Publié par :
Published by:
Publicación de la:

Faculté des sciences de l'administration
Université Laval
Québec (Québec) Canada G1K 7P4
Tél. Ph. Tel. : (418) 656-3644
Télec. Fax : (418) 656-7047

Édition électronique :
Electronic publishing:
Edición electrónica:

Aline Guimont
Vice-décanat - Recherche et affaires académiques
Faculté des sciences de l'administration

Disponible sur Internet :
Available on Internet
Disponible por Internet :

http://rd.fsa.ulaval.ca/ctr_doc/default.asp
rd@fsa.ulaval.ca

DOCUMENT DE TRAVAIL 2006-013

OPTIMAL AND HEURISTIC SOLUTION PROCEDURES FOR A TRIP PACKING PROBLEM

Fayez F. BOCTOR
Jacques RENAUD
Fabien CORNILLIER

Version originale :
Original manuscript:
Version original:

ISBN – 2-89524-265-8

Série électronique mise à jour :
On-line publication updated :
Seria electrónica, puesta al dia

06-2006

Optimal and heuristic solution procedures for a trip packing problem

Fayez F. Boctor, **Jacques Renaud** and **Fabien Cornillier**
Centre de recherche sur les technologies de l'organisation réseau, Université Laval, Québec, Canada

Abstract

The trip packing problem is the problem of assigning a number of trips to a fleet of vehicles. Each trip has a specific duration and the working time of vehicles is limited. This paper considers a version the trip packing problem where the fleet is composed of a limited number of non identical vehicles and the revenue of each trip depends on the vehicle selected to do it. The paper provides a mathematical formulation of the problem and proposes some simple local search heuristics to solve it. A set of benchmark problem instances is created and used to analyse the performance of the proposed heuristics.

Keywords: Scheduling, Mixed integer models, Heuristics.

1. INTRODUCTION

The *trip packing problem* (TPP) usually arises as a part of the *multi-trip vehicle routing problem* (MTVRP). While some contributions try to solve the MTVRP as a whole (see Brandão and Mercer 1997 and 1998), others solve the problem in two steps: first generate the set of routes to be used and then pack these trips into subsets such that each subset can be performed within a working day (see Fleischmann 1990, Taillard *et al*. 1996 and Prins 2002).

Precisely, the trip packing problem is that of assigning a number of trips to a fleet of vehicles where each trip has a specific duration and the working time of vehicles is limited. Consequently the sum of durations of trips assigned to a given vehicle should not exceed the allowable working time of the vehicle. This working time includes regular working hours and a limited number of overtime hours. The objective is to minimize the total operations cost or to maximize the net revenue. Operation costs include overtime cost, if overtime is allowed, as well as variable and fixed operating costs of the used vehicles.

In the case of homogeneous unlimited fleet where trip cost (and revenue) is independent of vehicles and no overtime allowed, the TPP reduces to the standard bin packing problem where the objective is to minimize the number of vehicles to be used in order to minimize vehicle fixed costs. For a survey of the literature related to the bin packing problem see Coffman *et al*. (1996).

Allowing for overtime reduces the TPP to a more general bin packing problem where each bin has a minimal and a maximal size. Also, having non identical vehicles (bins) implies that some of the trips (objects) can be assigned to some but not all vehicles (bins). Finally, if trips revenue depends on vehicle selection, we should maximize revenues rather than minimizing costs. To the best of our knowledge these versions of the bin packing problem has never been the subject of any publication.

Very few contributions to the solution of the trip packing problem are reported in the open literature. Fleischmann (1990) studied the MTVRP problem and suggested using bin packing in the final part of the algorithm. Taillard *et al*. (1996) start by generating a large number of routes satisfying the VRP constraints and finish by assembling routes with a bin packing heuristic into feasible working days. In both cases, the considered fleet is composed of unlimited number of identical vehicles and the objective is to minimize the number of vehicles. Prins (2002) considered the case of heterogeneous unlimited fleet and used an adaptation of the first fit decreasing algorithm to pack trips into a minimum number of trucks.

