Smart management system for electric bicycles loan

Jorge Revuelta Herrero[1], Juan Francisco de Paz Santana[1], Gabriel Villarrubia González[1], Juan Manuel Corchado Rodríguez[1]

Computer Science, Salamanca University.
Plaza de la Merced s/n. 37008, Salamanca, España
{jrevuelta,fcofds,gvg,corchado}@usal.es

Abstract. Nowadays, in many urban areas we can find vehicle renting services, whose members can rent and deposit a vehicle. These services operate both nationally and internationally and the most commonly rented vehicle is the bicycle, this is because it is more personal, the hiring cost is low and it has no negative impact on the environment.
This work presents an optimization method that seeks to minimize the time a bicycle user has to wait until they can travel to their desired destination; this will be done by adapting the stations at which the users deposit their bikes at the end of their trips.

Key words: Bicycles, BSS, Electric Bicycle

1 Introduction

The main problem facing bicycle rental services is providing service and satisfying the demands of a great numbers of platform users.

In the literature, the main way of tackling this problem is the use of a bicycle carrying trailer of a particular size, that would be in charge of redistributing vehicles at the service´s rental stations throughout the day.

This work focuses on the creation of an optimization method that will help minimize the time platform users have to wait in order to rent a bike. This will be done along the day, by altering in a controlled way, the parameters of a series of trips known by some users. The method is based on the implementation of two individual algorithms: a heuristic that allows to alter the destination of the trips made by the users and a second heuristic that allows to alter both the place of origin and destination of the trips made by the users. Alterations that can cause an unforeseen shift in the user´s trip, should be minimal, since they penalize the performance of the solution which must be able to satisfy user needs in a short period of time.

In the 2 section, we review previous work related to optimization in the field of bicycle rental services. The section 3, presents the proposed methods, the theoretical bases that underpin it and the methodology they conduct. In the 4 section, the proposed methods are applied to a specific problem. The section 5 presents the results of the application of the proposed methods on the case study, as well as the conclusions made.

2 State of the art

In the literature there are numerous works and articles related to optimization problems in the field of bicycle rental systems. Each of them deals with the same issue: the optimization of resources so that the service provided to the users could be more efficient and have less setbacks.

The work proposed by [3] raises the problem of Urban Bicycle Renting Systems. The infrastructure of the vehicle routes that connect the different stations and storage centers requires optimization. In the study, it is proposed to solve the *VRP* (Vehicle Routing Problem) with multiple storage areas, to which a fusion of evolutionary, nature inspired algorithms are applied, [2] with *Ant Colony Systems* [5].

The work proposed by [11] addresses a crucial factor in the optimization of bicycle rental systems; the user´s ability to pick up the bicycle at any station that he chooses as the point of origin and deposit the bike at any station that is the closest to the desired destination. To achieve this, the authors apply balancing systems to the *VRP* problem, considering a static case. The vehicles make routes through the stations and they must return to specific locations that are known a priori, and each and every station can be visited only once by each vehicle. The problem is addressed as a *traveling salesman problem* [10] with additional constraints.

In the work of [4] the authors propose the rebalancing of a bicycle rental system through an algorithm of destruction and repair. The metaheuristic of destruction and repair is combined and with a constructive heuristic and several local search procedures. The proposed algorithm is adapted to solve the *one-commodity Pickup and Delivery Vehicle Routing Problem with Maximum Duration (1-PDVRPD)*, which is the variant of the *Bike Sharing Problem* where the main restriction is the maximum duration for each route.

Other works, such as the one proposed by [6], also focusing on the lack of resources in the bicycle renting systems, approach the issue by constructing the stations from a series of maximum resources available, optimizing their locations according to the demands of the particular geographic location, so that the quantities of bicycles distributed are adequate to the demands in that area. The proposed method can be used on a restricted budget and makes use of optimization in order to maximize user demand coverage in bicycle rental systems. It uses strategic decisions in order to locate bicycle stations and define the size of the system (number of stations and bicycles) combined with operational decisions (bicycle relocation). The final model determines the optimal location of the stations and the number of bicycles per station.

The article by [1] divides the bicycle rental problem into two main phases: the first phase makes an estimate of the future unfulfilled demands at each of the stations, for a certain period of time, and the possible number of bicycles at the beginning of each period. The second phase uses these estimates to guide the proposed redistribution algorithms. The proposed quality of service parameter uses known data which is provided by the rental system and which, given the initial number of bicycles in the establishment, performs approximations and

statistical predictions for the future. This quality information measurement is applied in the second phase, a *VRP* in which the routes of the carrying trailers, which relocate the bicycles, so that they first reach the stations with the highest bicycle demand, they also collect the damaged bicycles and return them to the store.

