



# An Improved Biogeography-Based Optimization for the Long-Term Carpooling Problem

Rachid Kaleche, Zakaria Bendaoud & Karim Bouamrane

**To cite this article:** Rachid Kaleche, Zakaria Bendaoud & Karim Bouamrane (2021) An Improved Biogeography-Based Optimization for the Long-Term Carpooling Problem, *Applied Artificial Intelligence*, 35:10, 745-764, DOI: [10.1080/08839514.2021.1935586](https://doi.org/10.1080/08839514.2021.1935586)

**To link to this article:** <https://doi.org/10.1080/08839514.2021.1935586>



Published online: 17 Jun 2021.



Submit your article to this journal [↗](#)



Article views: 1323



View related articles [↗](#)



View Crossmark data [↗](#)






Citing articles: 2 View citing articles [↗](#)

RESEARCH ARTICLE



# An Improved Biogeography-Based Optimization for the Long-Term Carpooling Problem

Rachid Kaleche <sup>a</sup>, Zakaria Bendaoud <sup>b</sup>, and Karim Bouamrane <sup>a</sup>

<sup>a</sup>Laboratoire Informatique d'Oran (LIO), Ahmed Ben Bella University Oran 1, Oran, ALGERIA; <sup>b</sup>Universite Dr Tahar Moulay De Saïda, Saïda, Algeria

## ABSTRACT

The increasing number of vehicles on the road produces negative effects for health, the environment, quality of life, and the economy, among other areas. To address this problem, an important key is carpooling private vehicles from different homes to a common destination. This paper specifically addresses the long-term carpooling problem, which is an NP-complete problem. The proposed approach is a modified biogeography-based optimization metaheuristic, which is hybridized with a variable neighborhood search. Comparisons with efficient known approaches indicate the effectiveness of the proposed approach for large-scale long-term carpooling problems.

## ARTICLE HISTORY

Received 1 August 2020  
Revised V22 may 2021  
Accepted 21 October 2020

## Introduction

Vehicle use has increased, which has caused negative effects such as traffic congestion, air and noise pollution, insufficient parking places, and general degradation of quality of life. As a result, human health, the environment, and the economy are negatively affected. Although public transportation is an adequate response to this problem, it is not a sufficient response. Therefore, carpooling should be established through carpooling incentive policies.

Carpooling exists in two forms, the daily carpooling problem (DCPP) and the long-term carpooling problem (LTCPP). In the DCPP, a set of users or servers is declared each day. Each server picks up its colleagues or clients for the particular day. The problem is to generate for each server a set of clients that fits car capacities and time windows constraints. The DCPP objective is to minimize the total traveled distance. The DCPP is a particular case of the dial-a-ride problem (Ho et al. 2018). In the LTCPP, however, the objective is to connect each user to a subgroup called a pool for a long time. In the pool, each user, in turn, acts as a server and picks up the other users or clients of its pool from their homes and carries them to a common destination. The user then performs the inverse route under car capacities and time windows constraints.

In the LTCPP, the pools are more stable than in the DCP, and the objective is to reduce the number of pools and the total traveled distance. The LTCPP is a multi-objective problem; it was proven to be NP-complete (Varrentrapp, Maniezzo, and Stützle 2002).

Moreover, the LTCPP was tackled by exact and approximate methods. Baldacci, Maniezzo, and Mingozzi (2004) addressed LTCPP with an exact method. They have exposed two formulations of the LTCPP, one with the clustering approach and one with the partitioning approach of set partitioning. The dual problem of the second formulation, dual set partitioning, was resolved with the column generation method. The benchmark was composed of 35 instances with a maximum size of 250 employees.

Given that the LTCPP is an NP-complete problem, approximate methods are more appropriate to resolve it, especially when the problem is large. Thus, the LTCPP has been addressed using specific heuristics and metaheuristics. The heuristics include the saving functions heuristic (Ferrari et al. 2003), the simulation-based algorithm (Correia and Viegas 2008), and the multi-matching system (Shangyao, Yan, Chen, and Lin 2011). More specifically, the saving functions heuristic is based on a matching preference of two users to be pooled together; this preference corresponds to the saved cost, two users by two. The authors of the simulation-based algorithm adopted a divide and conquer approach. As such, the first stage of the algorithm is based on a  $k$ -means clustering algorithm. This clustering is based on minimizing the sum of square distances between the users and the centroid of each corresponding cluster. The second phase is treated by an optimization programme that searches to combine pools in order to obtain the smallest number of groups with high numbers of users. The authors of the multi-matching system treated a different problem, the long-term carpooling many-to-many carpooling problem. In this case, a derived Lagrangian problem is constructed and resolved to produce a lower bound. The upper bound is computed by using a Lagrangian heuristic. The sub-gradient method is used to adjust the Lagrangian multipliers. An iterative process is performed until an acceptable convergent solution is obtained or until a fixed maximum number of iterations is reached. This approach is time consuming, especially for large-scale problems.

A number of metaheuristics were utilized to address the LTCPP. Maniezzo et al. (2004) have examined the LTCPP with two metaheuristics inspired from ant colony optimization algorithm (ACO). For one metaheuristic, the solution is constructed completely. In the other, ants construct component solutions as pools that are later combined through an integer solver programme. Guo, Goncalves, and Hsu (2012) have proposed an approach based on the transformation of ACO to a clustering approach. They named this approach the clustering ant colony (CAC). Each ant in the CAC constructs the pools guided by preference information during their tour. When all ants have accomplished

their tours, the obtained solutions are improved by a variable neighborhood search (VNS) (Mladenovic and Hansen 1997). Guo, Goncalves, and Hsu (2011) also developed an approach known as the guided genetic algorithm (GGA). In this algorithm, the initial population is generated by a sweep heuristic (Gillett and Miller 1974). The crossover and mutation operators are guided by preference information, which is constituted from the best fits, and the local search is performed by mutation. The preference information is updated at each iteration, and it accomplishes two important roles: it avoids infeasible solutions which helps repair them, and it guides crossover and mutation GGA operators. Mlayah, Boudali, and Tagina (2018) have suggested an approach based on VNS hybridizing with Tabu search (TS) (Glover 1977) that they named HVNTS. In this approach, the initial population is generated by using a sweep heuristic. Then VNS is used as the diversification process, while TS acts as the intensification process. Su, Zhou, and Yu (2019) have treated the LTCPP problem different from the one treated by the aforementioned authors. In the model of Su, Zhou, and Yu, not all users have cars. Additionally, the model considers other parameters, including the number of days and the number of servers for each day. The authors have suggested the hybrid metaheuristic approach of artificial bee colony (ABC) with a VNS TL (VNSTL). They named this approach the ABC-VNSTL. The VNSTL performs the employee and onlooker phases, while the scout phase is conducted through an approach known as the scout diversity protection. Instead of modifying one dimension at each step, as in classical ABC, the ABC-VNSTL modifies several dimensions at the same time. As a result, its convergence is accelerated.