2. THE CONSIDERED PROBLEM

This paper considers a special version of the trip packing problem arose in solving the petrol station replenishment problem (see Taqa Allah *et al*. 2000, Malépart *et al*. 2003 and Cornillier *et al*. 2005, Cornillier *et al*. 2006). In this real-life problem, we have a limited number of non identical tank-trucks and a limited number of overtime hours are allowed. Having a heterogeneous fleet implies that the quantities to be delivered, and consequently revenue generated, by a given trip may depend on the truck to which the trip is assigned. Heterogeneous fleet also implies that it may not be feasible to assign some trips to a given vehicle as the corresponding demand may exceed the capacity of the vehicle.

In summary, we consider the trip packing problem under the following assumptions:

- Limited number of non identical trucks. Trucks are owned by the transportation company and, consequently, trucks fixed cost is paid whether they are used or not,
- In addition to regular working hours, each truck can work a limited number of overtime hours.

- A fraction of an overtime hour is paid as one hour,
- For each truck, some trips are feasible and others are not depending on the truck capacity,
- A trip revenue depend on the truck to be used for the trip.

3. MATHEMATICAL FORMULATIONS

The following notation will be used to formulate the trip packing problem:

I	set of trips to be packed; $\|I\|=n$
J	set of available trucks; $\|J\|=m$
i	trip index, $i \in I$
j	truck index; $j \in J$
L	maximum regular work hours
M	maximum number of overtime hours
C	overtime marginal cost per hour
d_i	duration of trip i (in hours)
J_i	set of trucks that can perform trip i, $\|J_i\|=m_i$
I_j	set of trips that can be done by truck j
V_{ij}	revenue of trip i if assigned to truck j
x_{ij}	binary equals 1 if trip i is assigned to truck j
A_j	overtime hours for truck j

Using this notation, our version of the trip packing problem can be formulated as follows.

***Formulation 1* (*F*1)**

Find: A_j integer ≥ 0 and $x_{ij} \in \{0,1\}$, $\forall i \in I$ and $j \in$ J that:

Maximize: $$Z_1 = \sum_{i \in I} \sum_{j \in J_i} V_{ij} x_{ij} - C \sum_{j \in J} A_j$$

Subject to:

$$\sum_{j \in J_i} x_{ij} = 1 \qquad ; i \in I$$

$$\sum_{i \in I_j} d_i x_{ij} - A_j \leq L \quad ; j \in J$$

$$A_j \leq M \qquad ; j \in J$$

In this formulation, the objective is to maximize the overall revenue minus the overtime marginal cost. Notice that A_j should take an integer value as a fraction of an over time hour is paid the same amount of money as a whole hour. The first set of constraints assigns each trip to one and only one truck. The second set allows determining the overtime hours for each truck and the third set puts an upper limit on the overtime hours.

In case where revenues do not depend on trucks to which trips are assigned, the problem reduces to minimizing overtime hours under the same set of constraints. Thus the formulation reduces to:

***Formulation 2* (*F*2)**

Find: A_j integer ≥ 0 and $x_{ij} \in \{0,1\}$, $\forall i \in I$ and $j \in$ J that:

Minimize: $Z_2 = \sum_{j \in J} A_j$

Subject to:

$$\sum_{j \in J_i} x_{ij} = 1 \qquad ; i \in I$$

$$\sum_{i \in I_j} d_i x_{ij} - A_j \leq L \quad ; j \in J$$

$$A_j \leq M \qquad ; j \in J$$

Also, in case where either overtime is not allowed or all working hours are considered as regular and we pay only for real working hours, the problem reduces to maximize total revenue. In addition, the constraints can be simplified as follows:

***Formulation 3* (*F*3)**

Maximize: $Z_3 = \sum_{i \in I} \sum_{j \in J_i} V_{ij} x_{ij}$

Subject to:

$$\sum_{j \in J_i} x_{ij} = 1 \qquad ; i \in I$$

$$\sum_{i \in I_j} d_i x_{ij} \leq L \qquad ; j \in J$$

The second and third sets of constraints of previous models are replaced by one set (the second set) which insures that the sum of the duration of trips assigned to truck *j* is less than or equal to *L*, the maximum available working hours. Recall that in this formulation, all working hours are regular hours.

