constant and $g_i(\cdot)$ is again a positive non-decreasing function. This more general case can also be addressed by the proposed methodologies but requires a specialized dynamic programming algorithm developed by Ioachim *et al.* [50] which can handle linear decreasing node costs.

Time- and Load-Dependent Costs. The VRPTW can be extended to include arc costs $z_{ij}(\cdot)$, $(i,j) \in A$, that depend on time and load variables. Indeed, soft time windows can be viewed as yielding such arc costs when $j \in N$:

$$z_{ij}(w_{jk}) = c_{ij} + c_j(w_{jk}).$$

Another example is provided by Desaulniers, Lavigne, and Soumis [32] for the m-TSPTW with linear waiting costs. For that problem, the arc costs are given by:

$$z_{ij}(w_{ik}, w_{jk}) = c_{ij} + \omega(w_{jk} - w_{ik} - s_i - t_{ij}),$$

where ω is a positive constant corresponding to the cost charged for waiting one unit of time, and $w_{jk} - w_{ik} - s_i - t_{ij}$ indicates the amount of time spent waiting on arc (i,j). As mentioned by the authors, such waiting costs can be taken into account similarly in other routing problems with time windows such as the VRPTW.

Arc costs depending on load variables were considered in the extension of the Vehicle Routing Problem with Pickup and Delivery involving time window constraints proposed by Dumas, Desrosiers, and Soumis [38]. In that version of the problem, the cost of using an arc (i,j) depends on the load l_{ik} of the vehicle k traversing it:

$$z_{ij}(l_{ik}) = g_i(l_{ik})c_{ij},$$

where $g_i(\cdot)$ is a positive non-decreasing function. Such load-dependent arc costs can easily be transferred to the VRPTW.

Driver Considerations. In order to devise vehicle routes that do not incur excessive driver costs or infeasible driver schedules, some aspects of the driver scheduling problem can be considered while solving the VRPTW. For instance, assuming that each driver is assigned to a single vehicle route, the following three driver scheduling aspects are of interest: a guaranteed minimum number of hours credited per route, a maximum number of hours worked per route, and break periods of minimal duration within long routes. As shown in Desaulniers *et al.* [30], the first two aspects can be modeled using resource variables which are handled similarly to the time and load variables. The last aspect can be treated by considering a multiple stage network where each stage contains a copy of the customer nodes and a partial path through the nodes of the same stage corresponds to a partial vehicle route without break periods. The maximum duration of these partial routes is controlled through the use of resource variables. Arcs imposing a break of minimum duration are defined from the nodes of each stage to the nodes of the next stage. Other driver considerations can also be integrated into an extended VRPTW model.

Summary. The exact methodology presented in Sections 3 and 4 is general enough to effectively contend with the VRPTW as well as a wide variety of supplementary issues. In fact, as long as the extended model falls within the unified framework proposed by Desaulniers *et al.* [29], the same methodology can be applied. This is a major advantage over the heuristic methods presented in Section 2 which most of the time require substantial effort to accommodate new situations.

7 Computational Results

In this section, we review computational experience with leading algorithms. To date, the optimal algorithm of Kohl *et al.* [58] solved 70 of the 87 Solomon benchmark short horizon problems (Desrochers, Desrosiers and Solomon [33]) to optimality. Recently, four additional problems were solved by Larsen [63], and six more by Cook and Rich [22] and Kallehauge, Larsen, and Madsen [52]. In particular, the new sequential implementation by Cook and Rich [22] of Kohl *et al.*'s [58] algorithm using the 2-path cuts succeded in solving three additional problems. Their computational experience indicates that the marginal benefit of considering 3-path cuts in the sequential algorithm was an improved value of the LP relaxation in several problems. Yet, these cuts did not lead to any additional problems being solved or improvements in solution time. Three more problems were solved by using the parallel version with up to 16 processors. Several unsolved instances that exhibited attractive integrality gaps were resolved using 32 processors and increasing the maximum value of κ to 6. Three other problems were solved this way. Insight gained from this phase also led the authors to increase the time limit for the 16-processor version and solve one additional instance.

Larsen [63] was the first to provide exact solutions to any of the 81 Solomon long horizon problems. He solved 17 problems in this set. Cook and Rich [22] solved 13 additional ones, while 16 more problems were solved by Kallehauge, Larsen, and Madsen [52]. Tables 1, 2, and 3, provide the cost of the best solutions, in terms of total distance, identified by either Kohl *et al.* [58] (KDMSS), Larsen [63] (L), Kallehauge, Larsen, and Madsen [52] (KLM) or Cook and Rich [22] (CR). The column NV indicates the number of vehicles used in the solution. These solutions were computed with approximate distances obtained by multiplying the real distances by 10 and truncating the result. Hence, some routes may not satisfy all time window constraints if real distances were used.

