Contents lists available at GrowingScience

Decision Science Letters

homepage: www.GrowingScience.com/dsl

A heuristic algorithm based on tabu search for vehicle routing problems with backhauls

Jhon Jairo Santa Chávez[a], John Willmer Escobar[b*], Mauricio Granada Echeverri[c] and César Augusto Peñuela Meneses[d]

[a]Programa de Ingeniería Eléctrica, Universidad Libre Seccional Pereira and Universidad Tecnológica de Pereira, Colombia
[b]Department of Industrial and Civil Engineering, Pontificia Universidad Javeriana Cali, Colombia
[c]Programa de Ingeniería Eléctrica, Universidad Tecnológica de Pereira, Colombia
[d]Universidad Libre Seccional Pereira

C H R O N I C L E

Article history:
Received January 16, 2017
Received in revised format:
May 22, 2017
Accepted May 28, 2017
Available online
May 28, 2017

Keywords:
Freight transportation
Vehicle routing problem
Mathematical modeling
Exact model
Combinatorial optimization
Tabu search
Computational simulation
Backhauling

A B S T R A C T

In this paper, a heuristic algorithm based on Tabu Search Approach for solving the Vehicle Routing Problem with Backhauls (VRPB) is proposed. The problem considers a set of customers divided in two subsets: Linehaul and Backhaul customers. Each Linehaul customer requires the delivery of a given quantity of product from the depot, whereas a given quantity of product must be picked up from each Backhaul customer and transported to the depot. In the proposed algorithm, each route consists of one sub-route in which only the delivery task is done, and one sub-route in which only the collection process is performed. The search process allows obtaining a correct order to visit all the customers on each sub-route. In addition, the proposed algorithm determines the best connections among the sub-routes in order to obtain a global solution with the minimum traveling cost. The efficiency of the algorithm is evaluated on a set of benchmark instances taken from the literature. The results show that the computing times are greatly reduced with a high quality of solutions. Finally, conclusions and suggestions for future works are presented.

© 2018 Growing Science Ltd. All rights reserved.

1. Introduction

The Vehicle Routing Problem (VRP) belongs to the classic logistic problems considered in the literature. The VRP consists on carrying products from a depot (S) to a set of customers (D), by a set of edges with associated costs c_{ij} ($i \in S \cup D$ and $j \in S \cup D$). The objective is to find a set of routes, which begin and end at the depot minimizing the traveling cost. The demand for all customers must be satisfied (Oliveira, 2004). Several exact algorithms have been proposed for solving the classic version

* Corresponding author.
E-mail address: jwescobar@javerianacali.edu.co (J.W. Escobar)

© 2018 Growing Science Ltd. All rights reserved.
doi: 10.5267/j.dsl.2017.6.001

of the VRP efficiently (Deif & Bodin, 1984). However, exact algorithms have proved to be optimal only on small and medium size instances for different variants of the VRP.

Several variants and considerations of the VRP have been proposed in the literature, such as capacity of the fleet of vehicles, heterogeneous fleet of vehicles, time windows for each customer, pickup and delivery nodes (Laporte, 2009; Aguado, 2009). Therefore, the vehicle routing problem becomes a more complicated problem that frequently exceeds the capacity of classical solution strategies, i.e. nonlinear techniques (Bazaraa et al., 2006). Then, the computational complexity is increased, and more sophisticated algorithms for solving the VRP are required (Dorigo & Gambarella, 1997). Successful applications of VRP variants have been proposed by Chávez et al. (2016), Escobar & Linfati (2012), Escobar (2014), Escobar et al. (2013), Escobar et al. (2014a), Escobar et al. (2014b) and Escobar et al. (2015).

In order to reduce the number of empty travels performed by each vehicle, the idea of pick-up and delivery of goods has been considered in the literature (Lu & Dessouky, 2004). However, this consideration requires special attention to the handling of cargo inside the vehicles. Therefore, a smart design of corridors to allow the easy access to all the products stored in the vehicles should be required. One of the variants of the VRP considers that each vehicle must deliver all cargo until it is empty. Then, each vehicle may travel to the nodes for which the demand of the collected products exist in order to transport these products to the starting point of the route (Deif & Bodin, 1984). This kind of problem is known as Vehicle Routing Problem with Backhauling (VRPB).