Numerical illustration

Let us consider an example where we need to assign 10 trips (n=10) to 4 trucks (m=4) over a period including 12 regular hours (L =12) and at most 3 overtime hours (M=3). The marginal cost of overtime hours is 1 (C=1). Trip durations and revenues are given in Table 1 where **NF** indicates unfeasible assignments.

Table 1: Duration and revenues for the numerical example

i	1	2	3	4	5	6	7	8	9	10
d_i	3.5	5.5	3	4.5	7	7.5	2	9	4.5	3.5
V_{i1}	10	8	**NF**	10	9	**NF**	9	**NF**	8	9
V_{i2}	9	10	**NF**	9	10	9	**NF**	8	10	**NF**
V_{i3}	**NF**	**NF**	9	8	10	8	10	9	**NF**	8
V_{i4}	9	10	8	10	**NF**	10	**NF**	9	9	**NF**

For this numerical example, the optimal solution of $F1$ is:

- Trips 1, 4 and 10 assigned to truck 1 (total duration 11.5 hours);
- Trips 5 and 9 assigned to truck 2 (total duration 11.5 hours);
- Trips 3, 7 and 8 assigned to truck 3 (total duration 14 hours); and
- Trips 2 and 6 assigned to truck 4 (total duration 13 hours).

This solution gives total revenue of 97 and 3 overtime hours. Thus Z_1^* =94.

4. PROPOSED HEURISTICS

In this paper we propose two heuristic approaches to solve the TPP as expressed by $F1$. The first approach is a multi-start local improvement heuristic and the second one is an adaptation of simulated annealing algorithm with several reheating cycles (see Boctor 1996).

4.1. MULTI-START LOCAL IMPROVEMENT HEURISTIC

The general framework of the multi-start local improvement heuristic (MSLIH) is as follows:

Repeat ***P*** times the following steps and retain the best obtained solution:

- Select a new construction heuristic or change the parameters of the used one and use it to construct an initial solution,
- Apply some improvement heuristics (one by one) until no further improvement can be reached by any of them.

In the following we present the steps of two construction heuristics and those of four improvement methods. All the proposed construction heuristics belong to the greedy heuristics family and all the proposed improvement heuristics are position exchange heuristics. Construction heuristics consider trips in a given order and may fail to find a feasible solution with the selected order. That is why steps 2 and 3 of these heuristics are repeated at most $\boldsymbol{Q_{max}}$ times using a different order each time. Afterwards, if no solution is found, we consider that the heuristic fails to find a feasible solution.

Construction heuristics

***Heuristic C*1**

1. Number trips according to some characteristic (e.g., in the descending order of their duration d_i or in the ascending order of m_i, the number of trucks able to perform each of them). Set Q=0.
2. Increment Q. If $Q>Q_{max}$ then the heuristic fails to find a feasible solution. Otherwise, set all trips free (unassigned) and $u_j = 0$, $\forall j \in J$ (u_j is the sum of the duration of trips assigned to truck j). Go to step 3.
3. For i=1 to n, do
   - 3.1 Assign trip i to the truck that can perform it within its regular hours (i.e., such that $u_j+d_i \leq L$,) while adding the largest value to the objective function. Let k be the selected truck, update u_k (i.e., set $u_k = u_k+d_i$.)
   - 3.2 If there is no such truck, assign trip i to the truck k such that:

     $$k = \arg\max_j \{V_{ij} - C(u_j+d_i-L) \mid u_j+d_i \leq L+M\}.$$

     In case of tie choose the truck having the smallest number. Update u_k.
   - 3.3 If there is still no such truck then find all other already-assigned trips ℓ that can be reassigned to another truck, say k (*i.e.*, $u_k+d_\ell \leq L+M$), leaving enough time to allow adding trip i to the original truck of ℓ, say truck j (*i.e.*, such that $u_j+d_i-d_\ell \leq L+M$). Let ℓ^* be the best among these trips (*i.e.*, the one leading to the highest net revenue), j^*its original truck and k^* the truck where it can be added. Move trip ℓ^* to k^* and add trip i to j^*. Update u_{k^*} and u_{j^*}.
   - 3.4 If trip i is still free, move it to the top of the trips list and go back to step 2.

Heuristic C2

Heuristic *C*2 is similar to *C*1 except that we modify step 3.1 as follows:

3.1 If trip *i* is free, assign it to the truck that can do it within its regular hours (i.e., such that $u_j+d_i \leq L$, where u_j is the sum of the duration of trips assigned to truck *j*) while adding the largest value to the objective function. Update u_j. and if there is a trip such that its duration equals $L-u_j$, then assign it to *j*. In case of tie assign the one adding the highest net revenue.

Versions of the proposed construction heuristics

Several versions of the proposed construction heuristics can be used. We call *C*1*d*, the version of *C*1 where trips are numbered in the descending order of their duration, *C*1*m*, the version where trips are numbered in the ascending order of the number of trucks able to perform them, and *C*1*r*, the version where trips are numbered randomly. Similarly, we will use *C*2*d*, *C*2*m* and *C*2*r*.

Improvement Heuristics

Heuristic I1: Until no better feasible solution can be reached:

- For every trip that can be moved to another truck, calculate the corresponding improvement of the objective function.
- Move the trip leading to the largest positive improvement while the solution remains feasible.

Heuristic I2: Until no better feasible solution can be reached:

- For every pair of trips that can exchange positions, calculate the corresponding improvement of the objective function.
- Exchange the pair leading to the largest positive improvement while the solution remains feasible.

Heuristic I3: Until no better feasible solution can be reached:

- For every triplet of trips (*i*,*j*,*k*) where *i* and *j* are assigned to the same truck and *k* to another one, calculate the change of the objective function if *i* and *j* are moved to the tuck of *k* and vice-versa.
- Exchange positions of the triplet leading to the largest positive improvement while the solution remains feasible.

Heuristic I4: Until no better feasible solution can be reached:

- For every trip triplet (*i*,*j*,*k*) assigned to 3 different trucks, calculate the change of the objective function if *i* replace *j*, *j* goes to the truck of *k* and , *k* replaces *i*.
- Exchange positions of the triplet leading to the largest positive improvement while the solution remains feasible.

Numerical illustration

In the following we apply some of the proposed heuristics to the numerical example presented in section 3.

Heuristic C1d

In step 1 we arrange trips in the ascending order of their duration. In step 2 we set all trips free and all trucks unused. The needed iterations of step 3 are summarized in Table 2.

Table 2: Step 3 of heuristic *C*1*d*

Trip	Duration	Selected Truck	Revenue minus overtime cost	Remaining regular (used overt time) hours of truck			
				1	2	3	4
8	9	3	9	12	12	3	12
6	7.5	4	10	12	12	3	4.5
5	7	2	10	12	5	3	4.5
2	5.5	1	8	6.5	5	3	4.5
4	4.5	1	10	2	5	3	4.5
9	4.5	2	10	2	0.5	3	4.5
1	3.5	4	9	2	0.5	3	1
10	3.5	1	9-2	(1.5)	0.5	3	1
3	3	3	9	(1.5)	0.5	0	1
7	2	3	10-2	(1.5)	0.5	(2)	1
Total			90				

Improvement: ***I1*** does not lead to any improvement. ***I2*** brings us to move trip 1 to truck 1 and trip 2 to truck 4 leading to total revenue of 94. ***I3*** and ***I4*** do not lead to any improvements.

