PERGAMON

Computers & Operations Research 26 (1999) 1153–1173

computers & operations research

A tabu search heuristic for the heterogeneous fleet vehicle routing problem

Michel Gendreau[a,b], Gilbert Laporte[a,c,*], Christophe Musaraganyi[a,d], Éric D. Taillard[a,e]

[a] *Centre de recherche sur les transports, Université de Montréal, Case postale 6128, succursale "Centre-ville", Montréal, Canada H3C 3J7*

[b] *Département d'informatique et de recherche opérationnelle, Université de Montréal, Case postale 6128, succursale "Centre-ville", Montréal, Canada H3C 3J7*

[c] *Service de l'enseignement des méthodes quantitatives de gestion, École des Hautes Études Commerciales, 3000 chemin de la Côte-Sainte-Catherine, Montréal, Canada H3T 2A7*

[d] *Service de Mathématiques de la Gestion, Université Libre de Bruxelles, CP 210/01, Boulevard du Triomphe, B-1050, Brussels, Belgium*

[e] *IDSIA, Corso Elvezia 36, 6900 Lugano, Switzerland*

Received April 1998; received in revised form October 1998

Abstract

The Heterogeneous Fleet Vehicle Routing Problem (HVRP) is a variant of the classical Vehicle Routing Problem in which customers are served by a heterogeneous fleet of vehicles with various capacities, fixed costs, and variable costs. This article describes a tabu search heuristic for the HVRP. On a set of benchmark instances, it consistently produces high-quality solutions, including several new best-known solutions.

Scope and purpose

In distribution management, it is often necessary to determine a combination of least cost vehicle routes through a set of geographically scattered customers, subject to side constraints. The case most frequently studied is where all vehicles are identical. This article proposes a solution methodology for the case where the vehicle fleet is heterogeneous. It describes an efficient tabu search heuristic capable of producing high-quality solutions on a series of benchmark test problems.

Keywords: Heterogeneous fleet vehicle routing problem; Fleet mix; Tabu search heuristic

*Corresponding author. Tel.: + 1-514-343-6143; fax: + 1-514-343-7121; e-mail: crt@crt.umontreal.ca

PII: S0305-0548(98)00100-2

1. Introduction

The purpose of this paper is to describe a tabu search algorithm for the *Heterogeneous Fleet Vehicle Routing Problem* (HVRP) defined as follows. Let $G = (V, A)$ be a directed graph where $V = \{v_0, v_1, \ldots, v_n\}$ is the vertex set and $A = \{(v_i, v_j)\colon v_i, v_j \in V,\ i \neq j\}$ is the arc set. Vertex v_0 represents a *depot* at which is based a fleet of vehicles, while the remaining vertices correspond to *cities* or *customers*. Each customer v_i has a non-negative demand q_i. There are several vehicle types. Denote by f_t the fixed cost of a vehicle of type t, by g_t its variable cost per distance unit, and by Q_t its capacity. The number of vehicles of each type is assumed to be unlimited. With each arc (v_i, v_j) is associated a distance c_{ij}. The HVRP consists of designing a set of vehicle routes, each starting and ending at the depot, and such that each customer is visited exactly once, the total demand of a route does not exceed the capacity of the vehicle assigned to it, and the total cost is minimized. The HVRP includes as a special case the version of the classical *Vehicle Routing Problem* (VRP) on which there is an unlimited number of identical vehicles. It is therefore NP-hard.

Our version of the HVRP is the most commonly studied in the Operations Research literature. Exceptions are the papers of Golden et al. [1] and of Salhi et al. [2] which both consider vehicle independent variable costs, and the paper by Taillard [3] in which the number of vehicles of type t is not unlimited, but equal to some constant m_t. Our problem is therefore to determine the vehicle fleet composition best suited to a particular setting while Taillard's problem is to make the best possible use of a given fleet.

As far as we are aware, no exact algorithm has ever been developed for the HVRP. Several approximation algorithms have been proposed, most derived from classical VRP heuristics, see, e.g., Golden et al. [1], Gheysens et al. [4, 5], Desrochers and Verhoog [6], and Salhi and Rand [7]. The algorithm of Osman and Salhi [8] is different in that it is based on tabu search. For a survey of some of these methods, see Salhi and Rand [7].

In this study, we develop a new tabu search heuristics for the HVRP. It applies to planar problems, i.e., to problems where vertices correspond to locations in the Euclidean plane. The algorithm itself is described in Section 2, followed by computational results in Section 3, and by the conclusion in Section 4.

2. Algorithm

The tabu search algorithm we propose for the HVRP is quite elaborate. First it makes use of GENIUS, a generalized insertion heuristic developed for the Traveling Salesman Problem (TSP) by Gendreau et al. [9]. Second, it incorporates several strategies, each of which requires some explanations. Third, the algorithm is itself embedded within a so-called *Adaptive Memory Procedure* (AMP), a search technique developed by Rochat and Taillard [10] in the context of the VRP. We will therefore organize the presentation of the algorithm as follows: brief description of GENIUS, components of the tabu search algorithm, step-by-step description of the tabu search algorithm, embedding of the tabu search algorithm within an AMP.