In this paper, we propose a heuristic algorithm based on a Tabu Search Approach for solving the VRPB. The proposed algorithm considers intensification of local information and the proper exploitation of local search procedures combined with exploratory stochastic strategies. The solution strategy considers the design of separate routes for both delivery and collection of goods. In order to find high quality solutions the algorithm tries to find the best sequence of customers for each route. Additionally, the best connection among *linehaul routes* and *backhaul routes* have been performed. The proposed methodology is implemented and tested on well-known instances from the literature (Mingozzi et al., 1999; Toth & Vigo, 1999a, 1999b). The proposed algorithm obtains good quality solutions in short computing times.

2. Mathematical model for the VRPB

The VRPB could be modeled by the formulation of a binary programming model, as described in Baldacci et al. (1999). The formulation is described bellow: let L be the set of customers with the requirement of products from the depot (Linehauling), and B be the set of customers with the requirement of collection of products (Backhauling). Let us define the set of feasible routes G_L and G_B. Each route $i \in G_L$ only contains nodes belonging to L, and it starts at the depot and finishes at a given linehaul customer. On the other hand, each route $j \in G_B$ only contains customers belonging to B, starts at a given backhaul customer and ends at the depot. In addition, let L_c be the set of routes on G_L, which cross the customer $c \in L$; and L_c^T be the set of routes on G_L, which finish at customer $c \in L$. Furthermore, Bc denotes the set of routes in G_B passing through the customer $c \in B$, while B_c^I considers the set of routes of G_B, which finish at customer $c \in B$. Finally, let consider ξ_{ij} be the set of the edges, which allow joining the customer $i \in L$ and the customer $j \in B$. The goal of the VRPB is to determine the routes to be performed in order to fulfill the requirements of the customers with the minimum traveling cost. Each customer must be visited exactly once by a single route and the total demand of each route must not exceed the vehicle capacity Q.

In the following formulation, three types of binary variables are considered. x_i is equal to1, if the route $i \in G_L$ is selected as part of the solution; otherwise assumes the value of 0. y_j takes the value of 1, if the route $j \in G_B$ is selected as part of the solution; otherwise takes the value of 0. Finally, z_{ij} assumes the value of 1 if the edge connecting $i \in G_L$ with the route $j \in G_B$ belongs to the solution, and 0 otherwise. The mathematical model for the VRPB is formulated by equations (1) – (7), where M is the number of available homogeneous vehicles at the depot; $cost\_l_i$ is the overall cost of the route $i \in G_l$, $cost\_b_i$ is the overall cost of the route $j \in G_B$, and $cost\_z_{ij}$ is the cost of traveling through the edge $(i,j) \in \xi_{ij}$ which connects the node $i \in L$ with the node $j \in B$.

$$\min z = \sum_{i \in G_L} cost\_l_i \cdot x_i + \sum_{j \in G_B} cost\_b_i \cdot y_l + \sum_{i \in L} \sum_{j \in B} cost\_z_{ij} \cdot z_{ij} \tag{1}$$

Subject to

$$\sum_{l \in L_c} x_l = 1, \qquad \forall\, c \in L \tag{2}$$

$$\sum_{j \in B_c} y_j = 1, \qquad \forall\, c \in B \tag{3}$$

$$\sum_{l \in L_c^T} x_l - \sum_{j \in B} z_{cj} = 0, \qquad \forall\, c \in L \tag{4}$$

$$\sum_{l \in B_c^I} y_l - \sum_{i \in L} z_{ic} = 0, \qquad \forall c \in B \tag{5}$$

$$\sum_{i \in L} \sum_{j \in B} z_{ij} = M \tag{6}$$

$$x_l \in \{0,1\}\,,\; y_l \in \{0,1\},\; z_{ij} \in \{0,1\} \tag{7}$$

3. Proposed algorithm based on Tabu Search

The mathematical model described by Eqs. (1-7) could be solved efficiently by using exact methods when the number of customers in the system is relatively small. If the number of customers to be fulfilled is increased, then the space of solution grows exponentially. In these cases, the implementation of approximation algorithms (heuristics or metaheuristics) could be a good option to solve the VRPB. In fact, these types of approaches reduce the complexity of the search process based on optimality conditions (Baldacci et al. 1999). In particular, we have applied an algorithm based on Tabu Search heuristic (Glover, 1989) to solve the VRPB. The Tabu Search (TS) is characterized by solving complex combinatorial optimization problems by using local search criteria. For this reason TS is also known as a neighborhood-based approach.

3.1 Encoding the solution

Fig. 1 sketches the representation for any alternative of solution in the proposed algorithm. Here, the position x_{ik} with $k = 1,2,..,L_i$ and $i = 1,2,..,K_L$, stores the customer to be visited at the turn k by the route i, where Li denotes the number of customers to be served by the linehaul route i.