Homberger [47] has extended the Solomon test problems to sizes of up to 1000 customers. Cook and Rich [22] and Kallehauge, Larsen, and Madsen [52] have solved seven problems with 200 customers (one r-problem and six c-problems). The latter authors have also solved to optimality two additional c-problems, one with 400 customers and the other with 1000 customers.

Several researchers have derived excellent near optimal results on Solomon's test problems. In particular, high quality solutions were obtained in reasonable computing times by the metaheuristics of Rochat and Taillard [74] and Taillard *et al.* [87]. The heuristics of Homberger and Gehring [48] were also competitive and improved several previously best known solutions. The approach of Kilby, Prosser, and Shaw [54] generated particularly good results on problems with few vehicles and long routes. Similar results were reported by Chiang and Russell [18]. Finally, Cordeau, Laporte, and Mercier [25] produced new best solutions for a number of instances and competitive results for the others, even though their metaheuristic was designed to primarily address various multi-level generalizations. Table 4 provides the best known solutions obtained by these heuristics. Distances with at least three decimal places were used. In addition, the heuristics considered a hierarchical objective function where solutions with a smaller number of vehicles and larger total distance dominate those with more vehicles and shorter distances. The authors are denoted as follows: Rochat and Taillard [74] by RT, Chiang and Russell [17] by CR2, Taillard *et al.* [87] by TBGGP, Homberger and Gehring [48] by HG, Kilby, Prosser and Shaw [54] by KPS and Cordeau, Laporte, and Mercier [25] by CLM, respectively.

The tables highlight the best known solutions that we are aware of at the time of writing this paper. As the interest in this area will only continue to grow as industry is emphasizing responsiveness, we would like researchers to help us keep current the best solutions found for the Solomon problems available on this author's web page.

Problem	NV	Distance	Authors	Problem	NV	Distance	Authors
r101.25	8	617.1	KDMSS	r201.25	4	463.3	CR+KLM
r101.50	12	1044.0	KDMSS	r201.50	6	791.9	CR+KLM
r101.100	20	1637.7	KDMSS	r201.100	8	1143.2	KLM
r102.25	7	547.1	KDMSS	r202.25	4	410.5	CR+KLM
r102.50	11	909	KDMSS	r202.50	5	698.5	CR+KLM
r102.100	18	1466.6	KDMSS	r202.100			
r103.25	5	454.6	KDMSS	r203.25	3	391.4	CR+KLM
r103.50	9	772.9	KDMSS	r203.50			
r103.100	14	1208.7	CR+L	r203.100			
r104.25	4	416.9	KDMSS	r204.25			
r104.50	6	625.4	KDMSS	r204.50			
r104.100				r204.100			
r105.25	6	530.5	KDMSS	r205.25	3	393.0	CR+KLM
r105.50	9	899.3	KDMSS	r205.50	5	690.9	L+KLM
r105.100	15	1355.3	KDMSS	r205.100			
r106.25	3	465.4	KDMSS	r206.25	3	374.4	CR+KLM
r106.50	5	793	KDMSS	r206.50			
r106.100	13	1234.6	CR+KLM	r206.100			
r107.25	4	424.3	KDMSS	r207.25	3	361.6	KLM
r107.50	7	711.1	KDMSS	r207.50			
r107.100	11	1064.6	CR+KLM	r207.100			
r108.25	4	397.3	KDMSS	r208.25	1	330.9	KLM
r108.50	6	617.7	CR+KLM	r208.50			
r108.100				r208.100			
r109.25	5	441.3	KDMSS	r209.25	2	370.7	KLM
r109.50	8	786.8	KDMSS	r209.50			
r109.100	13	1146.9	CR+KLM	r209.100			
r110.25	4	444.1	KDMSS	r210.25	3	404.6	CR+KLM
r110.50	7	697.0	KDMSS	r210.50			
r110.100	12	1068.0	CR+KLM	r210.100			
r111.25	5	428.8	KDMSS	r211.25	2	350.9	KLM
r111.50	7	707.2	CR+KLM	r211.50			
r111.100	12	1048.7	CR+KLM	r211.100			
r112.25	4	393	KDMSS				
r112.50	6	630.2	CR+KLM				
r112.100							

Table 1: Optimal (total distance) solutions on the $r1$- and $r2$-problems.

