A Tabu Search Algorithm for Solving a Transportation Problem of Patients between Care Units

Yannick Kergosien, Christophe Lenté and Jean-Charles Billaut
Laboratoire d’Informatique, Université François-Rabelais de Tours
64 avenue Jean Portalis, 37200 Tours, France
{yannick.kergosien, christophe.lente, jean-charles.billaut}@univ-tours.fr

Abstract. In a hospital environment, patients have to be transferred to another hospital or another care unit. Patients may also be taken at home or brought from the hospital back home. For the Hospital Centre of Tours (France), the response to this transportation demand, which is partially known in advance, is ensured by the staff and vehicles of the Hospital Centre managed by the ambulance central station. The demands are not all equivalent since some of them require specific vehicles. The purpose is to satisfy the transportation demands with minimum cost. We propose an integer linear programming formulation and a Tabu Search algorithm for solving the problem. Computational results show the efficiency of the method. In: Shihabani K (ed), Proceedings of the 1st International Conference on Applied Operational Research – ICAOR (2008), pp 18–31. Lecture Notes in Management Science Vol. 1. ISSN 2008-0050.

Keywords: Transportation, Tabu search, healthcare services, vehicle routing problem.

1 Introduction

Our case of study relates to the ambulance central station of the Hospital Centre of Tours (France). Several types of vehicles compose the station: ambulances, light medical vehicles, vehicles for transporting people with reduced mobility, and ambulances with specific equipments. The ambulance central station only treats two types of transportation demands: those that are known in advance and those that are non-urgent and come in real time.

In our study, we consider only the problem of regulation of planned transport. In Section 2, we describe the problem and the context of the study. In Section 3 we present a literature review. In Section 4 we propose an integer linear programming formulation of the problem, which has been tested with solver software. In Section 5 a Tabu Search algorithm is described. Computational experiments are reported in Section 6. Finally, we conclude the study and propose some further research directions.
2 Problem presentation

The ambulance central station of the Hospital Centre of Tours is composed by 25 vehicles and 61 persons among them 55 are ambulance drivers. Vehicles equipments are not all the same and a given vehicle cannot transport any patient. More than 55000 transportations are performed every year with a mean value of 130 daily transportations. Some transportation demands may be delegated to private companies which generates an additional cost.

Transportation demands come from different services belonging to the hospital. Any demand contains all the important information about the transport, such as:
- the transportation constraints (stretcher, wheelchair, …) if any,
- patient infectiousness, if any,
- specific equipment needed (drip, oxygenation, ...), if any,
- etc.

We assume in our study that the crews are already constituted and that there is no availability problem of ambulance drivers, which is verified in practice.

The demands are scheduled by a team of two or three persons called regulator of the ambulance service. These persons have to decide which vehicle is assigned to each demand, considering vehicles availability, time constraints, geographical situations, equipment needed and the set of demands to satisfy. They also have the possibility to call a private ambulance company if necessary. The aim of the study is to propose an aid to the regulator persons.

Each transport demand generates a cost, which is related to the distance between the starting point and the arrival point of each vehicle, plus an additional cost if the vehicle belongs to a private company.

Given the transportation demands to perform daily, the problem is to propose an assignment of vehicles to the demands (with starting times) so that all the demands are satisfied with minimum cost.

3 Literature review

This problem is a particular case of Dial A Ride Problem (DARP), which is a particular case of Vehicle Routing Problem with Pickup and Delivery (VRP-PD).

One of the first studies of this problem is given in [Stein, 1978]. VRP-PD is related to transportation of goods whereas DARP deals with transportation of people, a specificity being that a person has a unique departure point and a unique destination.

