Fuzzy Greedy Randomized Adaptive Search Procedure and Simulation Model to Solve the Team Orienteering Problem with Time Windows

Airam Expósito-Márquez^(✉), Christopher Expósito-Izquierdo,
Belén Melián-Batista, and José Marcos Moreno-Vega

Universidad de La Laguna, Avenida Astrofísico s/n, La Laguna, Spain
{aexposim,cexposit,mbmelian,jmmoreno}@ull.edu.es

Abstract. Tourism is a relevant economic activity that provides important resources (income, employment,...) to countries. When a tourist visits a country or city, he/she wants to know their points of interest. To do this, he/she must select some of the places according to his/her preferences and design routes to visit them. This problem can be adequately modeled as a Team Orienteering Problem with Time Windows (TOPTW). In this paper we propose a fuzzy GRASP and a multi-agent simulation model to solve the TOPTW. Our proposal incorporates two criteria to build the restricted candidate list. The computational results obtained show the validity of the proposal.

Keywords: Tourism · GRASP · Fuzzy · Simulation model

1 Introduction

Tourism plays an important role in the economy of the countries and represents an important source of income. According to the World Tourism Organization¹, 1400 million people moved around the world in 2019. A significant percentage of these tourists want to visit the points of interest of the country or city they visit. Travel agencies and specialized companies offer routes to visit these points that generally do not satisfy the preferences of all tourists. An interesting route for one tourist may not be suitable for another. It is therefore necessary to design tools that provide routes adapted to the preferences of each tourist. In addition to tourist preferences, other factors must be considered to build customized routes. Some of the most relevant are the importance of the points of interest, the duration of the routes and the opening hours.

The previous problem can be modeled as a Team Orienteering Problem with Time Windows (TOPTW). The TOPTW is an extension of the Team Orienteering Problem (TOP) [4], in which, given a set of visiting points with a score

¹ <https://www.unwto.org>.

and service time, the objective is to maximize the sum of the collected scores in a given number of routes, while guarantying a maximum travel time in each route. For solving the TOP, different exact and heuristic algorithms have been proposed [6]. In the TOPTW, each visiting point has to be visited within a given time window.

In this paper we present a Fuzzy Greedy Randomized Adaptive Search Procedure (GRASP) to solve the TOPTW. GRASP is a metaheuristic which consists of two phases. In the constructive phase, a solution is built by iteratively selecting an element of a restricted candidate list. In the improvement phase, the above solution is improved by applying a local search. These steps are repeated until a stopping rule is met. In our case, the restricted list of candidates is formed by the appropriate points of interest to be included in some route. Unlike the standard GRASP, Fuzzy GRASP uses two criteria to build this restricted candidate list. The first criterion evaluates the point of interest by attending to time increment of the route. The second criterion uses the score to evaluate the point of interest. On the other hand, we also propose a multi-agent simulation model aimed at handling the randomness of the optimization problem in practical scenarios. We consider that the scores associated to the points of interest are not deterministic, but can have low, moderate and high levels of variability instead. This means that some points have been clearly bad or good for most visitors, but others have not.

We use a classic set of instances to test our proposal. The obtained computational results show its good behavior. Fuzzy GRASP efficiently identifies high quality solutions in insignificant computational time.

2 Problem Description

The problem addressed in this paper is modelled as a multiple route-planning problem, specifically as a TOPTW. It consists of a set I of n points of interests (POIs), each with a given score, and a set K of m routes with the aim of visiting some of them in a given limited time. Each POI has associated a score or profit, a visit time, and a time windows. The starting and ending points are fixed, and represent tourists place of stay. However, not all POIs have to be visited. The number of routes corresponds to the number of stay days at destination. The objective function is to maximize the total collected score.

