Logist. Res. (2010) 2:3–11
DOI 10.1007/s12159-010-0028-3

ORIGINAL PAPER

A hybrid tabu search to solve the heterogeneous fixed fleet vehicle routing problem

Jalel Euchi · Habib Chabchoub

Received: 18 August 2009 / Accepted: 6 April 2010 / Published online: 28 April 2010

Abstract One of the most significant problems of supply chain management is the distribution of products between locations. The delivery of goods from a warehouse to local customers is a critical aspect of material logistics. The *Heterogeneous Fixed Fleet Vehicle Routing Problem* (HFFVRP) is a variant of the *Vehicle Routing Problem* (VRP) that aims to provide service to a specific customer group with minimum cost using a limited number of vehicles. We assume that the number of vehicles is fixed. We must decide how to make the best use of the fixed fleet of vehicles. In this paper we describe a Tabu Search algorithm embedded in the Adaptive Memory (TSAM) procedure to solve the HFFVRP. Computational experiments indicating the performance of the algorithm concerning quality of solution and processing time are reported.

Keywords Fixed fleet · Vehicle routing problem · Tabu search · Adaptive memory

1 Introduction

Much of the traditional logistics literature focuses on the outbound distribution of products to distribution centres and customers. Logistics may be defined as the provision of goods and services from a supply point to various demand points, Eilon et al. [1]. A complete logistic system involves transporting raw materials from a number of suppliers or vendors, delivering them to the factory plant for manufacturing or processing and eventually distribution to customers. Both the supply and the distribution procedures require effective transportation management. Good transportation management can save for private companies a considerable portion of their total costs. Potential cost savings include: lowered trucking cost due to better optimization of routes and shorter distances, reduced in-house space and related costs and less penalty incurred due to untimely delivery. One of the most significant measures of transportation management is effective vehicle routing.

The operations research community shows that the *Vehicle Routing Problem* (VRP) is one of its great success stories. The interplay between theory and practice is recognized as a major driving force for this success. Many variants and extensions of VRP have been subject of research during the last four decades. Some well-studied characteristics include a fixed fleet and heterogeneous vehicles.

The VRP was first introduced by Dantzig and Ramser [2] and since then it has been widely studied. It is a complex combinatorial optimization problem. The problem involves a fleet of vehicles set-off from a depot to serve a number of customers at different geographic locations with various demands. Several authors have made a literature review that deals with vehicle routing these include those of Bodin et al. [3], Laporte [4–6], and Toth and Vigo [7]. Vehicle routing problems are divided into various areas. Problems related to providing service through fixed fleets are complicated in comparison with unlimited fleet vehicle routing problems. The Heterogeneous Vehicle Routing Problem (HVRP) is studied in two different ways. On the one hand, some researchers make an assumption that there are an unlimited number of vehicles of each type and they try to find the optimal set of vehicles to be scheduled in the

J. Euchi (✉) · H. Chabchoub
Quantitatives Methods, GIAD Laboratory, Faculty of Economics and Management, Sfax University, Route de l'aéroport km 4.5, B.P 1088, Sfax 3018, Tunisie
e-mail: jalel.euchi@fsegs.rnu.tn

problem. This is called the Fleet Size and Mix VRP (FSMVRP). On the other hand, other researchers study the case where there is a fixed fleet of vehicles and try to schedule this fleet to the customers in an optimal way. This problem is called Heterogeneous Fixed Fleet VRP (HFFVRP).

In the literature, three variants of VRP with heterogeneous fleet (HFVRP) have been studied. The first one is introduced by Golden et al. [8], in which variable costs are uniformly given over all vehicle types with the number of available vehicles assumed to be unlimited for each type. The second version considers the variable costs depending on vehicle type, which is neglected in the first version. The third one, called Heterogeneous Fixed Fleet VRP (HFFVRP), generalizes the second version by limiting the number of available vehicles of each type. This version is the one dealt with in this paper.

When the number of vehicles is unlimited, we must determine the best composition of the fleet. We are not studying this version of the HFVRP in this paper and refer the reader to the paper of Gendreau et al. [9] and Choi and Tcha [10]. In this work, we are interested in the *Heterogeneous Fixed Fleet Vehicle Routing Problem* (HFFVRP) that involves a limited number of vehicles which can be heterogeneous or homogeneous. In this problem, the aim is to provide service to the customer group with minimum cost.