3 Proposed method

In this section we describe the methods and algorithms proposed, in the light of their limitations and the different points of view on the problem that we are dealing with and which has already been discussed in the previous section.

The main issue that impedes the implementation of the proposed method is the fact that the search in the solution space is discrete. Only the coordinates of the search space that correspond to stops in the different temporary instants (which are also discrete), are possible solutions or steps in the resolution of the problem.

In the same way, the time component that adds one more dimension to the problem causes the application of some of the previously presented algorithms to be discarded. The dependencies between the different solutions made for each set of trips are propagated in time in a chained way until the final solution is obtained.

It is for this reason, that the heuristic proposed by the method must be implemented iteratively, so that the solution to the problem does not belong to a single instance of the algorithm (local solution) but to a finite set of them, be solved in parallel and be able to obtain the best solution among all of them as a global solution to the problem.

The method used may be similar to the one applied by *Scatter Search* [7], [8], [9], since the solutions belong to the discrete point domain of the search space (coordinates of the service stops) along with a discrete-space search implementation similar to that proposed in *Swarm Particle Optimization* [12], where it is the set of particles that individually looks for candidate solutions problem and not the individual knowledge of the swarm of each where the solution to the problem is found.

The basic idea of the optimization procedure is that a set of users who make a specified route, at predetermined departure hours, can travel to their destination in the shortest time possible, and even though all routes will be executed, situations where the user cannot find a bicycle at the point of origin of their route, will be avoided.

The system can be modeled in the form of a fully connected graph, in which each of the nodes is a bicycle station and links the possible routes that can be traveled by users.

Each of the nodes on the graph, which represents a station in our system, contains information about its status, as well as the inclusion of states derived from the demand of the users when making routes. It is for this reason that the nodes contain a table of resources and demands that allow to manage the

system´s capacities for each one of the stations so that the best global solution is obtained. Resources and demands are represented are always represented in relation to the time frame which allows the system to place in discrete time units the arrival and departure of both resources and demands.

Fig. 1. Example of resource and demand tables for a stop in the system.

As can be seen in the figure 1, each of the stations is defined at the initial point of the route traveled through the demand made by the user on the travel platform, where the starting point and the finishing point of the trip can be seen and an established departure hour. In addition, the stations have bicycles that each user of the platform travels from a Point to an end point at a specific time at the beginning of the route. In addition, the stations have bicycles available for temporary use for the members of the platform.

It is easy, therefore, to be in a situation where a user who wants a trip at a given time, finds that there are no bicycles available at their place of origin. The objective of the proposed method is, therefore, to minimize the users´ non-useful time, understanding non-useful time when a user has to wait for an available bicycle; sum of the waiting times of the users who do not find a bicycle in their place of origin at the moment of demand and the time it takes them to reach a different station, recommended by the system.

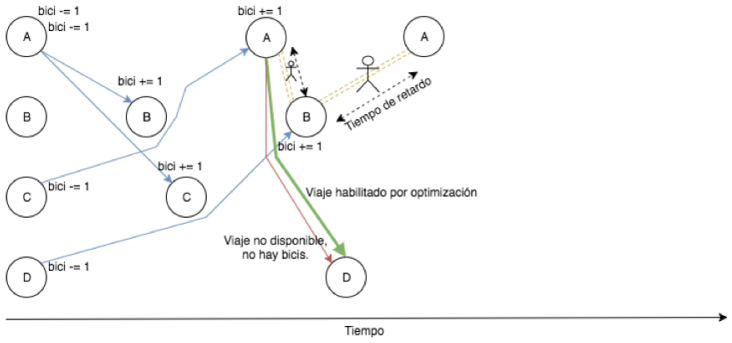

Fig. 2. Graphical representation of the proposed method.

As can be seen in the illustration 2, the proposed method allows the system to send users to stations that do not match the preferences chosen by the user in their route, thus allowing the system to restock the bicycles at other stations which have high demands and minimizing global waiting times. This movement produced by the adjustment of the method considers that takes into account that the user must travel by default more slowly between the user preference and the system recommendation, so that time cost is also used as a measure to be minimized in the algorithm.