Heuristic C2d

Again, in step 1 we arrange trips in the ascending order of their duration. In step 2 we set all trips free and all trucks unused. The needed iterations of step 3 are summarized in Table 3. The second row of the table shows that trip 3 was selected in application of step 3.1 as its duration equals the remaining time of truck 3 (d_3=L-u_3).

Table 3: Step 3 of heuristic *C2d*

Trip	Duration	Selected Truck	Revenue minus overtime cost	Remaining regular (used overt time) hours of truck			
				1	2	3	4
8	9	3	9	12	12	3	12
3	3	3	9	12	12	0	12
6	7.5	4	10	12	12	0	4.5
4	4.5	4	10	12	12	0	0
5	7	2	10	12	5	0	0
2	5.5	1	8	6.5	5	0	0
9	4.5	2	10	6.5	0.5	0	0
1	3.5	1	10	3	0.5	0	0
10	3.5	1	9-1	(0.5)	0.5	0	0
7	2	3	10-2	(0.5)	0.5	(2)	0
Total			92				

Improvement: ***I*1** does not lead to any improvement. ***I*2** brings us to move trip 2 to truck 4 and trip 4 to truck 1 leading to total revenue of 94. ***I*3** and ***I*4** bring no improvement.

4.2. ADAPTATION OF THE SIMULATED ANNEALING ALGORITHM

The general framework of the proposed adaptation of the simulated annealing (SA) algorithm is given in Figure 1. Within this adaptation we execute H heating cycles and within each cycle we start with an initial temperature T_{max} and end with a final temperature T_{min}. The temperature is reduced using a reduction coefficient α once we reach R_{max} iterations without improving the best feasible solution found so far. It was shown (Boctor 1996) that performing several heating cycles improves the final results.

At each iteration we test a neighbour of the current solution. This neighbour is either feasible or unfeasible but a penalty p_1 is added if a trip is assigned to a truck that cannot curry out it and another penalty p_2 is added for every working hour that exceeds the allowable hours. Three neighbourhood structures are used and we randomly select one of them. Then we randomly select

one neighbour in the randomly selected neighbourhood. These neighbourhood structures are: (1) move a randomly selected trip to randomly selected truck, (2) permute the trucks of two randomly selected trips, and (3) circularly permute the trucks of three randomly selected trips. This neighbour selection procedure allows investigating larger number of different neighbours and reduces the chance that the search procedure be trapped in a local optimum.

- Use a construction heuristic to construct an initial feasible solution
- Store the obtained solution as the current solution and as the best solution found so far
- Initialize the heating cycle counter h:=0
- Repeat until $h=H$
    - Initialize the cooling temperature $T=T_{max}$
    - Repeat until $T<T_{min}$
        - Initialize the repetition counter r:=0
        - Repeat until $r=R_{max}$
            - Randomly select a neighbourhood definition among those to be used
            - Randomly select a neighbour of the current solution from the selected neighbourhood
            - Let δ be the difference between the value associated to the neighbour solution and that of the current solution
            - If $\delta>0$ or a randomly drawn value is less than $e^{-\delta/T}$ then
                - Store the neighbour solution as the new current solution
                - If the neighbour solution is feasible and better than the best found so far store it as the new best solution and set r:=0
            - End If
        - End Repeat
        - Set $T=\alpha T$
    - End Repeat
- End Repeat

Figure 1: General Framework of the proposed adaptation of the simulated annealing algorithm

5. COMPUTATIONAL RESULTS

One hundred randomly-generated instances of the problem were used to asses the quality of the solutions produced by the proposed heuristics. Each of these instances includes 50 trips and 15 trucks. Durations are drawn from a uniform distribution between 30 and 390 minutes while revenues are drawn from a uniform distribution between 700 and 1200. Regular working hours are 12 and there is a maximum of 3 overtime hours. Additional overtime hourly cost is 60.