One of the first studies about DARP is presented in [Psaraftis, 1980] for the case of one vehicle, and in [Jaw, al., 1986] for the case of multiple vehicles. An extensive state-of-the-art of models and algorithms for DARP can be found in [Cordeau, al., 2007]. Several problems related to the DARP have been studied in the case of multiple vehicles. In [Cordeau, al., 2003], the authors consider time windows, vehicles of limited capacities and maximal travel durations for persons as for vehicles. They propose a Tabu Search algorithm for solving the problem.
Another study with the same constraints is presented in [Rekiek, al., 2006] for the transportation of disabled persons. The objective is to minimize the size of the fleet. A mathematical formulation in association with a branch-and-price procedure are presented in [Cordeau, 2006] and [Ropke, al., 2007]. Notice that all the DARP presented in these papers consider a homogeneous fleet, which is not the case in our study.

The case of a heterogeneous fleet is treated in few papers. In [Dumas, al., 1989], [Desrosiers, al., 1991] and [Ioachim, al., 1995], the authors propose two-stage heuristics based on (a) a grouping technique of demands and (b) a routing procedure. In [Borndörfer, al., 1997] the authors study the same problem applied to transportation of disabled persons. Other similar studies are available in [Xiang, al., 2006] or [Wong, al., 2006]. In [Melachrinoudis, al, 2007], the authors consider the DARP in a context of health care organization. In this problem, the same vehicle is used twice for the same patient: the first time for driving the patient to the hospital, the second time for driving the patient back to the starting point. The authors propose an integer programming formulation and a Tabu Search algorithm.

Several transportation problems exist in connection with health care organizations. In [Banerjea-Brodeur, al., 1998] the authors study the periodic VRP of transports between hospitals and the laundry. They propose a Tabu Search algorithm inspired by [Cordeau, al., 1997].

Another research field is the problem of emergency vehicle location (see [Al-salloum, al., 2006], [Araz, al., 2007] or [Rajagopalan, al., 2008]). The problem is to optimize the covering zone for emergency services. In [Shang, al., 1996], the authors consider the transportation of medical artefacts, mostly medical records, which is similar to a VRP-DP with time windows and without capacity constraints. Finally in [Beaudry, al., 2006] the authors develop a heuristic for a dynamic DARP. The objective is to insert new demands in existing routes respecting some constraints like patient isolation or required medical items. The authors present a heuristic by insertions followed by a Tabu Search algorithm.

The Tabu Search presented in this paper is based on the one described in [Taillard, al., 1997]. In this paper, the problem considered is a vehicle routing problem with soft time windows (noted VRP-STW): given a set of customers to serve and a set of vehicles positioned in a unique depot, the problem is to determine the route of each vehicle in order to satisfy the demands and to minimize two criteria: the total distance travelled and a lateness penalty.

4 Modeling

4.1 TSP model and notation

Let $R$ be a set of $n$ demands, each demand being characterized by an initial location $a$ and a final location $b$. The travel time between $a$ and $b$ is denoted by $t_{ab}$ and does not depend on the vehicle. We use the following notations:
- \( i \): index for a demand (or a patient), characterised by an initial location \( a \) and a final location \( b \).
- \( k \): index for a vehicle and \( m \) their number,
- \( u \): index for a “specificity” or “skill” required by a patient or possessed by a vehicle (for example wheelchair),
- \( p_i \): the time needed to prepare demand \( i \), which includes the patient preparation and the time needed before the travel can start,
- \( t_{ab} \): the travel time between \( a \) and \( b \); it doesn’t depend on the vehicle used,
- \( e_i \): the earliest date of departure of demand \( i \); the real date of departure should be as close as possible to (and no smaller than) \( e_i \),
- \( D_k \): the location of vehicle \( k \) (same location for the vehicles of the ambulance central station and several locations for private companies),
- \( T_k \): the daily maximum time of use of vehicle \( k \),
- \( s_{uk} \): is equal to 1 if vehicle \( k \) has skill \( u \), and 0 otherwise,
- \( h_{ik} \): is equal to 1 if patient \( i \) needs skill \( u \), and 0 otherwise,
- \( c_{ab}^k \): the cost of using vehicle \( k \) between locations \( a, b \). If \( a \) is equal to location \( D_k \), then \( c_{ab}^k \) includes an additional cost due to the vehicle use (important for private vehicles).