Problem	NV	Distance	Authors	Problem	NV	Distance	Authors
c101.25	3	191.3	KDMSS	c201.25	2	214.7	CR+L
c101.50	5	362.4	KDMSS	c201.50	3	360.2	CR+L
c101.100	10	827.3	KDMSS	c201.100	3	589.1	CR+KLM
c102.25	3	190.3	KDMSS	c202.25	2	214.7	CR+L
c102.50	5	361.4	KDMSS	c202.50	3	360.2	CR+KLM
c102.100	10	827.3	KDMSS	c202.100	3	589.1	CR+KLM
c103.25	3	190.3	KDMSS	c203.25	2	214.7	CR+L
c103.50	5	361.4	KDMSS	c203.50	3	359.8	CR+KLM
c103.100	10	826.3	KDMSS	c203.100	3	588.7	KLM
c104.25	3	186.9	KDMSS	c204.25	2	213.1	CR+KLM
c104.50	5	358.0	KDMSS	c204.50	2	350.1	KLM
c104.100	10	822.9	KDMSS	c204.100			
c105.25	3	191.3	KDMSS	c205.25	2	214.7	CR+L
c105.50	5	362.4	KDMSS	c205.50	3	359.8	CR+KLM
c105.100	10	827.3	KDMSS	c205.100	3	586.4	CR+KLM
c106.25	3	191.3	KDMSS	c206.25	2	214.7	CR+L
c106.50	5	362.4	KDMSS	c206.50	3	359.8	CR+KLM
c106.100	10	827.3	KDMSS	c206.100	3	586.0	CR+KLM
c107.25	3	191.3	KDMSS	c207.25	2	214.5	CR+L
c107.50	5	362.4	KDMSS	c207.50	3	359.6	CR+KLM
c107.100	10	827.3	KDMSS	c207.100	3	585.8	CR+KLM
c108.25	3	191.3	KDMSS	c208.25	2	214.5	CR+L
c108.50	5	362.4	KDMSS	c208.50	2	350.5	CR+KLM
c108.100	10	827.3	KDMSS	c208.100	3	585.8	KLM
c109.25	3	191.3	KDMSS				
c109.50	5	362.4	KDMSS				
c109.100	10	827.3	KDMSS				

Table 2: Optimal (total distance) solutions on the $c1$- and $c2$-problems.

Problem	NV	Distance	Authors	Problem	NV	Distance	Authors
rc101.25	4	461.1	KDMSS	rc201.25	3	360.2	CR+L
rc101.50	8	944	KDMSS	rc201.50	5	684.8	L+KLM
rc101.100	15	1619.8	KDMSS	rc201.100	9	1261.8	KLM
rc102.25	3	351.8	KDMSS	rc202.25	3	338.0	CR+KLM
rc102.50	7	822.5	KDMSS	rc202.50			
rc102.100	14	1457.4	CR+KLM	rc202.100			
rc103.25	3	332.8	KDMSS	rc203.25	2	356.4	KLM
rc103.50	6	710.9	KDMSS	rc203.50			
rc103.100	11	1258.0	CR+KLM	rc203.100			
rc104.25	3	306.6	KDMSS	rc204.25			
rc104.50	5	545.8	KDMSS	rc204.50			
rc104.100				rc204.100			
rc105.25	4	411.3	KDMSS	rc205.25	3	338.0	L+KLM
rc105.50	8	855.3	KDMSS	rc205.50	5	631.0	KLM
rc105.100	15	1513.7	KDMSS	rc205.100			
rc106.25	3	345.5	KDMSS	rc206.25	3	324.0	KLM
rc106.50	6	723.2	KDMSS	rc206.50			
rc106.100				rc206.100			
rc107.25	3	298.3	KDMSS	rc207.25	3	298.3	KLM
rc107.50	6	642.7	KDMSS	rc207.50			
rc107.100				rc207.100			
rc108.25	3	294.5	KDMSS	rc208.25			
rc108.50	6	598.1	KDMSS	rc208.50			
rc108.100				rc208.100			

Table 3: Optimal (total distance) solutions on the $rc1$- and $rc2$-problems.

