Available online at www.sciencedirect.com
SCIENCE DIRECT®
PERGAMON
Omega 31 (2003) 41–53
omega
The International Journal of Management Science
www.elsevier.com/locate/dsw

A problem generator-solver heuristic for vehicle routing with soft time windows

George Ioannou*, Manolis Kritikos, Gregory Prastacos

Management Sciences Laboratory, Graduate Program in Decision Sciences, Athens University of Economics and Business, 47A Evelpidon and 33 Lefkados St. 8th Floor, Athens 113-62, Greece

Received 6 April 2001; accepted 11 October 2002

Abstract

In this paper we consider the vehicle routing problem with soft time window constraints (VRPSTW), in which vehicles are allowed to service customers before and after the earliest and latest time window bounds, respectively. This relaxation comes at the expense of appropriate penalties that reflect the effect that time window violations have on the customers' satisfaction. The problem is of particular importance for fleet planning as it allows decision-makers from both the logistics and marketing-sales side to determine minimal fleet sizes by appropriate contract negotiations for order delivery times. To solve the problem, we couple the nearest-neighbour method with a problem generator that provides, in a structured manner, numerous instances that result from the manipulation of the level of time window violations and the population of customers that allow such violations. The proposed heuristic results in solutions that reduce the number of vehicles required for the hard case and provide minimal violations of time windows. Computational results on a set of benchmark problems show that our method outperforms previous approaches to the vehicle routing problem with soft time windows, and that it can serve as the basis for efficient and effective fleet planning.

Keywords: Routing; Heuristics; Logistics; Transportation; Fleet planning

1. Introduction

Typical fleet planning occurs when decision makers from both the logistics and the marketing-sales department confer in order to determine the minimum fleet size required to service the customers to which the sales department has promised deliveries within specific time windows, i.e., time bounds that constrain the earliest and the latest time that deliveries can take place. Customers have enforced the use of time windows during the last few years in all aspects of distribution, motivated by the Just-in-Time principles and the awareness of the competitiveness arising from adopting such principles, e.g., minimum inventory, reduced cycle times, just-in-time production, etc. [1]. Time windows establish hard constraints to the delivery problem, and consequently, according to well-known optimization results, the optimal solution to this problem provides an upper bound on that of the unconstrained one. Thus, due to time windows, distribution companies or manufacturers sustaining their own delivery vehicles, have to increase their fleet size in order to cope with hard time window requirements. The latter can only be affected during the sales negotiation phase, where all issues pertaining to orders and contracts take place. Providing effective tools to the sales-marketing department of a company, which allow for time window adjustments that can reduce vehicle fleet size during sales negotiation, is of paramount importance. In this paper we propose an approach to derive exactly this result, i.e., relax some time windows and reduce the vehicle fleet size compared to the fleet size

* Corresponding author. Tel.: +30-1-8203449; fax: +30-1-8828078.
E-mail address: ionnou@aueb.gr (G. Ioannou).

0305-0483/03/$ - see front matter © 2002 Elsevier Science Ltd. All rights reserved.
PII: S0305-0483(02)00064-6

required in the hard case, while keeping time window violations to a bare minimum.

The problem we address can be accurately described as follows. Consider a central depot consolidating several items that have to be delivered to a set of customers. The latter are geographically dispersed within a distance radius that allows for demand to be satisfied through daily deliveries. We assume that customer demand is known when a delivery schedule is determined, as is the distance and travel time between the depot and each customer locations, as well as between each pair of customers' locations. In addition, the time interval during which the delivery has to take place is also known (fixed when sales are finalized by the Sales and Marketing department). This interval is bounded by the earliest and latest time of the day that the delivery to a particular customer has to be completed. In contrast to the vehicle routing problem with hard time windows that should not be violated in any feasible solution, in the problem we tackle, time windows are "soft", i.e., they can be violated in the earliest or latest service times during the solution stage, by introducing appropriate penalties to reflect a measure of customer "non-satisfaction" if such violations occur. This means that when solving the problem, if a vehicle arrives too early or too late compared to the time window bounds, it may start the service of the respective customer, at the expense of a penalty that is proportional to the extent of the time window violation. The penalty is assumed to be a linear function of the amount of time window violation [2] and, in practice, includes the cost of warehousing, effect on customer satisfaction, etc.

The VRPSTW is a relaxation of the vehicle routing problem with time windows (VRPTW), which in turn, is a generalization of the classical vehicle routing problem (VRP). Because of the wide applicability in practical cases of the VRP and the VRPTW, both these problems have been exhaustively studied during the last couple of decades. The interested reader can find surveys of the published work in the areas of VRP and VRPTW composed by Golden and Assad [3], Gendreau et al. [4] and Laporte [5], concerning solution methods developed for these problems in the 1980s and 1990s. In contrast, the archival literature reveals very little published work for the vehicle routing problem with soft time windows. Below, we examine the limited research efforts for the VRPSTW.

Balakrishnan [2] formulated the VRPSTW, using a linear penalty function for each customer to define the permissible limits of the left and right violation of the customers' earliest and latest service times, respectively. The resulting model was solved using either a transformed nearest-neighbour method or a penalty-expanded savings method, which provide low-cost schedules involving fewer vehicles compared to the case of hard time windows. The author presented the computational results of the proposed methods on the example sets R101, R102, R103, R-109, RC101, RC102, RC103 and RC106 of Solomon [6]. However, the results on the remaining problem sets of Solomon [6] were not reported, and time window violations were quite severe for some of the example problems solved.