The objective is to find a routing that satisfies all demands and which minimizes the total cost and the maximal waiting time of the patients.

This problem is equivalent to a multiple TSP in a directed asymmetrical and strongly connected graph \( G=(V,E) \). The vertices of the graph are divided into two subsets: a subset \( D \) of vertices for the depot of vehicles and a subset \( R \) of vertices for the transportation demands.

An edge between two demands \( i \) and \( j \) indicates the consecutive treatment of demand \( i \) and demand \( j \). To each edge \((i,j)\) are associated:
- a length \( d_{ij} = t_{ab} + p_i + t_{bc} \) with \( a \) the location of demand \( i \), \( b \) the destination of demand \( i \), and \( c \) the location of demand \( j \).
- a cost \( w_{ij}^k = c_{ab}^k + c_{bc}^k \) which depends on the vehicle that is used.

An edge between a depot and a demand \( i \) indicates that demand \( i \) is the first satisfied by the vehicle. The length of this edge is equal to the travel duration from the depot to the location of demand \( i \), and the cost is equal to \( c_{ab}^k \), which is vehicle dependent.

An edge between the last demand and the depot has a length equal to the travel duration.

Each salesman of the multiple TSP represents a vehicle \( k \) and is subject to a daily maximum travel time denoted by \( T_k \). No salesman can pass by all the vertices (respect of the characteristics of the routes), and vertex \( i \) can only be crossed by a travelling salesman after date \( e_i \). There is only one depot per vehicle and a vehicle may stay at the depot if necessary.

The objective of this multiple TSP is to find the routes of the vehicles so that each demand is satisfied by exactly one vehicle and the constraints are respected. The two objective functions to be minimized are the total cost and the maximum tardiness of patients. We propose to solve the problem by using the “epsilon-constraint” approach [Steuer, 1986] which consists in minimizing one criterion and bounding the others. In our case, we bound the maximum lateness. This is
equivalent to associate to each vertex (to each demand) \( i \) a time window \([e_i, e_i + \varepsilon]\) during which one salesman (one vehicle) has to come.

We propose now an exact resolution of this problem by using an integer linear programming formulation (ILP).

### 4.2 ILP

One of the first ILP models for TSP was formulated by [Miller, al., 1960]. This formulation was extended by [Svetska, al., 1973] to several salesmen. In [Kara, al., 2006], the authors propose to extend these models by integrating a minimum number of vertices to be visited by each salesman in the case of a single depot and multiple depots. In this model, HV denotes an High Value.

**Variables.**

- \( \forall i \in R \cup D, \forall j \in R \cup D, i \neq j, (i \notin D) \lor (j \notin D), \forall k \in D : x_{i,j}^k = 1 \) if vehicle \( k \) takes demand \( i \) and then demand \( j \), 0 otherwise.
- \( \forall i \in R : u_i \in \{1...n_{\max}\} : \) this variable makes it possible to eliminate the sub tours of one vehicle, the value is at least equal to the number of vertices of \( R \) visited by the vehicle, with \( n_{\max} \) the maximum number of vertices that the same vehicle can visit. This value is determined by pre-processing data.
- \( \forall i \in R : z_i \geq 0 : \) starting date of the vehicle for demand \( i \).