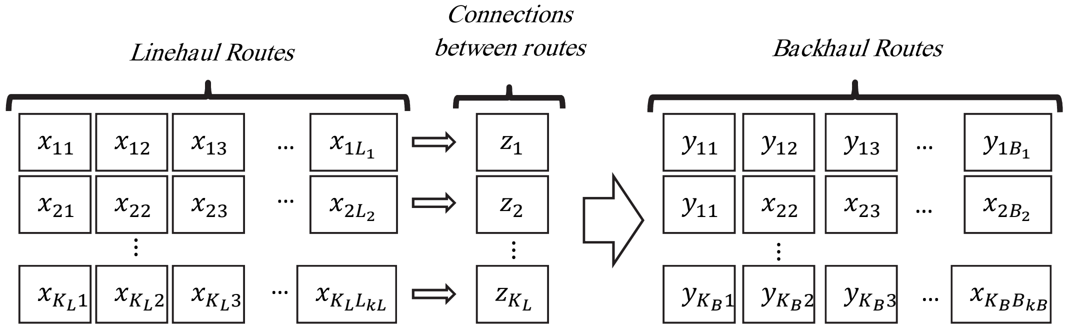
Linehaul Routes
Connections between routes
Backhaul Routes
x_{11} x_{12} x_{13} … x_{1L_1}
x_{21} x_{22} x_{23} … x_{2L_2}
x_{K_L1} x_{K_L2} x_{K_L3} … $x_{K_LL_{kL}}$
z_1
z_2
z_{K_L}
y_{11} y_{12} y_{13} … y_{1B_1}
y_{11} x_{22} x_{23} … x_{2B_2}
y_{K_B1} y_{K_B2} y_{K_B3} … $x_{K_BB_{kB}}$

Fig. 1. Encoding of one alternative of solution

Similarly, the position y_{jl} with $l = 1,2,..,B_j$ and $j = 1,2,..,K_B$, stores the customer to be visited at the turn l by the route j. Furthermore, Bj determines the number of customers to be served by the backhaul route j. Finally, the position z_i stores the index of the backhaul route that is merging with the linehaul route i. If z_i assumes a zero value then the vehicle in the route i return to the depot without serving any pickup demand. Such case only happen when $K_L > K_B$.

3.2 Initial Solution

The initial solution for the proposed algorithm is obtained by a constructive approach based on the nearest neighborhood, i.e. the lower distance to the current location. Hence, starting from the depot, each route receives the customer with the lower distance to the depot among the customers not assigned yet. This rules is sequentially repeated until all delivery customers have been assigned. However, the distance to be considered is the lower distance to the last customer belonging to the current route. Then, backhaul customers are connected to the last linehaul customer for each route. At this point, backhaul customer could be assigned to the route. Finally, when all customers have been assigned, the routes are connected to the depot.

3.3 Neighborhood Criteria

Given a solution X, the algorithm moves to a new solution into the neighborhood of X, denoted as $N(X)$. The neighborhood is created by slight variations in the attributes of X, which follow predefined rules of generation of movements. In the proposed algorithm, three different neighborhoods are considered. They are shown in Figure 2 to Figure 4, where a series of continuous lines indicate the new edges added to the solution as a consequence of the movement. On the contrary, the removed edges have been indicated by dashed lines.

- *Insertion move*: A customer is relocated to another position belonging either to the same route or to a different route. If the customer has delivery demand then the relocation is only performed between Linehaul routes. Otherwise, if the customer has recollection demand, the relocation is only performed between Backhaul routes. The Figure 2 shows an example of Insertion move.

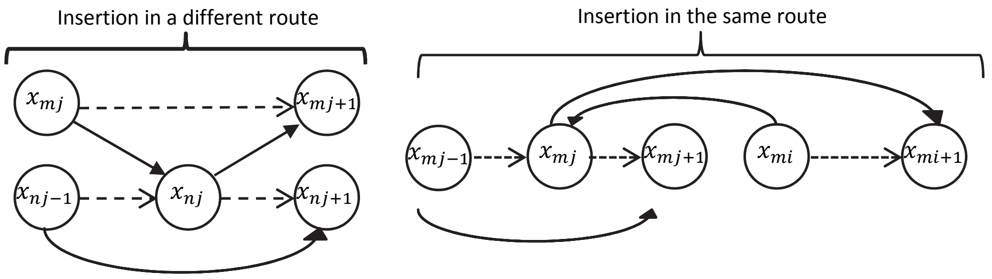

Fig. 2. Examples of Insertion moves on Linehaul routes

- *Swap move*: Two customers belonging to the same type of demand (Linehaul customers or Backhaul customers) exchange their positions on the routes. The Figure 3 shows an example of Swap move.