Problem	NV	Distance	Authors	Problem	NV	Distance	Authors
r101	19	1650.80	RT	r201	4	1252.37	HG
r102	17	1486.12	RT	r202	3	1197.66	CLM
r103	13	1292.85	HG	r203	3	942.64	HG
r104	10	982.01	RT	r204	2	849.62	CLM
r105	14	1377.11	RT	r205	3	998.72	KPS
r106	12	1252.03	RT	r206	3	912.97	RT
r107	10	1113.69	CLM	r207	2	914.39	CR2
r108	9	964.38	CLM	r208	2	731.23	HG
r109	11	1194.73	HG	r209	3	910.55	HG
r110	10	1125.04	CLM	r210	3	955.39	HG
r111	10	1099.46	HG	r211	2	910.09	HG
r112	9	1003.73	HG				
c101	10	828.94	RT	c201	3	591.56	RT
c102	10	828.94	RT	c202	3	591.56	RT
c103	10	828.06	RT	c203	3	591.17	RT
c104	10	824.78	RT	c204	3	590.60	RT
c105	10	828.94	RT	c205	3	588.88	RT
c106	10	828.94	RT	c206	3	588.49	RT
c107	10	828.94	RT	c207	3	588.29	RT
c108	10	828.94	RT	c208	3	588.32	RT
c109	10	828.94	RT				
rc101	14	1696.94	TBGGP	rc201	4	1406.94	CLM
rc102	12	1554.75	TBGGP	rc202	3	1389.57	HG
rc103	11	1262.02	RT	rc203	3	1060.45	HG
rc104	10	1135.48	CLM	rc204	3	799.12	HG
rc105	13	1637.15	HG	rc205	4	1302.42	HG
rc106	11	1427.13	CLM	rc206	3	1156.26	KPS
rc107	11	1230.54	TBGGP	rc207	3	1062.05	CLM
rc108	10	1139.82	TBGGP	rc208	3	832.36	CLM

Table 4: Best known solutions identified by heuristics

8 Conclusions

In the previous sections, we have highlighted the remarkable evolution of VRPTW methodologies for the last two decades. The models and algorithms presented are the stepping stones on which progress in this area spiraled upward. Several have been successfully applied in practice.

Optimization methods have relied on the intelligent exploitation of special problem structures and have benefited from the constant advances in computing technology. Presently, optimal algorithms using branching and cutting on solutions obtained through Dantzig-Wolfe decomposition are leading the field (Kohl *et al.* [58], Kallehauge, Larsen, and Madsen [52], and Cook and Rich [22]). Their success exemplifies that valid inequalities are a compelling way to strengthen the lower bounds for the VRPTW. These advances should create further interest in solving optimally the problems with many customers per route. Nevertheless, the results reported by Cook and Rich [22] illustrate that even powerful valid inequalities coupled with parallelism are not sufficient to solve all Solomon problems. Additional research on polyhedral structures should prove valuable for this. Another direction worth pursuing involves acceleration strategies. In a different context, du Merle *et al.* [36] show that a stabilization method significantly decreases CPU time at the master problem level. It is based on the use of bounded perturbation variables (i.e., bounded slack variables) that virtually eliminate degeneracy, and estimates of dual variables that make it unnecessary to solve the problem to optimality. Its adaptation to the VRPTW would lead to more and larger problems to be solved.

Decomposition algorithms are also easily adaptable to other settings. This is because they comprise modules, such as dynamic programming, that can handle a variety of objectives. Lateness, for one, is becoming an increasingly important benchmark in today's supply chains that emphasize on time deliveries. Moreover, they can be run as optimization-based heuristics by means of early stopping criteria.

Research on approximation methods has also substantially increased in scope and maturity. Metaheuristics have led the way in generating near optimal solutions as illustrated by the results of Rochat and Taillard [74], Homberger and Gehring [48], and Cordeau, Laporte, and Mercier [25] among others. Parallelism could resolve some of the efficiency issues. Recent composite heuristics, such as that of Cordone and Wolfler Calvo [27], are also showing much promise. They provide competitive solutions while being much faster. As heuristics need to be especially effective for very large-scale problems, we expect work on these to intensify. There is also a continuing need for standardization of the computational experiments. This should involve the data used - real or integer, the degree of approximation in the travel time calculations, and the reporting of results - whether best or average values are presented. An additional step in this direction could be for authors to report the itineraries obtained. This would ensure that identical solutions do not seem unequal simply because of differences in data type or its management.

Given the success to date of both optimization and approximate methods, we envision that hybrid methods, blending aspects of each, will constitute an important direction for future research. In addition, the theoretical and practical importance of the above developments can be further appreciated by realizing that they have also constituted the backbone for much more complex models for fleet planning, crew scheduling and crew rostering problems. It is our hope that this chapter has provided valuable insights for the pursuit of solutions to many current and future challenging problems.