Koskosidis et al. [7] treated time windows as soft constraints that could be violated at a cost, and decomposed the optimization problem to a generalized assignment/clustering component and a series of routing and scheduling components, which were identical to travelling salesman problems with time windows. The proposed solution method was tested on the benchmark problems of Solomon [6], as well as on randomly generated problem instances. The results indicated that the algorithm compares well to simple heuristic methods for the hard case; however no extensive results were obtained to guarantee the quality of the approach for real-life large vehicle routing problems. Finally, Taillard et al. [8] applied tabu search to the VRPSTW. The method was shown to generate very good results for the hard case but no concrete computational tests were performed to illustrate the effect of "soft" time windows on the solution of the routing problem and of the resulting fleet size.

The approach proposed in this paper includes a generator of various instances of the vehicle routing problem, each of which is characterized by a certain number of customers for which the time windows can be violated. The number of such "soft" customers varies from a small percentage of the total customer population, to the cardinality of the set of customers. Each soft problem is solved using the nearest-neighbour heuristic [6], in which the customer selection and insertion criteria are transformed to account for the effect of time window violations through a penalty added to the objective function. The approach, although based on a simple solution procedure, provides results that consistently outperform previously developed methods and significantly reduces the vehicle fleet size with a small number of time window violations and an acceptable average length of time bound expansions. The implementation scheme is effective, can provide attractive alternative solutions, and can be employed as a decision-making tool at the sales and contract negotiation phase. Our key research contribution, therefore, is the mechanism for generating soft-problems and the integration of the nearest-neighbour heuristic within an iterative solution scheme, which provides better results than the previously proposed methods.

The remainder of the paper is organized as follows: In Section 2 we discuss the VRPSTW, while in Section 3 we develop the proposed solution method. Section 4 presents the computational results of the new heuristic on literature and web data sets, and compares these results with previously published work. Finally, in Section 5 we summarize our conclusions and provide pointers to further research.

2. Problem formulation

From a modelling perspective the VRPSTW can be stated as follows: Find a set of closed routes, for a fleet of $|V|$ identical vehicles (V = set of available vehicles, i.e., the

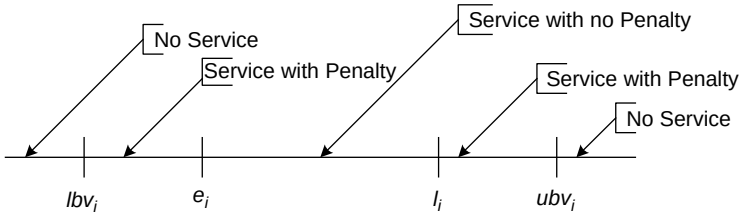

Fig. 1. Lower and upper bounds of time window violations.

maximum number of vehicles that can be used for deliveries) with known capacity C, servicing a set of customers, from a central depot at minimum cost. The number of customers is $n-1$, i.e., $|L|-1=n-1$, where L is the set of customers including the depot, which is a distinct node of the underlying connected graph. Indices i, j and u refer to customers and take values between 2 and n, while index $i=1$ refers to the depot; an additional index k counts the vehicles ($k=1,\ldots,|V|$). Vehicles are initially located at the central depot. Each customer i poses demand q_i, and is bounded by time window $[e_i, l_i]$ that models the earliest and latest time that customer i can be serviced by a vehicle. These are soft bounds that can be violated in the final solution. Furthermore, a service time, s_i, is required for each customer i. Each vehicle route originates and terminates at the central depot, while each customer is serviced by exactly one vehicle. There is a cost tc_{ij}, a travel time t_{ij} and a distance d_{ij} associated with the path from customer i to customer j. Furthermore, a cost w_k is relevant to the activation of vehicle $k \in V$; this is a one-time cost and it is related to the fixed costs for the acquisition or activation of vehicle k.

The mathematical programming formulation of the VRPTW requires two groups of variables. The first group models the sequence in which vehicles visit customers, and is defined as follows:

$$x_{ij}^k = \begin{cases} 1 & \text{if customer } j \text{ follows customer } i \text{ in the sequence visited by vehicle } k, \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

The second group of variables, z_k, are binary and defined as follows:

$$z_k = \begin{cases} 1 & \text{if vehicle } k \text{ is active,} \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

Note that a vehicle is active when it services at least one customer.

For each customer i, let a_i denote the time at which service begins, c_{ei} the unit penalty for service initiated before its earliest service time, and c_{li} the unit penalty for service initiated after its latest service time. The values of coefficients c_{ei} and c_{li} may vary according to customer importance/criticality (i.e., very large values if a customer is critical, forcing the solution to respect its time windows), and by setting c_{ei} and c_{li} equal to infinity the VRPSTW can be transformed into the hard VRPTW.

In addition, let lbv_i be a user-defined lower bound on the allowable time window violation, i.e., service may begin between lbv_i and e_i with penalty, but not before lbv_i even if the vehicle arrives at customer i before that time; in the latter case, the vehicle must wait until lbv_i to begin service. A similar bound (ubv_i) is imposed on the right side of the time window, i.e., on the latest service time; thus, a vehicle that arrives between l_i and ubv_i services the customer with a penalty, while it is not feasible to service the customer if the vehicle arrives after ubv_i. Finally, a vehicle that arrives between e_i and l_i services the customer without any penalty. The maximum limits on the allowable time window violations through lbv_i and ubv_i are imposed to contain possible large violations that could be unacceptable for customers. Fig. 1 illustrates the time bounds described above.