The POIs are identified by an index i , $i = 1, 2, \dots, n$. Indexes 1 and n represent the starting and ending POIs. Each route is represented by an index k , $k = 1, 2, \dots, m$. The score obtained in each POI is s_i , $i = 1, 2, \dots, n$. The time spent by the tourist when visiting POI i is r_i , $i = 1, 2, \dots, n$. The travel time from POI i to POI j is denoted by t_{ij} , $i, j = 0, 1, \dots, n$. T_{max}^k is the limit time for route k , $k = 1, \dots, m$, considering travel, visit and waiting times. m is the number of tourist stay days. The time windows are represented by e_i, l_i , the opening and closing times of a POI i , respectively; $i = 1, 2, \dots, n$.

The variable decisions of the mathematical model are: x_{ij}^k , y_i^k , and a_i^k , $i, j = 1, \dots, n$, $k = 1, 2, \dots, m$. x_{ij}^k is a binary variable that takes value 1 if route k goes

from POI i to j , and 0 otherwise. The binary variable y_i^k is equal to 1 if POI i is visited in route k and $y_j^k = 0$ otherwise. The arrival time at POI i in route k is contained in variable a_i^k .

On the other hand, the TOPTW can be formulated as a linear programming problem (LP) as follows:

Maximize:

$$\sum_{k=1}^m \sum_{i=1}^n s_i y_i^k \quad (1)$$

$$\sum_{k=1}^m \sum_{j=1}^n x_{0j}^k = \sum_{k=1}^m \sum_{i=1}^n x_{i0}^k = m \quad (2)$$

$$\sum_{k=1}^m y_i^k \leq 1 \quad i \in I \quad (3)$$

$$\sum_{j=0}^n x_{ji}^k = \sum_{j=0}^n x_{ij}^k = y_i^k \quad k \in K, i \in I \quad (4)$$

$$a_i^k + r_i + t_{ij} - a_j^k \leq M(1 - x_{ji}^k) \quad k \in K, i, j \in I^0 \quad (5)$$

$$a_i^k + t_{i0} \leq T_{max}^k \quad k \in K, i \in I \quad (6)$$

$$e_i \leq a_i^k \leq l_i \quad k \in K, i \in I \quad (7)$$

$$x_{ij}^k \in \{0, 1\} \quad k \in K, i, j \in I^0 \quad (8)$$

$$y_i^k \in \{0, 1\} \quad k \in K, i \in I \quad (9)$$

$$a_i^k \geq 0 \quad k \in K, i \in I \quad (10)$$

The objective function is represented in Eq. (1). The (2) constraints guarantee that routes start and end at point 0. The constraints (3) ensure that every location is visited at most once. The subtour elimination and flow conservation rule in constraints (4) and (5) establish the connectivity and time of each tour, where M is a large constant. The constraints (6) guarantee the maximum time for each tour. The time windows constraints are identified in (7). The condition variables are defined in (8), (9) and (10).

The values of x_{ij}^k , $k = 1, 2, \dots, m$, $i, j = 0, 1, \dots, n$ variables define the routes. With these variables, the selected POIs can be obtained by Eqs. (4). The combination of (5) and (7) leads to the arrival time at each POI j is iteratively given by

$$a_j^k = \max\{a_i^k, e_i\} + r_i + t_{ij} \quad (11)$$

for the previously point i in this day k ; i.e., such that $x_{ij}^k = 1$.

3 Literature Review

The Team Orienteering Problem with Time Windows (TOPTW) [18] has been extensively addressed in the literature over the last decade. For a comprehensive

survey on variants of the orienteering problem, we refer the reader to the paper by Gunawan et al. [6].

Due to the complexity of the TOPTW, most papers in the scientific literature tackle it by means of heuristic algorithms. Among the metaheuristic algorithms proposed for solving the TOPTW, there are implementations of Iterated Local Search, Local Search algorithm, Simulated Annealing, Variable Neighborhood Search, Large Neighborhood Search and Greedy Randomized Adaptive Search (GRASP).

