A Multiobjective Evolutionary Algorithm for Personalized Tours in Street Networks

Ivanoe De Falco, Umberto Scafuri, and Ernesto Tarantino(✉)

ICAR-CNR, Via P. Castellino 111, 80131 Naples, Italy
{ivanoe.defalco,umberto.scafuri,ernesto.tarantino}@na.icar.cnr.it

Abstract. The paper presents a novel optimizer to plan multiple–day walking itineraries, tailored to tourists' personal interests, in a street network modeled as a graph. The tour is automatically designed by maximizing the number of the Points of Interest (*POI*s) to visit as a function of both tourists' preferences and requirements, and constraints such as opening hours, visiting times and accessibility of the *POI*s, and weather forecasting. Since this itineray planning is classified as an NP–complete combinatorial optimization problem, a multiobjective evolutionary optimizer is here proposed. Such an optimizer is proven to be effective in designing personalized multiple–day tourist routes.

Keywords: Multiple–day orienteering problem with time windows · Personalized tour · Tourism · Multiobjective evolutionary algorithm

1 Introduction

A street network, namely a system of interconnecting lines and points that represent a system of streets or roads for a given area, provides the foundation for network analysis; for example, finding the best route. One of the most remarkable Operational Research (OR) problems related to a street network is that faced by a tourist who has to visit an area of a city in a limited amount of time. In this case the aim is that of selecting the most interesting places to visit on the basis of their personal interests, and of creating the shortest suitable route connecting them. In OR such a problem is usually modeled as a graph.

To plan a feasible tour the tourist has to collect information from different sources (websites, magazines or guidebooks) about the different Points of Interest (*POI*s), make a selection among these *POI*s and plan an itinerary connecting the selected points, considering the available time, the opening hours and visiting times of the different *POI*s, the weather forecasting and other different kinds of constraints [1]. Given the complexity of the problem, the building of the tour by combining all the preferences and constraints presents considerable difficulties.

In the last years several Personalized Electronic Tourist Guides (PETGs) relying on mobile computing have been developed to better perform the task fulfilled by the local tourist organizations.

A.M. Mora and G. Squillero (Eds.): EvoApplications 2015, LNCS 9028, pp. 115–127, 2015.
DOI: 10.1007/978-3-319-16549-3_10

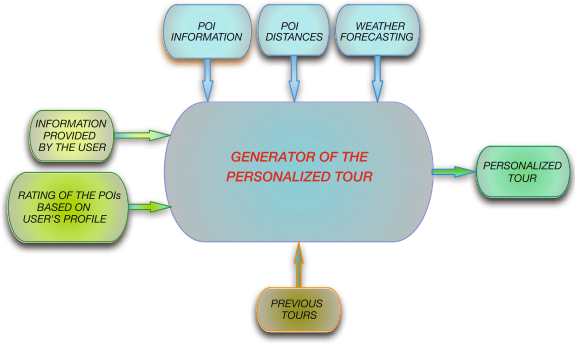

Fig. 1. The architecture of the PETG.

Within this paper the design of a personalized multiple–day walking tour in old city centres, which takes into account a set of *POI*s with a score, a set of waiting and visiting times, and a set of daily opening hours of these *POI*s, is investigated. Combined with the tourist's trip constraints and environmental contexts, these sets allow defining a problem which is an extension of the well–known Team Orienteering Problem with Time Windows (TOPTW) where the number of team members is replaced by the number of days available for the tourist to stay, the start and the arrival positions are not necessarily coincident in each day of the tour, and can change from day to day [2,3].

Since TOPTW is a highly–constrained combinatorial optimization problem [4] that cannot be solved in polynomial time [5,6], a multiobjective evolutionary optimizer is proposed to find in a reasonable computational time near–optimal solutions to the planning of a multiple–day route. Such an optimizer is innovative as it considers a higher number of objectives and of features with respect to some recent tools for building walking–only itineraries [2,7,8].

The paper is organized as follows: Sect. 2 presents a description of the optimizer used to deal with the problem under investigation. Section 3 describes the multiobjective evolutionary algorithm employed to find multiple–day tours. In Sect. 4 the findings of the proposed approach are shown and discussed. Finally, the last section contains conclusion remarks and future works.