This paper contributes to the field by proposing an efficient modified biogeography-based optimization (BBO) to resolve large-scale LTCPPs. The rest of this paper is organized in five sections. The problem is defined in the six section. The second section explains the mathematical model. The classical BBO metaheuristic (Simon 2008) is introduced in the third section. The new approach of a modified BBO is detailed in the fourth section. The fifth section provides a comparison of this approach with other efficient approaches, while, the sixth section presents the conclusion.

## Problem Definition

In the LTCPP, users must reach their common destination by sharing their cars. Each user within a pool, in turn, picks up the other members of the same pool in route toward the common destination. Each user has the following specified constraints:

- A limited car capacity when the user plays the server role
- A maximum driving time when the user plays the server role
- An earliest leaving time from home

- A maximum arriving time at work
- A penalty cost when a user travels alone

The LTCPP is a multi-objective problem that first clusters users in subgroups called pools such that the sum of traveled routes must be the shortest; second, the number of pools must be the fewest. The objective of the LTCPP is to minimize the total traveled distance by all users and to minimize the number of used cars under car capacities and time window constraints.

## Mathematical Model

The mathematical model of the LTCPP is presented below (Guo, Goncalves, and Hsu 2012). The LTCPP can be modeled by means of a directed graph  $G = (U \cup \{0\}, A)$ , where  $U$  is the set of users,  $\{0\}$  is the common destination, and  $A$  is the set of arcs between each user and the others and between each user and the common destination. The mean shortest paths of a pool  $k$  is defined by equation (1).

$$cost(k) = \begin{cases} \sum_{i \in k} \frac{cost(\min\_path(i, k))}{|k|}, & \text{if } |k| > 1, \\ \sum_{i \in k} cost_{i0} + p_i, & \text{otherwise.} \end{cases} \quad (1)$$

Where  $i$  is the user that acts as a server,  $cost(\min\_path(i, k))$  is the cost of the shortest path starting from  $i$  and ending at 0 by connecting all users of the pool  $k$ ,  $|k|$  is the size of the pool  $k$ ,  $cost_{i0}$  is the cost of the direct travel of the server  $i$  to the common destination. When a user  $i$  travels alone, a penalty  $p_i$  is added to the travel cost.

Since an LTCPP solution is composed from  $K$  pools, the total cost of a solution is the sum of the costs of  $K$  pools. The LTCPP mathematical model notations are defined as follows:

- $x_{ij}^{hk}$ : A binary variable that equals 1 if the arc  $(i, j)$  is traveled by a server  $h$  of a pool  $k$ ; it equals 0 otherwise.
- $y_{ik}$ : A binary variable that equals 1 if the user  $i$  belongs to a pool  $k$ ; it equals 0 otherwise.
- $\xi_{ij}$ : A binary variable that equals 1 if a user  $i$  travels alone.
- $s_i^h$ : A positive variable indicating the pickup time of a user  $i$  by a server  $h$ .
- $f_i^h$ : A positive variable denoting the arrival time at the common destination of a user  $i$  when picked up by the server  $h$ .
- $cost_{ij}$ : A positive variable indicating the travel cost between two users  $i$  and  $j$ .
- $t_{ij}$ : A positive variable indicating the travel time between two users  $i$  and  $j$ .

- $Q_k$ : A positive variable indicating the minimum car capacity of a pool  $k$ .
- $T_i$ : A positive variable indicating the maximum driving time a user  $i$  can accept.
- $e_i$ : A positive variable indicating the earliest leaving time accepted by a user  $i$ .
- $r_i$ : A positive variable indicating the latest arriving time at the common destination for a user  $i$ .
- $p_i$ : A positive variable indicating the penalty for a user  $i$  when traveling alone.
- $K$ : Index set of pools.
- $U$ : Index set of users.
- $A$ : Index set of arcs.

The objective function is:

$$f_{LTCPP} = \min \left( \sum_{k \in K} \frac{\sum_{h \in U} \sum_{(i,j) \in A} cost_{ij} x_{ij}^{hk}}{\sum_{i \in U} y_{ik}} + \sum_{i \in U} p_i \xi_i \right) \quad (2)$$

Subject to the following constraints:

$$\sum_{j \in U/\{h\}} x_{ij}^{hk} = y_{ik} i, h \in U, k \in K \quad (3)$$

$$\sum_{j \in U} x_{ji}^{hk} = y_{ik} i, h \in U, k \in K \quad (4)$$

$$\sum_{j \in U} x_{ij}^{hk} = \sum_{j \in U} x_{ji}^{hk} i, h \in U, k \in K \quad (5)$$

$$\sum_{k \in K} y_{ik} + \xi_i = \sum_{j \in U} x_{ji}^{hk} i, i \in U \quad (6)$$

$$\sum_{(i,j) \in A} x_{ij}^{hk} \leq Q_k h \in U, k \in K \quad (7)$$

$$\sum_{(i,j) \in A} x_{ij}^{hk} t_{ij} \leq T_h h \in U, k \in K \quad (8)$$

$$s_i^h \geq e_i i, h \in U \quad (9)$$

$$s_j^h - s_i^h \geq t_{ij} - M \left( 1 - \sum_{k \in U} x_{ij}^{hk} \right) (i, j) \in A, h \in U \quad (10)$$

$$f_i^h \geq s_i^h + t_{i0} - M \left( 1 - \sum_{k \in U} x_{i0}^{hk} \right) i, h \in U \quad (11)$$

$$f_i^h \leq r_i + M \left( 1 - \sum_{k \in K} \sum_{j \in U} x_{ij}^{hk} \right) i, h \in U \quad (12)$$

Equations (3) and (4) require user  $i$  to be affiliated with pool  $k$ , and if a path begins with user  $h$ , only one of the arcs  $(i, j)$  or  $(j, i)$  can be traveled. Equation (5) guarantees the continuity constraint. Equation (6) assumes that each user  $i$  traveling alone is penalized, thus privileging user  $i$  being pooled with other users. Equations (7) and (8) are car capacity and maximum accepted travel time, respectively. The feasible pickup times are guaranteed by equations (9) and (10), while the minimum and maximum values of arrival times are assured by equations (11) and (12), respectively.