A simple, fast and effective iterated local search that combines insert and shake steps to escape from local optima is presented in [18]. A variable neighborhood search algorithm for a multiple time windows extension has been addressed in [17]. A hybrid algorithm that combines GRASP with the evolutionary local search is designed in [10]. A Local Search algorithm is proposed in [9]. Two versions of a simulated annealing algorithm are developed in [11]. A variable neighborhood search procedure based on exploring granular instead of complete neighborhoods is presented in [9]. Granular exploration allows to obtain an efficient and effective algorithm. A Simulated Annealing with restart strategy is proposed in [12] for a multiple time windows extension. A Capacitated TOPTW, in which customers have associated also demands is presented in [2]. An iterated local search and a hybrid algorithm based on simulated annealing and iterated local search are proposed in [7]. A new variant of the TOPTW that includes mandatory visits is considered in [13]. In order to solve this problem, a multi-start simulated annealing is developed. A large neighborhood search algorithm is proposed in [15]. A multi-objective version of the TOPTW, in which the score has to be maximized and the time needed for the itinerary of the tourist is minimized, is presented in [8]. It is developed an iterated local search into adjustment iterated local search, in which the construction phase is performed heuristically. The uncertainty theory is used in [19] to consider uncertain travel times when solving the TOPTW. In this regard, simulation is a powerful tool to handle these scenarios. They have been successfully applied to a wide variety of environments. This is illustrated in [5] and [3], among others. Due to its versatility and capability to model specific features of the environment under analysis, simulation based on multiple agents [1] has become in one of the most relevant paradigms. For this reason, this is the approach considered in this paper.

In this work, we propose a Fuzzy GRASP for solving the TOPTW, in which the restricted candidate list used in the construction phase of GRASP is the fuzzy set of the high quality points of interest according to their travel time and score. Furthermore, we propose a multi-agent simulation model to deal with the randomness associated with the optimization problem in practical scenarios.

4 Solution Approach

4.1 Solver

A Fuzzy GRASP (Greedy Randomized Adaptive Search Procedure) is proposed in order to solve the proposed problem and obtain high-quality solutions in

reasonable time. The GRASP metaheuristic [14], in its standard version, consists of two phases; a first construction phase, and a second local search improvement phase.

The construction phase obtains a feasible solution by iteratively selecting at random a new location from a Restricted Candidate List (RCL) with size given by the parameter $RCLsize$. Finally, the neighborhood of the solution is explored until a local minimum is found in the second phase. The Algorithm 1 shows the GRASP phases cited above with $maxIterations$ as the maximum number of iterations of the GRASP procedure.

Algorithm 1. GRASP

```

1: function GRASP( $maxIterations, RCLsize$ )
2: readInput()
3: for  $i:=1$  to  $maxIterations$  do
4:    $solution = \text{GRASPConstructionPhase}(RCLsize)$ 
5:    $solution = \text{localSearch}(solution)$ 
6:    $\text{updateSolution}(solution, bestSolution)$ 
7: end for
8: return  $bestSolution$ 
9: end GRASP

```

In Algorithm 2 the construction phase of GRASP is shown. The initial solution has $m = |K|$ empty routes. The procedure starts with the initial solution, and subsequently the construction mechanism adds step-by-step a new POI to the current partial solution guaranteeing the solution feasibility.

The mechanism inserts each new no visited POI i in the best possible position guaranteeing the feasibility of the partial route. The several feasible positions in which insert a new POI i is defined as the (i, j, k) triplet with $y_j^k = 1$. For the partial route k , the mentioned insertion consists in inserting the POI i after j . The variables y_i^k , x_{ji}^k and x_{ih}^k are set to 1 in order to get the new solution. Then for the only index h such that x_{jh}^k is 1; this variable x_{jh}^k is also set to 0.

The variables a_i^k for this route k have to be computed using (11) and feasibility tested by the corresponding time limit constraints (6) and time window constraints (7). A greedy travel time function f computes the travel time increase in the route produced by an insertion represented by the triplet defined previously. The f function is defined as $f(i, j, k) = t_{ji}^k + t_{ih}^k - t_{jh}^k$ for the above index h .