2.1. The GENIUS algorithm

The GENIUS algorithm consists of a tour construction phase (GENI – Generalized Insertion) and of an improvement phase (US – Unstringing and Stringing). Starting from three arbitrary selected vertices, GENI gradually constructs a tour by inserting at each step a new vertex into a partial tour. Contrary to what happens in standard insertion heuristics, each GENI insertion is accompanied by a local reoptimization of the tour. At a general step of the algorithm, the current solution is a tour over a subset of V. For a given orientation of the tour, denote by v_{i+1} the successor of a vertex v_i on the tour and by v_{i-1} its predecessor. Also, let $N_p(v)$ be the set of the p closest neighbours, already on the tour, of any vertex v. GENI constructs an initial solution made up of any three vertices. It then inserts an unrouted vertex v at each subsequent step. For this, select $v_i, v_j \in N_p(v)$ and $v_k \in N_p(v_{i+1})$, with $v_k \neq v_i$ and $v_k \neq v_j$. Two types of insertion are considered. In Type I insertions, delete the arcs (v_i, v_{i+1}), (v_j, v_{j+1}) and (v_k, v_{k+1}), and replace them with (v_i, v) (v, v_j), (v_{i+1}, v_k) and (v_{j+1}, v_{k+1}). This implies that the two paths $(v_{i+1}, \ldots, v_j)$ and $(v_{j+1}, \ldots, v_k)$ are reversed. In Type II insertions, a fourth vertex $v_l \in N_p(v_{j+1})$ is selected and $v_k \neq v_j$, $v_k \neq v_{j+1}$, $v_l \neq v_i$, $v_l \neq v_{i+1}$. Inserting v into the tour implies the deletion of arcs (v_i, v_{i+1}), (v_{l-1}, v_l), (v_j, v_{j+1}) and (v_{k-1}, v_l) and their replacement with (v_i, v), (v, v_j), (v_l, v_{j+1}), (v_{k-1}, v_{j-1}), and (v_{i+1}, v_k). The two paths $(v_{i+1}, \ldots, v_{l-1})$ and $(v_l, \ldots, v_j)$ are reversed. At each step, $O(p^4)$ choices of v_i, v_j, v_k, v_l are therefore considered. The partial tour including v and corresponding to the best combination of a vertex choice, an orientation of the tour and an insertion type is then selected. GENIUS is obtained by performing after GENI a postoptimization phase called US (for Unstringing and Stringing). In US, each vertex is in turn removed from the tour using the reverse of GENI, and reinserted as in GENI, until no further improvement can be achieved. On test problems, GENIUS has produced highly competitive results. Computation times are dependent on p. A choice of p between 3 and 7 seems appropriate for $n \leqslant 500$. For further details on GENIUS, the reader is referred to [9].

2.2. Components of the tabu search algorithm

As for all tabu search algorithms, the success of the procedure we propose requires the incorporation of several devices that exploit the characteristics of the problem at hand. Some of these are borrowed from search strategies developed for the classical VRP Gendreau et al. [11], while others are new. We examine these in turn.

2.2.1. Penalized objective function

One basic characteristic of tabu search is that is allows deteriorations of the objective functions to occur during the search procedure. The idea behind this is to prevent the search to become trapped in a poor quality local optimum. A natural extension of this principle is to allow some infeasible solutions during the course of the search. More specifically, let $f_1(s)$ denote the objective function value of a solution s, and by $O(s)$, the total vehicle overcapacity, if any, associated with this solution. The algorithm works with the artificial objective $f_2(s) = f_1(s) + \alpha O(s)$, where α is a nonnegative penalty factor dynamically adjusted throughout the search. Initially, set $\alpha := 1$. Then, every ξ iterations, set $\alpha := 2\alpha$ if all ξ previous solutions were infeasible, and $\alpha := \alpha/2$ if they were all feasible. In our implementation, we used $\xi = 6$. This diversification device, initially proposed by Gendreau et al. [11] has proved highly effective in a number of different contexts (see e.g., Cordeau et al. [12]) as it produces an interesting mix of feasible and infeasible solutions.

2.2.2. Initial solution

In a first phase, construct m vehicle routes, where m is selected in the interval $[1, 4]$ according to a discrete uniform distribution. If the problem in nonplanar, arbitrarily assign customers to vehicles. Apply the following sweep procedure. Define a ray with one extremity at the depot and the other at an arbitrary customer, and rotate it counterclockwise as long as the accumulated weight of all customers in the current route does not exceed $\beta \sum_{i=1}^{n} q_i/3.7$, where β is randomly selected in the interval $[1, 4]$; then initialize a new route. Every time a customer is reached, it is inserted into the current route by means of GENI. At the end of this phase, each route is reoptimized by means of US.

In a second phase, a vehicle is assigned to each route by solving a shortest path problem. Denote by $v_{i_1}, \dots, v_{i_t}$ the t customers of a route. Define an auxiliary graph $G' = (V', A')$ where $V' = \{v_{i_1}, \dots, v_{i_t}\}$, $A = \{(v_{i_r}, v_{i_s}): 1 \leqslant r < s \leqslant t\}$ and assign to arc (v_{i_r}, v_{i_s}) a cost d_{rs} equal to the cost of serving the sequence $(v_0, v_{i_r}, \dots, v_{i_s}, v_0)$ using the cheapest feasible vehicle. Then the best combination of vehicles is obtained by determining a least cost path from v_{i_1} to v_{i_t} on G', using the costs d_{rs}.

2.2.3. Neighbourhood structure

At a general iteration, let s be the current solution and m the number of vehicle routes. To define neighbour solutions, randomly select min $(n, 5m)$ vertices and successively attempt to insert them in a route containing one of the their five closest neighbours. If a vertex v is moved from route r to route s, a check is made whether it would be preferable, in terms of the artificial objective f_2, to use a different vehicle in routes r and s (typically a smaller vehicle on route r and a larger one in route s). Whenever it is profitable to do so, a new vehicle is assigned. As is often done in tabu search [11–13], f_2 is replaced with $f_2' = f_2 + \Delta_{\max} \sqrt{m} \rho \varphi_v$, where $\Delta_{\max}$ is the largest observed variation in f_2 between two successive iterations, ρ is a scaling factor equal to 0.0001 in our implementation, and φ_v is the number of times vertex v has been moved. This scheme diversifies the search by penalizing solutions involving frequently moved vertices. At each iteration, the best non-tabu move is performed and route s is reoptimized using US.

2.2.4. Tabu status and aspiration criterion

Whenever a vertex v is moved from route r to route s at iteration λ, it may not be reinserted into route r until iteration $\lambda + \theta$, where θ is randomly selected on in some interval $[\underline{\theta}, \overline{\theta}]$. In our implementation, we used $\underline{\theta} = 5$ and $\overline{\theta} = 10$. This tabu tenure mechanism was first suggested by Gendreau et al. [11] and virtually eliminates the probability of cycling. As is common in tabu search, the algorithm uses an aspiration criterion that overrides the tabu status of a vertex whenever moving it results in a new best value for f_2.