Acknowledgments. This research was supported by the Quebec Government (Fonds pour la Formation de Chercheurs et l'Aide à la Recherche) and by the Natural Sciences and Engineering Council of Canada. We would also like to thank Oli B.G. Madsen from the Technical University of Denmark, Lyngby, for all the fruitful discussions regarding the paper.

References

[1] L.H. Appelgren. A column generation algorithm for a ship scheduling problem. *Transportation Science*, 3:53–68, 1969.

[2] L.H. Appelgren. Integer programming methods for a vessel scheduling problem. *Transportation Science*, 5:64–78, 1971.

[3] S. Arunapuram, K. Mathur, and D. Solow. Vehicle routing and scheduling with full truck loads. Technical report, Operations Research and Operations Management, Case Western Reserve University, Cleveland, OH, 1997.

[4] P. Badeau, M. Gendreau, F. Guertin, J.-Y. Potvin, and E. Taillard. A parallel tabu search heuristic for the vehicle routing problem with time windows. *Transportation Research C*, 5:109–122, 1997.

[5] E. Baker and J. Schaffer. Computational experience with branch exchange heuristics for vehicle routing problems with time window constraints. *American Journal of Mathematical and Management Sciences*, 6:261–300, 1986.

[6] R. Battiti and G. Tecchiolli. The reactive tabu search. *ORSA Journal on Computing*, 6(2):126–140, 1994.

[7] L. Bianco, A. Mingozzi, and S. Ricciardelli. An exact algorithm for combining vehicle trips. In J.R. Daduna, I. Branco, and J. Paixão, editors, *Computer-Aided Transit Scheduling, Lecture Notes in Economics and Mathematical Systems 430*, pages 145–172. Springer-Verlag, Berlin, 1995.

[8] R. Bixby, W. Cook, A. Cox, and E. Lee. Parallel mixed integer programming. Technical report CRPC-TR95554, Center for Research on Parallel Computation, Rice University, Houston, TX, 1995.

[9] J.L. Blanton and R.L. Wainwright. Multiple vehicle routing with time and capacity constraints using genetic algorithms. In *Proceedings of the 5th International Conference on Genetic Algorithms*, pages 452–459, San Mateo, CA, 1993.

[10] J. Braca, J. Bramel, B. Posner, and D. Simchi-Levi. A computerized approach to the New York City school bus routing system. *IIE Transactions*, 29:693–702, 1997.

[11] J. Bramel and D. Simchi-Levi. A location based heuristic for general routing problems. *Operations Research*, 43:649–660, 1995.

[12] J. Bramel and D. Simchi-Levi. Probabilistic analyses and practical algorithms for the vehicle routing problem with time windows. *Operations Research*, 44:501–509, 1996.

[13] J. Bramel and D. Simchi-Levi. *The logic of logistics*. Springer-Verlag, New York, 1997.

[14] J. Bramel and D. Simchi-Levi. On the effectiveness of set covering formulations for the vehicle routing problem with time windows. *Operations Research*, 45:295–301, 1997.

[15] J. Brandão. Metaheuristic for the vehicle routing problem with time windows. In S. Voss, S. Martello, I.H. Osman, and C. Roucairol, editors, *Meta Heuristics: Advances and Trends in Local Search Paradigms for Optimisation*, pages 19–36. Kluwer, Boston, 1998.

[16] W.B. Carlton. *A tabu search approach to the general vehicle routing problem.* Ph.D. Thesis, The University of Texas at Austin, Austin, TX, 1995.

[17] W.-C. Chiang and R.A. Russell. Simulated annealing metaheuristics for the vehicle routing problem with time windows. *Annals of Operations Research*, 63:3–27, 1996.

[18] W.-C. Chiang and R.A. Russell. A reactive tabu search metaheuristic for the vehicle routing problem with time windows. *INFORMS Journal on Computing*, 9:417–430, 1997.

[19] N. Christofides, A. Mingozzi, and P. Toth. State-space relaxation procedures for the computation of bounds to routing problems. *Networks*, 11:145–164, 1981.

[20] G. Clarke and J.W. Wright. Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, 12:568–581, 1964.

[21] T. Cook and R.A. Russell. A simulation and statistical analysis of stochastic vehicle routing with timing constraints. *Decision Science*, 9:673–687, 1978.

[22] W. Cook and J. L. Rich. A parallel cutting plane algorithm for the vehicle routing problem with time windows. Technical report, Computational and Applied Mathematics, Rice University, Houston, TX, 1999.