**Constraints.**

\[
\forall k \in D : \quad \sum_{i \in R} x_{i,j}^k \leq 1
\]
\[
\forall i \in R : \quad \sum_{k \in D} x_{i,j}^k + \sum_{k \in D} \sum_{j \in R} x_{j,i}^k = 1 \tag{2}
\]
\[
\forall i \in R, \forall k \in D : \quad x_{i,j}^k + \sum_{j \in R} x_{j,i}^k - x_{i,k}^k - \sum_{j \in R} x_{i,j}^k = 0 \tag{3}
\]
\[
\forall i \in R, \forall j \in R, i \neq j : \quad u_i - u_j + n \sum_{k \in D} x_{i,j}^k \leq n - 1 \tag{4}
\]
\[
\forall i \in R : \quad e_i \leq z_i \tag{5}
\]
\[
\forall i \in R : \quad z_i - e_i \leq \varepsilon \tag{6}
\]
\[
\forall i \in R, \forall k \in D : \quad d_{i,j} x_{i,j}^k \leq z_j \tag{7}
\]
\[
\forall i \in R, \forall j \in R, i \neq j : \quad z_i + d_{i,j} \leq z_j + (1 - \sum_{k \in D} x_{i,j}^k).HV \tag{8}
\]
\[
\forall i \in R, \forall j \in R, i \neq j, \forall k \in D : \quad z_i + d_{i,k} - z_j + d_{k,j} \leq T_k + (2 - x_{i,k}^k - x_{k,j}^k).HV \tag{9}
\]
The notation indicates a value presumably infinite.

The constraints (1) impose only one exit of vehicle by depot. The constraints (2) and (3) impose that all the vertices must be crossed exactly once by all the vehicles and in a continuous route: when a vehicle enters a vertex, it has to exit. The constraints (4) are the Subtour Elimination Constraints (SEC). They are used to prevent building a cycle that does not contain the depot of the corresponding vehicle. The constraints (5) and (6) limit the value of \( z_e \) for each vertex. The constraints (7) and (8) formulate the routes durations between two vertices. Constraints (9) prevent each vehicle \( k \) from exceeding the legal driving duration.

**Objective function.** The lateness being bounded, only the transportation cost has to be minimized:

\[
\sum_{k \in D} \sum_{i \in R \setminus D} \sum_{j \in \mathcal{E} \setminus \{i\}} w_{i,j}^k x_{i,j}^k
\]  

(10)

5 Tabu Search based algorithm

The following Tabu Search algorithm is inspired from [Taillard, al., 1997] with different hypotheses: multi depot, hard time window, no commodity to serve, adequacy between patient and vehicle (notion of skill) and two different criteria (see [Taillard, al., 1997] for more details).

The general method used for solving the problem is the following:

- **Step 1:** Initialization of the adaptive memory
- **Step 2:** For \( j \in \{1..J\} \) Do
  - **Step 3:** Construct a solution \( S_{init} \) from the routes of the adaptive memory; \( S = S_{init} \)
- **Step 4:** For \( i \in \{1..I\} \) Do
  - **Step 5:** Decompose the set of routes of solution \( S \) into two subsets of disjoint routes.
  - **Step 6:** Apply the Tabu Search on each subset separately to obtain two new subsets of routes.
  - **Step 7:** Recompose these two new subsets, leading to the new solution \( S \).
- **Step 8:** Add the routes of the best solution found in the adaptive memory.
- **Step 9:** post optimization.
5.1 Initialization of the adaptive memory

In Step 1, a procedure is used to build $p$ different initial solutions. This procedure works as follows:
- vehicles are sorted in their cost of use in increasing order (vehicles of the ambulance central station first and private ambulances last),
- demands are treated in an arbitrary order; a demand is assigned to the first vehicle in the list that can take the demand on time and that respects the constraints (skills and maximum time of use).

This procedure allows generating a high variety of routes. After that, for each initial solution, a Tabu Search (described later in Section 4.6) is applied with a small number of iterations. A route in a solution is the list of demands performed by one vehicle. All the routes of the improved solutions are inserted in an “adaptive memory”.

5.2 Adaptive memory

The adaptive memory is a set of routes, sorted according to the quality of the solution they come from. Two important actions are noteworthy in the adaptive memory:
- adding the routes of one solution (Step 1 or Step 8),
- constructing a solution from the routes of the adaptive memory (Step 3).

The adaptive memory always contains a fixed number of different routes, coming from the better solutions obtained during the search procedure. These routes are sorted in the quality of their parent solution in decreasing order.