2.2.5. Post-optimization and fleet change

The first phase of search process ends after ω consecutive iterations, without improvement. We have used $\omega = 20$ for $n < 50$ and $\omega = 30$ for $n \geqslant 50$. When the termination criterion is satisfied, a second phase is entered on which two attempts are made to improve the best-known solutions. The first is an exchange procedure that swaps two vertices belonging to two neighbour routes, where centres of gravities of customers are used to compute distances between routes. The second attempt is a fleet change procedure that works as follows. Consider a route r and split it into two routes r' and r'' by using the best possible vehicle combination. If r contains at least five customers, then we impose that r' and r'' must each contain at least two customers. Then, while forbidding the fusion of two routes that have just been created as the result of a split, perform the following operations as long as f_2 improves: select two routes in the neighbourhood of r' and r''; perform customer moves and exchanges between them, the limiting case being the fusion of the two routes.

The second phase of the search process just described acts as a strong diversification device. In particular, it enables fleet changes that are not so frequently performed in the first phase, especially if vehicle fixed costs are high. Splitting route r into r' and r'' in the second phase may result on a worsening in f_2, but when the first phase is reentered, better incumbents are often identified.

2.3. Step-by-step description of the tabu search algorithm

We are now in a position to provide a step-by-step description of our tabu search algorithm. In our implementation, the value of η is equal to 5. The current number of routes is denoted by m.

Step 0 (*Initial solution*). Determine δ initial solutions. In our implementation, δ is equal to 6. Initialize s^* and $\bar{s}^*$, the best known solutions with respect to f_1 and f_2: $s^* := \bar{s}^* := s$. Initialize the iteration count $\lambda := 1$. Set $\alpha := 1$. No vertex is tabu. *Execute Steps* 1 *and* 2 *a total of* η *times.*
Step 1 (*Main search*). Set $\lambda := \lambda + 1$. Evaluate f_2' for all min $(n, 5m)$ neighbours of s. If the best solution s' is feasible and $f_1(s') < f_1(s^*)$, set $s := s^* := s'$; if it is infeasible and $f_2(s') < f_2(\bar{s}^*)$, set $s := \bar{s}^* := s'$. Otherwise, let s' be the non-tabu solution minimizing f_2'; set $s := s'$. Apply US to the routes of s different from those of the previous solution. If λ is a multiple of ξ, update the penalty coefficient α. If s^* and $\bar{s}^*$ have not changed for ω consecutive iterations, go to Step 2; otherwise, repeat Step 1.
Step 2 (*Postoptimization and fleet change*). Attempt to improve upon s^* by exchanging customers between neighbour routes. On s^*, perform the fleet change procedure, customer moves and customer exchanges to obtain a diversified solution s'. If $f_1(s') < f_1(s^*)$, set $s^* := s'$.

2.4. Adaptive memory procedure

The adaptive memory procedure (AMP), also known as probabilistic diversification and intensification, was introduced by Rochat and Taillard [10] in the context of the VRP. It works with a pool of full or partial solutions in a constantly updated memory. It is used as an initial solution generator for the tabu search algorithm. At each step, the procedure extracts a number of elements from the memory and combines some of their best structures to generate new solutions which are then improved through a local search process.

To keep the size of the memory under control, its worst elements are periodically discarded and replaced by new ones. The AMP can be viewed as a generalization of genetic search (see e.g., Goldberg [14]) in which two offspring are created from two parents; in the AMP, the offspring are created from several parents.

Like tabu search, the AMP is controlled by several problem dependent rules and parameters. Here is how it was applied to the HVRP.

Step 0 (*Memory initialization*). The adaptive memory is a set T of vehicle routes, where $|T| = 300$ in our implementation. During the tabu search algorithm, vehicle routes are labeled according to the objective value of the solution to which they belong and the $|T|$ routes containing at least two vertices and having the lowest labels are inserted in the memory. Initialize the iteration count $\mu := 1$.

Step 1 (*New solution*). Set $\mu := \mu + r1$. Construct a new solution s by combining elements of T. For this set $T' := T$ and select a route from T'. The selection process is biased so as to give a larger probability to routes having lower labels. Include route r into solution s and remove from T' all routes having at least one vertex in common with those of r. Repeat this operation as long as $T' \neq \emptyset$. If all vertices of V belong to s, go to Step 2. Otherwise include all missing vertices into s by creating a return route for each of these vertices.

Step 2 (*New solution improvement*). Improve s by means of the tabu search algorithm described in Section 2.3.

Step 3 (*Memory update*). Label the routes of s and insert them into T. Remove the worst elements from T to keep its size constant. If $\mu < \gamma$, where γ is a user controlled parameter, go to Step 2. Otherwise terminate. In our implementation, we used $\gamma = 7$.

3. Computational results

We now summarize the results of tests performed to calibrate the various parameters used in our algorithm and we present computational results on test problems.

3.1. Parameter calibration

Our algorithm, like most tabu search implementations, contains several user controlled parameters which require calibration. Parameter values were determined by using a sequential process as opposed to a statistical experimental design scheme. Sensitivity analyses were performed for all main parameters. No claim is made that our choice of parameter values is the best possible and should be the same for all instances of the problem. However, they seem to work well on our test problems, and they should also yield good results on instances of similar size and characteristics. Our algorithm contains nine user controlled parameters:

- p neighbourhood size n GENIUS (Section 2.1)
- ξ update frequency for the penalty parameter α (Section 2.2.1)
- ρ scaling parameter in the continuous diversification scheme (Section 2.2.3)
- $\underline{\theta}$ and $\overline{\theta}$ bounds for the tabu tenure (Section 2.2.4)
- ω maximum number of tabu iterations without improvement (Section 2.2.5)

- δ number of initial solutions (Section 2.3)
- η number of executions of the main tabu search routine (Section 2.3)
- $|T|$ adaptive memory size (Section 2.4)
- γ number of executions of the adaptive procedure (Section 2.4).