### Biogeography-Based Optimization (BBO)

Biogeography is the study of the distribution of species on earth, their migrations between habitats, and their extinctions. Dan Simon (2008) has proposed BBO as a population-based metaheuristic for global optimization based on this phenomenon. To carry out the equilibrium principal (MacArthur and Wilson 1967) and optimization process (Ma 2010; Volk 2003), BBO uses a mathematical model where:

- Each habitat represents a solution;
- Each habitat has a habitat suitability index (HSI) to represents its quality and, thus, the fitness of a solution;
- Each habitat is characterized by  $m$  suitable index variables (SIV), which represent variables of the addressed problem;
- Each habitat  $i$  has an immigration rate  $\lambda_i$  and an emigration rate  $\mu_i$ ;
- The two migration rates are functions relevant to the number of species;
- The species, which are SIV(s), migrate from one habitat to another according to migration rate probabilities;
- Each habitat (solution) is subject to mutation on its SIV(s).

After generating an initial population of habitats, BBO uses three operators, migration, mutation, and elitism. The migration operator performs information exchange between habitats. Rather than producing new individuals as evolutionary algorithms with a crossover operator, BBO modifies the existing individuals by using a migration operator (Simon 2008). The mutation operator perturbs the individuals to diversify the population and escape the local

optima traps. The elitism mechanism guides the intensification process. The pseudo code of the BBO algorithm is as follows:

Algorithm 1: BBO algorithm

Initialization

**While** stop criteria not verified

Evaluate the quality (*HSI*) of each solution

Memorize the *K* solutions having best quality (having the higher level of *HSI*)

*Migration: migrate randomly the habitats (solutions) by using the rates  $\lambda$  and  $\mu$*

*Mutation: mutate the solutions which don't belong to elite*

Replace the population of solutions by the descendants

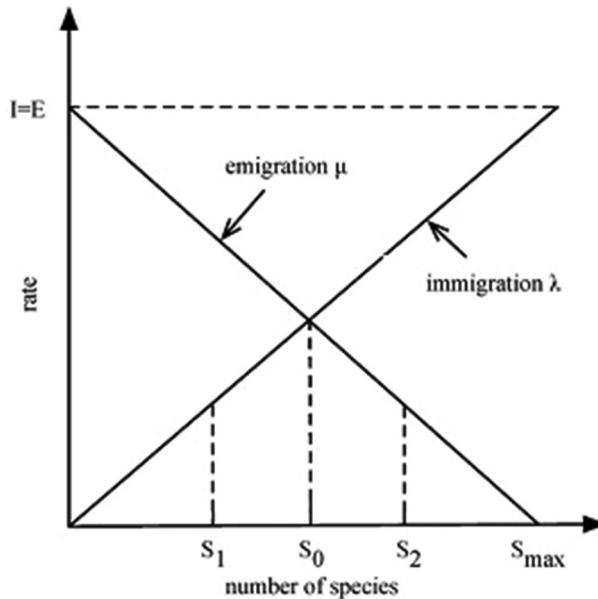
Implementing elitism: replace worst habitats by the bests known

**End While**

The BBO migration process is relevant to the immigration and emigration rates, which are computed by equations (13) and (14), respectively. These two equations define the linear model represented by Figure 1.

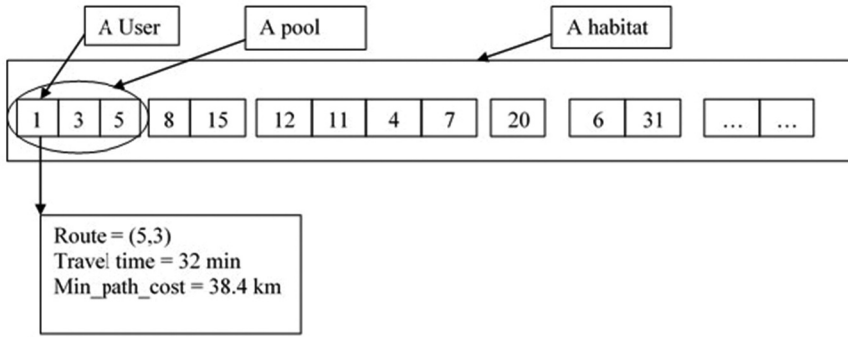
$$\lambda_s = I \left( 1 - \frac{S}{S_{max}} \right) \quad (13)$$

$$\mu_s = E \left( \frac{S}{S_{max}} \right) \quad (14)$$



**Figure 1.** Correlation between migration rates and species number.





**Figure 2.** A habitat and its SIV(s) in BBO representation of an LTCPP solution.

Where  $I$  and  $E$  are the initial immigration and emigration rates,  $S$  is the number of species of habitat  $S$ , and  $S_{max}$  is the maximum number of species. The linear model was compared with other models, including constant, trapezoidal migration models among others (Ma 2010).

Algorithm 2: BBO migration algorithm

Select  $H_i$  with a probability  $\alpha\lambda_i$

If  $H_i$  is selected then

For  $j = 1$  to  $n$  do

Select  $H_j$  with probability  $\alpha\mu_j$

If  $H_j$  is selected then

Select randomly  $SIV_a$  from  $H_j$

Replace a random SIV in  $H_i$  by  $SIV_a$

End if

End for

End if

$$m(S) = m_{max} \left( \frac{1 - P_s}{P_{max}} \right) \quad (15)$$

Where  $m_{max}$  is a user parameter,  $P_s$  the probability that a habitat has  $S$  species, and  $P_{max}$  is the probability that a habitat has the maximum number of species.