The algorithm is based on a three-dimensional space where the two-dimensional spatial planes *(x, y)* are parallel to the third dimension *(t)*, representing the geographical locations of the stations in our problem as seen in Figure 3. Each demand of a user generates a service request layout coinciding with the user´s route demand and intends to find a P' for each P result of completing the trip and that it sums up the successive delays produced iteratively for each demand.

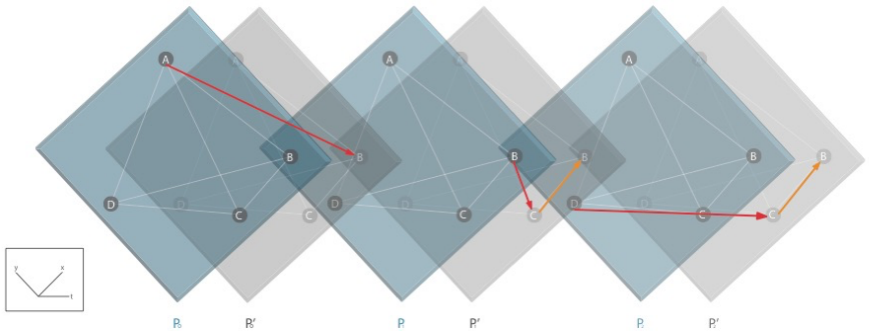

Fig. 3. Graphical 3D representation of the proposed optimization method.

The result of the algorithm is the set of demand plans $P_0, ..., P_n$ and target plans $P'_0, ..., P'_n$ representing the destinations chosen and summing up all the delays produced T_d.

The first of the proposed methods only considers the change of the destination to which the user is guided, for each of the system demands, as shown below:

1. A $2D$ space is generated, of size $m * n$ (where m and n are the dimensions of the coordinate system) in which stations are placed randomly k with b of initial bikes each.

$$\begin{pmatrix} (k_i|b)_{0,0} & \cdots & (k_i|b)_{0,n} \\ \vdots & \ddots & \vdots \\ (k_i|b)_{m,0} & \cdots & (k_i|b)_{m,n} \end{pmatrix}, k_i \in K = k_0, ..., k_k \tag{1}$$

2. A vector of a defined size is generated with d demands on coordinates (x, y) corresponding to any of the $k_n \in K$ stations.

$$D = d_0, ..., d_n, \qquad d_i = (x_i, y_i), t_i \tag{2}$$

3. Also, a discrete instant is generated t for each $d \in D$ which represents the discrete instant in which the request for the trip is made.

$$\forall d_i \in D \, \exists \, t_i \in \mathbb{N} \tag{3}$$

4. The algorithm must therefore optimize, so that the delay time in the demands is minimal in D taking into account the limitations of b_k bikes available at the k stations and the experienced changes in time due to the execution of the requests in D.
5. For each demand $d_i \in D$:
6. Choose the layout $(P|P')_k$ nearest temporally prior to layout P_i as the initial state or choose the initial state P_0 if no other exists.
7. If it is possible to travel from $d_i \exists \, k_d \mid b_d > 0$ this means, available bicycles, it is used as origin and subtracts a bicycle from k_d.
8. A random deviation coordinate, as shown in Figure 4, is generated on the ideal destination coordinate for the request, following a Gaussian distribution with center components x, y of the destination coordinate and configurable deviation, and obtains the closest station to the generated coordinate.
   If the obtained station has enough resources (parking spaces available), it is used as a destination. In the case of the lack of spaces, the standard deviation configured in one unit of the Gaussian distribution is expanded and another coordinate is obtained.
   This step is repeated until a station with sufficient resources is found, different from the origin of the request or until a maximum number of iterations u is made.
   In case of not having obtained solution in the search of destination, the raised solution is discarded like candidate solution.

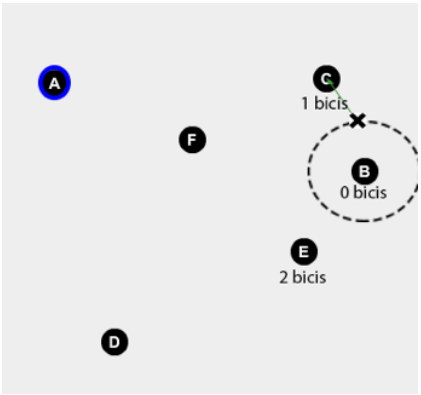

Fig. 4. Random generation of locations by standard deviation ranges.