A new route is inserted in the memory at the right place; if the route already exists, this route is moved to a new position in the list if better.

The construction of a new solution from the adaptive memory works as follows:
- a fixed probability of selection noted $\rho$ is associated to the routes
- the routes are considered in the memory order and a route is selected if a randomly generated number is greater than or equal to $\rho$.
- after a selection of one route, the routes that have at least one patient in common with the selected route, or those that are assigned to the same vehicle are eliminated for the next selection.
- if the set of candidate routes is empty and if there are remaining demands, they are assigned to vehicles as in the initialization procedure.

5.3 Diversification

During the search, some solutions can be explored several times (in Steps 4, 5, 6, 7 for instance). To avoid this, a treatment for diversification is inserted at Step 3.
When the best solution is generated three times, the selection order of routes in the
adaptive memory is reversed. This step allows forcing the exploration of another part of the solution space.

5.4 Decomposition and reconstruction of solutions

As explained before, the Tabu Search is used for improving several initial solutions. A solution is a collection of disjoint routes and a neighbour is obtained by crossing pairs of routes. The neighbourhood is obtained by considering all crossing pairs.

In order to reduce the computation time, the set of routes is split in two subsets, which is the decomposition phase of a solution.

A solution can be decomposed in two ways: a geographical approach and a time approach. At iteration \( i \) (step 4), the route followed by the \( i^{th} \) vehicle is chosen. Then, the \( m/2 \) routes which are the more similar to the chosen route – either in terms of geographical location or in terms of timing – are gathered. Other routes are gathered in a second set and the Tabu Search is used on each subset. Geographical location grouping and timing grouping are used alternatively.

Reconstructing a solution consists in making the union of the two subsets of routes, improved by the Tabu Search.

5.5 Post optimization

At the end of the algorithm (Step 9), a post-optimization step is applied to the best found solution. This step uses the same process as for the initialization, i.e. a small number of iterations of the Tabu Search (similar to a local search). This optimization is sometimes efficient for improving solutions because the Tabu Search works on all routes simultaneously.

5.6 Tabu Search

The Tabu Search algorithm is classical [Glover, 89] and has the same structure as in [Taillard, al., 1997]. The implemented algorithm can be summarized as follows:

- While the stopping criterion is not met Do
  - Generate the neighbourhood of the current solution by applying the operator CROSS exchange.
  - Select the best solution, which is not in the Tabu list, in this set of neighbours
  - If this solution is better than the best known solution Then
    - Save this solution as the best solution
  - Update the Tabu list

Here, a solution is a subset of routes from the adaptive memory. We now describe the major elements of this Tabu Search algorithm.
**Stopping criterion.** The stopping criterion is the same as in [Taillard, al., 1997]: a maximum number of iterations defined by:

\[ A \times \left( 1 + \frac{DR - 1}{B} \right) \]  \hspace{2cm} (11)

Where A and B are two parameters and DR is the number of iterations for the decomposition and reconstruction phases of a set of routes of a solution.

The stopping criterion depends on the decomposition and reconstruction phases because, with each run of the Tabu Search, the quality of the initial solution is improved. Thus the Tabu Search needs more operations to escape from a possibly local optimum.

**The operator CROSS exchange.** The neighbourhood is determined starting from the operator CROSS exchange. As in [Taillard, al., 1997], this operator is particularly well adapted for a VRP with time windows, but it needs to be adapted to our problem (competences, particular time window...). It is also a generalization of well-known operators like 2-opt or Or-opt. This operator consists in exchanging two segments between two routes as presented in Fig. 1.

![Figure 1](image)

In Fig. 1, X1' is the direct successor of X1, X2' of X2, Y1' of Y1 and Y2' of Y2. The following algorithm describes the general method used to calculate the neighbourhood between two routes:

- **Step 1:** For all sub-segments S1=[X1',Y1] in route of vehicle 1
- **Step 2:** If S1 contains only vertices for which vehicle 2 has the required equipment:
  - **Step 3:** For all sub-segments S2=[X2',Y2] in route of vehicle 2
  - **Step 4:** If S2 contains only vertices for which the first vehicle has the required equipment:
    - **Step 5:** If condition 1 is valid (see below) and the exchange improves the best known solution in the neighbourhood
    - **Step 6:** exchange these two segments between the two routes
- Step 7: If this new solution is feasible and better than the best known solution.
- Step 8: Save this solution as the best known.