Algorithm 3: BBO mutation algorithm

**For**  $i = 1$  to  $N$  do

Compute from  $\lambda_i$  and  $\mu_i$  the probability  $P_i$

Compute the mutation rate  $m_i$  with equation (15)

// Browse SIV(s) of the solution to mutate

**For**  $j = 1$  to  $D$  do

**If**  $rand(0, 1) < m_i$  alors

Generate randomly a feasible  $SIV_a$

// Replace the SIV of the solution to mutate

$S_i(j) = SIV_a$

**End If**  
**End for**  
**End for**

BBO is a performant metaheuristic. Therefore, it has been widely used in different areas of transport, such as VRP problems (Berghida 2015), image processing (Zheng et al. 2016), power and electricity (Chatterjee et al. 2012), robotics (Zhang, Wang, and Chen 2019), wireless networks (Zhang et al. 2015), data mining (Zhao et al. 2019), economics and social sciences (Du and Simon 2013), and the internet of things (Cao, Wang, and Li 2014), among other domains.

## Hybrid Modified Mutation BBO

### *Solution Encoding*

A habitat (solution) can be implemented by an indirect (binary) or direct (real) representation; in this case, the direct representation was chosen. The direct or real representation is advantageous given that it does not require an interpretation of intermediate solutions or a translation of the final solution. In addition, Guo, Goncalves, and Hsu (2012) as well as Mlayah, Boudali, and Tagina (2018) have used this representation.

Two levels of information emerge from the problem: the clustering of users into independent pools and within each pool, and the route that each user browses when they play the role of server. To address the LTCPP by using BBO, the following correspondences were conducted (Figure 2):

- A habitat represents a solution;
- The total cost of a solution is equal to the HSI of a habitat;
- A pool represents a species and, thus, an SIV;
- In turn, each user in a pool plays the role of server, who picks up the rest of the pool users.

### *The Hybrid Modified Mutation BBO Algorithm*

The difficulty of using any metaheuristic is balancing diversification and intensification; this balancing is an open problem (Yang 2014). Therefore, the BBO algorithm structure was modified to address the balance problem and the LTCPP problem. The new algorithm was named the hybrid modified mutation BBO (HMMBBO). The global idea of HMMBBO is to use as an intensification process migration combined with elitism and to allow a modified mutation operator to aid the diversification process. Instead of mutating a large number of species, as in the BBO, the mutation is applied to

a reduced number of pools (species). The HMMBBO performs small and large mutations in the beginning of the process and only small mutations at the end. This strategy favors diversification at the beginning and intensification at the end. After the mutation, a migration combined with an improvement that is performed through a VNSTL of a randomly selected habitat. In VNSTL, two TLs were used. The first TL was used to avoid selecting the same solution twice, while the second was used to list the pools that could not be improved. As in nature, mutated species migrate, interact, and attempt to improve themselves; the latter phenomenon is simulated by implementing VNSTL. The HMMBBO algorithm is defined in algorithm (4).

Important differences exist between a classical BBO and the HMMBBO. First, elitism is not applied when a mutation is performed. In other words, elitism is only used in the intensification phase. The aim of this modification is to allow mutated individuals to interact among themselves and with the rest of the individuals. Another difference is the use of the mutation operator only after a number of fails and not at each iteration, as in classical BBO. The third difference is the use of migration as a propagator of new information provided by mutation in the diversification phase. The last difference is the hybridization with VNSTL to improve the new habitats and their interactions. In the next section, useful reduce functions are defined before detailing migration and mutation operations.

Algorithm 4: HMMBBO algorithm for LT CPP

Load instance data

Initiate the table of allowed and not allowed user pairs to be pooled one with each other

Initiate BBO parameters: population size  $N$ , migration rates  $\lambda$  and  $\mu$ , elite size  $K$

Generate Initial population composed from structured and random  $H_i, i = 1, \dots, N$

**While** stop criteria not verified

Evaluate the quality ( $HSI$ ) of each solution

Memorize the  $K$  solutions having best quality (having the lowest  $HSI$ )

Compute the rates of immigration ( $\lambda$ ) and emigration ( $\mu$ ) for each solution

Compute the probability for each number of species  $P$

*Migration satisfying problem constraints*

**If** ( $nbfails > m$  and  $rand(0,1) < \alpha$ )

**If** ( $(|H_k - H_0|/H_0) < \omega$ ) and  $(iter/nbTotIter) < r$  and  $nbMutation > C_1$  and  $nbRenew < C_2$ )

*//Mutation: used as a renew of population (LM: Large Mutation)*

*Modified Mutation (minLM, maxLM)*

$nbRenew = nbRenew + 1$

$nbMutation = 0$

**Else**

```

//Mutation: mutate a reduced number of species
Modified Mutation (minRM, maxRM)
nbMutation = nbMutation + 1
nbRenew = Max (0,nbRenew - 1)
End if
// Propagation of information
Migration
//Improve m solutions randomly
VNSTL
nbfails = 0
Else
Replace the worst solutions by the elite solutions of precedent generation
End if
If ( $|H_k - H_0|/H_0 < \omega$ )
nbfails = nbfails + 1
Else
nbfails = 0
End if
iter = iter + 1
End While

```

### Useful Reduce Functions

The search space of LTCPP is large and contains many infeasible solutions. Avoiding such solutions maximizes the amount of computing time spent exploring more feasible solutions. Therefore, useful reduce functions (Baldacci, Maniezzo, and Mingozzi 2004) were used. These functions were used for the following purposes:

- To construct a table of potentially allowed users to be pooled together when equation (16) is satisfied; they are not allowed to be pooled together otherwise;
- To identify users that must be alone if equation (17) is violated;
- To control during the resolution process if the addition of a user to a pool violates time window constraints before controlling them for each route; this control is performed with equations (18) and (19).

$$e_i + t_{ij} + t_{j0} < \min(r_i, r_j) \quad (16)$$

$$e_i + t_{i0} < r_i \quad (17)$$

$$\min(t_{ipoolk}) + shortTime_{poolk} \leq T_i \quad (18)$$

Where  $\min(t_{ipoolk})$  is the shortest time route from a user  $i$  to all users of the pool  $k$ , and  $shortTime_{poolk}$  is the shortest time route between traveled routes by all users of pool  $k$ .

$$\max(e_i + \min(t_{ipoolk}), \min(e_{poolk})) + shortTime_{poolk} \leq \min(r_i, \min(r_{poolk})) \quad (19)$$

Where  $\min(e_{poolk})$  is the earliest leaving time of all users of pool  $k$ ,  $\max(e_i + \min(t_{ipoolk}), \min(e_{poolk}))$  is the potential earliest leaving time of user  $i$  when playing the role of server and if added to pool  $k$ . Therefore, the first term of equation (19) added to the shortest travel time of pool  $k$  must be less than or equal to the minimal last arrival time of user  $i$  and of all users of pool  $k$ .