9. The deposited resource (the bicycle left at the station) is added to the destination station chosen and the destination is verified so that it matches the destination chosen by the user. If the destinations do not match, the travel time is obtained (by the Manhattan distance in the location matrix) from the selected destination to the desired destination, by multiplying the speed difference factor between cycling and walking, and it is added to T_d or global *delay time* of the problem.

$$\begin{aligned} T_d &= T_d + Tv_{fd_i,fe_i} * k_{pedestrian-bicycle} \\ \text{where} fe_i &= \text{chosen destination to } d_i \\ \text{where} fd_i &= \text{ideal destination to } d_i \end{aligned} \tag{4}$$

10. The time taken to travel from the origin to the selected destination is calculated using Manhattan and the layout P_i' is located at the time instant with respect to the origin P_i plus the calculated travel time.
11. Once steps 5 ... 10 are performed for each d_i the final value to be reduced T_d is obtained.

In addition, an alternative to the previous methods is proposed which allows to adjust the location origin of the set of travel requests made by the users.

The following are the steps that comprise it:

1. Steps 1 to 4 are applied in the same way as in the previous method.
2. In the main loop that travels $d_i \in D$, a random coordinate is generated first, following the same rules as those used to generate the target. The restriction applied in this case is, that there are enough bicycles at the chosen station of origin.
3. The travel time is calculated for the station which is different to the one selected by the user, as done in step 10 of the previous method.
4. The remaining steps are performed identically as in the previous method.

4 Case study

In order to validate the methods proposed above, a simulated case study has been developed with plausible data which could correspond to reality.

Variables:

- Number of trips: 50, 100, 150.
- Period of time in which they occur: 12 hours.
- Number of stations: 12
- Initial bicycles per station: 7
- Maximum bicycles per station: 12
- Number of particles per problem: 100

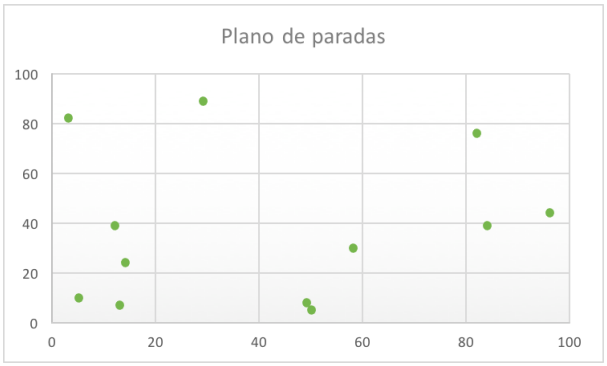

Fig. 5. Locations of the case study stations.

The implementation of the methods has been done using the programming language *Python*, the 2.7 version using a plane of dimensions $100x100$ as we can see in the figure 5 on which the distances are calculated using *Manhattan* (for its resemblance to apples in a city).

5 Results and conclusions

The generation of circle trips was the first case that was tested in order to verify the validity of the proposed methods. That is, to generate a number of trips equal to the number of available bicycles, all destined to the same station and from that stop to generate the same number of trips, thus passing by all the stations and finishing with the starting one. The optimum result is that for this type of travel arrangement, cumulative delays should not occur in any case, since there will always be sufficient resources and any alteration that would cause the user to walk, would penalize the result.

The results were satisfactory, obtaining a $T_d = 0$ in all cases.

Also, in order to test both of the methods proposed, they have been applied to the same data set for implementation and in different amounts of routes requested, ranging from 50 to 150 trips.

The generation of the data for the execution of the different tests shown below is the generation of N trips within the available time range in seconds (therefore, for this case: $[0, 43200]$) with random origin between the set of stations described in the previous section and random destination different from the origin between all the stops of the defined set and that is identical for all tests executed.

Five pairs of tests have been performed for each set of trips of the same number and the same random data have been generated for each pair, the pair composed being the best solution for the data of the algorithm *Destination* and the best solution for the data of the algorithm *Origin-Destination*.

Also, it should be noted that the pairs of tests have been ruled out in cases where neither algorithm offers a solution (i.e., infinite solution).

As can be seen in the figures 6, 7 and 8, the method that considers the different route destination requests and their modification, achieves better optimization results in all cases, in this regard, the method also allows to change the point of origin of the request, according to the needs of the system.