2 The Generator of the Personalized Tour

The core of the hypothesized PETG, schematized as in Fig. 1, is our optimizer, i.e., the generator of the personalized tour. Such an optimizer is supposed to interact with several input modules to gather information about the tourist's interests and requirements, and environmental constraints, and with an output module to endow the tourist with the planned multiple–day itinerary.

Our optimizer needs the following input information:

– *Information provided by the user:* the user identifier, the average moving speed (young, old, family with children, etc.), the wheelchair or disabled access, the

day of tour beginning, the number of days available for the tour, number and identification codes of the *POI*s that the user requests to be included in the visit. Moreover, for each day the optimizer needs the start/arrival positions and the start and finish times of programmed pauses.

The start and arrival positions, represented by the GPS coordinates, i.e., longitude and latitude, are selected by the user either from the map, or from the list of tourist attractions or from a set of locations ('access doors') corresponding to the public transportation stations/stops close to the area.

- *Rating of the POI*s. The evolutionary optimizer hypothesizes a recommender system which assigns the 'score' or 'rating' to each *POI* on the basis of the personal preferences of the tourist. The rating, which measures the *POI* attractiveness for that specific user, is supposed to be estimated by a profiling phase previously effected either through the data extracted from the social media or from a questionnaire which the same user is invited to fill during the preliminary connection phase.
- *POI information.* The information related to each *POI*, to the distance and routes between *POI*s, and to the context is stored in suitable databases. Specifically, the data supposed to be known for each *POI* are the *POI* identifier, the name of the *POI*, the position in terms of GPS coordinates, the average waiting and visiting times, the opening hours for each day of the tour, the accessibility for disabled people, a flag as outdoor or indoor site.
- *POI distances.* The *POI* distances are represented by a real–valued triangular matrix with a dimension equal to the number of *POI*s. The value of each cell (i, j) of the matrix indicates the distance between the *POI*s i and j measured on the basis of the shortest route connecting them. This means that it is not the Euclidean or Manhattan distance evaluated on the basis of the GPS coordinates of the two *POI*s, rather it accounts for the actual structure of the streets in the area containing the *POI*s.
- *Weather table.* A weather table is used to propose, as far as possible, itineraries which take into account the weather conditions. The hourly forecasting is extremely complex, so a granularity equal to three hours (one hour before or after the time interval indicated) is considered As a consequence, for each day of the tour this table contains the weather forecasting subdivided in a number of items equal to that of the three–hour time slots in which the tour falls.
- *Previous tours.* Particularly useful is the data file which keeps track of the previous tours effected by the tourists in case of either a multiple–day tour or a tour requested by a user who previously visited the same area. In both cases the knowledge of previous tours avoids proposing an itinerary including already visited *POI*s, unless the user explicitly requires to visit again some such *POI*s. Each record of this file is related to a tour made by a single user and reports the user identifier, the day(s) of the tour, the number and the identifiers of the visited *POI*s.

2.1 Output of the Optimizer

The output of our evolutionary optimizer is the personalized tour automatically saved in a .csv file, named *tour.csv*, that is processed by an API devised to

visualize the corresponding optimized tour on a map directly on the user's mobile device. The data stored in *tour.csv* are:

- The start time.
- For each *POI*: the identifier of the *POI*, the walking time between the current position and the *POI*, the arrival time at the *POI*, the waiting time and the finish time of the visit.
- For each pause: the start and finish times.
- The walking time to the final destination and the related arrival time.

Moreover a summary of the more relevant information related to the proposed tour is reported in terms of the total number of visited *POI*s specifically requested by the user, the total number of visited *POI*s, the total time employed for the tour including waiting, visiting and transfer times, the total covered distance and the score of the complete tour measured as the average of the score of the single *POI*s included in the tour.

2.2 Problem Statement

To generate personalized multiple–day itineraries respecting predefined time windows, the problem is modeled as a graph consisting of a set of locations and a set of the paths between each pair of these locations. A non–negative score representing the rating of the tourist for that location and a set of time windows is associated with each vertex. A positive weight corresponding to the walking time is associated with each path.