### Initial Population

An initial population of habitats is generated from structured and randomized individuals. The generation of the structured individuals is based on a modified version of the heuristic used by Guo (2012). The heuristic has been modified by merging pools with one user when feasible. The goal of the modified heuristic is to provide, from the beginning of the research process, a heterogeneous population and hence a diversity in terms of pool composition.

### Migration

The migration operator has an important function in BBO. It is the operator through which the exchange of information between habitats is performed. First, a habitat  $H_i$  receiving species, which are pools containing information, is selected according to a probability based on the immigration rate  $\lambda_i$ . Then, each habitat  $H_j$  from which a species or pool emigrates is selected according to a probability based on the emigration rate  $\mu_j$ . One pool emigrates from each selected habitat. This process does not create a new descendant but modifies existing ones, as illustrated in Figure 3. The migration operator is implemented in algorithm (5).

The migration of a  $pool_k$  from a habitat  $H_j$  to a habitat  $H_i$  causes user redundancy in the pool of  $H_i$ . Therefore, the solution  $H_i$  becomes inconsistent.



Figure 3. Migration of pools.

The response to this situation is to remove these users from their pools. After doing so, some affected pools may contain one user, which penalizes the pool cost and, consequently, the entire solution. As a result, the algorithm attempts to merge the penalized pools. Figure 4 provides an example of the migration of one pool according to the following description.

- Step 1: Select a habitat  $H_i$  to receive a pool from a habitat  $H_j$ .
- Step 2: Select the pool containing users 8, 12, 10, and 6 from the habitat  $H_j$ .
- Step 3: Migrate the selected pool to habitat  $H_i$  and remove redundant users from other pools of  $H_i$ .
- Step 4: Merge user 19, who is alone due to migration operation.

Algorithm 5: HMMBBO migration algorithm for LTCPP

Select  $H_i$  with a probability  $\alpha\lambda_i$

**If**  $H_i$  is selected then

**For**  $j = 1$  to  $n$  do

Select  $H_j$  with probability  $\alpha\mu_j$

**If**  $H_j$  is selected then

Select randomly  $pool_k$  with a size greater than 1 from  $H_j$

**If**  $pool_k$  not exist in  $H_i$  then

Add  $pool_k$  to  $H_i$

Remove from other pools of  $H_i$  users having the same users of  $pool_k$

Attempt to merge affected pools of  $H_i$  which their sizes = 1

**End if**

**End if**

**End for**

**End if**

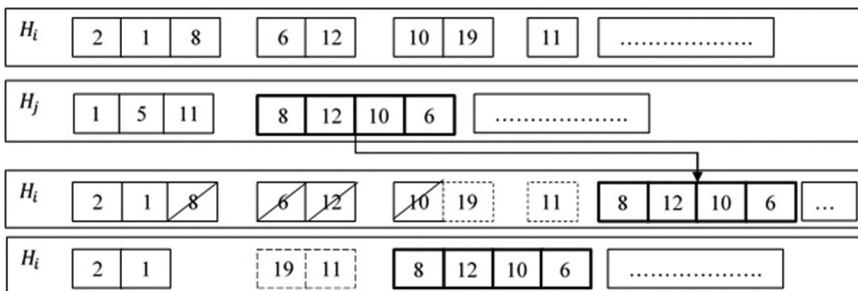


Figure 4. BBO migration example for LTCPP.

### **Mutation**

In any metaheuristic the diversification process is an important process. On the one hand, it allows escape from local optima traps and encourages exploration of new areas. On the other hand, it could prevent the improvement of local optima if used prematurely or frequently. Therefore, the modified mutation operator is performed only under the conditions defined in algorithm (4). Moreover, in HMMBBO mutation is modified to control the size of mutation by two parameters that indicate the range of mutations by habitat. As a result, the number of species to mutate depends on the two parameters in a random manner. The aim of this modification is to define the move size to another search space and, therefore, attempt to escape local optima traps and to explore other areas.

In BBO, the mutation operator guaranteed the diversification process and had to be used for this purpose. In the modified mutation operator of HMMBBO, different operators were used to address the LTCPP. The operators 'swap,' 'merge,' 'move,' and 'divide' were used in the intensification process to improve the existing solutions in the CAC and GGA approaches of Guo, Goncalves, and Hsu (2012, 2011) as well as the HVNTS approach of Mlayah, Boudali, and Tagina (2018). In contrast, the operators in the proposed approach were used for diversification regardless of the improvement of existing solutions. The operators were defined and used as follows:

- **Swap:** A number of pools with more than one user are randomly selected. For each selected pool, the swap is performed with another pool of the same habitat; the last pool is selected randomly and can have any size. This operation is repeated until all pools are visited or until a valid swap is identified.
- **Merge:** A portion of unfilled pools are randomly selected. Each pool is merged with another of the same habitat, and the last pool is selected randomly. Similar to the swap operator, this operation ends at the first valid merge.
- **Move:** A pool with more than one user is randomly selected. Then, another unfilled pool of the same habitat is randomly selected. The operator moves one user of the first pool into the second pool, and the operation is repeated until a valid move is reached or until all pools are visited. This operator is applied to a portion of pools.
- **Divide:** A full pool is randomly selected and divided into two pools; each must contain more than two users. This condition avoids the penalty over cost in the case of a pool with one user.
- **Divide-Merge:** This creates a new pool with two users from two different pools. First, one pool containing more than two users is randomly selected. Then, the pools of the same habitat are sorted according to the gravity center of the first selected pool. This operator associates one user

from the first selected pool with another user of the other pools, in the above-mentioned order.

- **MoveAll:** The goal of this operator is to move each user of a selected pool to another pool of the same habitat. If one user of the concerned pool is not successfully moved, the operation fails. This operator is applied to a portion of pools.
- **BrokeAloneUser:** This operator is applied to pools with one user. It first attempts to merge the pool; if the merge fails, it attempts to move a user from another pool with more than one user. If both of these operations fail, a swap is attempted.

These neighborhood operators are used as described by algorithm (6). The operators ‘divide’ and ‘divide-merge’ are used less frequently than the others in the mutation. Their less-frequent use is attributed to the fact that these two operators increase the number of pools that can degrade a number of solutions. As such, they could result in the intensification process being penalized.