A Restricted Candidate List (RCL) with candidate POIs to be visited in the solution is constructed using the greedy function f . The standard version of GRASP designs the RCL introducing $RCLsize$ feasible insertion triplets (i, j, k) with the best quality values for function f . In this paper a fuzzy version of the GRASP is proposed.

The proposed fuzzy mechanism considers the fuzzy set of the high quality POIs of the RCL based on their score. $\mu(\cdot)$ is the membership function of the

Algorithm 2. Construction Phase of the GRASP

```

1: function GRASPConstructionPhase( $RCLsize$ )
2: Initialize the partialSolution with  $m$  empty routes
3: while (it is possible to visit new locations) do
4:    $CL = \emptyset$ 
5:   for all location  $i \in I$  do
6:     Find the best feasible triplet  $(i, j, k)$  to insert this new location  $i$  in
       partialSolution according to greedy time function  $f(i, j, k)$ 
7:     Add the feasible triplets  $(i, j, k)$  to the Candidate List  $CL$ 
8:   end for
9:   Create the Restricted Candidate List,  $RCL$ , with the best  $RCLsize$  triplets
        $(i, j, k)$  from  $CL$  according to  $f$ 
10:  Select a random triplet  $(i, j, k)$  from  $RCL$ 
11:  Update the variables of route  $k$  by inserting the location  $i$  at position  $j$ 
12: end while
13: return partialSolution
14: end GRASPConstructionPhase

```

set stated by $\mu(i) = s_i/s_{max}$ with s_i the score related with POI i and s_{max} the highest score in RCL. RCL^* is the set with the best locations in RCL created by the α -cut of RCL related with $\mu(\cdot)$, with $\alpha \in [0, 1]$.

$$RCL^* = \{(i, j, k) \in RCL : \mu(s_i) \geq \alpha\}$$

The new POI inserted in the solution is randomly selected from RCL^* . The decision maker is responsible of fixing the α parameter. The choice of this parameter influences the quality of the POI that will be inserted into the solution. The Algorithm 3 shows the fuzzy GRASP construction phase.

Basically, the selection of the best insertion position of POIs in a tour composes the candidate list. The greedy function f evaluates the increase in time due to the POI insertion in partial route. The value of f determine the position of candidate in the list. The best position to insert a candidate POI for all routes is located by the procedure. The goal of the best position is minimize the total time after the insertion of POI. Finally the list is ordered ascending by the increase in time such that lowest time increase candidates are placed at candidate list top. The candidate list size is settled in $RCLsize$ due to build the RCL. Subsequently, the α -cut, that recognizes the best score POIs, conforms RCL^* . The decision maker fix the α value.

The proposed GRASP mix the minimization of travel times in the RCL composition and maximization score in the RCL^* composition. Therefore the RCL^* contains the best POIs that combines the two criteria mentioned. The POIs that will be insert in current partial solution are those with low travel time increase and high score. Later, in RCL^* a POI is randomly selected and inserted into partial solution. The inserted POI is banned for future candidate lists. Finally, the feasible solution is obtained and construction phase ends.

The next phase is an improvement phase that uses a local search for this purpose. Commonly, a local search procedure performances iteratively by replacing