The parameter p was set equal to 5, a value which has already produced good results on the VRP [11] and which seems to offer a good compromise between computing time and solution quality. The value of ξ was set equal to six after some testing, but the performance of the algorithm is not overly affected by the value of this parameter. Five values were considered for ρ: 0, 0.0001, 0.001, 0.01, and 1. Selecting too large a value tends to over penalize some good moves, while taking too low a value may not have the desired effect. We found that selecting $\rho = 0.0001$ yielded the best results. The choice of a good tabu tenure interval is important. In particular, if $\underline{\theta}$ is too large, too many moves are tabu and good solutions can be forbidden. If $\bar{\theta}$ is too small, the search is not sufficiently diversified and cycling can also occur. We tested $[\underline{\theta}, \bar{\theta}] = [5, 10]$, $[10, 20]$ and $[15, 30]$ and found that $[5, 10]$ was the best choice. Several values were tested for ω, δ and η. Again, the aim is to find a good compromise between execution time and solution quality. After testing, we have opted for $\omega = 20$ for $n \leqslant 50$, $\omega = 30$ for $n > 50$, $\delta = 6$ and $\eta = 5$. Memory size was fixed at $|T| = 300$ after some testing. Selecting too large a value results in computational inefficiency. Selecting too small a value increases the risk of repeatedly producing the same solution, thus hindering the desired diversification effect. Finally, γ was set equal to 7 after some experimentation.

3.2. Results on test problems

The algorithm described in Section 2 was run with the same parameter values on twenty instances described in the literature. The first 12 instances (Tables 1 and 2) are taken from Golden et al. [1]. We use the same numbering system as these authors. These instances are described in Table 1 and they contain fixed costs only (e.g., the columns g_A to g_F must be ignored as they only contain unit values). In addition, we have also solved the eight instances used by Taillard (Table 3). These correspond to instances 13–20 of Table 1, except that this time, columns f_A to f_F must be ignored as these problems have no fixed costs, but only variable vehicle costs. In all test problems, the distance matrix satisfies the triangle inequality.

These instances were solved on a Sun Sparc 10 station using ten different runs of our tabu search algorithm. We report in Table 2 results corresponding to the first twelve instances (without variable costs), and we compare our solution values with those obtained by Osman and Salhi [8] and by Taillard [3]. The column headings relative to our results are as follows:

- Average value: average solution value over ten runs;
- Best value: best solution value over ten runs;
- Seconds (best): CPU time corresponding to the best solution.

Values in bold characters correspond to best-known solutions. Values in parentheses were obtained using a non-standard version of the algorithm. Full solutions are provided in Appendix A.

Table 1
Description of the 12 instances used as test problems

Inst. #	n	A			B			C			D			E			F		
		Q_A	f_A	g_A	Q_B	f_B	g_B	Q_C	f_C	g_C	Q_D	f_D	g_D	Q_E	f_E	g_E	Q_F	f_F	g_F
3	20	20	20		30	35		40	50		70	120		120	225				
4	20	60	1000		80	1500		150	3000										
5	20	20	20		30	35		40	50		70	120		120	225				
6	20	60	1000		80	1500		150	3000										
13	50	20	20	1.0	30	35	1.1	40	50	1.2	70	120	1.7	120	225	2.5	200	400	3.2
14	50	120	100	1.0	160	1500	1.1	300	3500	1.4									
15	50	50	100	1.0	100	250	1.6	160	450	2.0									
16	50	40	100	1.0	80	200	1.6	140	400	2.1									
17	75	50	25	1.0	120	80	1.2	200	150	1.5	350	320	1.8						
18	75	20	10	1.0	50	35	1.3	100	100	1.9	150	180	2.4	250	400	2.9	400	800	3.2
19	100	100	500	1.0	200	1200	1.4	300	2100	1.7									
20	100	60	100	1.0	140	300	1.7	200	500	2.0									

Table 2
Computational results for the 12 instances with fixed costs but without variable costs

Instance number	n	Osman and Salhi [8]	Taillard [3]	Average value	Best value	Seconds (best)
3	20	965	**(961.03)**	961.03	**961.03**	164
4	20	6445	(**6437.33**)	6441.01	**6437.33**	253
5	20	1009	(1008.59)	1008.72	**1007.05**	164
6	20	**6516**	(6516.47)	6517.98	**6516.46**	309
13	50	2437	2413.78	2424.88	**2408.41**	724
14	50	9125	**9119.03**	9121.98	**9119.03**	1033
15	50	2600	**2586.37**	2590.68	**2586.37**	901
16	50	2745	**2741.50**	2743.96	**2741.50**	815
17	75	1762	**1747.24**	1752.29	1749.50	1022
18	75	2412	**2373.63**	2392.57	2381.43	691
19	100	8685	**8661.81**	8682.50	8675.16	1687
20	100	4166	**4047.55**	4100.20	4086.76	1421

Using the same conventions, we display in Table 3 our computational results for instances with variable costs, but no fixed costs, and we compare them with those of Taillard [3].

Results presented in Tables 2 and 3 indicate that our algorithm always generates solutions that are almost as good as those of Osman and Salhi [8] and Taillard [3] or even better. Given that no

Table 3
Computational results for the eight instances with variable costs, but without fixed costs

Instance number	n	Taillard [3]	Average value	Best value	Seconds (best)
13	50	1494.58	1494.21	**1491.86**	626
14	50	603.21	603.33	**603.21**	669
15	50	1007.35	1001.80	**999.82**	736
16	50	1144.39	1137.01	**1136.63**	852
17	75	1044.93	1046.36	**1031.00**	1453
18	75	1831.24	1812.00	**1801.40**	1487
19	100	110.96	1117.09	**1105.44**	1681
20	100	1550.36	1553.72	**1541.18**	1706

strong lower bounds are available for the HVRP, comparisons with other authors is the only way we have to compare solutions. On the first set of instances, our results are better than that of Osman and Salhi, but are outperformed by those of Taillard, though not by a wide margin. Our algorithm has a better performance on instances with variable costs, but without fined costs. Here, our average solution values are better than those of Taillard in half the cases, and our algorithm always succeeds in obtaining a best known solution. On the smaller instances ($n = 50$), our computation times vary between 600 and 1050 s; using the same machine, Taillard obtained times smaller by 50%, between 350 and 570 s. On larger instances ($n = 75$ and 100), our computation times vary between 700 and 1700 s, while those of Taillard are higher, between 2000 and 12500 s.