All the 100 instances were solved to optimality by the commercial MIP code Cplex. The average computation time for Cplex is 5 120 seconds with a minimum of 5 seconds, a maximum of 61 829 seconds and a standard deviation of 10 995 seconds.

Construction heuristics and best parameters

Table 4 presents the results obtained by each of the proposed construction heuristics. Several values of Q_{max}, the number of times the trip list is reordered, were tested and we found that it was possible to obtain feasible solutions for all problems with Q_{max}=2. Average computational time (clock time) for all these heuristics is less than 0.01 seconds. The Table shows that best results were obtained by *C*1*m* followed by *C*1*r*. Table 4 also shows that all versions of *C*1 produced better results than the corresponding versions of *C*2.

Table 4: Results obtained by construction heuristics

Heuristic	*C*1*d*	*C*2*d*	*C*1*m*	*C*2*m*	*C*1*r*	*C*2*r*
Average deviation from the optimum	4.34%	4.82%	2.99%	3.28%	3.21%	3.61%
Standard deviation	1.18%	1.31%	0.89%	0.98%	0.91%	1.01%
Minimum deviation from the optimum	1.89%	1.94%	1.30%	0.91%	1.24%	0.91%
Maximum deviation from the optimum	7.84%	8.39%	4.69%	6.03%	5.53%	6.86%
Number of times it produced the best solution	12	4	36	22	32	18
Number of times it was the only to produce the best solution	8	0	23	9	22	8
Number of times it produced the worst solution	25	58	8	6	5	17
Number of times it was the only to produce the best solution	10	43	6	4	2	14

Multi-start local improvement heuristic and best parameters

This heuristic approach requires selecting the value of *P* the number of restarts. Table 5 presents the results obtained by the heuristic with *P*=1 (i.e. only one start) in function of the heuristic used to construct the initial solution. Although the average deviation from the optimum is quite small, no one of the different versions of the MSLIH ever produced the optimal solution. Average computational time (clock time) for all these heuristics is between 1.37 and 1.54 seconds. Results show that using *C*1*d*, *C*2*d* or *C*1*r* to construct the initial solution produced the best results in average. However, using the means-comparison statistical test, we can conclude that there is no significant difference between the obtained average deviations.

It is worth noting that if we apply the composite MSLIH 6 times while starting each time with a different one of the 6 proposed construction heuristics, we obtain an average deviation from the

optimum of 0.21% in less than 9 seconds in average. The corresponding minimum and maximum deviation from the optimum are 0.01% and 0.53% respectively.

Table 5: Results obtained by multi-start local improvement heuristic with P=1

Initial solution by	*C*1*d*	*C*2*d*	*C*1*m*	*C*2*m*	*C*1*r*	*C*2*r*	*Best*
Average deviation from the optimum	0.40%	0.40%	0.43%	0.42%	0.40%	0.43%	0.21%
Standard deviation	0.20%	0.22%	0.21%	0.23%	0.21%	0.22%	0.11%
Maximum deviation from the optimum	1.08%	1.60%	1.06%	1.15%	1.11%	1.11%	0.53%
Minimum deviation from the optimum	0.05%	0.08%	0.01%	0.01%	0.03%	0.02%	0.01%
Number of times it produced the best solution	25	22	16	17	23	18	-
Number of times it was the only to produce the best solution	14	11	11	12	18	13	-
Number of times it produced the worst solution	16	15	20	22	21	28	-
Number of times it was the only to produce the worst solution	9	6	14	16	14	21	-
Average computation time (clock time) in seconds	1.53	1.54	1.43	1.45	1.37	1.46	8.78

Table 6 presents the results obtained for different values of P. The initial solution was always constructed using *C*1*r*. As shown in the table very good results were obtained with P=20. More generally, results improve as P increases. However, computation time also increases as P increases. Thus the value of P to be used should be chosen in function of the user preferences.