Due to the complexity of the HFFVRP, no exact algorithms have ever been presented for it. It is widely studied by heuristic design as those proposed in Salhi et al. [11] and Osman and Salhi [12]. Recently, the solution methods for the HFFVRP have substantially progressed in Taillard [13], Taillard and Rochat [14], Taillard [15], Tarantilis et al. [16], Li et al. [17]. Classical heuristics for the HFFVRP including the saving-based algorithms are presented in Desrochers et al. [18]. Golden et al. [8] develop a saving heuristic to solve the Fleet Size and Mix Vehicle Routing Problem as well as techniques for generating a lower bound and an underestimate of optimal solutions. Gendreau et al. [9] have proposed the Tabu Search (TS) algorithm for the Fleet Size and Mix VRP (FSMVRP). Also, Choi and Tcha [10] propose a column generation method to solve the HFVRP.

Taillard [15] developed a heuristic column generation method (HCG) to solve the HFFVRP. A new metaheuristic called Back-tracking Adaptive Threshold Accepting (BATA) was developed by Tarantilis et al. [16] in order to solve the HFFVRP. Recently, Li et al. [17] developed a record-to-record travel algorithm for the HFFVRP. They have built an integer programming model and solved the linear relaxation by column generation.

Tabu Search was introduced by Glover [19] in which he also coined the term metaheuristics and defined these as strategies designed to guide inner heuristics aimed at specific problems. Tabu Search is an extension of classical local search methods typically used to find approximate solutions to difficult combinatorial optimization problems. In order to improve the solution, the TS is embedded in the Adaptive Memory Procedure (AMP).

The main idea in the AMP is to record in a structure the individual components (the vehicle routes) making up elite solutions as they are found. These components are kept sorted in the AMP with respect to the objective function value of the solution to which they belong.

Our contribution is twofold. First, in the presence of the limited fleet constraints, the problem becomes more complex, which implies that the choice of a good metaheuristic can provide a good result. Our most interesting contribution is the introduction of three constructive initial solutions. Our algorithm will allow the possibility to start with multiple constructive methods, at each step we try to improve the solution constructed with another constructive method.

Second, a major contribution of the paper is the development of the efficient hybrid metaheuristic based on the AMP with high-quality solution produced.

The remainder of this paper is organized as follows: In Sect. 2, we describe the HFFVRP. In Sect. 3 we give the main paradigm of TS. Section 4 presents the details of our metaheuristic procedure. Section 5 explains how we solve the HFFVRP. Section 6 presents our computational study and provides the analysis of our results. Finally, Sect. 7 concludes this paper.

2 The heterogeneous fixed fleet vehicle routing problem

The HFFVRP can be described as follows: Let $N = \{1,\ldots,n\}$ be the set of customers and $G = (V, A)$ be a directed graph where $V = \{v_0, v_1, v_2, v_3, \ldots, v_n\}$ is the vertex set and $A = \{(v_i, v_j) : v_i, v_j \in V, i \neq j\}$ is the arc set. The vertex v_0 represents a depot at which is grouped a fleet of vehicles while the remaining vertices correspond to cities or customers. Each customer v_i has a non-negative demand q_i. Denote by z_k the fixed cost of a vehicle k, g_k its variable cost per distance unit, and Q_k its capacity. c_{ijk} represents the cost of the travel from customer i to j with a vehicle of type k. There are several types of vehicles, with T denoting the set of routes of such types. n_k is the number of vehicles of type k. In this version of the HFFVRP, the values of n_k are fixed. Then, the number of vehicles of type k is limited and the fleet is known in advance. Let $m = |T|$ represent the sum of routes realized of n_k for all types of vehicles. With each arc (v_i, v_j) is associated a distance d_{ij}.

A route l_k is feasible with respect to the route length constraint if $\sum_{(v_i,v_j)\in A} \delta_{ij}^k c_{ijk} \leq c_{\max}$ where $\delta_{ij}^k =$

$\begin{cases} 1 & \text{if}\,(i,j) \in \text{route}\, l_k \\ 0 & \text{otherwise} \end{cases}$ and $d_{\max}$ is the maximum cost of the travel.

The HFFVRP consists of designing a set of vehicle routes, each starting and ending at the depot such that each customer is visited exactly once by exactly one vehicle of the available fleet, the total demand of a route does not exceed the capacity of the vehicle assigned to it, the route length constraint is maintained, and the total cost is minimized.