Each route must satisfy capacity constraints, which state that the total quantity of goods delivered cannot exceed the vehicle capacity C. The time window constraints, on the other hand, can be relaxed via appropriate penalties in order to reduce the total number of vehicles that are required. The selection and context of the penalty function is critical for the formulation and solution of the VRPSTW, since it affects the weight of time window violations and, thus, the potential for reducing the required vehicle fleet size. The formula of the penalty function is

$$P_i = \begin{cases} \infty & \text{if } a_i < lbv_i, \\ c_{ei}\,(e_i - a_i) & \text{if } lbv_i \leqslant a_i < e_i, \\ 0 & \text{if } e_i \leqslant a_i \leqslant l_i, \\ c_{li}\,(a_i - l_i) & \text{if } l_i < a_i \leqslant ubv_i, \\ \infty & \text{if } ubv_i < a_i. \end{cases} \quad (3)$$

Furthermore, we impose a maximum limit $wt_{\max}$ on the waiting time of a vehicle at any customer, to contain possible high levels of waiting times before customer service begins, i.e.

$$a_j - (a_i + s_i + t_{ij}) \leqslant wt_{\max}, \quad \text{if } x_{ij}^k = 1. \quad (4)$$

Given the above definitions, we can proceed to the formulation of the overall objective function for the VRPSTW, which should include three components: route cost, vehicle activation cost and time window violation cost. These components are obvious in the objective function of (5):

$$\text{Minimize} \quad \sum_{k=1}^{|V|}\sum_{i=1}^{n}\sum_{j=1}^{n} tc_{ij}x_{ij}^{k} + \sum_{k=1}^{|V|} w_k z_k + \sum_{i=1}^{n} P_i. \qquad (5)$$

The problem constraints are identical to those of the typical vehicle routing problem with time windows—except (4) above and the time window bounds—and are, thus, omitted here (see, e.g. [9]). The effect of time window violations can be expressed in terms of the total average time window deviation per customer, which is given by:

$$\text{TATWD} = \left(\sum_i \max\{0, e_i - a_i\} + \sum_i \max\{0, a_i - l_i\}\right) \Big/ n. \qquad (6)$$

The measure of (6) is critical since it provides an indication of the size of the time window violations.

3. Solution method

The new heuristic we propose for the solution of VRPSTW incorporates a problem instance generator and a solution engine. The problem generator repeatedly produces soft time window problem instances, feasible to the specifications of the original problem, which differ in terms of the number of customers that have soft time windows and the allowable maximum time window violation. For the solution engine, we employ the simple mechanism of the nearest-neighbour heuristic (NNH). We implement NNH, incorporating the penalty function of (3) as a factor in the customer selection criterion.

3.1. The instance generator engine

The VRPSTW is a multi-dimensional problem based on the different size violations of time windows. For example, allowing 0%, 5%, or 10% violation of the time window of customer i, we can produce three different problem sets. Also, if we allow only some customers to have soft time windows, while the remaining ones have hard time windows, we can produce various problem instances as well. To describe the problem instance generator engine we use the following definition:

Definition. A λ-*soft* problem is an instance in which the time windows of the first λ% of the customers may be violated.

Thus, a λ-*soft* problem has at least (100-λ)% customers with non-violated time windows, since at most λ% time windows may be violated in a VRPSTW solution. After generating a λ-*soft* problem, the solution engine we describe in Section 3.2 is evoked, and the resulting solution is examined. From the customers with violated time windows, the generator selects customer i that has the minimum violation, which is less than a tightness coefficient ε. The latter coefficient allows the method to distinguish between small time window violations that may be reversible, i.e., not violated without affecting the vehicle fleet size. The time window of customer i is then fixed (hard time window) and the problem instance is resolved. Thus, the generator engine allows violations for the first $\lceil n\lambda/100 \rceil$ customers and selects customer j, which satisfies the property below, for time window fixing:

$$\underset{1 \leqslant i \leqslant \lceil n\lambda/100 \rceil, a_i \notin [e_i, l_i]}{\text{Min}} \{\max\{e_i - a_i, a_i - l_i\}\} \leqslant \varepsilon. \qquad (7)$$

It is obvious that our problem generator provides a wealth of problem instances based on the values of parameters λ and ε, and allows the solution method to search for a result with minimal number of non-violated time windows. Note that the procedure of fixing the values of problem parameters for candidates that have small ε values on a given solution has been employed with excellent results in other problem areas, e.g., the fixed charge capacitated network design problem [10].

3.2. The solution engine

As our solution engine we have selected the simple and fast mechanism of the nearest-neighbour heuristic expanded with the penalty factor of (3). At every solution step, the heuristic selects customer j with the lowest cost C_j for inclusion after customer i in the route under construction. The cost C_j can be mathematically expressed as:

$$C_j = b_d t_{ij} + b_a f_{ij} + b_u g_{ij} + b_p P_j. \qquad (8)$$

In (8), the weights b_d, b_a, b_u, and b_p define the relative contribution of each individual metric to the overall selection criterion, while $b_d + b_a + b_u + b_p = 1$, and $b_d, b_a, b_u, b_p \geqslant 0$. We can define the last three sub-metrics of (8) as follows:

$f_{ij} = a_j - (a_i + s_i)$	the time difference between the completion of service at customer i and the beginning of service at customer j.
$g_{ij} = l_j - (a_i + s_i + t_{ij})$	the urgency of delivery to customer j as expressed by the time remaining until the customer's latest service time.
P_j	the penalty of time window violation for customer j.

The criterion of (8) ensures that a customer j selected for inclusion in the route under construction will be closest to the last selected customer (in terms of time and time window influence) and will result in a small time window violation penalty.

The procedure of customer insertion is repeated until no further non-routed customer can be inserted into the route under construction. In this case, a new route is initialized with a different customer and the loop is performed until all customers are assigned to routes. NNH terminates by providing the number of routes, the number of active vehicles (equal to the number of routes), the customers that are assigned to each vehicle, the sequence in which customers are visited by vehicles, the number of non-violated time windows, the cost of violations, the average of earliest time violation, the average of latest time violation, and the average of total violation. Note that in our NNH implementation, various values of the parameters b_d, b_a, b_u, and b_p are examined using a step of 0.1 for each of them.