Table 6: Results obtained using different values of P (initial solution by *C*1*r*)

P	1	2	3	5	9	15	20
Average deviation from the optimum	0.40%	0.32%	0.25%	0.19%	0.16%	0.12%	0.09%
Standard deviation	0.21%	0.21%	0.14%	0.13%	0.10%	0.09%	0.07%
Maximum deviation from the optimum	1.11%	1.38%	0.69%	0.65%	0.63%	0.43%	0.30%
Minimum deviation from the optimum	0.03%	0.02%	0.00%	0.00%	0.00%	0.00%	0.00%
Number of times it produced the optimal solution	0	0	1	5	3	7	10
Average computation time (clock time) in seconds	1.37	2.74	4.28	5.07	12.47	21.49	28.06

Simulated annealing and best heating and cooling schedules

This heuristic approach requires selecting the value several parameters: H the number of heating cycles, T_{max} the initial temperature, T_{min} the final temperature, α the temperature reduction coefficient and R_{max} the number of repetitions without improving the best obtained

solution. We also need to fix the infeasibility penalties p_1 and p_2. A very large number of parameter combinations were tested and Table 7 presents the results obtained by some of the best combinations in the ascending order of their average computation time. For all these results we used: T_{max} =16, T_{min} =1, α =0.5, p_1=10000 and p_2=600.

From Table 7 we can see that average percentage deviation decreases as computation time increases. Also, comparing the results of Tables 6 and 7 we can conclude that, using the same computation time, the multi-start local improvement heuristic produces better results than our adaptation of the simulated annealing algorithm (see Figure 2). For example, with about 5 seconds of computing time, the multi-start local improvement heuristic produced an average deviation of 0.19% while the simulated annealing adaptation produced an average deviation of 0.24% in about 6.5 seconds. With about 12.5 seconds of computing time the average deviations are respectively 0.16% and 0.17%.

Table 7: Results obtained the simulated annealing adaptation in function of its parameters

H	6	6	10	6	10	6	10	10
R_{max}	10 000	15 000	10 000	20 000	15 000	25 000	20 000	25 000
Average deviation from the optimum	0.24%	0.23%	0.22%	0.17%	0.18%	0.17%	0.16%	0.14%
Standard deviation	0.17%	0.16%	0.17%	0.14%	0.13%	0.13%	0.13%	0.13%
Maximum deviation from the optimum	0.86%	0.89%	1.06%	0.77%	0.80%	0.65%	0.70%	0.76%
Minimum deviation from the optimum	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Number of times it produced the optimal solution	2	2	2	9	4	5	8	12
Average computation (clock) time in seconds	6.54	9.61	10.30	12.64	15.40	15.77	19.68	25.50

6. CONCLUSIONS

This paper presented a more general version of the trip packing problem, gave some mathematical formulations of the problem, developed heuristic solution procedures to solve it and presented a computational experiment to assess the quality of the results obtained by these heuristics.

Mainly, we show that with little computational effort we can produce solutions quite close to the optimum.