In a Tabu Search algorithm, one of the most expensive elements in terms of computation time is the exhaustive neighbourhood exploration. We use a necessary condition to avoid building some unfeasible neighbours (condition 1 of step 5):

\[
\begin{align*}
    e_{X^2} + \varepsilon & \geq e_{X^1} + d_{X_1,X^2} \wedge e_{Y^1} + \varepsilon & \geq e_{Y^2} + d_{X_2,Y^1} \\
    \wedge e_{Y^2} + \varepsilon & \geq e_{Y^1} + d_{Y_1,Y^2} & \geq e_{Y^2} + d_{Y_2,Y'}
\end{align*}
\]  

(12)

This condition checks if vehicle 1 has enough time to pass from X1 to X2' and from Y2 to Y1' (vehicle 2 from X2 to X1' and Y1 to Y2' respectively).

Moreover, the objective value is calculated without performing the exchange. The movements which are not improving the solution are not kept. The aim is to improve the research speed. The evaluation of a movement can be calculated by:

\[
f(S') = f(S) - \sum_{i \in X_1}^{I_1} w_{i,\text{succ}(i)} + \sum_{i \in X_2}^{I_2} w_{i,\text{succ}(i)} + \sum_{i \in X'}^{\text{pred}(i)} w_{i,\text{succ}(i)} + \sum_{i \in X'}^{\text{pred}(i)} w_{i,\text{succ}(i)}
\]  

(13)

where \(\text{succ}(i)\) (\(\text{pred}(i)\)) is the successor (predecessor) of vertex \(i\), \(f(S)\) the value of the objective function of solution \(S\) and \(f(S')\) the value of neighbour \(S'\).

These conditions allow reducing the remaining neighbourhood between two routes. However, a solution obtained after a movement is not necessary valid for a given vehicle. Indeed, each vehicle has a maximum travel time which implies to check the feasibility of the two new routes.

**Tabu list.** As in [Taillard, al., 1997], the objective value of a solution is saved during a given number of iterations (half the maximum number of iterations of the Tabu Search) and a solution is Tabu when its objective value is in the Tabu list. This definition of a Tabu list is not classical but can be simply explained. Two solutions with the same objective value have a high probability for being identical. This is because of the important number of values of \(w_{i,j,p}\), which depends on vehicle \(k\), on the starting point of demand \(i\) and demand \(j\). Furthermore, determining if a solution is Tabu can be done very quickly. The size of the Tabu list is supposed to be sufficiently large for being considered of infinite size.

### 6 Computational experiments

In order to adjust the parameters and for testing the resolution algorithm, we have generated random instances with the following characteristics:
- the period is one day, between 7:00 pm (time 0) and 7:00 am (time 720),
- vehicles cannot be used more than 6 to 8 hours per day, the total number of vehicles \( m \) belongs to \{5,10,15,20,30,35,50,75\}; the number of private vehicles is equal to the half (truncated).
- public ambulances start from the same depot whereas private ambulances are divided geographically in \( m/3 \) sites.
- private ambulances are more expensive than public ambulances,
- transport demands have at least their starting point or their arrival point in one of the four hospitals of Tours (randomly selected with equiprobability),
- the waiting time of patients is bounded to \( \varepsilon = 15 \) minutes.
- the number of skills is equal to 3.
- the date of a transportation demands is generated either in the morning or in the afternoon (equiprobability). For the morning, the date is generated following a normal distribution \( \mathcal{N}(180,60) \) and for the afternoon \( \mathcal{N}(540,90) \). The aim is to respect the peaks of demands.