Using binary decision variable

$$x_{lk} = \begin{cases} 1 & \text{if the } l\text{th route is selected and performed by} \\ & \text{a vehicle of type } k \\ 0 & \text{otherwise} \end{cases}$$

Taillard [15] gives the following mathematical formulation of the problem:

$$\text{Min} \sum_{k=1}^{K} \sum_{l=1}^{m} c_{lk}\, x_{lk}$$

subject to

$$\sum_{k=1}^{k} \sum_{l=1}^{m} a_{il} x_{lk} = 1 \quad i = 1, \ldots, n$$

$$\sum_{l=1}^{m} x_{lk} \le n_k \quad k = 1, \ldots, K$$

$$x_{lk} \in \{0, 1\} \quad l = 1, \ldots, m \quad k = 1, \ldots, K$$

where c_{lk} denotes the cost of the lth route performed by a vehicle of type k, and where $a_{il} = \begin{cases} 1 & \text{if the customer } i \in l\text{th route of } T \\ 0 & \text{otherwise} \end{cases}$.

3 Tabu Search algorithm

The TS is described, introduced, and refined by Glover [19]. Tabu Search is a type of metaheuristic that has been widely used to solve complex combinatorial optimization problems. As many other metaheuristics, the success of TS is, in large part, due to its ability to steer the search process from getting stuck in a local optimum. This is achieved by allowing a move to a neighbouring solution that may result in deterioration in the objective value but that simultaneously avoids cycling back through previous moves. Tabu Search procedures exploit the short-term memory, i.e. the Tabu list, which keeps track of recently visited solution or their attributes. A move to a neighbouring solution is permitted if the neighbouring solution is neither contained in the Tabu List (TL) nor possesses an identical attribute (e.g. objective value) to a solution in that list. However, a move to a neighbouring solution could be selected based on some aspiration criteria even if it is prohibited by the TL.

Tabu Search explores different kinds of memories in the search such as *recency* based (short-term), *frequency* based, *long-term* memories, etc. Usually it uses one neighbourhood structure and, with respect to that structure, performs descent and ascent moves building a trajectory.

4 Metaheuristic procedure

Our proposed solution algorithm is based on the AMP (Golden et al. [8], Arntzen et al. [20]) to solve the HFFVRP. The AMP was first proposed by Taillard and Rochat [14] as an enhancement of TS to solve the Vehicle Routing Problem (VRP). It was motivated by the work of Glover regarding surrogate constraints (Glover [21]). An important principle behind AMP is that good solutions may be constructed by combining different components of other good solutions. A memory containing components of visited solutions is kept. Periodically, a new solution is constructed using the data in the memory and improved by a local search procedure. The improved solution is then used to update the memory.

Our algorithm uses a TS embedded in AMP. The choice of our solution procedure is based on the success of TS to solve a wide range of challenging problems. A key feature of TS is its use of AMP to enhance a search strategy.

A pseudo-code of the Adaptive Memory Procedure (AMP) is given below:

1. Initialize the memory M.
2. While a stopping criterion is not met, do the following:
   a. Construct a new solution combining components of M.
   b. Apply a Tabu Search algorithm to (let be the improved solution).
   c. Update M using components of

We develop a TS algorithm applied in the local search phase of the adaptive memory.

5 Hybrid tabu search algorithm to solve the HFFVRP

In the sequel, we investigate and develop a TS heuristic embedded in Adaptive Memory Procedure (TSAM) to solve the HFFVRP.

The TSAM proposed herein for the HFFVRP can be roughly characterized into four steps: Initialization, generation of an initial solution, solution improvements, and updating the AMP.

The details of each step of the TSAM are detailed below:

5.1 Initialization (Step 1 in the adaptive memory)

To begin with, a certain amount of storage space called adaptive memory is allocated within the AMP. A set of routes is generated as described below and stored in the adaptive memory. The feasibility in the initialization is made with respect to the route length constraint and by avoiding the violation of capacity constraints.

For each arc (v_i, v_j) in A, we receive a profit value based on the distance travelling from v_j to v_j and the number of times this arc appears in the solution denoted by $g(v_i, v_j)$ if the vehicle of type k travels from customers v_i to v_j $((v_i, v_j) \in l_k)$.

We are going to skew the selection in favour of the most profitable vertices.

To generate an initial solution, define an advantage to insert the arc (v_i, v_j) in the solution by:

$$av_{ijk} = \begin{cases} \frac{(g(v_i,v_j)-c_{lk})}{c_{ijk}} & \text{if } (v_i, v_j) \in \text{route } l_k \\ \frac{1}{c_{ijk}} & \text{otherwise} \end{cases}$$

To choose an arc of a set $\bar{A} \subseteq A$, let $av_{\bar{A}k} = \sum_{(v_i,v_j)\in \bar{A}} av_{ijk}$.