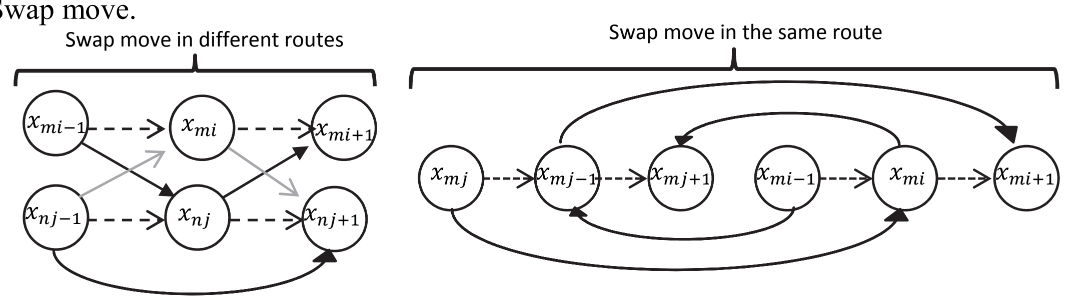

Fig. 3. Examples of swap moves on *linehaul routes*

- Swap-connections: The edges connecting the *Linehaul routes* with the *Backhaul routes* exchange their positions on the routes. The Figure 4 shows an example of swap-connections move.

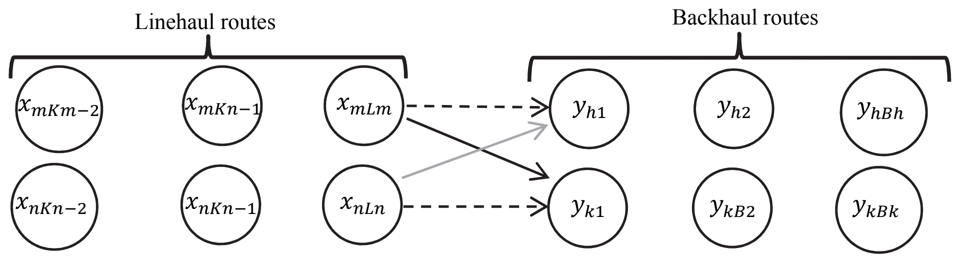

Fig. 4. Example of Swap-connections move.

3.4 Tabu List Design

The Tabu search algorithm defines a list of attributes in order to prevent the exploration to solutions that have already been visited during the search. In particular, we have defined an attribute as a vector of three positions to store the sequence of customer which linked to the moved customer, i.e. the previous one already visited, the moved customer, and the next customer to be served. In Figure 3, for example, the swap move on the same route is performed by permuting the customers x_{mi} and x_{mj}. In such case two attributes would be tested: $A_1 = \left(x_{m\,i-1}, x_{m\,j}, x_{m\,i+1}\right)$ and $A_2 = \left(x_{m\,j-1}, x_{m\,i}, x_{m\,j+1}\right)$.

Let us define a matrix TL, with NxNxN dimensions, to identify all the possible attributes that could be obtained from the problem. This matrix could be used to represent generically the Tabu List of the algorithm, as the values stored in each position determine the iteration in which an attribute were accepted, i.e. the iteration which the movement to generate the attribute were performed. If the movement was recently removed from the solution then it is banned and other movement must be checked. Thus, the criterion for acceptance or rejection of any movement is stated by the rule described in (8), where TL_A is the position of the matrix *TL* defined by the components stored in the vector of attribute A. Then, given a determined number of iterations (N_{Tabu}), the stored attributes are released and could be considered for the new solutions.

$$\begin{cases} Accepted & (TL_A = 0) \lor (Ite - TL_A) < N_{Tabu} \\ Rejected & Otherwise \end{cases} \tag{8}$$

Once the movement is selected to be performed, the components of the matrix *TL* to be updated are the positions defined by the attributes already removed from the solution. For example, in the Figure 3, the attributes $A_3 = (x_{m\,i-1}, x_{m\,i}, x_{m\,i+1})$, and $A_4 = \left(x_{m\,j-1}, x_{m\,j}, x_{m\,j+1}\right)$ are updated to the value of the current iteration. So, they only could be accepted again when N_{Tabu} additional iterations were performed. The proposed approach to validate the movements enables a fast checking of the attributes already banned. However it increases the memory requirements in the Tabu List of the algorithm. This fact did not spoil the global execution of the approach, as we have proven in the next section