The Objectives. The goal of our optimizer in building a personalized multiple-day itinerary is to optimize five contrasting objectives, namely,

- maximize the score of the proposed *POI*s;
- visit as many *POI*s as possible among those explicitly requested by the user;
- visit as many *POI*s as possible among the recommended ones in addition to those specifically included by the user;
- complete the tour within the time limit fixed by the user respecting the following commitments: opening and closing hours of the *POI*s, average visit duration and pause times (rest, lunch, shopping, ...) required by the user;
- minimize the covered distance;
- respect other constraints such as: accessibility to the *POI*s for people with disabilities and usability of the outdoor *POI*s in case of rain, high wind, etc.

Differently from the opening times of the *POI*s, which are hard constraints, the pause times requested by the user are handled as being soft temporal constraints, whose satisfaction can happen in suitable neighborhoods of the times requested by the user if this flexibility allows better encountering her/his preferences in terms of the accessibility and of the number of visited *POI*s. A detailed description of all the objectives is reported in the following items.

– *Rating of the tour by the tourist side.* As each generic *POI* i is characterized by a real value $a(i)$ which represents its attractiveness for a specific user, the rating Φ_1 of the complete route *tour* is equal to:

$$\Phi_1(tour) = \frac{1}{POI\_in\_tour} \sum_{i=1}^{POI\_in\_tour} a(i)$$

that is the average of the scores of the single *POI*s. POI_in_tour represents the number of *POI*s comprised in the proposed tour. This objective is to be maximized.

– *Number of the POIs explicitly required by the user and actually present in the proposed tour.* Denoting with POI_req the number of the *POI*s that the user requires to be comprised in the tour, an ideal tour will be one which includes all the POI_req. Nevertheless, due to the constraints, the tours proposed by the optimizer could include an actual number of required *POI*s, named POI_act, lower or equal to POI_req. In formula:

$$\Phi_2(tour) = POI\_req - POI\_act$$

This objective is to be minimized.

– *Number of total POIs included in the tour.* The user is naturally interested in visiting as many *POI*s as possible in the time available for her/his tour. Therefore the optimizer, in addition to the POI_act, has to propose a tour able at the same time to assure the greatest rating and to contain the highest number of *POI*s encountering at best user's preferences. So we can define the following objective:

$$\Phi_3(tour) = POI\_in\_tour$$

This objective is to be maximized and must respect all the constraints in terms of time, tour length, presence of all the desired *POI*s and so on.

– *Total duration of the tour.* It is supposed that the user desires to spend as much time as possible for the visit of the attractions, saving time for pauses (rest, lunch, shopping). To evaluate a generic tour the proposed optimizer takes into account the following variables:

- $t\_tran(i, j)$ indicates the time needed to transfer from a *POI* i to j where j represents the *POI* successive to i in the tour under examination (in the case of the first *POI* it represents the time to reach the first *POI* from the start point chosen by the user);
- $t\_wait(j)$ represents the waiting time to access *POI* j, due to queues, or other constraints;
- $t\_vis(j)$ denotes the visiting time of the *POI* j;
- $t\_tot\_vis(j)$ accounts for the total of the three above–mentioned times;
- t_arr represents the time to reach the finish point chosen by the user from the last *POI* in the proposed tour;
- $t\_pau(k)$ represents the duration of the generic pause k required by the user (let Tot_pau be the total number of these pauses);

- *max_time_tour* denotes the time limit declared by the user to complete the tour.

This premised, the total time of the tour Φ_4 is given by:

$$\Phi_4(tour) = t\_arr + \sum_{j=1}^{POI\_in\_tour} t\_tot\_vis(j) + \sum_{k=1}^{Tot\_pau} t\_pau(k)$$

This quantity is to be maximized while respecting the constraint $\Phi_4(tour) \leqslant max\_time\_tour$.