Algorithm 6: HMMBBO mutation algorithm for LTCPP

**Mutation (int minMutations, int maxMutations)**

**For**  $i = 1$  to  $N$  **do**

    Compute the mutation rate  $m_i$  with equation (15)

    // define the max number of iterations between *minMutations* and *maxMutations*

$F = \text{random}(\text{minMutationstomaxMutations})$

**For**  $j = 1$  to  $F$  **do**

**If**  $\text{rand}(0, 1) < m_i$  **then**

            //  $l \in \text{Neighbourhoodoperators}(1, \dots, n)$

**For**  $l = 1$  to  $n$  **do**

                //Select randomly  $m$  distinct pools between  $m_1$  and  $m_2$

                // according operator  $l$

**For**  $k = 1$  to  $m$  **do**

                    // Apply a neighborhood operator  $l$  on each selected pool

**Operator**<sub>1</sub>( $H_i, \text{pool}_k$ )

**End for**

**End For**

**End If**

**End For**

**End for**

Information contained in the mutated habitats is propagated by the migration operator and improved by the VNSTL, as described in the section dedicated to the HMMBBO algorithm. The VNSTL uses the same operators as mutation, although in this case they are used to improve and not to diversify. Therefore, each neighborhood operator succeeds if it enhances the quality of the habitat to which it is applied. Two user parameters are defined



for VNSTL, the number of solutions for enhancement and the number of authorized successive fails for each neighborhood operator.

## Results and Comparisons

Experiments have been conducted by using the new approach based on an improved BBO algorithm. The objects of these experiments have included those derived by Guo (2012) and from VRP hard instances (Baldacci, Maniezzo, and Mingozzi 2004). The results of HMMBBO were compared to those obtained by Guo (2012). The instances are composed from two types, clustered and randomized. Each type is then composed of nine instances divided into three groups of three instances of 100, 200, and 400 users. The HMMBBO was implemented in Java, and experiments were conducted on a laptop equipped with an Intel processor Core™ i5-4210 U 1.7 Mhz CPU. All results were obtained after 30 runs.

The results from the new approach were compared with those obtained by Guo (2012) and the approaches multi-agent self-adaptive genetic algorithm (AGA), GGA, and CAC. Two versions of HMMBBO were performed, HMMBBO1 and HMMBBO2. These versions are different on the VNSTL parameters. For HMMBBO1, the number of modified solutions by VNTSL for each iteration is one, and the number of authorized fails for each neighborhood operator is one. The values of these two parameters for HMMBBO2 are a random number between one and two, and five respectively.

In addition, HMMBBO1 outperforms the three other approaches on five clustered instances, and it is outperformed only by AGA in the instances of C202, C203, and C402. In contrast, HMMBBO1 is outperformed by all the approaches in C201. With HMMBBO2, the obtained results from HMMBBO1 are improved in all clustered instances. In fact, HMMBBO2 outperforms AGA on C202 and C402. On the other hand, AGA outperforms HMMBBO2 in C201 and C203, yet the latter is closest to CAC for this instance. Table 1 provides the results for clustered instances. Furthermore, HMMBBO1 and HMMBBO2 are more efficient for random instances, as HMMBBO1 outperforms AGA, GGA, and CAC in all

**Table 1.** Results for clustered instances.

Instance	Size	HMMBBO1		HMMBBO2		AGA		GGA		CAC	
		Best	Avg	Best	Avg	Best	Avg	Best	Avg	Best	Avg
C101	100	1150.2	1164.3	<b>1148.4</b>	<b>1159.3</b>	1585.5	1585.5	1585.5	1599.3	1585.5	1593.4
C102	100	1334.8	1350.9	<b>1330.6</b>	<b>1345.9</b>	1701.9	1704.1	1701.9	1712.0	1706.8	1728.2
C103	100	<b>1312.3</b>	1331.7	1316.8	<b>1331.1</b>	1508.6	1511.6	1513.7	1543.9	1508.6	1527.5
C201	200	2749.9	2824.2	2734.4	2787.3	<b>2626.8</b>	<b>2671.5</b>	2672.2	2749.4	2703.1	2717.7
C202	200	2761.5	2822.4	<b>2749.7</b>	<b>2792.1</b>	2806.7	2811.9	2836.7	2876.5	2879.2	2892.9
C203	200	2794.2	2868.8	2780.5	2832.3	<b>2716.0</b>	<b>2724.6</b>	2716.0	2891.8	2769.3	2834.1
C401	400	4947.1	5007.7	<b>4838.3</b>	<b>4912.2</b>	5425.9	5448.9	5489.4	5690.6	5533.3	5618.6
C402	400	4492.7	4573.9	<b>4418.7</b>	<b>4509.6</b>	4518.2	4538.0	4548.3	4786.4	4518.2	4760.3
C403	400	5327.6	5429.2	<b>5306.3</b>	<b>5364.2</b>	5725.9	5796.2	5909.6	6085.2	5930.7	6046.4

instances except R402, for which HMMBBO1 and AGA demonstrate the best result. However, HMMBBO2 outperforms HMMBBO1, AGA, GGA, and CAC in all random instances. The result comparisons for randomized instances are reported in Table 2. On the other hand, HMMBBO1 and HMMBBO2 are less time consuming than the other approaches. While the CPU time increases quickly for the three other approaches, HMMBBO1 and HMMBBO2 are less time consuming. Figures 5 and 6 illustrate the CPU time progression for clustered and randomized instances.

Conclusion

This paper proposes the HMMBBO approach to address the LTCPP by using migration and elitism as intensification processes and performing the diversification process with a modified mutation operator. Moreover, the diversification phase has been combined with a propagation phase based on the use of a migration operator and an improvement with VNSTL, without using elitism. As a result of these two phases and the absence of elitism, HMMBBO avoids losing new information provided by the mutation operator. Consequently,

Table 2. Results for randomized instances.