3.5 Criterion for acceleration of convergence

The proposed algorithm tries to evaluate intensively small regions in the space of solutions and to determine the next region to be explored. In theory, the algorithm shall move from solution X_1 to another solution X_2 only if X_2 improves the objective function. However, the neighborhood could not produce any improvement of the objective function. In order to avoid local stagnation, the algorithm could accept some level of degradation in the quality of solution. In the proposed approach, the level of degradation to be expected is controlled by a dynamic factor calculated through the equation (9), where $IteBest$ is the number of iterations with no improvement in the quality of the global solution (i.e. the incumbent), with a limit of iterations given by the parameter Ite_best_Max. Similarly, Ite is the counter of iterations of the algorithm, as the maximum of iterations have been set with parameter Ite_Max. On the other hand, k_f is a parameter to be optimized in accordance with the instance.

$$k_{accept} = 1 + e^{\left(\frac{IteBest}{Ite\_best\_Max}\right)} \cdot e^{-\left(k_f \cdot \frac{Ite}{Ite\_Max}\right)} \tag{9}$$

The factor k_{accept} speeds up the convergence of the algorithm when the approach is closing up to the iteration bound. On the other hand, the same factor enables the degradation of the objective function when several iteration have been performed since the last updating of the incumbent. Hence, the exploration is favored at the initial phase of the algorithm, and the exploitation is reinforced in the final phase.

3.6 Flow chart of the algorithm

Fig. 5 shows the main steps executed by the proposed algorithm. Note that the solution X_{ite}, is only updated if exists one solution $X_j \in N_\varepsilon(X_{ite})$ such that it improves the value of the objective function $f(X_{Ite})$ times a factor K_{accept}, and the attributes needed to reach the solution X_j from X_{ite} are not prohibited in the Tabu List. However, the aspiration criterion considers the change in the current solution when the current incumbent is improved by X_j, even with banned attributes. The best solution found by the algorithm is stored in X_{best}. The convergence criterion of the algorithm is determined by the number of iterations Ite_best_Max and Ite_Max.

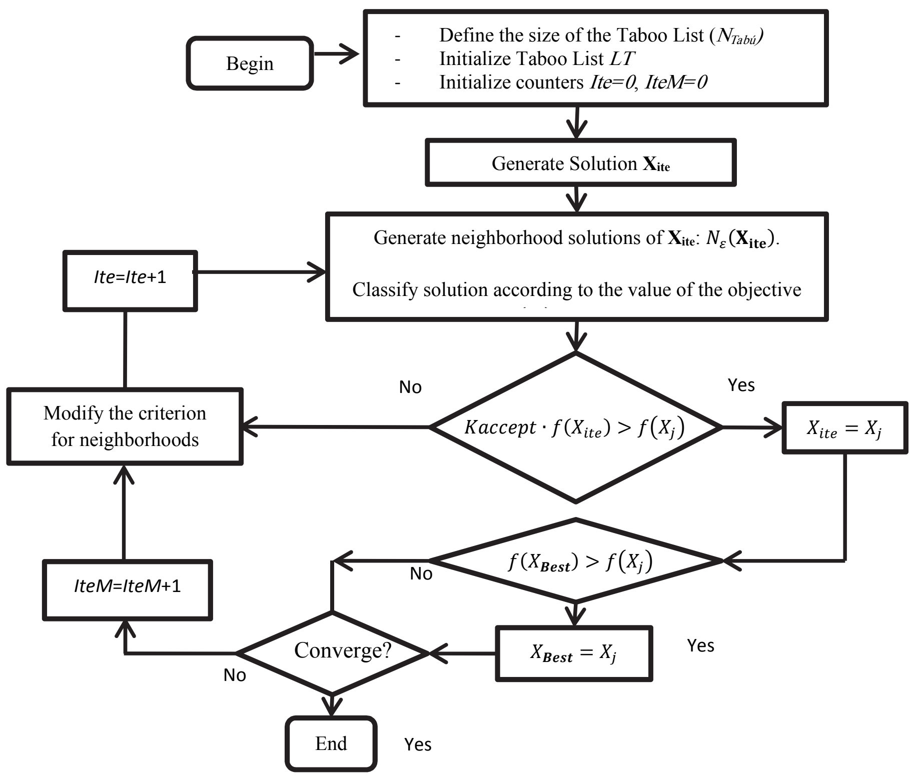
Begin
- Define the size of the Taboo List ($N_{Tabú}$)
- Initialize Taboo List LT
- Initialize counters $Ite=0$, $IteM=0$
Generate Solution $\mathbf{X}_{ite}$
Generate neighborhood solutions of $\mathbf{X}_{ite}$: $N_\varepsilon(\mathbf{X}_{ite})$.
Classify solution according to the value of the objective
$Ite=Ite+1$
No
Yes
Modify the criterion for neighborhoods
$Kaccept \cdot f(X_{ite}) > f(X_j)$
$X_{ite} = X_j$
$f(X_{Best}) > f(X_j)$
No
$IteM=IteM+1$
Converge?
$X_{Best} = X_j$
Yes
No
End
Yes