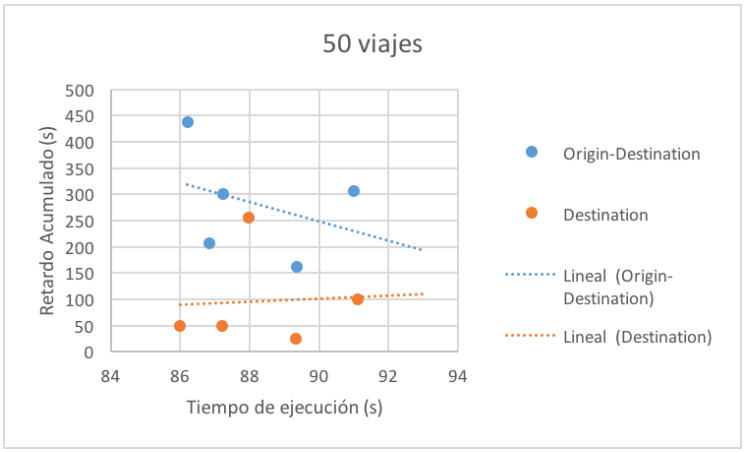

Fig. 6. Results for 50 trips.

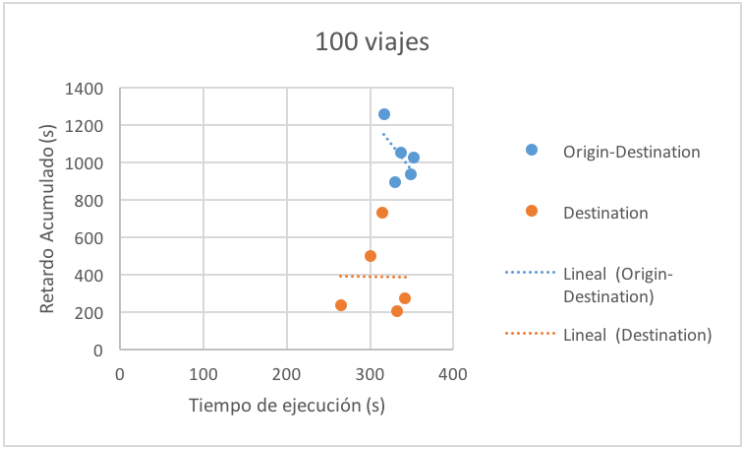

Fig. 7. Results for 100 trips.

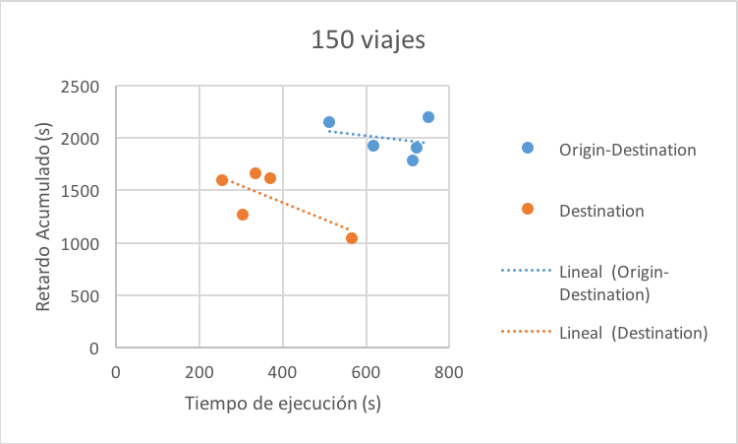

Fig. 8. Results for 150 trips.

Likewise, the 1 table shows the results of the Student T-Test, which allows us to determine if there is a significant difference between the means of the performance values calculated by the groups that shape the execution of each of the two variants in the proposed method, through the probability that two results of the different groups are identical.

The null hypothesis H_0 would therefore, confirm that the results delivered for each number of trips and for the two methods proposed are similar. Since the values obtained are all < 0.05, we can reject the null hypothesis H_0 and assert that the two samples are significantly different and that therefore the results delivered by the two methods differ from each other (take H_1 as true).

Table 1. T-Test results for performance in different sets of trips.

Viajes	50	100	150
T-Test	0,0186	0,0006	0.0047

It should be noted that as the number of journeys is increased, the *Destination* method may not find a valid solution (i.e., infinite solution) to the problem due to constraints caused by the lack of resources available at stations), while the *Origin-Destination* method yields results (albeit with a lower efficiency) in all the problems generated. Another aspect to consider is the execution time of both methods, for a reduced number of trips (which does not exceed the number of global resources of the system) the end solutions of both methods have very proximate time margins. This, however, is not adhered to when the number of resources in the problem are less than the number of demands in the problem; in this case the Destination method obtains results (when it is able to obtain

them) in a shorter time than the Origin-Destination method as shown in figure 9.