		HMMBBO1		HMMBBO2		AGA		GGA		CAC	
Instance	Size	Best	Avg	Best	Avg	Best	Avg	Best	Avg	Best	Avg
R101	100	1781.3	1793.9	<b>1778.2</b>	<b>1788.3</b>	2207.1	2214.7	2207.1	2235.9	2223.1	2283.2
R102	100	<b>1443.6</b>	1463.4	<b>1443.6</b>	<b>1453.2</b>	1824.5	1835.7	1824.5	1867.5	1841.4	1874.3
R103	100	<b>2130.9</b>	2146.7	<b>2130.9</b>	<b>2145.6</b>	2200.3	2226.9	2209.1	2286.0	2235.9	2313.4
R201	200	3810.3	3909.0	<b>3790.7</b>	<b>3864.0</b>	3966.2	4117.1	4034.8	4188.3	4156.1	4231.8
R202	200	3311.3	3404.1	<b>3292.1</b>	<b>3364.2</b>	3646.8	3666.7	3646.8	3751.7	3717.2	3824.2
R203	200	3329.4	3388.6	<b>3292.7</b>	<b>3342.8</b>	3923.2	3982.1	3923.2	4158.4	4164.7	4304.6
R401	400	7226.1	7327.8	<b>7047.2</b>	<b>7176.6</b>	7354.2	7405.1	7514.9	7799.5	7891.4	8033.7
R402	400	6158.4	6232.2	<b>6060.5</b>	<b>6127.8</b>	6172.7	6222.5	6172.7	6254.0	6365.2	6559.7
R403	400	7481.8	7587.3	<b>7330.1</b>	<b>7425.1</b>	7602.0	7695.0	7670.2	7872.9	8023.4	8129.5

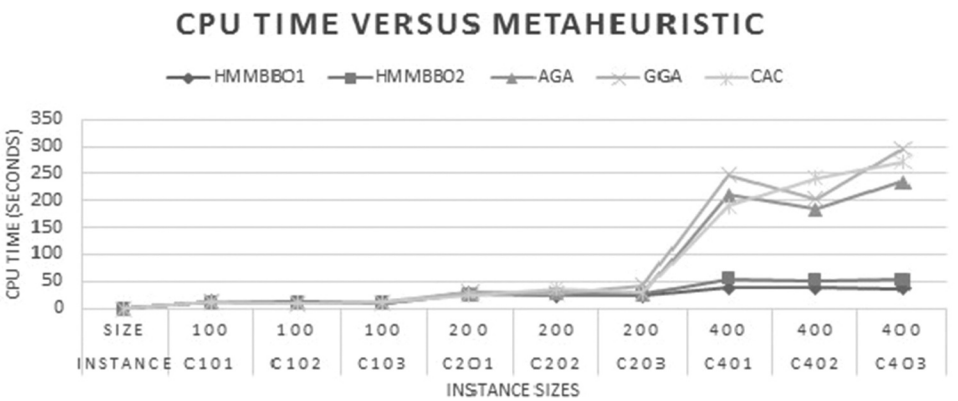
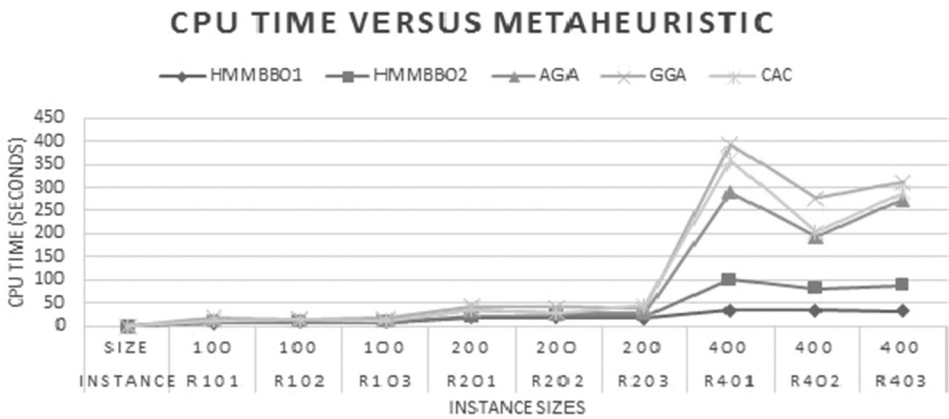


Figure 5. CPU time progression for clustered instances.



**Figure 6.** CPU time progression for randomized instances.

HMMBBO has the advantages of BBO, thus escaping local optima traps by disabling the negative effects of elitism.


In fact, HMMBBO has been demonstrated to be more efficient than the well-known approaches of AGA, GGA, and CAC in 16 instances and in 18 for its second version, HMMBBO2. In the other instances, HMMBBO provides promising and competing results compared to those obtained by AGA, GGA, and CAC. These results have been achieved with the two HMMBBO versions in less time than with AGA, GGA, and CAC.

However, the HMMBBO results from two instances have been outperformed by the results provided by AGA. Therefore, the results for these two instances should be improved, in addition to improving HMMBBO in general. Any future research must be extended to include other metaheuristics that address the LTCPP.

### Acknowledgments

Our sincere thanks to the Directorate General for Scientific Research and Technological Development (DGRSDT), Ministry of Higher Education and Scientific Research for its support of this work.

### ORCID

Rachid Kaleche  <http://orcid.org/0000-0002-9326-2944>  
Zakaria Bendaoud  <http://orcid.org/0000-0003-3091-5044>  
Karim Bouamrane  <http://orcid.org/0000-0002-6953-0339>