To better understand the comparison of our results with those of Taillard, it is important to know that Taillard's algorithm generates a large family of routes and then selects some of them by solving a set partitioning problem. With this approach, the interaction between vehicles in only considered in a global sense when the set partitioning is solved. This method works particularly well on instances containing only vehicle fixed costs since these costs become critical at the set partitioning stage. Our algorithm is more global and is able to oscillate between various fleet compositions. It therefore tends to perform better on instances that combine fixed and variable vehicle costs. Our method also takes more time than Taillard's algorithm when $n = 50$, but is faster when $n = 75$ and 100. This is a direct consequence of the fact that the set partitioning phase of Taillard's algorithm becomes rather time consuming when n becomes large.

4. Conclusion

We have described an efficient and competitive tabu search algorithm for the HVRP. This problem is particularly difficult to solve using a local search technique since a natural tendency of the search process is to move towards a local optimum with the wrong fleet composition. To circumvent this difficulty it is necessary to diversify the search by embedding within the algorithm a fleet change mechanism. We believe this feature of our tabu search implementation is largely responsible for its success.

Appendix A

We now present the full solutions obtained with our algorithm for the Golden et al. [1] and the Taillard [3] instances.

A.1. Solutions for the 12 Golden et al. [1] *instances*

In these instances, the solution cost is the sum of the total vehicle cost and of the total distance traveled.

Instance 3

Route number	Demand	Vehicle type	Fixed vehicle cost	Distance	Customer sequence
1	40	C	50	37.36	5 - 11
2	30	B	35	47.58	8 - 1
3	118	E	225	91.65	18 - 4 - 19 - 13 - 14 - 6
4	29	B	35	16.12	12
5	118	E	225	125.47	3 - 20 - 2 - 16 - 9 - 10 - 15 - 17
6	19	A	20	52.84	7

Solution cost : 961.03

Instance 4

Route number	Demand	Vehicle type	Fixed vehicle cost	Distance	Customer sequence
1	60	A	1000	67.08	8 - 1 - 2
2	60	A	1000	84.48	12 - 17 - 15 - 19 - 4
3	56	A	1000	65.24	5 - 10 - 9 - 11
4	59	A	1000	80.11	16 - 20 - 3
5	60	A	1000	71.18	7 - 18
6	59	A	1000	69.24	13 - 14 - 6

Solution cost : 6437.33

Instance 5

Route number	Demand	Vehicle type	Fixed vehicle cost	Distance	Customer sequence
1	67	D	120	51.99	13 - 18 - 17
2	18	A	20	30.66	4 - 19
3	29	B	35	32.56	12
4	120	E	225	141.83	10 - 9 - 16 - 2 - 20 - 3 - 6
5	120	E	225	125.01	14 - 7 - 8 - 1 - 11 - 5 - 15

Solution cost : 1007.05

Instance 6

Route number	Demand	Vehicle type	Fixed vehicle cost	Distance	Customer sequence
1	58	A	1000	136.00	7 - 8 - 9 - 10
2	59	A	1000	118.21	3 - 20 - 16
3	59	A	1000	90.28	17 - 11 - 2 - 1
4	59	A	1000	71.51	6 - 14 - 13
5	59	A	1000	43.91	4 - 18 - 19
6	60	A	1000	56.55	15 - 5 - 12

Solution cost : 6516.47

Instance 13

Route number	Demand	Vehicle type	Fixed vehicle cost	Distance	Customer sequence
1	199	F	400	109.68	44 - 3 - 49 - 24 - 18 - 50 - 25 - 9 - 32 - 40
2	200	F	400	110.27	2 - 28 - 22 - 1 - 43 - 42 - 41 - 23 - 16 - 33 - 6
3	200	F	400	122.97	29 - 5 - 48 - 21 - 47 - 36 - 37 - 20 - 15 - 13 - 8 - 7
4	40	C	50	34.14	45 - 34
5	18	A	20	12.17	26
6	200	F	400	111.86	12 - 39 - 31 - 10 - 38 - 11 - 14 - 19 - 35
7	30	B	35	14.14	4
8	27	B	35	22.36	46
9	39	C	50	44.90	27 - 30
10	20	A	20	16.12	17

Solution cost : 2408.62

Instance 14

Route number	Demand	Vehicle type	Fixed vehicle cost	Distance	Customer sequence
1	114	A	100	51.12	12 - 39 - 9 - 40 - 17
2	119	A	100	90.18	38 - 11 - 14 - 19 - 13
3	120	A	100	101.14	16 - 23 - 49 - 24 - 18 - 50 - 44 - 3
4	117	A	100	45.52	7 - 35 - 8 - 46 - 34 - 4
5	156	B	1500	78.65	2 - 28 - 21 - 36 - 47 - 48 - 30
6	120	A	100	82.77	6 - 33 - 1 - 43 - 41 - 42 - 22
7	111	A	100	92.29	26 - 10 - 31 - 25 - 32
8	116	A	100	77.61	45 - 29 - 5 - 37 - 20 - 15 - 27

Solution cost : 9119.28

Instance 15

Route number	Demand	Vehicle type	Fixed vehicle cost	Distance	Customer sequence
1	50	A	100	39.59	4 - 18
2	49	A	100	47.60	25 - 14
3	48	A	100	67.53	8 - 31 - 28
4	48	A	100	47.61	6 - 23 - 48
5	50	A	100	69.36	34 - 30 - 46
6	50	A	100	31.42	12 - 5
7	49	A	100	47.07	11 - 16 - 38
8	100	B	250	100.06	37 - 15 - 45 - 33 - 39 - 10 - 49 - 9
9	99	B	250	109.50	22 - 3 - 36 - 35 - 20 - 29 - 21 - 50
10	98	B	250	105.95	13 - 41 - 40 - 19 - 42 - 44 - 17
11	47	A	100	88.81	24 - 43 - 7 - 26
12	40	A	100	34.20	27 - 47
13	49	A	100	47.66	32 - 2 - 1