[23] J.-F. Cordeau, M. Gendreau, and G. Laporte. A tabu search heuristic for periodic and multi-depot vehicle routing problem. *Networks*, 30:105–119, 1997.

[24] J.-F. Cordeau and G. Laporte. A tabu search algorithm for the site dependent vehicle routing problem with time windows. Technical report CRT-00-04, Centre for Research on Transportation, Montréal, Canada, 2000.

[25] J.-F. Cordeau, G. Laporte, and A. Mercier. A unified tabu search heuristic for vehicle routing problems with time windows. Technical report CRT-00-03, Centre for Research on Transportation, Montréal, Canada, 2000.

[26] R. Cordone and R. Wolfer Calvo. Note on time window constraints in routing problems. Internal report 96.005, Politechnico di Milano, Dipartimento di Electronica e Informazione, Milan, Italy, 1996.

[27] R. Cordone and R. Wolfer Calvo. A heuristic for vehicle routing problem with time windows. Internal report 97.012, Politechnico di Milano, Dipartimento di Electronica e Informazione, Milan, Italy, 1997.

[28] G. B. Dantzig and P. Wolfe. The decomposition algorithm for linear programming. *Operations Research*, 8:101–111, 1960.

[29] G. Desaulniers, J. Desrosiers, I. Ioachim, M.M. Solomon, F. Soumis, and D. Villeneuve. A unified framework for deterministic time constrained vehcile routing and crew scheduling problem. In T.G. Crainic and G. Laporte, editors, *Fleet Management and Logistics*, pages 57–93. Kluwer, Boston, 1998.

[30] G. Desaulniers, J. Desrosiers, A. Lasry, and M.M. Solomon. Crew pairing for a regional carrier. In N. Wilson, editor, *Computer-Aided Scheduling of Public Transport 7*, pages 19–41. Springer-Verlag, Berlin, 1999.

[31] G. Desaulniers, J. Desrosiers, M.M. Solomon, and F. Soumis. Daily aircraft routing and scheduling. *Management Science*, 43:841–855, 1997.

[32] G. Desaulniers, J. Lavigne, and F. Soumis. Multi-depot vehicle scheduling with time windows and waiting costs. *European Journal of Operational Research*, 111:479–494, 1998.

[33] M. Desrochers, J. Desrosiers, and M.M. Solomon. A new optimization algorithm for the vehicle routing problem with time windows. *Operations Research*, 40:342–354, 1992.

[34] M. Desrochers and F. Soumis. A generalized permanent labeling algorithm for the shortest path problem with time windows. *INFOR*, 26:191–212, 1988.

[35] J. Desrosiers, Y. Dumas, F. Soumis, and M.M. Solomon. Time constrained routing and scheduling. In M.O. Ball et al., editor, *Network Routing, Handbooks in OR & MS*, pages 35–139. Elsevier Science, Amsterdam, 1995.

[36] O. du Merle, D. Villeneuve, J. Desrosiers, and P. Hansen. Stabilized column generation. *Discrete Mathematics*, 194:229–237, 1999.

[37] Y. Dumas, J. Desrosiers, E. Gélinas, and M.M. Solomon. An optimal algorithm for the traveling salesman problem with time windows. *Operations Research*, 43:367–371, 1995.

[38] Y. Dumas, J. Desrosiers, and F. Soumis. The pickup and delivery problem with time windows. *European Journal of Operational Research*, 54:7–22, 1991.

[39] M.L. Fisher. Optimal solution of the vehicle routing problems using minimum k-trees. *Operations Research*, 42:626–642, 1994.

[40] M.L. Fisher and R. Jaikumar. A generalized assignment heuristic for vehicle routing. *Networks*, 11:109–124, 1981.

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

[42] S. Gélinas, M. Desrochers, J. Desrosiers, and M.M. Solomon. A new branching strategy for time constrained routing problems with application to backhauling. *Annals of Operations Research*, 61:91–109, 1995.

[43] M. Gendreau, G. Hertz, G. Laporte, and M. Stan. A generalized insertion heuristic for the traveling salesman problem with time windows. *Operations Research*, 43:330–335, 1998.

[44] A.M. Geoffrion. Lagrangean relaxation for integer programming. *Mathematical Programming Study*, 2:82–114, 1974.