To choose an arc (v_i, v_j) in $\bar{A}$ we proceed by first selecting randomly a number $\alpha \in \lfloor 0, av_{\bar{A}k} \rfloor$. Then, select the arc (v_{i_τ}, v_{j_τ}) such that τ is the smallest integer value such that $\sum_{l=1}^{\tau} av_{i_l j_l k} \geq \alpha$.

The procedure to generate m routes of the adaptive memory sums up as follows:

Step 1: Create m vehicle routes containing only an arc (v_0, v_0). Also create the set $M = \{(v_i, v_j) : c_{v_0 v_i k} + c_{v_i v_j k} + c_{v_j v_0 k} \leq c_{\max}\}$. Let $k = 1$ and move to Step 2.

Step 2: If $\bar{A}$ contains only one arc (v_0, v_0), select by roulette wheel an arc (v_i, v_j) from M and insert it into route l_K. Let $\bar{A} = A - (v_i, v_j)$ and $c_k = c_{v_0 v_i} + c_{v_i v_j} + c_{v_j v_0}$. Otherwise, continue the construction of route l_K as follows:

- Choose with roulette wheel an arc $(v_i, v_j) \in \bar{A}$
- Insert an arc (v_i, v_j)and select two vertices p and q in l_K, so that the evaluation functions $\{(av_{v_p v_i k} + av_{v_i v_j k} + av_{v_j v_q k} - av_{v_p v_q k}) : c_{v_p v_i k} + c_{v_i v_j k} + c_{v_j v_q k} - c_{v_p v_q k} \leq c_{\max}\}$ will be maximal.

If such (v_i, v_j) is not found (because $c_K + c_{v_p v_i k} + c_{v_i v_j k} + c_{v_j v_q k} - c_{v_p v_q k} \geq c_{\max}$), go to Step 3.

- Insert (v_i, v_j) between p and q and adjust the length of the route: $l_K = c_{ijk} + c_{v_p v_i k} + c_{v_i v_j k} + c_{v_j v_q k} - c_{v_p v_q k}$.
- Let $\bar{A} = A - (v_i, v_j)$
- If $\bar{A} \neq \phi$, repeat Step 2

Step 3: Let $k = k + 1$. If $k \leq m$, go back to Step 2.

The Hybrid TSAM in the initialization phase may be started from heuristic created solution. In the first step we consider that all vehicles are at the depot. Second, for every step we select a customer based in the constructive methods described above and we insert it in the best position that minimizes the total cost. In the next phase we try to construct a solution and to repair and improve the solution constructed from the routes generated in the AMP.

5.2 Construction of solution (Step 2 in the adaptive memory)

The TSAM procedure starts from an initial solution s constructed in the AMP. Then to improve the solution we use the regret heuristic used by Potvin and Rousseau [22], Liu and Shen [23]. Generate an order $\{a_{k_1}, a_{k_2}, \ldots, a_{k_m}\}$ in which the routes of the AMP are considered.

The regret heuristic works in the following way:

- Initialization

- For every artificial vertex (chosen vertex) $v_i \in V$: find the closest transport vertex v_j and the second closest transport vertex v_z.
- Calculate a regret value $REG_i = c_{iz} - c_{ij}$.
- Sort the regret value in descending order.
- Allocate the closest transport vertex to the artificial vertex according to this order; if a transport vertex is the closest to two or more artificial vertex, it is assigned to the one with the highest regret value.
- Continue with the same procedure until all of the transport vertices are assigned to a route. Always find the closest and second closest transport vertex to the last included vertex.

5.3 Solution improvements (Step 3 in the adaptive memory)

In order to improve the solution, we propose to use a TS algorithm as a local search. We give a pseudo-code of the proposed algorithm in Fig. 1.