Algorithm 3. Construction Phase of the Fuzzy GRASP

```

1: function GRASPFuzzyConstructionPhase( $RCLsize, \alpha$ )
2: Initialize the partialSolution with  $m$  empty routes
3: while (it is possible to visit new locations) do
4:    $CL = \emptyset$ 
5:   for all location  $i \in I$  do
6:     Find the best position  $j$  to insert  $i$  in a partial route  $k$  of the partialSolution
       according to greedy time function  $f(i, j, k)$ 
7:     Add the feasible triplets  $(i, j, k)$  to the Candidate List  $CL$ 
8:   end for
9:   Create the Restricted Candidate List,  $RCL$ , with the top  $RCLsize$  triplets
        $(i, j, k)$  from  $CL$ 
10:  Get  $RCL^* = \{(i, j, k) \in RCL : \mu(s_i) \geq \alpha\}$ 
11:  Select a random triplet  $(i, j, k)$  from  $RCL^*$ 
12:  Update the variables of route  $k$  by inserting the location  $i$  at position  $j$ 
13: end while
14: return partialSolution
15: end GRASPFuzzyConstructionPhase

```

the current solution with a improved solution achieved in the neighborhood. The stop criteria of local search is not to find a better solution in the neighborhood. The Algorithm 4 presents a standard local search procedure. The proposed local search applies exchange movements between POIs in order to minimize the total travel time. The POIs can be selected in the same or different routes. A best-improving strategy is used, so that the search explores all neighborhoods in order to replace the current solution by the best solution found. Once the total travel time of the solution has been improved, the procedure tries to insert a new POI in the solution with the aim of maximize the solution score.

Algorithm 4. FUZZY GRASP improvement phase

```

1: function localSearch(solution)
2:  $s = solution$ 
3: repeat
4:   Find the best neighbor  $n$  of current solution  $s$  according the total time
5:   if ( $TotalTime(n) \leq TotalTime(s)$ ) then
6:      $s = n$ 
7:   end if
8: until  $TotalTime(n) \geq TotalTime(s)$  for all neighbor  $n$ 
9: return  $s$ 
10: end localSearch

```

Overall, the aim of the proposed solver is to maximize the total solution score. The procedure is an two-phases iterated algorithm that progress until the stop criteria is met. The introduced fuzzy mechanism in the standard GRASP

allows to consider two criteria in construction of the RCL list; the minimization of travel time and the maximization of score.

4.2 Simulation Model

In this paper, we also propose a simulation model aimed at handling randomness associated with the optimization problem in practical scenarios. This randomness is derived from the potential changes in the preferences of the visitors, inadequate treatment of the staff in the points or unadvertised changes in the time windows, among others. Thus, the scores of the points of interest are considered as random variables instead of constant values, as done in previous works found in the scientific literature so far.

We design a multi-agent simulation approach that uses asynchronous time and receives a feasible solution of the problem. This solution is provided by the solver previously described. It also receives the random variables that defines the behaviour of the scores. The model is composed of a hierarchy of agents, The base agent is governed by the state machine shown in Fig. 1 and is aimed at simulating a route of the solution. The nodes in the state machine represent states of the visitor during its route, whereas the arcs represent transitions between pairs of consecutive states. This way, a transition is applied when the triggering event occurs. As can be checked, the visitor moves to the next point of interest and waits until it is open. This is visited once the point is open. Then, next point is checked or the visitor goes to its end point. It is worth mentioning that the score obtained when visiting a point of interest is derived from sampling the random variable that defines it. At the same time, there is a manager agent which is responsible for assign routes to the base agents and managing the performance metrics (*i.e.*, number of points visited, time in each point, etc.). Thus, the model is composed of one manager agent and m base agents.

Fig. 1. State machine of an agent aimed at simulating a route

5 Computational Experiments

5.1 Solver Computational Experiments

This section describes the results from the first computational experiment carried out in our study. The aim of this experiment is to evaluate the accuracy of the Fuzzy GRASP metaheuristic proposed to solve the TOPTW and get high-quality solutions.

The benchmark suite tackled along this section is based on the classical set of instances introduced by [16] for Vehicle Routing Problem with Time Windows. Vansteenwegen et al. [18] provide a set of locations with a given score which can be visited. A time windows and service time have been defined for each location. The limit number of routes match with the number of vehicles of the original Solomon data set. The number of locations used is equal to 100 and the limit time per route varies according to the specific instance. This benchmark considers travel times equal to euclidean distances. Regarding the geographical data, there are three kinds of instances: instances with clustered customers (C), instances with random customers (R) and a mixture of random and clustered customers (RC).