Fig. 5. Flow chart for the proposed algorithm

4. Computational results

The proposed algorithm has been implemented in Delphi 7.0, and its performance has been evaluated by considering benchmark instances for the VRPB proposed by Baldacci et al. (1999) and Toth & Vigo (1999b). For each instance, we have performed 100 executions in order to obtain the average of both the execution time and the objective function, as well as the standard deviation of the results. In order to obtain results through the proposed approach, the parameters were set as: $k_f = 20, Ite\_best\_Max = 2000$, $Ite\_Max = 10000$, and $N_{Tabu} = 5$. Theses parameter were fixed for all the executions and the total of instances. The results have been compared with a heuristic algorithm showed by Toth and Vigo (1999a). In addition, the results obtained by the exact algorithm proposed by Deif and Bodin (1984) has been used, as this reference matches the results in the lowest computational requirements. Hence, Table 1 shows the results obtained over the considered instances. The first columns (Instance, NL, NB, k_L, k_B) describe the characteristics of the instances. The CPU used for the proposed algorithm is AMD10 processor with 2.3 GHz and 8 MB of RAM.

Table 1
Results found by the proposed algorithm on VRPB Instances

Instance	NL	NB	K_L	K_B	Proposed Algorithm				Deif-Bodin		Toth –Vigo		% ratio	
					Average Value	% Stand. Deviation	Best Value	Time (ms)	Best Value	Time (s)	Best Value	Time (s)	Value	Time
EIL2250A	11	10	3	2	399	7.5	378	100	429	0.1	371*	5.1	**101.9**	100.0
EIL2266A	14	7	3	1	435	4.5	430	104	424	0.2	366*	4.8	117.5	52.0
EIL2280A	17	4	3	1	426	3.9	411	166	375	0.2	375*	7.0	109.6	83.0
EIL2350A	11	11	2	1	788	2.7	767	84	708	0.2	682*	2.6	112.5	42.0
EIL2366A	15	7	2	1	745	2.6	724	253	711	0.2	649*	5.5	111.6	126.5
EIL2380A	18	4	2	2	640	3.5	**623**	314	695	0.2	623*	3.9	**100.0**	157.0
EIL3050A	15	14	2	2	524	7.8	**501**	371	577	0.2	501*	3.3	**100.0**	185.5
EIL3066A	20	9	3	1	617	1.9	610	316	594	0.2	539	7.4	113.2	158.0
EIL3080A	24	5	3	1	646	4.3	625	452	586	0.2	522	7.5	119.7	226.0
EIL3380A	26	6	3	1	855	3.6	806	623	926	0.3	761*	15.1	105.9	207.7
EIL3350A	16	16	3	2	793	7.7	**738**	315	892	0.2	742	1.9	**99.5**	157.5
EIL3366A	22	10	3	1	849	3.4	824	401	876	0.2	753	15.4	109.4	200.5
EIL3380A	26	6	3	1	850	3.4	805	615	926	0.7	761	15.9	105.8	87.9
EIL5150A	25	25	3	3	615	9.0	566	683	669	0.7	562	40.8	**100.7**	97.6
EIL5166A	34	16	4	2	637	10.9	573	607	641	0.7	553	48.5	**103.6**	86.7
EIL5180A	40	10	4	1	684	4.7	644	1288	655	1.9	574	53.1	112.2	67.8
EILA7650A	37	38	6	5	1039	23.7	777	413	840	2.6	756	164.3	**102.8**	15.9
EILA7666A	50	25	7	4	1105	20.5	854	402	907	2.7	780	148.3	109.5	14.9
EILA7680A	60	15	8	2	1198	11.7	966	448	913	2.7	833	238.2	116.0	16.6
EILB7650A	37	38	8	7	1102	26.0	841	248	957	2.8	825	240.5	**101.9**	8.9
EILB7666A	50	25	10	5	1423	21.6	953	199	984	2.7	891	241.0	107.0	7.4
EILB7680A	60	15	12	3	1257	21.7	990	325	1032	3.0	948	240.7	**104.4**	10.8
EILC7650A	37	38	5	4	908	18.9	732	757	861	2.7	715	110.5	**102.4**	28.0
EILC7666A	50	25	6	3	932	17.9	783	756	853	3.0	745	148.7	**105.0**	25.2
EILC7680A	60	15	7	2	997	17.8	778	605	838	2.8	759	219.4	**102.5**	21.6
EILD7650A	37	38	4	3	872	14.5	736	1631	806	2.8	691	93.7	106.5	58.3
EILD7666A	50	25	5	2	985	12.7	790	1308	835	3.0	717	89.7	110.2	43.6
EILD7680A	60	15	6	2	960	13.9	739	682	798	2.7	710	190.6	**104.1**	25.3
EILA10150A	50	50	4	4	1187	12.2	864	1952	913	6.8	852	213.5	**101.4**	28.7
EILA10166A	67	33	6	3	1265	11.4	939	1238	955	6.4	868	240.6	108.2	19.3
EILA10180A	80	20	6	2	1322	6.3	1096	1404	956	6.6	900	241.0	121.8	21.3
EILB10150A	50	50	7	7	1466	21.2	1027	492	1214	7.0	962	241.6	106.8	7.0
EILB10166A	67	33	9	5	1560	13.0	1292	513	1318	6.6	1040	241.3	124.2	7.8
EILB10180A	80	20	11	3	1479	19.5	1196	656	1174	7.6	1060	241.6	112.8	8.6