Solution cost : 2586.37

Instance 16

Route number	Demand	Vehicle type	Fixed vehicle cost	Distance	Customer sequence
1	79	B	200	56.25	38 - 16 - 2 - 11
2	79	B	200	101.17	13 - 41 - 40 - 19 - 42
3	80	B	200	91.62	15 - 45 - 33 - 39 - 10 - 49
4	75	B	200	39.62	47 - 4 - 18
5	80	B	200	97.51	29 - 20 - 35 - 36 - 3 - 1
6	75	B	200	77.36	8 - 26 - 31 - 28 - 22 - 32
7	79	B	200	56.84	25 - 14 - 6 - 27
8	78	B	200	59.1	12 - 17 - 44 - 37 - 5
9	73	B	200	80.58	24 - 43 - 7 - 23 - 48
10	79	B	200	81.35	46 - 9 - 30 - 34 - 21 - 50

Solution cost : 2741.50

Instance 17

Route number	Demand	Vehicle type	Fixed vehicle cost	Distance	Customer sequence
1	197	C	150	116.63	51 - 16 - 63 - 23 - 56 - 49 - 24 - 18 - 50 - 32 - 44 - 3
2	200	C	150	87.40	26 - 7 - 53 - 11 - 66 - 65 - 38 - 58 - 72 - 12
3	199	C	150	109.65	34 - 46 - 8 - 35 - 14 - 59 - 19 - 54 - 13 - 52 - 68
4	200	C	150	113.60	17 - 40 - 39 - 9 - 25 - 55 - 31 - 10 -67
5	198	C	150	97.44	6 - 33 - 73 - 1 - 43 - 41 - 42 - 64 - 22 - 62 - 2
6	200	C	150	126.90	30 - 74 - 28 - 61 - 21 - 69 - 36 - 71 - 60 -70 - 20 - 37 - 29
7	120	B	80	77.42	45 - 48 - 47 - 5 - 15 - 57 - 27
8	50	A	25	15.46	4 - 75

Solution cost : 1749.50

Instance 18

Route number	Demand	Vehicle type	Fixed vehicle cost	Distance	Customer sequence
1	95	C	100	65.18	8 - 19 - 54 - 13 - 27 - 52
2	100	C	100	73.53	45 - 29 - 5 - 37 - 36 - 47
3	99	C	100	105.55	57 - 15 - 20 - 70 - 60 - 71 - 69 - 48
4	50	B	35	15.46	75 - 4
5	100	C	100	77.51	74 - 21 - 61 - 28 - 62
6	48	B	35	35.88	30 - 2
7	148	D	180	97.33	11 - 65 - 66 - 59 - 14 - 35
8	150	D	180	123.41	7 - 53 - 38 - 10 - 31 - 55 - 25 - 44
9	95	C	100	63.54	32 - 9 - 39 - 72 - 58
10	49	B	35	31.32	12 - 40
11	100	C	100	80.97	33 - 1 - 56 - 23 - 63 - 51
12	48	B	35	17.12	26 - 67
13	46	B	35	23.42	46 - 34
14	20	A	10	16.12	17
15	19	A	10	18.44	6
16	97	C	100	87.04	16 - 49 - 24 - 18 - 50 - 3
17	100	C	100	94.62	73 - 43 - 41 - 42 - 64 - 22 - 68

Solution cost : 2381.43

Instance 19

Route number	Demand	Vehicle type	Fixed vehicle cost	Distance	Customer sequence
1	96	A	500	103.41	19 - 49 - 64 - 11 - 62 - 88
2	100	A	500	117.68	78 - 34 - 35 - 71 - 65 - 66 - 20 - 69
3	100	A	500	78.23	27 - 70 - 32 - 90 - 63 - 10 - 31
4	98	A	500	75.48	89 - 5 - 84 - 17 - 45 - 8 - 83 - 18
5	100	A	500	57.35	28 - 12 - 80 - 68 - 76 - 1
6	97	A	500	75.64	77 - 3 - 79 - 29 - 24 - 54 - 26
7	93	A	500	120.96	2 - 57 - 42 - 14 - 38 - 43 - 15 - 41 - 22
8	99	A	500	90.47	82 - 46 - 36 - 47 - 48 - 7 - 52
9	99	A	500	101.89	55 - 25 - 39 - 67 - 23 - 56
10	100	A	500	48.21	6 - 96 - 59 - 95 - 97 - 87
11	99	A	500	51.66	92 - 37 - 100 - 85 - 93 - 99
12	97	A	500	74.65	30 - 51 - 9 - 81 - 33 - 50
13	99	A	500	80.20	60 - 61 - 16 - 86 - 44 - 91 - 98
14	99	A	500	67.16	40 - 21 - 73 - 72 - 74 - 75 - 4
15	82	A	500	32.17	53 - 58 - 13 - 94

Solution cost : 8675.16

Instance 20

Route number	Demand	Vehicle type	Fixed vehicle cost	Distance	Customer sequence
1	139	B	300	72.57	53 - 58 - 21 - 73 - 74 - 72 - 4 - 54 - 26
2	59	A	100	37.69	40 - 13 - 94
3	60	A	100	73.04	87 - 42 - 43 - 15 - 57 - 2
4	140	B	300	131.06	18 - 82 - 46 - 36 - 49 - 64 - 63 - 90 - 32 - 30 - 1
5	59	A	100	41.13	99 - 93 - 59
6	140	B	300	90.07	12 - 80 - 68 - 24 - 29 - 34 - 78 - 79 - 3 - 77
7	138	B	300	89.76	60 - 83 - 8 - 45 - 17 - 84 - 5 - 61 - 85 - 98
8	140	B	300	114.03	41 - 22 - 75 - 56 - 23 - 67 - 39 - 25 - 55
9	140	B	300	117.56	33 - 81 - 9 - 35 - 71 - 65 - 66 - 20 - 51
10	134	B	300	83.65	52 - 7 - 48 - 47 - 19 - 11 - 62 - 88
11	54	A	100	54.75	31 - 10 - 70 - 69
12	58	A	100	40.04	28 - 76 - 50 - 27
13	139	B	300	99.02	6 - 16 - 86 - 38 - 14 - 44 - 91 - 100 - 37 - 92
14	58	A	100	42.39	97 - 95 - 96 - 89

Solution cost : 4086.76

A.2. Solutions for the 12 Taillard [3] instances

In these instances, the solution cost is the scalar product of the variable vehicle cost vector and of the distance vector.