3.3. The heuristic

For every solution, the problem generator determines a customer for which time windows should be fixed, using the Min-max criterion of (7). The revised problem is solved using NNH and the updated solution is stored. The procedure is repeated for a variety of values of the λ coefficients, starting from an initial λ and ending when $\lambda = 100$, using a parameter γ to increase λ at each loop. The heuristic method, which we entitle problem generator solver heuristic (PGSH), is as follows:

Algorithm PGSH

Step 0: *Initialization*
Read $n, \lambda, C, \varepsilon, wt_{\max}, b_d, b_a, b_u, b_p$
Read $t_{ij}, e_i, l_i, c_{ei}, c_{li}, lbv_i, ubv_i \ \forall i, j = 2, 3, 4, \ldots, n$

Step 1: *Soft Problem Generation*
Create a λ-*soft problem* using the problem generator

Step 2: *Penalty-expanded NNH Solution*
Solve the λ-*soft problem* using NNH with the criterion of (8)

Step 3: *Time window fixing*
Identify customer j according to the criterion of (7)
Set $P_j = 0$ producing a new (λ-1)-*soft problem*

Step 4: *Solution revisiting*
Solve the (λ-1)-*soft problem* using NNH with the criterion of (8)
If the number of vehicles has not increased, go to Step 3
Otherwise go to Step 5

Step 5: λ-*change*
Increase λ by a fixed value, γ ($\lambda = \lambda + \gamma$)
If $\lambda < 100$, go to Step 1
Otherwise go to Step 6

Step 6: *Terminate*
Output the following:
Number of routes
Number of non-violated time windows
Sequence of customers visited by each vehicle
Total distance (time)
Total cost
Average of total violation (TATWD)

Because the core of PGSH is the simple mechanism of NNH, the solution of the multitude of problems created by the instance generator increases the computational time of the heuristic only moderately.

4. Computational results

Our method (PGSH) was implemented on a Pentium IV, 1 GHz PC. PGSH was first tested on the classical data sets R1 and RC1 of Solomon [6]. Each data set contains problems with 100 customers. The Cartesian coordinates of customers in the R1 problems of are randomly generated from a uniform distribution, while the problems in set RC1 contain semi-clustered customers, i.e., a combination of clustered and randomly (uniformly) distributed customers. Sets R1 and RC1 have tight time windows (10 time units), short scheduling horizons (230 time units for R1 and 240 time units for RC1) and vehicle capacity $C = 200$ units, allowing few customers per route. Also, inter-customer travel time is assumed equal to the Euclidean distances. For additional information concerning the data sets, the reader is referred to the original paper of Solomon [6].

In all computational experiments we have set the maximum waiting time ($wt_{\max}$) and allowable penalty ($P_{\max}$) equal to 10% of the maximum route time allowed ($wt_{\max} = P_{\max} = 23$ for R1 and $wt_{\max} = P_{\max} = 24$ for RC1); $P_{\max}$ defines the values of the parameters lbv_i and ubv_i. Furthermore, the penalty coefficients c_{ei}, and c_{li} were set equal to 1 for each customer i, while the λ-increase parameter γ was set equal to 5 and the initial value of λ equal to 10. Finally, the values of parameter ε examined were $\{1, 2, 3, \ldots, 22, 23\}$.

Table 1 presents the results of PGSH for problems in set R1, while Table 2 those for set RC1. Three metrics for each data set are reported: (a) the number of vehicles reached by PGSH, (b) the percentage of non-violated time windows (TW) for each vehicle fleet size, and (c) the total average violation of time windows for each vehicle fleet size (TATWD). Note that we report several solutions for each problem set (e.g., for R103 we report 4 solutions with 14, 13, 12 and 11 vehicles, and the respective measures of % of non-violated time windows and TATWD). These solutions come from the various problem instances solved by PGSH. The last column of both Tables 1 and 2 includes the best solution for the respective problem reported to-date.

It is evident from the results of these two tables that PGSH determines solutions that reduce the number of routes (or of

Table 1
PGSH and best solutions for Solomon's benchmark problems—set R1

Problem	Metric	PGSH value								Best solution
R101	Number of vehicles	21	20	19	18	17	16	15	14	18
	% Non-violated TW	100	100	97	95	89	75	67	52	
	TATWD	0.0	0.0	0.1	0.3	1.0	3.7	3.8	5.3	
R102	Number of vehicles	19	18	17	16	15	14	13	12	17
	% Non-violated TW	100	100	98	93	89	81	74	62	
	TATWD	0.0	0.0	0.4	0.8	1.9	2.8	2.9	4.8	
R103	Number of vehicles	14	13	12	11					13
	% Non-violated TW	100	97	93	79					
	TATWD	0.0	0.1	1.1	2.2					
R104	Number of vehicles	11	10	9						10
	% Non-violated TW	100	96	79						
	TATWD	0.0	0.5	2.9						
R105	Number of vehicles	14	13	12						14
	% Non-violated TW	96	83	71						
	TATWD	0.6	2.3	3.5						
R106	Number of vehicles	12	11	10						12
	% Non-violated TW	95	80	55						
	TATWD	0.8	2.6	4.4						
R107	Number of vehicles	10								10
	% Non-violated TW	88								
	TATWD	1.9								
R108	Number of vehicles	10	9							9
	% Non-violated TW	100	85							
	TATWD	0.0	2.0							
R109	Number of vehicles	13	12	11						11
	% Non-violated TW	100	99	87						
	TATWD	0.0	0.1	1.5						
R110	Number of vehicles	11	10							11
	% Non-violated TW	100	82							
	TATWD	0.0	2.6							
R111	Number of vehicles	10								10
	% Non-violated TW	87								
	TATWD	1.4								
R112	Number of vehicles	10	9							10
	% Non-violated TW	100	85							
	TATWD	0.0	1.9							

vehicles) through small violations of the time windows. For example, in set R102, violating just 7% of the time windows results in only 16 required vehicles, while the best solution reported for the hard problem (no violations) is 17 vehicles. This is an important result that could indeed be exploited by sales when finalizing the contracts that set time windows.