* Optimal solution proved (Toth & Vigo, 1999b)

Fig. 6 shows the convergence of the algorithm during one execution of the algorithm at the instance IEL2380A. The first stage of the algorithm is featured by exploration, and high degradation of the objective function is permitted. This procedure enables to getting out from stagnation on local suboptimal. Otherwise, at the final stage, the degradation is highly controlled and the convergence is accelerated.

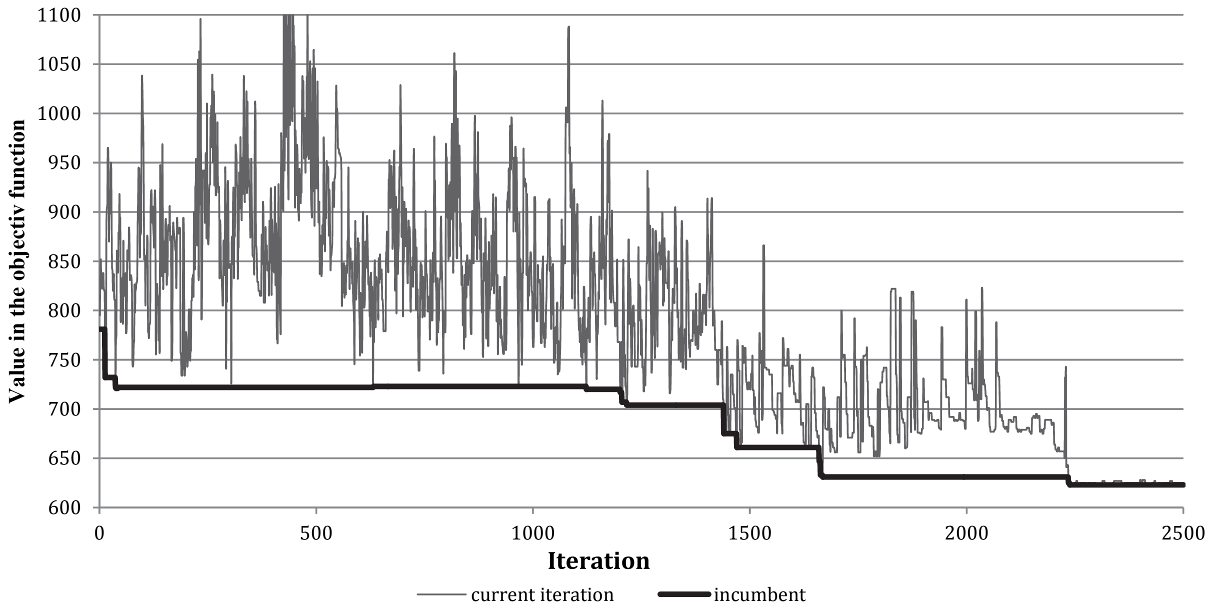

Fig. 6. Behavior of convergence of the algorithm. Instance evaluated EIL2380A

5. Concluding remarks

In this paper, an effective heuristic algorithm based on a Tabu Search has been proposed for the VRPB. The main goal was to find a set of routes by considering that the backhaul customers must be served only when the linehaul routes are performed. The results obtained over the benchmarking instances show the effectiveness of the proposed algorithm. The value of the objective function obtained in 14 instances are close to those obtained by Toth and Vigo (1999a), worsen as much as 5%. In fact, the average ratio in the objective function, considering all the instances, was lower than 10% compared with the benchmark. However, the average time required was just 70% of that shown by the algorithm proposed byDeif and Bodin (1984). The low time requirements exposed by the proposed algorithm could be applied to different variants of the VRPB, such as time windows, stochastic elements, multiple vehicle capacity, among others.