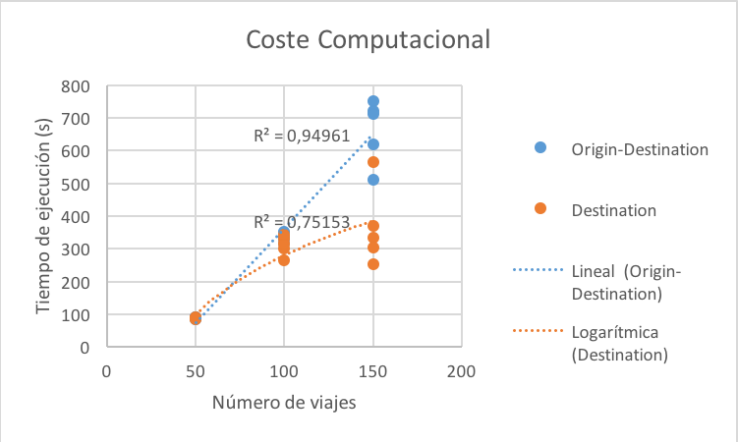

Fig. 9. Computational cost for executions.

The goal of future works will be to develop the methods by expanding shared knowledge, as applied in the *Ant Colony Optimization*, whose ants in addition to possessing individual and private information also share information about the acquired general knowledge and generate valid candidate solutions based on local minimum solutions.

6 Acknowledgements

This work was supported by the Spanish Ministry, Ministerio de Economía y Competitividad and FEDER funds. Project. SURF: Intelligent System for integrated and sustainable management of urban fleets TIN2015-65515-C4-3-R.

Bibliography

[1] Ramon Alvarez-Valdes, Jose M. Belenguer, Enrique Benavent, Jose D. Bermudez, Facundo Muñoz, Enriqueta Vercher, and Francisco Verdejo. Optimizing the level of service quality of a bike-sharing system. *Omega*, 62: 163–175, 2016. ISSN 03050483. doi: 10.1016/j.omega.2015.09.007.

[2] T Bäck, DB Fogel, and Z Michalewicz. Handbook of evolutionary computation. *Release*, 1997.

[3] Camelia Chira, Javier Sedano, José R Villar, Mónica Cámara, and Emilio Corchado. Urban bicycles renting systems: Modelling and optimization using nature-inspired search methods. 2014. doi: 10.1016/j.neucom.2013.07.051.

[4] Mauro Dell'Amico, Manuel Iori, Stefano Novellani, and Thomas Stützle. A destroy and repair algorithm for the Bike sharing Rebalancing Problem. *Computers & Operations Research*, 71:149–162, 2016. ISSN 03050548. doi: 10.1016/j.cor.2016.01.011.

[5] M. Dorigo and G. Di Caro. Ant colony optimization: a new meta-heuristic. In *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, volume 2, pages 1470–1477. IEEE, 1999. ISBN 0-7803-5536-9. doi: 10.1109/CEC.1999.782657.

[6] Ines Frade and Anabela Ribeiro. Bike-sharing stations: A maximal covering location approach. *Transportation Research Part A: Policy and Practice*, 82:216–227, 2015. ISSN 09658564. doi: 10.1016/j.tra.2015.09.014.

[7] F Glover. Heuristics for integer programming using surrogate constraints. *Decision Sciences*, 1977.

[8] F Glover. Tabu search-part I. *ORSA Journal on computing*, 1989.

[9] F Glover. Tabu search—part II. *ORSA Journal on computing*, 1990.

[10] Karla L. Hoffman, Manfred Padberg, and Giovanni Rinaldi. Traveling Salesman Problem. In *Encyclopedia of Operations Research and Management Science*, pages 1573–1578. Springer US, Boston, MA, 2013. doi: 10.1007/978-1-4419-1153-7_1068.

[11] Ahmed Abdelmoumene Kadri, Imed Kacem, and Karim Labadi. A branch-and-bound algorithm for solving the static rebalancing problem in bicycle-sharing systems. *Computers & Industrial Engineering*, 95:41–52, 2016. ISSN 03608352. doi: 10.1016/j.cie.2016.02.002.

[12] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95 - International Conference on Neural Networks*, volume 4, pages 1942–1948. IEEE, 1995. ISBN 0-7803-2768-3. doi: 10.1109/ICNN.1995.488968.