[45] B.L. Golden, E.A. Wasil, J.P. Kelly, and I-Ming Chao. Metaheuristics in vehicle routing. In T.G. Crainic and G. Laporte, editors, *Fleet Management and Logistics*, pages 33–56. Kluwer, Boston, 1998.

[46] K. Halse. *Modeling and solving complex vehicle routing problems.* Ph.D. Thesis, Institute of Mathematical Modelling, Technical University of Denmark, Lyngby, Denmark, 1992.

[47] J. Homberger. Extended Solomon's VRPTW instances (on web page). *http://www.fernuni-hagen.de/WINF/touren/inhalte/probinst.htm.*

[48] J. Homberger and H. Gehring. Two evolutionary metaheuristics for the vehicle routing problem with time windows. *INFOR*, 37:297–318, 1999.

[49] D.J. Houck Jr., J.-C. Picard, M. Queyranne, and R.R. Vemuganti. The travelling salesman problem as a constrained shortest path problem: Theory and computational experience. *Opsearch*, 17:93–109, 1980.

[50] I. Ioachim, S. Gélinas, J. Desrosiers, and F. Soumis. A dynamic programming algorithm for the shortest path problem with time windows and linear node costs. *Networks*, 31:193–204, 1998.

[51] B. Kallehauge. *Lagrangean duality and non-differentiable optimization applied on routing with time windows.* M.Sc. Thesis IMM-EKS-2000-13 [in Danish], Department of Mathematical Modelling, Technical University of Denmark, Lyngby, Denmark, 2000.

[52] B. Kallehauge, J. Larsen, and O.B.G. Madsen. Lagrangean duality and non-differentiable optimization applied on routing with time windows - experimental results. Internal report IMM-REP-2000-8, Department of Mathematical Modelling, Technical University of Denmark, Lyngby, Denmark, 2000.

[53] D.R. Karger. Global min-cuts in *rnc* and other ramifications of a simple mincut algorithm. In *Proceedings of the 4th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 84–93, ACM-SIAM, 1993.

[54] P.J. Kilby, P. Prosser, and P. Shaw. Guided local search for the vehicle routing problem with time windows. In S. Voss, S. Martello, I.H. Osman, and C. Roucairol, editors, *Meta Heuristics: Advances and Trends in Local Search Paradigms for Optimisation*, pages 473–486. Kluwer, Boston, 1998.

[55] G. Kindervater and M. Savelsbergh. Vehicle routing: Handling edge exchanges. In J. Lenstra and E. Aarts, editors, *Local Search in Combinatorial Optimization*, pages 337–360. Wiley, Chichester, 1997.

[56] K. Knight and J. Hofer. Vehicle scheduling with timed and connected calls: A case study. *Operational Research Quarterly*, 19:299–310, 1968.

[57] N. Kohl. *Exact methods for time constrained routing and related scheduling problems.* Ph.D. Thesis, Institute of Mathematical Modelling, Technical University of Denmark, Lyngby, Denmark, 1995.

[58] N. Kohl, J. Desrosiers, O.B.G. Madsen, M.M. Solomon, and F. Soumis. 2-Path cuts for the vehicle routing problem with time windows. *Transportation Science*, 33:101–116, 1999.

[59] N. Kohl and O.B.G. Madsen. An optimization algorithm for the vehicle routing problem with time windows based on Lagrangean relaxation. *Operations Research*, 45:395–406, 1997.

[60] A.W.J. Kolen, A.H.G. Rinnooy Kan, and H.W.J.M. Trienekens. Vehicle routing with time windows. *Operations Research*, 35:266–273, 1987.

[61] G. Kontoravdis and J.F. Bard. A GRASP for the vehicle routing problem with time windows. *ORSA Journal on Computing*, 7:10–23, 1995.

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

[63] J. Larsen. *Parallellization of the vehicle routing problem with time windows.* Ph.D. Thesis IMM-PHD-1999-62, Department of Mathematical Modelling, Technical University of Denmark, Lyngby, Denmark, 1999.

[64] A. Levin. Scheduling and fleet routing models for transportation systems. *Transportation Science*, 5:232–255, 1971.

[65] O.B.G. Madsen. Optimal scheduling of trucks - A routing problem with tight due times for delivery. In H. Strobel, R. Genser, and M. Etschmaier, editors, *Optimization applied to transportation systems*, pages 126–136. IIASA, International Institute for Applied System Analysis, Laxenburgh, 1976.

[66] A. Mingozzi, L. Bianco, and S. Ricciardelli. Dynamic programming strategies for the traveling salesman problem with time window and precedence constraints. *Operations Research*, 45:365–377, 1997.