Tables 3 and 4 compare the number of vehicles used and the percentage of non-violated windows of our method and two methods previously reported in the literature for the soft problem. The results of Balakrishnan [2] are presented in column 3 (BAL) and those of Koskosidis et al. [7] in column 4 (KPS). Note that there are no previous results by any

Table 2
PGSH and best solutions for Solomon's benchmark problems—set RC1

Problem	Metric	PGSH value				Best solution
RC101	Number of vehicles	16	15	14	13	14
	% Non-violated TW	100	100	92	87	
	TATWD	0.0	0.0	1.3	2.1	
RC102	Number of vehicles	14	13	12	11	13
	% Non-violated TW	100	98	89	80	
	TATWD	0.0	0.2	1.5	2.3	
RC103	Number of vehicles	13	12	11	10	11
	% Non-violated TW	100	100	93	81	
	TATWD	0.0	0.0	1.1	2.8	
RC104	Number of vehicles	11	10			10
	% Non-violated TW	100	93			
	TATWD	0.0	0.7			
RC105	Number of vehicles	13	12	11		13
	% Non-violated TW	89	76	60		
	TATWD	1.6	2.6	5.2		
RC106	Number of vehicles	13	12	11		12
	% Non-violated TW	100	99	87		
	TATWD	0.0	0.1	1.5		
RC107	Number of vehicles	11	10			11
	% Non-violated TW	94	60			
	TATWD	0.6	4.3			
RC108	Number of vehicles	11	10			10
	% Non-violated TW	100	90			
	TATWD	0.0	1.4			

Table 3
Comparison between literature heuristics and PGSH for R1 problems

Problem	Metric	BAL			KPS		PGSH		
R101	Number of vehicles	17	16	14	21	21	17	16	14
	% Non-violated TW	72	55	44	100	100	89	75	52
R102	Number of vehicles	19	13		19	19	13		
	% Non-violated TW	100	63		100	100	74		
R103	Number of vehicles	13	12		14	14	13	12	
	% Non-violated TW	86	68		100	100	97	93	
R108	Number of vehicles				10	10	9		
	% Non-violated TW				100	100	85		
R109	Number of vehicles	13	12	11	13	13	12	11	
	% Non-violated TW	100	90	67	98	100	99	87	

Table 4
Comparison between literature heuristics and PGSH for RC1 problems

Problem	Metric	BAL			KPS	PGSH		
RC101	Number of vehicles	16	15	14	16	16	15	14
	% Non-violated TW	100	96	68	95	100	100	92
RC102	Number of vehicles	14	13		14	14	13	
	% Non-violated TW	100	88		94	100	98	
RC103	Number of vehicles	13	12		13	13	12	
	% Non-violated TW	100	92		100	100	100	
RC106	Number of vehicles	13	12		13	13	12	
	% Non-violated TW	100	71		92	100	99	
RC108	Number of vehicles				11	11		
	% Non-violated TW				99	100		

Table 5
Comparison between PGSH and optimal solutions for hard problem

Problem	R101	R102	C101	C102	C106	C107	C108
Optimal	18	17	10	10	10	10	10
PGSH	18 (95)	17 (98)	10 (100)	10 (100)	10 (100)	10 (100)	10 (100)

of the two methods for the soft version of problems R104, R105, R106, R107, R110, R111, R112, RC105, RC105, and RC107. In addition, in Tables 3 and 4 we omit the value of TATWD, since neither Balakrishnan [2] nor Koskosidis et al. [7] report this metric. The results of Tables 3 and 4 clearly show that PGSH provides better solutions compared to the previously developed methods in all cases examined, since for the same number of vehicles, it results in violation of a smaller number of time windows. This represents a significant improvement over existing methods through the use of a very simple heuristic mechanism that is solely based on the primitive nearest-neighbour heuristic. One could easily infer that the application of a more sophisticated method could eventually produce even better solutions that violate a smaller number of time windows and use significantly reduced vehicle fleet sizes.

The results of the new heuristic were also compared to the existing optimal solutions reported by Desrochers et al. [11]. The optimal solutions are presented in the second row of Table 5, while the third row provides the PGSH solution (number of vehicles, and percentage of non-violated time windows in parentheses). From Table 5, it is evident that PGSH achieves the optimal solution with respect to the number of vehicles in five of the seven test problems without violating time windows.

To examine the effect of the various parameters of PGSH on the solution quality, we provide in Table 6, for two of the random and for two of the clustered problems, the values of ε, b_d, b_a, b_u, and b_p for which the solutions of Tables 1 and 2 were obtained. From the data of Table 6, we cannot infer a strong correlation between the quality of the results and the value of ε. However, we can observe that to obtain solutions with a small number of vehicles (i.e., smaller than the optimal number of vehicles in the VRP with fixed time windows—e.g., 14 vehicles for R101) we needed quite small values for this problem parameter ($\varepsilon=3$). Conversely, setting ε equal to large numbers compared to time windows (e.g., $\varepsilon = 23$ for R101) was adequate for obtaining solutions with larger vehicle fleet (21 vehicles). This was expected, since larger fleet sizes can be reached for the hard case as well, and large values of ε create almost hard problems (i.e., from the first iterations of PGSH, most time windows are fixed).

In addition, from the data of Table 6 we cannot extract concrete conclusions concerning parameters b_d, b_a, b_u, and b_p. Even a thorough examination of all the results we obtained during our computational experiments did not provide strong evidence about a consistently performing set of values for these parameters. Thus, we resorted to the recursive value-change to guarantee PGSA solution quality. The computational burden for PGSH is not unbearable, since NNH is very fast, thus, the total time required to complete all iterations is relatively small compared to complex evolutionary methods such as genetic algorithms or simulated annealing.