Instance 13

Route number	Demand	Vehicle type	Vehicle variable cost	Distance	Customer sequence
1	119	E	2.5	62.97	26 - 39 - 9 - 32 - 44 - 3
2	69	D	1.7	62.65	27 - 13 - 19 - 35 - 7
3	62	D	1.7	33.05	34 - 46 - 8
4	65	D	1.7	45.84	16 - 33 - 6
5	39	C	1.2	88.79	25 - 31
6	67	D	1.7	83.98	50 - 18 - 24 - 49
7	30	B	1.1	14.14	4
8	67	D	1.7	64.51	22 - 28 - 2
9	200	F	3.2	101.11	45 - 29 - 5 - 15 - 20 - 37 - 36 - 47 - 21 - 48 - 30
10	118	E	2.5	79.50	14 - 11 - 38 - 10
11	68	D	1.7	83.94	1 - 43 - 42 - 41 - 23
12	69	D	1.7	31.95	17 - 40 - 12

Solution cost : 1491.86

Instance 14

Route number	Demand	Vehicle type	Vehicle variable cost	Distance	Customer sequence
1	159	B	1.1	105.34	14 - 11 - 38 - 10 - 31 - 39
2	151	B	1.1	111.16	12 - 9 - 25 - 50 - 18 - 24 - 49 - 23 - 16
3	120	A	1.0	82.77	6 - 33 - 1 - 43 - 41 - 42 - 22
4	156	B	1.1	78.65	2 - 28 - 21 - 36 - 47 - 48 - 30
5	158	B	1.1	83.47	4 - 45 - 29 - 5 - 37 - 20 - 15 - 13 - 27
6	120	A	1.0	53.06	34 - 46 - 8 - 19 - 35 - 7 - 26
7	109	A	1.0	50.89	17 - 3 - 44 - 32 - 40

Solution cost : 603.21

Instance 15

Route number	Demand	Vehicle type	Vehicle variable cost	Distance	Customer sequence
1	99	B	1.6	73.54	49 - 30 - 34 - 50 - 9 - 38
2	50	A	1.0	69.07	41 - 13
3	49	A	1.0	47.60	25 - 14
4	47	A	1.0	88.81	24 - 43 - 7 - 26
5	48	A	1.0	47.61	6 - 23 - 48
6	49	A	1.0	78.92	46 - 16 - 21 - 29 - 22 - 1
7	31	A	1.0	28.44	11 - 32
8	160	C	2.0	105.37	2 - 20 - 35 - 36 - 3 - 28 - 31 - 8 - 27
9	95	B	1.6	36.89	18 - 47 - 12
10	99	B	1.6	97.07	5 - 10 - 39 - 33 - 45 - 15 - 44
11	50	A	1.0	96.62	37 - 17 - 42 - 40 - 19 - 4

Solution cost : 999.82

Instance 16

Route number	Demand	Vehicle type	Vehicle variable cost	Distance	Customer sequence
1	40	A	1.0	72.88	29 - 21 - 9 - 38
2	38	A	1.0	90.41	42 - 40 - 19 - 4
3	140	C	2.1	79.33	14 - 25 - 13 - 41 - 18
4	138	C	2.1	99.18	32 - 2 - 20 - 35 - 36 - 3 - 28 - 22 - 1
5	133	C	2.1	73.38	46 - 5 - 49 - 30 - 34 - 50 - 16 - 11
6	78	B	1.6	97.04	44 - 15 - 45 - 33 - 39 - 10
7	37	A	1.0	72.49	24 - 43 - 23
8	37	A	1.0	77.48	7 - 26 - 31
9	40	A	1.0	47.05	48 - 8
10	29	A	1.0	16.12	12
11	30	A	1.0	28.46	6 - 27
12	37	A	1.0	41.86	47 - 17 - 37

Solution cost : 1131.00

Instance 17

Route number	Demand	Vehicle type	Vehicle variable cost	Distance	Customer sequence
1	197	C	1.5	103.93	75 - 48 - 47 - 36 - 69 - 71 - 60 - 70 - 20 - 37 - 5 - 29 - 45
2	120	B	1.2	81.72	12 - 72 - 31 - 39 - 9 - 40
3	117	B	1.2	83.10	52 - 27 - 15 - 57 - 13 - 54 - 19 - 8
4	188	C	1.5	81.19	68 - 2 - 62 - 28 - 61 - 21 - 74 - 30 - 4
5	350	D	1.8	118.41	67 - 34 - 46 - 7 - 35 - 53 - 14 - 59 - 11 - 66 - 65 - 38 - 10 - 58 - 26
6	195	C	1.5	116.35	51 - 16 - 49 - 24 - 18 - 55 - 25 - 50 - 32 - 44 - 3- 17
7	197	C	1.5	116.98	6 - 33 - 63 - 23 - 56 - 41 - 43 - 42 - 64 - 22 - 1- 73

Solution cost : 1038.60

Instance 18

Route number	Demand	Vehicle type	Vehicle variable cost	Distance	Customer sequence
1	47	B	1.3	101.41	25 - 55- 31 - 72
2	98	C	1.9	80.99	8 - 19 - 54 - 13 - 57 - 15 - 27
3	395	F	3.2	142.69	17 - 3 - 44 - 32 - 9 - 39 - 58 - 10 - 38 - 65 - 66 - 11 - 59 - 14 - 53 - 35 - 7 - 26
4	98	C	1.9	84.71	50 - 18 - 24 - 49 - 16 - 51
5	233	E	2.9	120.14	6 - 33 - 63 - 23 - 56 - 41 - 43 - 42 - 64 - 22 - 1 - 73 - 2 - 68
6	49	B	1.3	31.32	40 - 12
7	100	C	1.9	77.51	74 - 21 - 61 - 28 - 62
8	95	C	1.9	30.86	67 - 46 - 52 - 34
9	249	E	2.9	104.51	4 - 45 - 29 - 5 - 37 - 20 - 70 - 60 - 71 - 69 - 36 - 47 - 48 - 30 - 75