5.3.1 Initial solution

The TS starts from an initial solution s constructed with the nearest neighbourhood method where customers are placed in an array sorted in the increasing order of demand. In this method, the customer with the biggest demand is appended to a route. When the next to-be inserted customer's distance exceeds the length of cycles on the current route, a new route is initiated.

```
Begin Tabu Search
   s_0 : initial solution
   s_current = s_0 ; s_best = s_0 ; θ = 0; TL = 0;
  for ( j = 1; j < nbclient; j + +)
     { //apply nearest neighbour
         s = Nearest _ Neighbour (s_current );
         for (i = 1; i < θ_max ; i + +)
           {//find the best non-tabu solution
               update TL;
               //use the permutation local search
               s′= permutation (s_current );
               s_current = 2 − move (s′);
               value = evaluate (s_current );
               if (value < s_best )
                 {
                   s_best = value;
                   θ + +;
                 }
           updateTL;
           }
     }
end Tabu Search
```

Fig. 1 Pseudo-code of Tabu Search algorithm

5.3.2 Neighbourhood structures

The TS algorithm that we have implemented uses two structures of neighbourhoods:

- Permutation-neighbourhood

Let v_i and $v_{i'}$ be two vertices on two different routes $l_i(s)$ and $l_{i'}(s)$. A permutation-move consists of replacing $l_{v_i}(s)$ and $l_{v_{i'}}(s)$ by $(l_{v_i}(s) - v_i) + v_{i'}$ and $(l_{v_{i'}}(s) - v_{i'}) + v_i$, respectively.

- two-move

In a two-move, vertex v_i is moved from its route to a route $l \neq l_{v_i}(s)$. Route l can be an empty route. Hence, $l_{v_i}(s)$ and l are replaced by $l_{v_i}(s) - v_i$ and $l + v_i$, respectively.

A conventional Tabu List (TL) contains pairs (i, l)with the condition that it is forbidden to move customer i to route l. A move (i, l) is considered as Tabu if $(i, l) \in TL$. The TS is stopped where $\theta_{\max}$ iterations have been performed without improving the best solution found s^*.

To improve the solution generated for each algorithm, we use two improvement procedures:

5.3.3 Exchanging of vertices between two routes

For two randomly selected routes l_1 and l_2 from the current solution s, these improvement procedures make one-vertex exchange between route l_1 and l_2 (Brandão [24]), starting with the first vertex on l_1. We scan from the first to the last arc on $l_2$to examine whether an exchange of the first vertex on l_1 with the current vertex on l_2 makes the total cost of the two routes shorter. The exchange is made immediately when such vertex on l_2 is found. Then, the procedure is repeated with the second vertex on l_1 and scanning vertex on l_2. This procedure stops when all one-vertex exchanges between l_1 and l_2 leading to improve route duration have been performed.

5.3.4 A random vertex-insertion procedure

For a selected route, this semi-greedy random insertion approach randomly removes a subset of vertices from the route and re-inserts them in the resulting partial route in a greedy way.

5.4 Updating the adaptive memory (Step 4 in the adaptive memory)

The strategy adopted in this paper is based in the framework of Tabu Search algorithm, but also borrows some heuristic ideas from the greedy constructive heuristics mentioned before. The main features of the algorithm are in the constructive greedy methods used in the different phases to improve the solution.

Below, the implementation of each part of the TSAM to solve the HFFVRP is described.

The Steps of the hybrid metaheuristic are summarized as follows:

Step 1: Generate m routes derived from solution nearest neighbour methods.
Start with $AMP = \phi$ (adaptive memory procedure).
Step 2: While a stopping criterion is not met, do the following:
Initialize the m routes to AMP', $s \neq \phi$.
Apply the constructive method mentioned in Sect. 5.1.
Repeat, while $AMP' \neq \phi$
Choose randomly a route $l \in AMP'$
Let $s' = s' \cup \{l\}$.
For each route $l' \in AMP'$, where $l \cap l' \neq \phi$
Let $AMP' = AMP' \backslash \{l'\}$.
Apply regret heuristic.
Step 3: Improve the new constructed solution.
For each route l in s, let $AMP = AMP \cup \{l\}$

Find a solution s^* (best solution) by considering the routes in
Apply Tabu Search algorithm
Exchange vertices between two routes and perform the random vertex-insertion procedure.
Step 4: Update the AMP by inserting the new constructed routes and removing routes (if necessary) which belong to the worst solutions.

6 Computational results

This section reports on the performance of the proposed TS heuristic embedded in Adaptive Memory (TSAM) over the benchmark test problems of Golden et al. [8] and Li et al. [17].

6.1 Implementation and instances

We consider two sets of instances to evaluate the performance of the TSAM algorithm. The first set is composed of the eight test problems developed by Golden et al. [8] for the vehicle fleet size and mix routing problem which can be viewed as a special case of HFFVRP where the travel costs are the same for all vehicle types and the number of vehicles of each type is limited. The specifications for the HFFVRP problem set are given in Table 1. We use the numbering scheme (problem 13 ⋯ problem 20) given by Golden et al. [8].