Table 6
PGSH parameters for sample data sets

Problem	R101							
Vehicles	21	20	19	18	17	16	15	14
ε	23	23	23	23	20	7	7	3
b_d	0.0	0.3	0.0	0.0	0.2	0.5	0.1	0.4
b_a	0.1	0.2	0.1	0.4	0.5	0.1	0.7	0.4
b_u	0.0	0.3	0.0	0.1	0.1	0.1	0.1	0.1
b_p	0.9	0.2	0.9	0.5	0.2	0.3	0.1	0.1
	R102							
	19	18	17	16	15	14	13	12
ε	20	19	20	13	10	7	11	4
b_d	0.2	0.2	0.5	0.3	0.4	0.1	0.0	0.3
b_a	0.1	0.4	0.2	0.1	0.3	0.4	0.4	0.2
b_u	0.1	0.1	0.1	0.1	0.1	0.2	0.2	0.2
b_p	0.6	0.3	0.2	0.5	0.2	0.3	0.4	0.3
	RC101				RC102			
	16	15	14	13	14	13	12	11
ε	17	22	16	10	12	23	14	10
b_d	0.1	0.2	0.0	0.3	0.0	0.3	0.1	0.4
b_a	0.2	0.1	0.7	0.4	0.3	0.1	0.7	0.4
b_u	0.1	0.1	0.1	0.2	0.1	0.1	0.1	0.1
b_p	0.6	0.6	0.2	0.1	0.6	0.5	0.1	0.1

To further examine the performance of PGSH with respect to the values of ε, we have plotted in Fig. 2 the PGSH results for various ε values on the data of set R101. The horizontal axis of the graph in Fig. 2 represents the value of ε and the vertical axis the percentage of non-violated time windows. The two data series in the graph reflect the results obtained concerning the percentage of non-violated time windows in solutions with 17 and 16 vehicles, for $\varepsilon \in \{1, 2, 3, \ldots, 22, 23\}$. Furthermore, the two horizontal lines at 55 and 72 represent the best solution for these two examples reported by Balakrishnan [2].

From the data series of Fig. 2 we can observe that PGSH outperforms the heuristic of Balakrishnan [2], irrespective of the value of ε (apart from ε=1 for 17 vehicles). Furthermore, the plot shows a rather steady performance of PGSH for all ε-values. The plot of Fig. 3, for 21 and 15 vehicles on set R101, further justifies this observation.

The number of iterations for a given ε are related to the solution of PGSH. Fig. 4 presents the number of vehicles and the percentage of violated time windows, at each of the iterations of PGSH for $\varepsilon = 23$. From the results of Fig. 4 we can infer the way PGSH proceeds: At the beginning, PGSH provides solutions close to the NNH results for the hard VRP case, without violating any time windows, while as the number of iterations increase, PGSH derives smaller vehicle fleets by violating some time windows. Similar curves were observed for all ε values. From a practical perspective, the evolution of PGSH can be bounded by the percentage of time window violations allowed; i.e., setting the maximum number of time window violations equal to 5%, the algorithm terminates with 18 vehicles after approximately 1500 applications of NNH (including the iterations due to the recursive change of the b-parameters). Note that the total time required to complete the results of Fig. 4 was less than 3 min, time acceptable even for daily vehicle scheduling.

Subsequently, we tested PGSH on larger problems obtained from the Web [12]. The data sets contain 200 and 400 customers and maintain the features of the original examples of Solomon [6]. Note that in the literature few heuristics and meta-heuristics are tested on large vehicle routing problems with time windows—see, e.g., Ioannou et al. [9]. We tested PGSH on three problems with 200 customers and three problems with 400 customers. The problem characteristics are shown in Table 7.

Table 8 shows the solutions obtained by PGSH, the original heuristic of Baker and Scaffer [13] noted by B&S, for the hard case, and the Lower Bound on the number of routes, for all problems of Table 6. The lower bound is given by the ratio of the total customer demand and the vehicle capacity [i.e., $\mathrm{LB} = (\sum q_i)/C$]. Note, that there are no published results for the VRPSTW for problems with more than 100 customers; thus we have used the B&S "hard" solutions for comparison purposes.

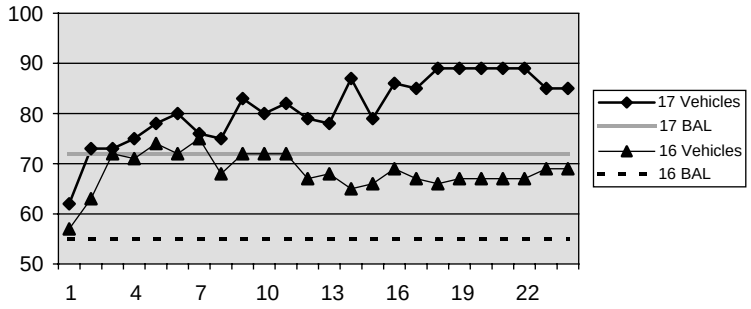

Fig. 2. PGSH results on R101 for 17 and 16 vehicles and various ε.

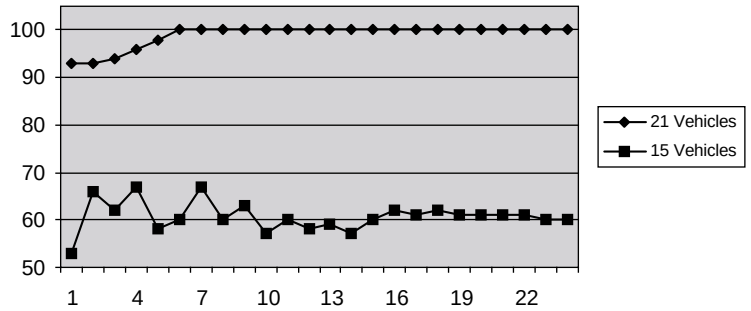

Fig. 3. PGSH results on R101 for 21 and 15 vehicles and various ε.