– *Total length of the tour.* Another important side to consider is the total covered distance of the tour which must be not excessively long in particular for some categories of potential users, as for example elderly people, families with children, or disabled people. It is to note that the classical Traveling Salesman Problem (TSP) which considers only the distances in order to find an itinerary cannot be used in this case. In fact, for a PETG, in addition to the spatial coordinates, also the temporal side is to be taken into account, as for example the opening and closing hours of the *POI*s. However, it is important also to account for a spatial objective with the aim to minimize the length of the tour proposed. Such a length is evaluated as:

$$\Phi_5(tour) = d\_init + \sum_{i=1}^{POI\_in\_tour} d(i, i+1) + d\_fin$$

where $d(i, i+1)$ is the distance between a generic *POI* i and the next *POI* $(i+1)$ in the considered tour, *d_init* is the distance between the start position declared by the user and the first *POI* in the tour under examination, d_fin is the distance between the last *POI* in the tour and the finish point declared by the user.

– *Other constraints.* The generated solutions must respect these further constraints: in case of rain or high wind outdoor *POI*s are to be excluded from the itineraries, unless explicitly requested by the user, and in case of tours required by disabled people the *POI*s with limited access are to be discarded.

3 The Multiobjective Evolutionary Optimizer

To find near–optimal personalized multiple–day tours a multiobjective evolutionary algorithm, able to satisfy all the contrasting objectives reported in Sect 2.2, is proposed.

3.1 Multiobjective Optimization Notions

To deal with the five contrasting objectives mentioned above, a multiobjective evolutionary algorithm based on the concept of the so–called *Pareto optimal set* is designed and implemented. To make this paper self–contained we report

some fundamentals of the multiobjective optimization. This technique relies on the notion of *dominance*: as an example, for a problem with two objectives to optimize, each represented by a fitness function Φ_i, representing the quality of the solution with respect to the objective, a solution $\boldsymbol{X}_1$ is said to dominate in the Pareto sense (P–dominate) another solution $\boldsymbol{X}_2$ if and only if, for any objective, the related $\Phi_i(\boldsymbol{X}_1)$ is not worse than $\Phi_i(\boldsymbol{X}_2)$ and is better for at least one of the objectives. A solution $\boldsymbol{S}^*$ is said Pareto–optimal if it belongs to Pareto optimal set. The Pareto–optimal set and the Pareto–optimal front are the sets of Pareto–optimal solutions in design variables and objective function domains, respectively. By doing so, at each generation a set of "optimal" solutions, namely the current Pareto front, emerges where none of them can be considered to be better than any other in the same set. As the number of generations increases the current Pareto front will shift, and will hopefully approach the Pareto–optimal front. Usually, at the end of the execution of the evolutionary algorithm the final Pareto front will be proposed to the user that, among the solutions contained therein, will choose the one which best suits his needs.

3.2 The Methodology

An evolutionary algorithm (EA) is a population–based optimization algorithm which uses mechanisms inspired by biological evolution to find approximate solutions for complex problems [9,10]. As in each EA, once initialized, a population of $NPOP$ solutions, called individuals, is let free to evolve from a generation g to the next one by means of the operators of selection, recombination, and replacement. The components of the EA within this paper are standard with the exception of the mutation operator that is specific for the faced problem, and of the selection which is typical of this multiobjective version. In our case the basic steps of the algorithm can be described as follows:

- Initialization: an initial population of individuals is randomly generated.
- Selection: the choice of the individuals which undergo recombination takes place by a random uniform selection among the non dominated solutions. At the end of each generation, it is important to sort the solutions to individuate those belonging to the Pareto front (non–dominated solutions).
- Recombination: given two elements selected among the non–dominated solutions, we apply as evolutionary operators the uniform crossover [11,12] and the mutation to recombine the solutions. As mutation the classical exchange [13,14] and 2–opt variants [13,15] are employed. As a result one offspring is obtained.
- Evaluation: being the problem structured as having the five objectives described in Sect. 2.2, the fitness of each solution will be evaluated on each of those five optimization criteria. Considered that this implies the resolution of a multiobjective problem we will make reference to the notion of *dominance* reported in the above Sect. 3.1.
- Replacement: the i–th offspring obtained by the recombination is compared with the i–th individual in the current population and the already present individual is replaced only if it is dominated by the new generated one.