The second set is composed by the five new test problems developed by Li et al. [17], selected from the large-scale vehicle routing problems with 200–360 customers from Golden et al. [25] and adapted to the HFVRP (Table 2).

These problems contain between 50 and 360 vertices, all randomly located over a square. They have fixed fleet, capacity restrictions, no route length constraints, and no service times at the vertices. Moreover, euclidean distances are used in the entire problem.

The algorithm described here has been implemented in C++ using Visual Studio C++ 6.0. Experiments are performed on a PC Pentium 4, 3 GHz with 512 MB of RAM. Thus, the TS algorithm runs 10 times on each instance, and all results presented below are averages over these 10 runs.

6.2 Parameter settings

The TS procedure employs a set of parameters whose values need to be set before the algorithm is run. These parameters include the number of Tabu iterations $N_{\max}$, route improvement frequency χ, route selection parameter

Table 1 Specifications of eight benchmark problems with at most six types of vehicles

Problem	n	Vehicle A				Vehicle B				Vehicle C				Vehicle D				Vehicle E				Vehicle F				%
		Q_A	f_A	α_A	n_A	Q_B	f_B	α_B	n_B	Q_C	f_C	α_C	n_C	Q_D	f_D	α_D	n_D	Q_E	f_E	α_E	n_E	Q_F	f_F	α_F	n_F	
13	50	20	20	1	4	30	35	1.1	2	40	50	1.2	4	70	120	1.7	4	120	225	2.5	2	200	400	3.2	1	95.39
14	50	120	100	1	4	160	1,500	1.1	2	300	3,500	1.4	1													88.45
15	50	50	100	1	4	100	250	1.6	3	160	450	2	2													94.76
16	50	40	100	1	2	80	200	1.6	4	140	400	2.1	3													94.76
17	75	50	25	1	4	120	80	1.2	4	200	150	1.5	2	350	320	1.8	1									95.38
18	75	20	10	1	4	50	35	1.3	4	100	100	1.9	2	150	180	2.4	2	250	400	2.9	1	400	800	3.2	1	95.38
19	100	100	500	1	4	200	1,200	1.4	3	300	2,100	1.7	3													76.74
20	100	60	100	1	6	140	300	1.7	4	200	500	2	3													95.92

Table 2 Specifications for five new test problems with at most six types of vehicles

Problem	n	Vehicle A				Vehicle B				Vehicle C				Vehicle D				Vehicle E				Vehicle F				%
		Q_A	f_A	α_A	n_A	Q_B	f_B	α_B	n_B	Q_C	f_C	α_C	n_C	Q_D	f_D	α_D	n_D	Q_E	f_E	α_E	n_E	Q_F	f_F	α_F	n_F	
H1	200	50	20	1	8	100	35	1.1	6	200	50	1.2	4	500	120	1.7	3	1,000	225	2.5	1					93.02
H2	240	50	100	1	10	100	1,500	1.1	5	200	3,500	1.2	5	500	120	1.7	4									96.00
H3	280	50	100	1	10	100	250	1.1	5	200	50	1.2	5	500	120	1.7	4	1,000	225	2.5	2					94.76
H4	320	50	100	1	10	100	200	1.1	8	200	400	1.2	5	500	120	1.7	2	1,000	225	2.5	2	1,500	250	3	1	94.12
H5	360	50	25	1	10	100	80	1.2	8	200	150	1.5	5	500	320	1.8	1	1,000	225	2.5	2	2,000	250	3	1	92.31

T, number of neighbourhood solutions generated in the TS β and maximum non-improvement iterations θ. These parameters were determined on the basis of a number of preliminary runs. The values of these parameters are defined as follows: $N_{\max} = 200$; $\chi = 6$; $T = 30$; $\beta = 200$, $\theta = 100$.

6.3 Evaluation method

The results produced by our algorithms have been compared with those produced by the algorithms of Taillard [15], Tarantilis et al. [16] and Li et al. [17].

We begin the presentation of the results by examining, in Table 3, the efficiency of the procedure of TSAM. This table shows the strong performance of the TSAM algorithm in the form of the quality of solution and in the best CPU time.

In Table 4 we describe the efficiency of the TSAM algorithm over other metaheuristics presented in the literature. It gives the comparison results between the TSAM and the other methods proposed by Taillard [15], Tarantilis et al. [16], and Li et al. [17]. We observe that in seven out of eight test problems, the TSAM finds a better solution.