After preliminary experiments, the parameters have been set to the following values: number of initial solutions \( p = 20 \); size of the adaptive memory \( = 50 \times m \); number of iterations \( J = 50 \) and \( I = m \); \( A = 50 \) and \( B = 10 \); \( \rho = 3 \) divided by the number of routes in the adaptive memory.

Several types of data files with 100 instances each have been created with different numbers of demands. First, we have tested a resolution of the ILP with solver CPLEX v10 plus pre-processing and cutting. The CPU time is bounded for instances with more than 50 demands. Then, we have tested the Tabu Search algorithm. Table 1 presents the average computation time and a relative deviation \( \Delta \) between the two resolution methods. \( f(TS) \) stands for the value of the Tabu Search algorithm and \( f(ILP) \) for the value of the ILP resolution with Cplex.

\[
\Delta = 100 \times \frac{f(TS) - f(ILP)}{\max\{f(TS); f(ILP)\}}
\]

Column \( \%\text{opt} \) indicates the ratio of optimal solutions returned by the Tabu Search (for small and medium instances only).

<table>
<thead>
<tr>
<th>Number of demands + vehicles</th>
<th>ILP (Cplex)</th>
<th>Tabu Search with adaptive memory</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CPU (s)</td>
<td>( \Delta ) (%)</td>
</tr>
<tr>
<td>10 + 5</td>
<td>0,05</td>
<td>0,43</td>
</tr>
<tr>
<td>20 + 10</td>
<td>1,80</td>
<td>0,41</td>
</tr>
<tr>
<td>30 + 15</td>
<td>165</td>
<td>0,64</td>
</tr>
<tr>
<td>40 + 20</td>
<td>4291</td>
<td>1,67</td>
</tr>
<tr>
<td>60 + 30</td>
<td>200*</td>
<td>-16,20</td>
</tr>
<tr>
<td>70 + 35</td>
<td>300*</td>
<td>-20,23</td>
</tr>
</tbody>
</table>
*: because the computation time is bounded, the solution returned by Cplex is not necessarily optimal. Moreover, it may happen that Cplex does not find any solution.

We notice that for small and medium instances, the Tabu Search often returns the optimal solution or near optimal solutions. Furthermore, the computation time is smaller than the one of CPLEX. For large instances, the Tabu Search returns better solutions than CPLEX in a reasonable computation time.

Table 2 presents the relative improvement of a solution for larger instances. The relative deviation $\Delta'$ is calculated between the best of the $p$ random initial solutions and the best solution found by this method. Table 2 shows an improvement of more than 30% of the best initial solution, which proves the interest of the Tabu Search algorithm. However, because of the number of demands and vehicles, the computation time is much longer than that of Table 1.

<table>
<thead>
<tr>
<th>Number of demands + vehicles</th>
<th>Tabu Search with an adaptive memory</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\Delta'$ (%)</td>
</tr>
<tr>
<td>100 + 50</td>
<td>31,0%</td>
</tr>
<tr>
<td>150 + 75</td>
<td>33,4%</td>
</tr>
</tbody>
</table>

7 Conclusion

We have considered in this paper a transportation problem of patients between care units of a French hospital. The objective function was to minimize the total travel cost subject to a bound on the maximum waiting time of patients. We have proposed a Tabu Search algorithm inspired by [Taillard, al., 1997]. This method supports an adaptive memory which stores routes and consists in performing a Tabu Search algorithm several times: for improving the set of initial solutions, for the neighbourhood exploration and finally for improving the final solution. This method has been compared to CPLEX solver for small instances and to the initial solutions for larger instances. The results show that the method returns good solutions in a small computation time.

One of the first future research directions is to develop another heuristic algorithm for better comparison. Column generation seems to be a very promising method. Another direction is to consider that some demands arrive in real time and that the regulator of the ambulance service has to take these demands and to modify in real time the planning. Such a problem has already been considered in the literature and would be of great interest for the Hospital.
References