The four last steps are repeated for each individual so that a new population is obtained. This procedure is repeated for a maximum number of generations g_{max}, with the aim to individuate the best Pareto front solution to be presented as the output of the optimizer. We consider this solution as the one with the lowest distance from the theoretically optimal solution, i.e., the one which perfectly satisfies all the five objectives. This "best" solution is generally located in the intermediate region of the front and is the one which yields a better balance in satisfying all the objectives.

Encoding. Each individual in the population represents a potential multiple–day tour and is encoded by a vector of integer values with dimension equal to the number of *POI*s. This dimension is denoted with NPOI. Each integer denotes a *POI* and is present only once in each solution. For example a solution with NPOI=10 is shown in Fig. 2.

Fig. 2. Example of a tour encoding.

This solution, starting from the position chosen by the user (not explicitly contained in such an encoding), proposes to reach the *POI* in the first cell at the left side of the vector (9 in the figure) and then proceed forward visiting the *POI* in the second cell (1 in the figure) and so on. Differently from the TSP in which all the points are visited in the order indicated by the solution, in our PETG it is highly probable that not all the *POI*s are effectively visited. This can happen for several reasons:

- remaining available time: the tour must terminate when the residual available time is only sufficient to reach the final point from the *POI* in which the user currently is;
- closing: the tour can lead the user to a *POI* during its closing time;
- previous tours: the *POI* has already been visited in a previous tour and thus it will be not considered if not expressly required again;
- multiple–day tour: if the user has required a tour programmed in several days, the *POI* will be discarded if already included in the tour of one of the previous days.

If a *POI* is actually visited in the tour, a flag will be set for it. This allows the management of a multiple–day tour. In fact, for the second day the examination of the solution restarts from the leftmost position in the vector as for the first day and all the *POI*s already visited in the first day are now skipped. Analogously, for each further day, the vector which represents the solution will be examined again, always starting from its leftmost position.

4 Experiments

The algorithm has been coded in Java language and all the tests have been performed on a MacBookPro4.1 Intel Core Duo 2.4 GHz, 2 GB RAM. After a preliminary tuning phase, the population size $NPOP$ has been set equal to 500, while the number of generations g_{max} has been set equal to 100. The crossover probability CR has been set equal to 0.4. For the mutation, the exchange probability EM has been fixed to 0.8 while the parameter FV for the 2-opt mutation has been set equal to 1.0. Since evolutionary algorithms are nondeterministic, to individuate the best tour for any given problem 10 runs are performed. The execution time for all the 10 runs is about 13 s.

The algorithm is able to provide multiple–day walking tours in any area once the needed information are made available. Within this paper its ability has been tested for the area of the old city centre of Naples, Italy, by considering 20 *POI*s. These *POI*s are listed in Table 1 together with some of the relevant information used for the building of the personalized tours. The list of the used information is not exhaustive. The waiting and visiting times are expressed in minutes. The possible start and finish points of the daily tours, outlined in Table 2, surround the selected area of interest and represent the 'access doors' to the old city centre through the local transportation means.

In the following, an example of a personalized tour generated as a function of the input information provided by the user is shown. The rating is quantified within the range [0, 100]. The value 100 is assigned to each *POI* that the user expressly requires as belonging to the proposed tour. Moreover, beside the waiting and visiting times, also the walking and the total tour times are reported in minutes, while the walking and the total covered distances are measured in meters. Lastly, the weather is considered sunny during the whole visit.

In the example the case is considered that the user wishes to perform a two–day tour and requires to visit five *POI*s as well.

The input information provided by the user is the following:

- Day of tour beginning: 01/05/2014
- average moving speed: 0.5 m/s
- disabled access request: no
- number of days available for the tour: 2
- number of *POI*s that the user declares to be included in the visit: 5
- identification codes of these *POI*s: 3 6 9 12 15
- For the first day of the tour:
  - start and arrival times: 9.00 am - 7.00 pm
  - start and arrival positions: Dante (M1) - Università (M1)
  - number of programmed rests: 3
  - For each pause:
    - start and finish times of pause 1: 10.45 am - 11.15 am
    - start and finish times of pause 2: 1.00 pm - 2.30 pm
    - start and finish times of pause 3: 5.00 pm - 5.30 pm
- For the second day of the tour:

Table 1. The *POI*s for the old city centre of Naples.