The instances data set used are described in Table 1 in more detail.

The parameter of the proposed Fuzzy GRASP are:

1. The α value of α -cut of $\mu(s_j)$
2. The size $sRCL$ of the GRASP restricted candidate list

The parameter values of the proposed solver assess during the computational experiments are shown in Table 2.

The GRASP procedure was run 1000 times for each instance and parameter combination used in computational experiments. The results are presented in Table 3 and show the comparison between the best solutions obtained by standard GRASP and Fuzzy GRASP, and the optimal solution. The score of the optimal solutions is the sum of scores for all locations. Column one and two give the name of the instance and the score of the optimal solution, respectively. The third and fourth columns show the best solutions scores of standard and Fuzzy GRASP, respectively. The gap between the best solutions of the standard and Fuzzy GRASP and the optimal solution are given by the fifth and sixth columns, respectively. Finally, the last column shows the execution time of Fuzzy GRASP. The computational experiments confirm that Fuzzy GRASP is highly efficient when solving the instances under analysis.

5.2 Simulation Analysis

We have carried out several experiments dedicated to check the suitability of the simulation model proposed in Sect. 4.2. In this case, the score associated with the points of interest is modelled as a random variable, S , that does not follow any arbitrary non-negative probability distribution. In particular, a Log-Normal

Table 1. Instances used in experimentation

Instance	Size	Max. routes	Max. time
<i>c101</i>	100	10	1236
<i>c102</i>	100	10	1236
<i>c103</i>	100	10	1236
<i>c104</i>	100	10	1236
<i>c105</i>	100	10	1236
<i>c106</i>	100	10	1236
<i>c107</i>	100	10	1236
<i>c108</i>	100	10	1236
<i>c109</i>	100	10	1236
<i>r101</i>	100	19	230
<i>r102</i>	100	17	230
<i>r103</i>	100	13	230
<i>r104</i>	100	9	230
<i>r105</i>	100	14	230
<i>r106</i>	100	12	230
<i>r107</i>	100	10	230
<i>r108</i>	100	9	230
<i>r109</i>	100	11	230
<i>r110</i>	100	10	230
<i>r111</i>	100	10	230
<i>r112</i>	100	9	230
<i>rc101</i>	100	14	240
<i>rc102</i>	100	12	240
<i>rc103</i>	100	11	240
<i>rc104</i>	100	10	240
<i>rc105</i>	100	13	240
<i>rc106</i>	100	11	240
<i>rc107</i>	100	11	240
<i>rc108</i>	100	10	240

Table 2. Parameters used by Fuzzy GRASP

Parameter	Values
α value of α -cut	$\alpha \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$
Size of RCL	$sRCL \in \{3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20\}$