[67] I. Or. *Travelling salesman-type combinatorial problems and their relation to the logistics of blood banking.* Ph.D. Thesis, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, Illinois, 1976.

[68] I.H. Osman. Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. *Annals of Operations Research*, 41:421–451, 1993.

[69] J.-Y. Potvin and S. Bengio. The vehicle routing problem with time windows - Part II: Genetic search. *INFORMS Journal on Computing*, 8:165–172, 1996.

[70] J.-Y. Potvin, T. Kervahut, B.L. Garcia, and J.-M. Rousseau. The vehicle routing problem with time windows - Part I: Tabu search. *INFORMS Journal on Computing*, 8:158–164, 1996.

[71] J.-Y. Potvin and J.-M. Rousseau. A parallel route building algorithm for the vehicle routing and scheduling problem with time windows. *European Journal of Operational Research*, 66:331–340, 1993.

[72] J.-Y. Potvin and J.-M. Rousseau. An exchange heuristic for routing problems with time windows. *Journal of Operational Research Society*, 46:1433–1446, 1995.

[73] H. Pullen and M. Webb. A computer application to a transport scheduling problem. *Computer Journal*, 10:10–13, 1967.

[74] Y. Rochat and E. Taillard. Probabilistic diversification and intensification in local search for vehicle routing. *Journal of Heuristics*, 1:147–167, 1995.

[75] R.A. Russell. An effective heuristic for the m-tour traveling salesman problem with some side conditions. *Operations Research*, 25:517–524, 1977.

[76] R.A. Russell. Hybrid heuristics for the vehicle routing problem with time windows. *Transportation Science*, 29:156–166, 1995.

[77] M.W.P. Savelsbergh. Local search in routing problems with time windows. *Annals of Operations Research*, 4:285–305, 1985.

[78] M.W.P. Savelsbergh. An efficient implementation of local search algorithms for constrained routing problems. *European Journal of Operational Research*, 47:75–85, 1990.

[79] M.W.P. Savelsbergh. The vehicle routing problem with time windows: Minimizing route duration. *ORSA Journal on Computing*, 4:146–154, 1992.

[80] J. Schulze and T. Fahle. A parallel algorithm for the vehicle routing problem with time window constraints. *Annals of Operations Research*, 86:585–607, 1999.

[81] F. Semet and E. Taillard. Solving real-life vehicle routing problems efficiently using tabu search. *Annals of Operations Research*, 41:469–488, 1993.

[82] M.M. Solomon. On the worst-case performance of some heuristics for the vehicle routing and scheduling problem with time window constraints. *Networks*, 16:161–174, 1986.

[83] M.M. Solomon. Algorithms for the vehicle routing and scheduling problem with time window constraints. *Operations Research*, 35:254–265, 1987.

[84] M.M. Solomon, E. Baker, and J. Schaffer. Vehicle routing and scheduling problems with time window constraints: Efficient implementations of solution improvement procedures. In B.L. Golden and A.A. Assad, editors, *Vehicle routing: Methods and studies*, pages 85–106. North-Holland, Amsterdam, 1988.

[85] M.M. Solomon and J. Desrosiers. Time window constrained routing and scheduling problems. *Transportation Science*, 22:1–13, 1988.

[86] A. Swersey and W. Ballard. Scheduling school buses. *Management Science*, 30:844–853, 1983.

[87] E. Taillard, P. Badeau, M. Gendreau, F. Guertin, and J.-Y. Potvin. A tabu search heuristic for the vehicle routing problem with soft time windows. *Transportation Science*, 31:170–186, 1997.

[88] É.D. Taillard. Parallel iterative search methods for vehicle routing problems. *Networks*, 23:661–673, 1993.

[89] S. R. Thangiah and P. Petrovic. Introduction to genetic heuristics and vehicle routing problems with complex constraints. In *Advances in computational and stochastic optimization, logic programming, and heuristic search, Oper. Res. / Comput. Sci. Interfaces Ser. 9*, pages 253–286. Kluwer, Boston, 1998.

[90] S.R. Thangiah, I.H. Osman, and T. Sun. Hybrid genetic algorithm, simulated annealing and tabu search methods for vehicle routing problems with time windows. Technical Report UKC/OR94/4, Institute of Mathematics and Statistics, University of Kent, Canterbury, UK, 1994.

[91] P.M. Thompson and H.N. Psaraftis. Cyclic transfer algorithms for multi-vehicle routing and scheduling problems. *Operations Research*, 41:935–946, 1993.