Identifier	Name of the *POI*	Waiting time	Visiting time	Opening hours	Indoor	Disabled access
1	Basilica of San Lorenzo Maggiore	0	40	9.30 am - 5.30 pm	*Y*	*Y*
2	National Archaeological Museum	10	120	9.00 pm - 7.30 pm	*Y*	*Y*
3	Church of Santa Chiara	10	60	7.00 am - 1.00 pm 4.30 pm ÷ 8.00 pm	*Y*	*Y*
4	Capuano Castle	0	30	8.00 am - 8.00 pm	*Y*	*Y*
5	Sant'Antoniello	0	30	9.00 am - 6.00 pm	*Y*	*Y*
6	Sansevero Chapel Museum	5	30	9.30 am - 6.30 pm	*Y*	*Y*
7	The Cathedral (Duomo)	0	40	8.30 am - 1.30 pm 2.30 pm - 8.00 pm	*Y*	*Y*
8	San Marcellino	0	20	8.00 am - 8.00 pm	*N*	*Y*
9	Roman Theater	15	60	10.00 am - 6.00 pm	*Y*	*N*
10	Church of San Gregorio Armeno	0	30	9.30 am - 5.00 pm	*Y*	*Y*
11	Church of Girolamini	5	50	9.30 am - 5.00 pm	*Y*	*Y*
12	Church of Santa Maria Donnaregina	0	40	9.30 am - 16.30 am	*Y*	*Y*
13	Basilica of San Paolo Maggiore	0	30	10.00 am - 6.00 pm	*Y*	*Y*
14	Diomede Carafa Palace	0	60	10.00 am - 1.30 pm	*Y*	*Y*
15	Filangieri Museum	10	90	9.00 am - 1.00 pm	*Y*	*Y*
16	Conservatory of San Pietro a Majella	0	45	10.00 am - 6.00 pm	*Y*	*Y*
17	Church of Gesù Nuovo	0	40	6.30 am - 1.00 pm 4.00 pm - 8.00 pm	*Y*	*Y*
18	Venezia Palace	0	60	9.00 am - 5.00 pm	*Y*	*Y*
19	Church of San Domenico Maggiore	0	35	9.30 am - 12.00 pm 4.30 pm - 7.00 pm	*Y*	*Y*
20	God Nile Statue	0	10	12.00 am - 12.00 am	*N*	*Y*

- start and arrival times: 9.30 am - 4.30 pm
- start and arrival positions: Duomo (M1) - Museo (M1/M2)
- number of programmed rests: 2
- For each pause:
  - start and finish times of pause 1: 11.00 am - 11:30 am
  - start and finish times of pause 2: 1.30 pm - 2.30 pm

The best tour determined in all the runs in accordance to the user information and the rating of the *POI*s derived from the user's profile is:

(17 18 3 2 15 6 7 9 12 11 5 4 16 1 13 18 20 19 10 4)

The tour provided by the algorithm for each day is shown in Table 3. The two-day tour associated to this output file is also graphically reported in Fig. 3 over a

Table 2. The access points for the old city centre of Naples.

Museo (M1/M2)
Dante (M1)
Montesanto (Cumana/Funicular of Montesanto/M2)
Toledo (M1)
Piazzetta Augusteo (Central Funicular)
Municipio (M1/M6/Port/Car Parking)
Università (M1)
Duomo (M1)
Garibaldi (M1/M2/Circumvesuviana/Car Parking)

Table 3. The output of the optimizer.