Table 3. Comparative results

Instance	Opt.	GRASP	Fuzzy GRASP	Gap (%) Fuzzy GRASP	Gap (%) GRASP	Fuzzy GRASP Execution time (ms)
<i>c101</i>	1810	1600	1700	6.1	11.6	7.07
<i>c102</i>	1810	1730	1780	1.7	4.4	4.17
<i>c103</i>	1810	1800	1810	0.0	0.6	5.32
<i>c104</i>	1810	1810	1810	0.0	0.0	71.45
<i>c105</i>	1810	1710	1770	2.8	5.5	4.66
<i>c106</i>	1810	1710	1770	2.8	5.5	3.93
<i>c107</i>	1810	1800	1810	0.0	0.6	3.84
<i>c108</i>	1810	1810	1810	0.0	0.0	4.18
<i>c109</i>	1810	1810	1810	0.0	0.0	4.94
<i>r101</i>	1458	1429	1447	0.7	2.0	4.01
<i>r102</i>	1458	1441	1444	0.8	1.2	4.95
<i>r103</i>	1458	1421	1443	1.0	2.5	5.45
<i>r104</i>	1458	1352	1403	3.8	7.3	5.32
<i>r105</i>	1458	1415	1435	1.6	2.9	3.93
<i>r106</i>	1458	1379	1418	2.6	5.4	4.77
<i>r107</i>	1458	1366	1405	3.8	6.3	5.51
<i>r108</i>	1458	1391	1418	2.0	4.6	5.55
<i>r109</i>	1458	1386	1403	3.3	4.9	4.38
<i>r110</i>	1458	1356	1401	3.5	7.0	5.29
<i>r111</i>	1458	1377	1421	3.6	5.6	5.87
<i>r112</i>	1458	1372	1414	3.6	5.9	5.58
<i>rc101</i>	1724	1658	1640	4.5	3.8	3.88
<i>rc102</i>	1724	1566	1677	4.4	9.2	4.48
<i>rc103</i>	1724	1628	1688	2.6	5.6	4.86
<i>rc104</i>	1724	1693	1696	1.8	1.8	5.70
<i>rc105</i>	1724	1636	1665	2.9	5.1	4.50
<i>rc106</i>	1724	1595	1631	5.1	7.5	4.02
<i>rc107</i>	1724	1652	1688	2.1	4.2	4.86
<i>rc108</i>	1724	1633	1674	3.0	5.3	5.77

distribution with $Var[S] = k \cdot E[S]$. $k = \{0.25, 1, 2\}$ to represent scenarios with low, moderate, and high levels of variability.

Table 4 shows the average computational results obtained by the proposed model when simulating the behaviour of 10 solutions selected at random. In this case, the score, travelled distance, total time, minimum time in the routes, maximum time in the routes, and slack time used to satisfy the time windows are shown when changing the level of variability in the scenarios. Due to the

fact that the variability impacts only on the score of the points of interest, the remaining metrics are not altered in the different scenarios and their values are only provided for further analysis. In this case, the results indicate that the score obtained when visiting the involved points of interest in the routes increases when the variability of the scenarios is higher.

Table 4. Computational results obtained by the simulation model

	$k = 0.25$	$k = 1$	$k = 2$
Score	1302.606	1304.991	1308.624
Distance	1100.106	1100.106	1100.106
Time	1970.106	1970.106	1970.106
Min. time	201.583	201.583	201.583
Max. time	226.534	226.534	226.534
Slack	6388.068	6388.068	6388.068

6 Conclusions and Future Research

This work proposes a Fuzzy GRASP metaheuristic and a multi-agent simulation model to solve the Team Orienteering Problem with Time Windows, in which the scores associated to the points of interest are not deterministic. In the first place, we develop a Fuzzy GRASP to solve the optimization problem with deterministic scores to obtain a set of high-quality solutions. Then, these solutions are used by the designed multi-agent simulation model to consider low, moderate and high levels of variability in the scores.

The computational experiments carried out in this work corroborate that the GRASP metaheuristic can reach a set of high-quality solutions in reasonable computational time. Moreover, the simulation model is a tool able to perform experiments about the impact of changes in the score of the points of interest on the quality of the solutions in realistic scenarios. However, there are still several promising lines for further research. In particular, uncertainty can be associated with travelling times between pairs of points of interest, and therefore the optimization and simulation approaches must be modified appropriately to handle this. At the same time, it is highly interesting to develop recommendation systems to propose visiting routes to the tourists according to their individual features.

Acknowledgment. Airam Expósito-Márquez would like to thank the Canary Government for the financial support he receives through his post-graduate grant.