The algorithm has exhibited a poor performance as the size of the problem was increased. It means that additional amends are needed in order to enhance the quality of the algorithm. On the other hand, the impact of the factor k_{accept} needs a complete examination in order to optimize their value for the overall instances. In addition, other diversification strategies could be applied in order to explore new parts of the space of solutions. These aspects may lead to better quality results and prevent the stagnation on suboptimal solutions.

Acknowledgement

The autors would like to thank the annonymous referees for constructive comments on earlier version of this paper.

References

Aguado, J. S. (2009). Fixed Charge Transportation Problems: a new heuristic approach based on Lagrangean relaxation and the solving of core problems. *Annals of Operations Research*, 172(1), 45.

Chávez, J. J., Escobar, J. W., & Echeverri, M. G. (2016). A multi-objective Pareto Ant Colony algorithm for the Multi-Depot Vehicle Routing problem with Backhauls. *International Journal of Industrial Engineering Computations*, 7(1), 35 – 48.

Escobar, J. W., & Linfati, R. (2012). Un algoritmo metaheurístico basado en recocido simulado con espacio de búsqueda granular para el problema de localización y ruteo con restricciones de capacidad. *Revista Ingenierías Universidad de Medellín*, 11(21), 139-150.

Escobar, J. W. (2014). Heuristic algorithms for the capacitated location-routing problem and the multi-depot vehicle routing problem. *4OR*, *12*(1), 99.

Escobar, J. W., Linfati, R., & Toth, P. (2013). A two-phase hybrid heuristic algorithm for the capacitated location-routing problem. *Computers & Operations Research*, *40*(1), 70-79.

Escobar, J. W., Linfati, R., Toth, P., & Baldoquin, M. G. (2014a). A hybrid granular tabu search algorithm for the multi-depot vehicle routing problem. *Journal of Heuristics*, *20*(5), 483-509.

Escobar, J. W., Linfati, R., Baldoquin, M. G., & Toth, P. (2014b). A Granular Variable Tabu Neighborhood Search for the capacitated location-routing problem. *Transportation Research Part B: Methodological*, *67*, 344-356.

Escobar, J. W., Linfati, R., & Adarme-Jaimes, W. (2015). A hybrid metaheuristic algorithm for the capacitated location routing problem. *Dyna*, *82*(189), 243-251.

Mingozzi, A., Giorgi, S., & Baldacci, R. (1999). An exact method for the vehicle routing problem with backhauls. *Transportation Science*, *33*(3), 315-329.

Bazaraa, M. S., Sherali, H. D., & Shetty, C. M. (2006). *Nonlinear Programming*. New Jersey: *A John Wily & Sons*, Inc., Publication.

Deif, I., & Bodin, L. (1984). Extension of the Clarke and Wright algorithm for solving the vehicle routing problem with backhauling. In Proceedings of the Babson conference on software uses in transportation and logistics management (pp. 75-96). Babson Park, MA.

Dorigo, M., & Gambardella, L. M. (1997). Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, *1*(1), 53-66.

Glover, F. (1989). Tabu search—part I. *ORSA Journal on computing*, *1*(3), 190-206.

Laporte, G. (2009). Fifty years of vehicle routing. *Transportation Science*, *43*(4), 408-416.

Lu, Q., & Dessouky, M. (2004). An exact algorithm for the multiple vehicle pickup and delivery problem. *Transportation Science*, *38*(4), 503-514.

Olivera, A. (2004). Heurísticas para problemas de ruteo de vehículos. *Reportes Técnicos* 04-08.

Toth, P., & Vigo, D. (1999). A heuristic algorithm for the symmetric and asymmetric vehicle routing problems with backhauls. *European Journal of Operational Research*, *113*(3), 528-543.

Toth, P., & Vigo, D. (1997). An exact algorithm for the vehicle routing problem with backhauls. *Transportation science*, *31*(4), 372-385.

© 2017 by the authors; licensee Growing Science, Canada. This is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC-BY) license (http://creativecommons.org/licenses/by/4.0/).