The tour proposed for the first day										
POI (*Pause*)	*Start time*	*Walking time*	*Waiting time*	*Visiting time*	*Visit end*	*Pause begin*	*Pause end*	*Distance*	*Total distance*	*Score*
Start	9.00 am							0	0	
17		7	0	40	9.47 am			239	239	61
18		5	0	60	10.52 am			175	414	22
(*Pause* 1)			30			10.52 am	11.22 am			
3		7	10	60	12.39 pm			219	633	100
(*Pause* 2)			90			12.39 pm	2.09 pm			
2		30	10	120	4.49 pm			923	1556	80
(*Pause* 3)			30			4.49 pm	5.19 pm			
6		23	5	30	6.17 pm			698	2254	100
20		5	0	10	6.32 pm			158	2412	19
Arrival		15			6.47 pm			453	2865	
The tour proposed for the second day										
POI (*Pause*)	*Start time*	*Arrival time*	*Waiting time*	*Visiting time*	*Visit end*	*Pause begin*	*Pause end*	*Distance*	*Total distance*	*Score*
Start	9.30 am							0	0	
15		5	10	90	11.15 am			158	158	100
(*Pause* 1)			30			11.15 am	11.45 am			
7		14	0	40	12.39 pm			430	588	89
(*Pause* 2)			60			12.39 pm	1.39 pm			
9		9	15	60	3.03 pm			278	866	100
12		20	0	40	4.03 pm			622	1488	100
Arrival		8			4.11 pm			266	1754	
Summary										
Required *POI*s			5	out of	5					
Visited *POI*s			10	out of	20					
Total tour time			988	out of	1020					
Total covered distance			4619							
Total score			77	out of	100					

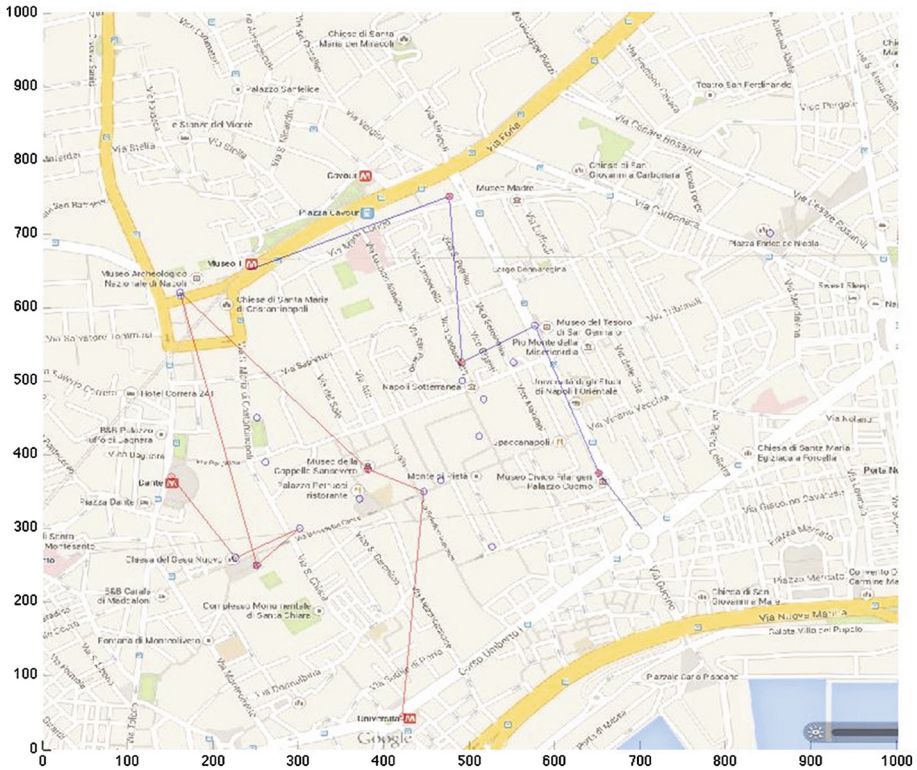

Fig. 3. The two-day tour proposed for the fourth example.

map of the area representing the old city centre of Naples. This figure evidences each day tour with a different color, namely the first day is reported in red whereas the second one is in blue. As it is simple to verify from Table 3, the tour proposed allows visiting all the POIs expressly required by the user. Moreover, the visualization reported in Fig. 3 demonstrates even more the effectiveness of our evolutionary optimizer. In fact, the proposed two-day tour carries out an "intelligent" optimization by automatically subdividing the area of the old city into two distinct zones, each visited in a different day. It is worth noting that on the first day the planned visit to *POI* 3 (Church of Santa Chiara) ends at 12.39 pm, i.e., in good time before this *POI* closes at 1 pm for lunch pause.