## References

- Baldacci, R., V. Maniezzo, and A. Mingozzi. 2004. An exact method for the car pooling problem based on lagrangean column generation. *Operations Research* 52 (3):422–39. doi:[10.1287/opre.1030.0106](https://doi.org/10.1287/opre.1030.0106).
- Berghida, Meryem, and Abdelmadjid Boukra. 2015. EBBO: An enhanced biogeography-based optimization algorithm for a vehicle routing problem with heterogeneous fleet, mixed backhauls, and time windows. *The International Journal of Advanced Manufacturing Technology* 77 (9–12): 1711–25.
- Cao, J., F. Wang, and P. Li. 2014. An improved biogeography-based optimization algorithm for optimal reactive power flow. *International Journal of Control and Automation* 7 (3):161–76. doi:[10.14257/ijca.2014.7.3.16](https://doi.org/10.14257/ijca.2014.7.3.16).
- Chatterjee, A., P. Siarry, A. Nakib, and R. Blanc. 2012. An improved biogeography-based optimization approach for segmentation of human head CT-scan images employing fuzzy entropy. *Engineering Applications of Artificial Intelligence* 25 (8):1698–709. doi:[10.1016/j.engappai.2012.02.007](https://doi.org/10.1016/j.engappai.2012.02.007).
- Correia, G., and J. M. Viegas. 2008. A structured simulation-based methodology for carpooling viability assessment. Transportation Research Board 87th Annual Meeting, Washington DC, USA.
- Du, D., and D. Simon. 2013. Complex system optimization using biogeography-based optimization. *Mathematical Problems in Engineering* 2013:1–17.
- Ferrari, E., R. Manzini, A. Pareschi, A. Persona, and A. Regattieri. 2003. The car pooling problem: Heuristic algorithms based on savings functions. *Journal of Advanced Transportation* 37 (3):243–72. doi:[10.1002/atr.5670370302](https://doi.org/10.1002/atr.5670370302).
- Gillett, B., and L. Miller. 1974. A heuristic algorithm for the vehicle dispatch problem. *Operations Research* 22 (2):340–49. doi:[10.1287/opre.22.2.340](https://doi.org/10.1287/opre.22.2.340).
- Glover, F. 1977. Heuristics for integer programming using surrogate constraints. *Decision Sciences* 8 (1):156–66. doi:[10.1111/j.1540-5915.1977.tb01074.x](https://doi.org/10.1111/j.1540-5915.1977.tb01074.x).
- Guo, Y. 2012. Metaheuristics for solving large size long-term car pooling problem and an extension. Ph.D. thesis, Universit'e Lille Nord de France
- Guo, Y., G. Goncalves, and T. Hsu. 2011. A guided genetic algorithm for solving the long-term car pooling problem, "2011 IEEE Workshop On Computational Intelligence In Production And Logistics Systems (CIPLS), pp. 1-7, doi: [10.1109/CIPLS.2011.5953357](https://doi.org/10.1109/CIPLS.2011.5953357).
- Guo, Y., G. Goncalves, and T. Hsu. 2012. A clustering ant colony algorithm for the long-term car pooling problem *International Journal of Swarm Intelligence Research*. 3 (2):39–62.
- Ho, S. C., W. Y. Szeto, Y.-H. Kuo, J. M. Y. Leung, M. Petering, and T. W. H. Tou. 2018. A survey of dial-a-ride problems: Literature review and recent developments. *Transportation Research Part B: Methodological* 111:395–421. doi:[10.1016/j.trb.2018.02.001](https://doi.org/10.1016/j.trb.2018.02.001).
- Ma, H. 2010. An analysis of the equilibrium of migration models for biogeography-based optimization. *Information Sciences* 180 (18):3444–64. doi:[10.1016/j.ins.2010.05.035](https://doi.org/10.1016/j.ins.2010.05.035).
- MacArthur, R., and E. Wilson. 1967. *The theory of Island biogeography*. Monographs in population biology. Princeton: Princeton University Press.
- Maniezzo, V., C. Antonella, D. Vigo, and H. Hildmann. 2004. An ANTS heuristic for the long – Term car pooling problem. *New Optimization Techniques in Engineering* 15:411-430.
- Mladenovic, N., and P. Hansen. 1997. Variable neighborhood search. *Computers & Operations Research* 24 (11):1097–100. doi:[10.1016/S0305-0548\(97\)00031-2](https://doi.org/10.1016/S0305-0548(97)00031-2).
- Mlayah, I., I. Boudali, and M. Tagina. 2018. A hybrid variable neighborhood tabu search for the long-term car pooling problem. In *Hybrid intelligent systems*. vol. 923, ed. A. M. Madureira, A. Abraham, N. Gandhi, and M. L. Varela, 481–90. Cham. Springer International Publishing.

- Simon, D. 2008. Biogeography-based optimization. *IEEE Transactions on Evolutionary Computation* 12 (6):702–13. doi:[10.1109/TEVC.2008.919004](https://doi.org/10.1109/TEVC.2008.919004).
- Su, S., F. Zhou, and H. Yu. 2019. An artificial bee colony algorithm with variable neighborhood search and tabu list for long-term carpooling problem with time window. *Applied Soft Computing* 85 (December):105814. doi:[10.1016/j.asoc.2019.105814](https://doi.org/10.1016/j.asoc.2019.105814).
- Varrentrapp, K., V. Maniezzo, and T. Stützle. 2002. The long-term car-pooling problem: On the soundness of the problem formulation and proof of NP-completeness. *Technische Universität Darmstadt*.
- Volk, T., *Gaia's Body*. 2003. *Toward a physiology of earth*. Cambridge, MA: MIT Press.
- Yan, S., C.-Y. Chen, and Y.-F. Lin. 2011. A model with a heuristic algorithm for solving the long-term many-to-many car pooling problem. *IEEE Transactions on Intelligent Transportation Systems* 12 (4):1362–73. doi:[10.1109/TITS.2011.2158209](https://doi.org/10.1109/TITS.2011.2158209).
- Yang, X. S. 2014. Swarm intelligence based algorithms: A critical analysis, *Evolutionary Intelligence*. 7 (1):17–28.
- Zhang, X., D. Wang, and H. Chen. 2019. Improved biogeography-based optimization algorithm and its application to clustering optimization and medical image segmentation. *IEEE Access* 7:28810–25. doi:[10.1109/ACCESS.2019.2901849](https://doi.org/10.1109/ACCESS.2019.2901849).
- Zhang, Y., S. Wang, Z. Dong, P. Phillip, G. Ji, and J. Yang. 2015. Pathological brain detection magnetic resonance imaging scanning by wavelet entropy and hybridization of biogeography-based optimization and particle swarm optimization. *Progress In Electromagnetics Research* 152:41–58. doi:[10.2528/PIER15040602](https://doi.org/10.2528/PIER15040602).
- Zhao, F., S. Qin, Y. Zhang, W. Ma, C. Zhang, and H. Song. 2019. A hybrid biogeography-based optimization with variable neighborhood search mechanism for no-wait flow shop scheduling problem. *Expert Systems with Applications* 126:321–39. doi:[10.1016/j.eswa.2019.02.023](https://doi.org/10.1016/j.eswa.2019.02.023).
- Zheng, Q., R. Li, X. Li, N. Shah, J. Zhang, F. Tian, J. Li, and J. Li. 2016. Virtual machine consolidated placement based on multi-objective biogeography-based optimization. *Future Generation Computer Systems* 54:95–122. doi:[10.1016/j.future.2015.02.010](https://doi.org/10.1016/j.future.2015.02.010).