In Table 5, we give the relative percentage deviation of each algorithm's solution from the best known solution. A simple criterion to measure the efficiency and the quality of an algorithm is to compute the relation percentage deviation of its solution from the best solution reported in the literature on specific benchmark instances. From this table we conclude that the solution quality of the algorithms is comparable with an average deviation that is between 1 and 5% for the eight test problems. Our algorithm still seems to be superior in terms of solution quality with an average deviation of 0.0222% (Table 5).

Finally, in Table 6 we report the comparative result on five new test problems proposed by Li et al. [17]. It is interesting to observe that over the five large instances, four new best solutions were produced with our algorithm. In the large test problems, the TSAM yields consistently better results than the HRTR metaheuristic of Li et al. [17].

Table 3 Computational results for TSAM algorithms on eight test problems

Problem number	n	Total vehicles used	Fixed cost	Variable costs	Time
TSAM (tabu search embedded in adaptive memory)					
13	50	15	1,650	1477.34	3
14	50	6	6,800	590.00	8
15	50	9	2,050	1019.69	3
16	50	9	2,200	1112.92	2.26
17	75	10	1,035	1022.31	31.19
18	75	13	1,930	1768.51	25.35
19	100	8	9,500	1104.87	82.94
20	100	13	3,200	1510.72	71.09

n number of nodes

TSAM Tabu Search embedded in Adaptive Memory

Table 4 A comparison of TSAM, HCG, BATA, and HRTR according to overall costs

Problem	Taillard		Tarantilis et al.		Li et al.		Our algorithm	
	HCG	Time (s)	BATA	Time (s)	HRTR	Time (s)	TSAM	Time (s)
13	1518.05	476	1519.96	843	1517.84	358	**1477.34**	3
14	615.64	575	611.39	387	607.53	141	**590**	8
15	1016.86	335	1015.29	368	1015.29	166	1019.69	3
16	1154.05	350	1145.52	341	1144.94	188	**1112.92**	2.26
17	1071.79	2,245	1071.01	363	1061.96	216	**1022.31**	31.19
18	1870.16	2,876	1846.35	971	1823.58	366	**1768.51**	25.35
19	1117.51	5,833	1123.83	428	1120.34	404	**1104.87**	82.94
20	1559.77	3,402	1556.35	1,156	1534.17	447	**1510.72**	71.09

The bold values are the best known solution

HCG Heuristic column generation solution from Taillard, Sun Sparc workstation, 50 MHz; *BATA* Backtracking Adaptive Threshold Accepting solution from Tarantilis et al. [16], Pentium II, 400 MHz, 128 MB RAM; *HRTR* Record-to-record travel solution from Li et al. [17], Athlon, 1 GHz, 256 MB RAMS; *TSAM* Tabu Search Adaptive Memory, Pentium IV, 3 GHz, 512 MB RAM

Table 5 Percent deviation results for HFFVRP algorithms on eight test problems

Problem	Best Known solution	Taillard HCG	Tarantilis et al. BATA	Li et al. HRTR	Our algorithm TSAM
13	**1477.34**	2.75	2.88	2.76	0.00
14	**590.00**	4.34	3.62	2.97	0.00
15	**1015.29**	1.51	1.35	1.35	0.0043
16	**1112.92**	3.69	2.92	2.87	0.00
17	**1022.31**	4.84	4.76	3.87	0.00
18	**1768.51**	5.74	4.40	3.11	0.00
19	**1104.87**	1.14	1.71	1.40	0.00
20	**1510.72**	3.24	3.02	1.55	0.00
Average	**1200.24**	3.41	3.08	2.48	0.0005

The bold values are the best known solution

Table 6 Comparative result on five new test problems

Problem	n	Li et al.		Our algorithm	
		HRTR	Average CPU (s)	TSAM	Average CPU (s)
H1	200	12067.65	687.82	**11742.62**	413.05
H2	240	10234.40	995.27	**10103.87**	724.00
H3	280	16231.80	1437.56	**16231.80**	1060.18
H4	320	17576.10	2256.35	**17529.21**	1755.37
H5	360	21850.41	3276.91	**20996.15**	2355.47

The bold values are the best known solution

HRTR Record-to-record travel solution from Li et al. [17], Athlon, 1 GHz, 256 MB RAMS; *TSAM* Tabu Search Adaptive Memory, Pentium IV, 3 GHz, 512 MB RAM

7 Conclusion

This work presents a TS heuristic embedded in Adaptive Memory. The AMP and its mechanism for updating stored solutions allow a comparatively large pool of good and diversified solutions to be stored and used during the search process, alternating between small and large neighbourhood stages during the course of the TS. The computational results obtained with the TSAM metaheuristic on a set of benchmark instances compare favourably to existing literature, both with respect to solution quality and to computation time. The results of this research show that the performance of the proposed metaheuristic (TSAM) is competitive when compared with other approaches presented in the literature.