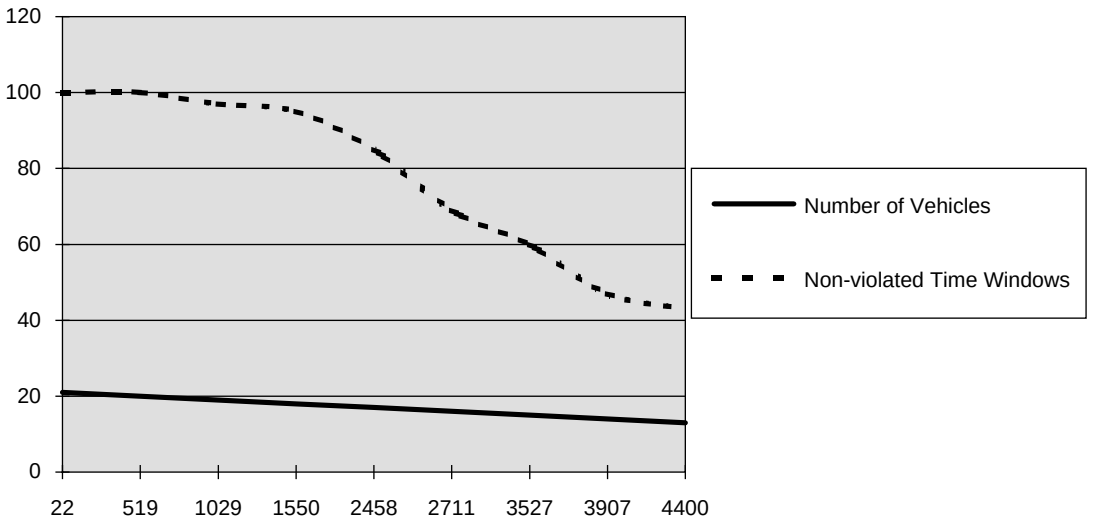

Fig. 4. PGSH results on R101 for $\varepsilon = 23$.

The results of Table 8 further support our observations from Tables 1 and 2, i.e., that with small violations of the time windows of very few customers the number of the vehicles required to service the customer demand is reduced; e.g., for problem R1_2_2, PGSH achieves a vehicle fleet size of 18, which is the lower bound on the number of required vehicles, by violating the time windows of only 6 customers (out of 200, i.e., only 3%).

Finally, we have applied PGSH on industrial data obtained from a delivery company, which supplies goods to a large

Table 7
Characteristics of large-scale problems

Data set (customers)	Total Cargo	Avg earliest time	Avg latest time	Average TW width	Time horizon	Service time	Percentage of time windows
R1_2_1 (200)	3513	255.7	265.7	10	634	10	100%
R1_2_2 (200)	3513	193.2	343.3	150.1	634	10	25%, 50 → (0, 570.3) 75%, 150 → (257.5, 267.5)
R1_2_3 (200)	3513	118.0	408.0	290.0	634	10	50%, 100 → (0, 570) 50%, 100 → (236.1, 246.1)
R1_4_1 (400)	7109	311.8	321.8	10.0	804	10	100%
R1_4_2 (400)	7109	231.7	419.8	188.1	804	10	25%, 100 → (0, 722) 75%, 300 → (308.9, 318.9)
R1_4_3 (400)	7109	153.0	518.8	365.8	804	10	50%, 200 → (0, 720.8) 50%, 200 → (306.8, 316.8)

Table 8
Results of PGSH on large-scale problems

Problem	Metric	B	Lower bound	PGSH		
R1_2_1	Number of vehicles	24	18	22	21	20
(200 customers)	Non-violated TW			197	195	189
	TATWD			1.0	0.6	1.2
R1_2_2	Number of vehicles	22	18	19	18	
(200 customers)	Non-violated TW			199	194	
	TATWD			0.3	0.7	
R1_2_3	Number of vehicles	23	18	19	18	
(200 customers)	Non-violated TW			199	195	
	TATWD			0.3	0.3	
R1_4_1	Number of vehicles	50	36	42	41	40
(400 customers)	Non-violated TW			389	382	370
	TATWD			1.0	1.8	2.9
R1_4_2	Number of vehicles	43	36	39	38	37
(400 customers)	Non-violated TW			399	385	365
	TATWD			0.2	1.2	2.0
R1_4_3	Number of vehicles	42	36	38	37	36
(400 customers)	Non-violated TW			397	393	374
	TATWD			0.3	0.8	2.6

number of retail outlets throughout the Athens Metropolitan area, in Athens, Greece. The number of customers is approximately two thousand and the company uses a fleet of 40 vehicles, with capacity of 1000 cartons, to perform all daily deliveries. Note that the number of vehicles was determined from a previous study we had performed for this company, the results of which are reported in [9]. Fig. 5 provides a partial view of the distribution of customers on a map of the Athens Metropolitan area.

To be consistent with the examples reported in the literature, we assumed that travel times are equivalent to the corresponding Euclidean distances. The time-window associated with the depot is 3300 time units, equivalent to one shift. A fixed service time of 10 time units for vehicle unloading is associated with each customer, and all customers have time windows of 150 time units. Typically, time windows are not violated by design but deliveries are performed before the earliest service time or after the latest service time in practice because of urgency and traffic conditions. We have asked the company to rank its customers with respect to importance and we appropriately fixed the penalties P_i to reflect this ranking.