References

1. Abar, S., Theodoropoulos, G.K., Lemarinier, P., O'Hare, G.M.P.: Agent based modelling and simulation tools: a review of the state-of-art software. *Comput. Sci. Rev.* **24**, 13–33 (2017)

2. Aghezzaf, B., El Fahim, H.: Solving the capacitated team orienteering problem with time windows through variable neighborhood search. *Int. Rev. Comput. Softw.* **10**(11), 1134–1142 (2015)
3. Amaran, S., Sahinidis, N.V., Sharda, B., Bury, S.J.: Simulation optimization: a review of algorithms and applications. *Ann. Oper. Res.* **240**(1), 351–380 (2015). <https://doi.org/10.1007/s10479-015-2019-x>
4. Chao, I.M., Golden, B.L., Wasil, E.A.: The team orienteering problem. *Eur. J. Oper. Res.* **88**(3), 464–474 (1996)
5. Günal, M.M., Pidd, M.: Discrete event simulation for performance modelling in health care: a review of the literature. *J. Simul.* **4**(1), 42–51 (2010)
6. Gunawan, A., Lau, H.C., Vansteenwegen, P.: Orienteering problem: a survey of recent variants, solution approaches and applications. *Eur. J. Oper. Res.* **255**(2), 315–332 (2016)
7. Gunawan, A., Lau, H.C., Vansteenwegen, P., Lu, K.: Well-tuned algorithms for the team orienteering problem with time windows. *J. Oper. Res. Soc.* **68**(8), 861–876 (2017)
8. Hapsari, I., Surjandari, I., Komarudin, K.: Solving multi-objective team orienteering problem with time windows using adjustment iterated local search. *J. Ind. Eng. Int.* **15**(4), 679–693 (2019). <https://doi.org/10.1007/s40092-019-0315-9>
9. Labadie, N., Mansini, R., Melechovsky, J., Wolfler Calvo, R.: The team orienteering problem with time windows: an LP-based granular variable neighborhood search. *Eur. J. Oper. Res.* **220**(1), 15–27 (2012)
10. Labadie, N., Mansini, J., Melechovsky, J., Wolfler Calvo, R.: Hybridized evolutionary local search algorithm for the team orienteering problem with time windows. *J. Heuristics* **17**(6), 729–753 (2011)
11. Lin, S.-W., Yu, V.F.: A simulated annealing heuristic for the team orienteering problem with time windows. *Eur. J. Oper. Res.* **217**(1), 94–107 (2012)
12. Lin, S.-W., Yu, V.F.: A simulated annealing heuristic for the multiconstraint team orienteering problem with multiple time windows. *Appl. Soft Comput. J.* **37**, 632–642 (2015)
13. Lin, S.-W., Yu, V.F.: Solving the team orienteering problem with time windows and mandatory visits by multi-start simulated annealing. *Comput. Ind. Eng.* **114**, 195–205 (2017)
14. Resende, M.G.C., Ribeiro, C.C.: *Optimization by GRASP: Greedy Randomized Adaptive Search Procedures*. Springer, New York (2016)
15. Schmid, V., Ehmke, J.F.: An effective large neighborhood search for the team orienteering problem with time windows. In: Bektaş, T., Coniglio, S., Martinez-Sykora, A., Voß, S., et al. (eds.) *ICCL 2017. LNCS*, vol. 10572, pp. 3–18. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68496-3_1
16. Solomon, M.M.: Algorithms for the vehicle routing and scheduling problems with time window constraints. *Oper. Res.* **35**(2), 254–265 (1987)
17. Tricoire, F., Romauch, M., Doerner, K.F., Hartl, R.F.: Heuristics for the multi-period orienteering problem with multiple time windows. *Comput. Oper. Res.* **37**(2), 351–367 (2010)
18. Vansteenwegen, P., Souffriau, W., Vanden Berghe, G., Van Oudheusden, D.: Iterated local search for the team orienteering problem with time windows. *Comput. Oper. Res.* **36**, 3281–290 (2009)
19. Wang, J., Guo, J., Chen, J., Tian, S., Gu, T.: Uncertain team orienteering problem with time windows based on uncertainty theory. *IEEE Access* **7**, 63403–63414 (2019)