5 Conclusions and Future Works

Within this paper a multiobjective evolutionary optimizer for solving a TOPTW, characterized by multiple and contrasting objectives, is proposed. Such an optimizer is innovative as regards the number of both optimization criteria and features considered. Its ability to generate personalized multiple–day walking

tours in a street network modeled as a graph, respecting user's interests and limitations, and environmental constraints, has been successfully tested for the old city centre of Naples. This optimizer will constitute the core of a PETG that will be distributed as a free app for use in the above mentioned area.

As future works, we aim to endow our optimizer with a higher flexibility by providing a route planning capable of adapting to new circumstances in real–time to assure an on–the–fly tour updating.

Acknowledgements. This work has been supported by the project "Organization of Cultural Heritage for Smart Tourism and Real–Time Accessibility (OR.C.HE.S.T.R.A.)" (PON04a2_D) financed within the 2012 "Smart Cities and Communities" call of the Italian Ministry for University and Research.

References

1. Brown, B., Chalmers, M.: Tourism and mobile technology. In: Proceedings of the 8th European Conference on Computer Supported Cooperative Work, pp. 335–354 (2003)
2. Vansteenwegen, P., Souffriau, W., Vanden Berghe, G., Oudheusden, D.V.: The city planner: an expert system for tourists. Expert Syst. Appl. **38**, 6540–6546 (2011)
3. Rodríguez, B., Molina, J., Pérez, F., Caballero, R.: Interactive design of personalised tourism routes. Tourism Manage. **33**, 926–940 (2012)
4. Vansteenwegen, P., Van Oudheusden, D.: The mobile tourist guide: an OR opportunity. OR Insight **20**, 21–27 (2007)
5. Souffiau, W., Vansteenwegen, P., Vertommen, J., Berghe, G.V., Van Oudheusden, D.: A personalised tour trip design algorithm for mobile tourist guides. Appl. Artif. Intell. **22**, 964–985 (2008)
6. Vansteenwegen, P., Souffriau, W., Oudheusden, D.V.: The orienteering problem: a survey. Eur. J. Oper. Res. **209**, 1–10 (2011)
7. Cotfas, L.A., Diosteanu, A., Dumitrescu, S.D., Smeureanu, A.: Semantic search itinerary recommender systems. Int. J. Comput. **5**, 370–377 (2011)
8. Gavalas, D., Kenteris, M., Konstantopoulos, C., Pantziou, G.: Web application for recommending personalised mobile tourist routes. IET Softw. **6**, 313–322 (2012)
9. Bäck, T., Fogel, D.B., Michalewicz, Z. (eds.): Handbook of Evolutionary Computation. Oxford University Press, Oxford (1997)
10. Goldberg, D.E.: Genetic algorithms in search, optimization, and machine learning. Addison-Wesley Professional, New York (1989)
11. Syswerda, G.: Uniform crossover in genetic algorithms. In: Proceedings of the 3rd International Conference on Genetic Algorithms, pp. 2–9 (1989)
12. Spears, W.M., De Jong, K.A.: An analysis of multipoint crossover. In: Proceedings of the Workshop of the Foundations of Genetic Algorithms, pp. 301–315 (1991)
13. Banzhaf, W.: The molecular traveling salesman. Biol. Cybern. **14**, 7–14 (1990)
14. Deep, K., Mebrahtu, H.: Combined mutation operators of genetic algorithms for the travelling salesman problem. Int. J. Comb. Optim. Probl. Inform. **2**, 1–23 (2011)
15. Chiang, C.W., Lee, W.P., Heh, J.S.: A 2-opt based differential evolution for global optimization. Appl. Soft. Comput. **10**, 1200–1207 (2010)