Acknowledgments We would like to thank the anonymous referees for their careful reading and very useful comments on earlier versions of this paper.

References

1. Eilon S, Watson-Grandy C, Christofides N (1971) Distribution management: mathematical modeling and practical analysis. Hafner, New York
2. Dantzig G-B, Ramser J-H (1959) The truck dispatching problem. Manage Sci 6(1):80–91
3. Bodin L-D, Golden B-L, Assad A-A, Ball M-O (1983) Routing and scheduling of vehicles and crews. The state of the Art. Comp Oper Res 10:69–211
4. Laporte G (1992) The vehicle routing problem: an overview of exact and approximate algorithms. Eur J Oper Res 59(3):345–358
5. Laporte G (1992) The traveling salesman problem: an overview of exact and approximate algorithms. Eur J Oper Res 59(2):247–291
6. Laporte G (1993) Computer aided routing CWI tract.75:M.W.P. Savelsbergh Mathematisch Centrum, Amsterdam, 1992, 134 pages, DFL.40.00, ISBN 90 6196 412 1. Eur J Oper Res 71(1):143
7. Toth P, Vigo D (2002) Models, relaxations and exact approaches for the capacitated vehicle routing problem. Discrete Appl Math 123:487–512
8. Golden B-L, Assad A-A, Levy L, Gheysens F-G (1984) The fleet size and mix vehicle routing problem. Comp Oper Res 11(1):49–66
9. Gendreau M, Laporte G, Musaraganyi C, Taillard E-D (1999) A Tabu search heuristic for the heterogeneous fleet vehicle routing problem. Comp Oper Res 26:1153–1173
10. Choi E, Tcha D-W (2007) A column generation approach to the heterogeneous fleet vehicle routing problem. Comp Oper Res 34:2080–2095
11. Salhi S, Sari M, Sadi D, Touati NAC (1992) Adaptation of some vehicle fleet mix heuristics. OMEGA 20:653–660
12. Osman I-H, Salhi S (1996) Local search strategies for the VFMP. In: Rayward-Smith VJ, Osman IH, Reeves CR, Smith

GD (eds) Modern heuristic search methods. Wiley, New York, pp 131–153
13. Taillard E-D (1993) Parallel iterative search methods for vehicle routing problems. Networks 23:661–673
14. Taillard E-D, Rochat Y (1995) Probabilistic diversification and intensification in local search for vehicle routing. J Heuristics 1:147–167
15. Taillard E-D (1999) A heuristic column generation method for the heterogeneous fleet VRP. RAIRO 33(1):1–14
16. Tarantilis C, Kiranoudis C, Vassiliadis V (2004) A threshold accepting metaheuristic for the heterogeneous fixed fleet vehicle routing problem. Eur J Oper Res 152:148–158
17. Li F, Golden B-L, Wasil E-A (2007) A record to record travel algorithm for solving the heterogeneous fleet vehicle routing problem. Comp Oper Res 34:2734–2742
18. Desrochers M, Verhoog T-W (1991) A new heuristic for the fleet size and mix vehicle routing problem. Comp Oper Res 18:263–274
19. Glover F (1989) Tabu search—Part I. ORSA. J Comp 1(3):190–206
20. Arntzen H, Hvattum L-M, Lokketangen A (2006) Adaptive memory search for multi demand multidimensional knapsack problems. Comp Oper Res 33:2508–2525
21. Glover F (1977) Heuristics for integer programming using surrogate constraints. Decis Sci 8(1):156–166
22. Potvin J-Y, Rousseau J-M (1993) A parallel route building algorithm for the vehicle routing and scheduling problem with time windows. Eur J Oper Res 66:331–340
23. Liu F-H, Shen S-Y (1999) A route-neighborhood-based metaheuristic for vehicle routing problem with time windows. Eur J Oper Res 118:485–504
24. Brandão J (2009) A deterministic tabu search algorithm for the fleet size and mix vehicle routing problem. Eur J Oper Res 195:716–728
25. Golden B, Wasil E, Kelly J, Chao I-M (1998) The impact of metaheuristics on solving the vehicle routing problem: algorithms, problem sets, and computational results. In: Crainic T, Laporte G (eds) Fleet management and logistics. Kluwer, Boston, pp 33–56