Solution cost : 1801.40

Instance 19

Route number	Demand	Vehicle type	Vehicle variable cost	Distance	Customer sequence
1	96	A	1.0	88.12	40 - 73 - 74 - 22 - 41 - 15 - 43 - 57 - 2 - 58
2	193	B	1.4	91.00	52 - 7 - 82 - 48 - 47 - 19 - 11 - 62 - 88 - 31 - 27
3	99	A	1.0	88.83	54 - 55 - 25 - 24 - 29 - 68 - 80 - 12
4	195	B	1.4	95.99	26 - 4- 39 - 67 - 23 - 56 - 75 - 72 - 21 - 53
5	98	A	1.0	75.48	18 - 83 - 8 - 45 - 17 - 84 - 5 - 89
6	100	A	1.0	131.38	60 - 46 - 36 - 49 - 64 - 63 - 90 - 32 - 10
7	199	B	1.4	119.89	28 - 76 - 77 - 3- 79 - 78 - 34 - 35 - 71 - 65 - 66 - 20 - 30 70
8	198	B	1.4	101.10	6 - 96 - 99 - 61 - 16 - 86 - 38 - 44 - 14 - 42 - 87 - 13
9	92	A	1.0	68.61	69 - 1 - 51 - 9 - 81 - 33 - 50
10	188	B	1.4	58.46	95 - 97 - 92 - 98 - 37 - 100 - 91 - 85 - 93 - 59 - 94

Solution cost : 1105.44

Instance 20

Route number	Demand	Vehicle type	Vehicle variable cost	Distance	Customer sequence
1	58	A	1.0	41.25	52 - 31 - 69 - 27
2	59	A	1.0	118.03	8 - 46 - 36 - 49 - 64 - 7
3	195	C	2.0	108.00	53 - 40 - 21 - 72 - 75 - 56 - 23 - 67 - 39 - 25 - 55 - 4
4	200	C	2.0	110.52	88 - 62 - 10 - 32 - 90 - 63 - 11 - 19 - 47 - 48 - 82 - 18
5	60	A	1.0	89.19	54 - 24 - 29 - 34 - 78 - 76
6	139	B	1.7	115.75	1 - 51 - 9 - 35 - 71 - 65 - 66 - 20 - 30 - 70
7	194	C	2.0	72.96	28 - 50 - 33 - 81 - 79 - 3 - 77 - 68 - 80 - 12 - 26
8	60	A	1.0	86.00	73 - 74 - 22 - 41 - 15 - 43 - 42
9	54	A	1.0	68.07	89 - 60 - 84 - 17 - 45 - 83
10	181	C	2.0	51.94	94 - 95 - 92 - 37 - 98 - 85 - 93 - 59 - 99 - 96 - 6
11	58	A	1.0	49.54	87 - 57 - 2 - 58
12	200	C	2.0	102.76	13 - 97 - 100 - 91 - 44 - 14 - 38 - 86 - 16 - 61 - 5

Solution cost : 1541.19

Acknowledgements

This work was partially supported by the Canadian Natural Sciences and Engineering Research Council under grants OGP0038816 and OGP0039682. Éric Taillard also benefitted from an NSERC International Postdoctoral fellowship and was partially funded by Strategic grant STR0149269. This support is gratefully acknowledged. Thanks are also due to two anonymous referees for their valuable comments.

References

[1] Golden BL, Assad AA, Levy L, Gheysens FG. The fleet size and mix vehicle routing problem. Computers & Operations Research 1984;11:49–66.
[2] Salhi S, Sari M, Saidi D, Touati NAC. Adaptation of some vehicle fleet mix heuristics. Omega 1992;20:653–60.
[3] Taillard ÉD. A heuristic column generation method for heterogeneous fleet. RAIRO (Recherche opérationnelle), 1999, forthcoming.
[4] Gheysens FG, Golden BL, Assad AA. A comparison of techniques for solving the fleet size and mix vehicle routing problem. Operations Research Spektrum 1984;6:207–16.
[5] Gheysens FG, Golden BL, Assad AA. A new heuristic for determining fleet size and composition. Mathematical Programming Study 1986;26:233–6.
[6] Desrochers M, Verhoog TW. A new heuristic for the fleet size and mix vehicle routing problem. Computers & Operations Research 1991;18:263–74.
[7] Salhi S, Rand RK. Incorporating vehicle routing into the vehicle fleet composition problem. European Journal of Operational Research 1993;66:313–30.
[8] Osman IH, Salhi S. Local search strategies for the vehicle fleet mix problem. In: Rayward-Smith VJ, Osman IH, Reeves CR, Smith GD, editors. Modern heuristic search methods. Chichester: Wiley, 1996:131–53.
[9] Gendreau M, Hertz A, Laporte G. New insertion and postoptimization procedures for the traveling salesman problem. Operations Research 1992;40:1086–94.
[10] Rochat Y, Taillard ÉD. Probabilistic diversification and intensification in local search for vehicle routing. Journal of Heuristics 1995;1:147–67.
[11] Gendreau M, Hertz A, Laporte G. A tabu search algorithm for the vehicle routing problem. Management Science 1994;40:1276–90.
[12] Cordeau J-F, Gendreau M, Laporte G. A tabu search heuristic for periodic and multi-depot vehicle routing problems. Networks 1997;30:105–19.
[13] Taillard ÉD. Robust taboo search for the quadratic assignment problem. Parallel Computing 1991;17:433–45.
[14] Goldberg DE. Genetic algorithms in search, optimization and machine learning. Reading, MA: Addison-Wesley, 1989.

Michel Gendreau is professor of Operations Research at the Université de Montréal. He is the Deputy Director of the Centre for Research on Transportation. His main research interests include transportation planning, combinatorial optimization and metaheuristics.

Gilbert Laporte is professor of Operations Research at the École des Hautes Études Commerciales de Montréal and member of the Centre for Research on Transportation and of the GERAD. His research interests include vehicle routing, location and scheduling.

Christophe Musaraganyi obtained an M.Sc. in Operations Research at the Université de Montréal. He is now employed by KPMG Orinoco, Belgium.

Éric D. Taillard obtained his Ph.D. in Mathematics from the École Polytechnique Fédérale de Lausanne. He is now a researcher at IDSIA, Lugano, Switzerland.