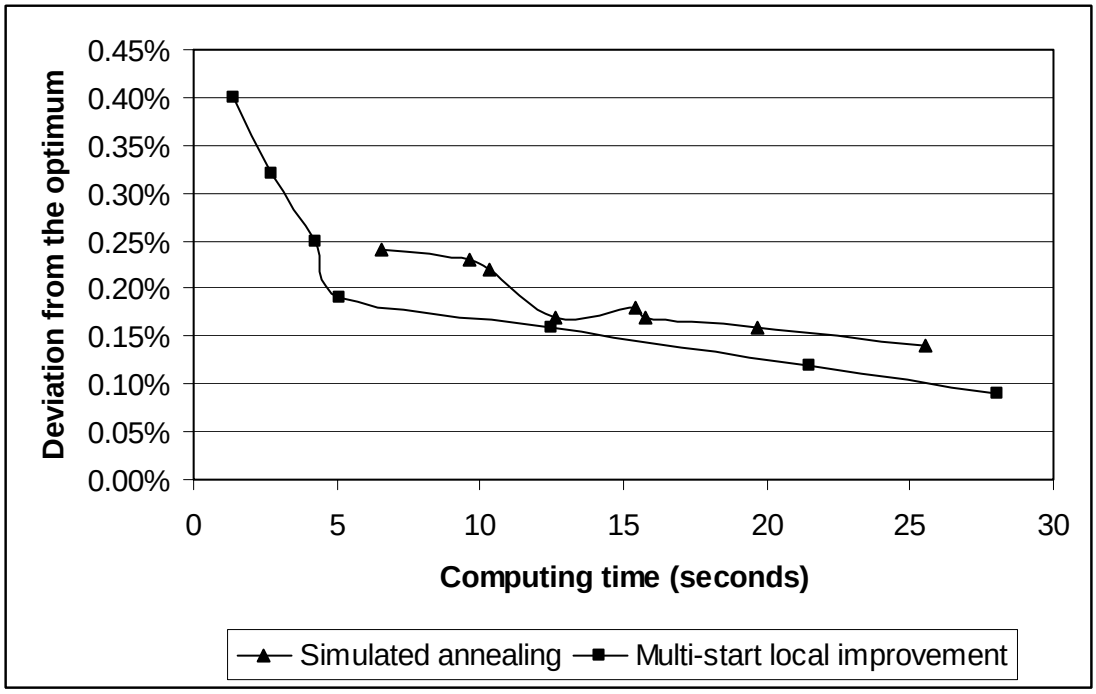

Figure 2: Deviation from optimum in function of computing time

Acknowledgement

This research work was partially supported by the *Canadian Natural Sciences and Engineering Research Council* (*NSERC*) under grants OGP0172633 and OPG0036509. This support is gratefully acknowledged.

References

Boctor, F. F. (1996) Resource-constrained project scheduling by simulated annealing, *International Journal of Production Research*, 34, 8, 2335-2351.

Brandão, J. and A. Mercer, (1997) A tabu search algorithm for the multi-trip vehicle routing and scheduling problem, *European Journal of Operational Research,* 100, 180–191.

Brandão, J. and A. Mercer, (1998) The multi-trip vehicle routing problem, *Journal of the Operational Research Society*, 49, 799–805.

Coffman, E., M. Garey and D. Johnson (1996) Approximation algorithms for bin packing: A survey, in D. Hochbaum (editor) *Approximation algorithms for NP-hard problems*, PWS Publishing, Boston, 46-93.

Cornillier F., F. Boctor, G. Laporte and J. Renaud (2005) An Exact algorithm for the petrol station replenishment problem, Working paper 2005-17, Faculté des sciences de l'administration, Université Laval.

Cornillier F., F. Boctor, J. Renaud and G. Laporte (2006) A heuristic for the multi-period petrol station replenishment problem, Working paper, Faculté des sciences de l'administration, Université Laval.

Fleischmann, B. (1990) The vehicle routing problem with multiple use of vehicles, Working paper, Fachbereich Wirtschaftswissenschaften, Universität Hamburg.

Malépart V., F. F. Boctor, J. Renaud, et S. Labilois. (2003) Nouvelles approches pour l'approvisionnement des stations d'essence. *Revue Française de Gestion Industrielle*, 22, 15–31.

Prins, Ch. (2002) Efficient heuristics for the heterogeneous fleet multi-trip VRP with application to a large-scale real case, *Journal of Mathematical Modelling and Algorithms,* 1, 135–150.

Taillard E., G. Laporte and M. Gendreau (1996) Vehicle routeing with multiple use of vehicles. *Journal of the Operational Research Society*, 47, 1065-1070.

Taqa Allah, D., J. Renaud et F.F. Boctor (2000) Le problème d'approvisionnement des stations d'essence, *Journal Européen des Systèmes Automatisés*, 34, 1, 11-33.