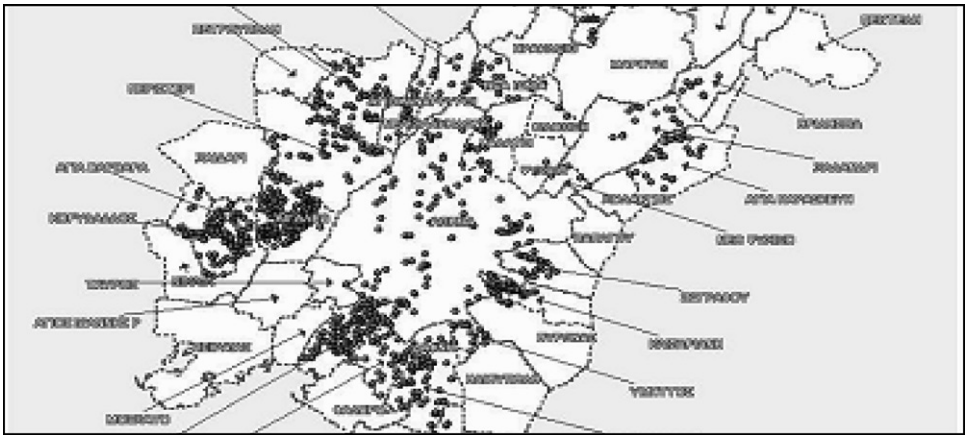

Fig. 5. Distribution of customers for industrial problem.

The application of PGSH on this industrial example further solidified our computational experiments. We were able to obtain a solution with 39 vehicles with only 12 time window violations (less than 1% of the total customer base), 38 vehicles with 50 time window violations (less than 3% of the total customer base) and 37 vehicles with 70 time window violations (less than 5% of the total customer base). Note also that a very small number of customers with violated time windows was allotted to each vehicle.

Furthermore, our interaction with the company's management provided an additional use of PGSH: Given a set of daily orders that must be satisfied, a set of available vehicles (which may be less than the vehicle fleet because of maintenance or employee unavailability), and an importance ranking of customers, which customers' time windows should be violated and by how much in order to complete all deliveries? This is a very frequent question in real-life situations, and PGSH can easily handle the underlying problem; i.e., search for a solution with the required number of vehicles and provide the minimum time window violations for this vehicle fleet, thus responding to dynamic changes in customer demand and variability of the vehicle fleet size.

5. Conclusions

In this paper we presented a new solution method for the vehicle routing problem with soft time windows. The solution engine of the method, which is based on the nearest-neighbour heuristic appropriately expanded to account for a penalty associated with time window violations, is applied on example sets created by our problem generator. The latter provides instances where vehicles are allowed to service some of the customers before and/or after their time windows. The problem is of particular importance for fleet planning and contract negotiations since it allows decision-makers to determine the best trade-off between time window expansion and number of required vehicles.

We have tested our method on benchmark problems with 100 customers form the literature and larger problems (200 and 400 customers) from the web. The results indicate that the method is very effective and outperforms previously developed approaches for the VRPSTW. The success of our method can be attributed to the wealth of problem instances solved in a structured manner (using the λ-soft problem generator).

The proposed approach could certainly benefit from the application of more advanced techniques for solving the VRP, such as tabu search or simulated annealing. However, one should be cautious in applying such methods in iterative solution schemes, since computational times could expand significantly. Nevertheless, incorporating into the proposed method more efficient solution engines is an open research avenue.

The real world is dynamic, thus, effective and efficient decision support tools are needed to address real word vehicle routing problems, taking into account the important benefits that emanate from the soft version of the VRPTW and exploiting the solutions provided by fast heuristics, to help sales and logistics people make better trade-offs during contract negotiations and vehicle fleet planning. The proposed approach, PGSH, as demonstrated by the industrial example, is a contribution in this pursuit.

Acknowledgements

The authors would like to thank the two anonymous referees for their constructive comments and suggestions that helped improve the content and the presentation of the paper.

References

[1] Hopp WJ, Spearman ML. Factory physics: foundations of manufacturing management. Chicago, IL: Irwin, 1996.
[2] Balakrishnan N. Simple heuristics for the vehicle routing problem with soft time windows. Journal of the Operational Research Society 1993;44(3):279–87.
[3] Golden B, Assad A. Vehicle routing: methods and studies. Amsterdam: Elsevier Science Publishers, 1988.
[4] Gendreau M, Laport G, Potvin J. Vehicle routing: modern heuristics. In: Aarts E, Lenstra JK, editors. Local search in combinatorial optimisation. New York: Wiley, 1997.
[5] Laport G. The vehicle routing problem: an overview of exact and approximate algorithms. European Journal of Operational Research 1992;59(2):345–58.
[6] Solomon MM. Algorithms for the vehicle routing and scheduling problems with time windows constraints. Operations Research 1987;35(2):254–65.
[7] Koskosidis YA, Powell WB, Solomon MM. An optimisation-based heuristic for vehicle routing and scheduling with soft time window constraints. Transportation Science 1992;26(2):69–85.
[8] Taillard E, Badeau P, Gendreau M, Guertin F, Potvin J-Y. A tabu search heuristic for the vehicle routing problem with soft time windows. Transportation Science 1997;31(2):170–86.
[9] Ioannou G, Kritikos M, Prastacos G. A greedy look-ahead heuristic for the vehicle routing problem with time windows. Journal of the Operational Research Society 2001;52(6):523–37.
[10] Herrmann JW, Ioannou G, Minis I, Proth JM. A dual ascent approach to the fixed-charge capacitated network design problem. European Journal of Operational Research 1995;95(4):476–90.
[11] Desrochers M, Desrosiers J, Solomon MM. A new optimisation algorithm for the vehicle routing problem with time windows. Operations Research 1992;40(2):342–54.
[12] Homberger J. Extended Solomon's VRPTW instances, 2000, http://www.fernuni0hagen.de/WINF/touren/inhalte/probinst.htm.
[13] Baker E, Schaffer JR. Solution improvement heuristics for the vehicle routing and scheduling problem with time window constraints. American Journal of Mathematical and Management Sciences 1986;6(3–4):261–300.