Optimizing Tourist Trip Design for Urban Sustainability

Marco Baioletti[1][0000−0001−5630−7173]
Fabrizio Fagiolo[2][0009−0003−0390−7855]
Valentino Santucci[2][0000−0003−1483−7998]

[1] University of Perugia, 06123 Perugia, Italy
marco.baioletti@unipg.it
[2] University for Foreigners of Perugia, 06123 Perugia, Italy
f.fagiolo@phd.unistrapg.it
valentino.santucci@unistrapg.it

Abstract. The rapid growth of mass tourism poses significant challenges to visitor experience and places substantial environmental and operational pressures on heritage sites and urban areas. This study examines the optimization of individual tourist trips and evaluates their broader urban sustainability impact, using Perugia, a historical city in central Italy, as a case study. In particular, we propose a recommendation system for tourist itineraries that, once configured with data on points of interest, takes tourist preferences as input and generates an optimized trip. Each trip is produced by solving a specifically formulated combinatorial optimization problem aimed at maximizing tourist satisfaction. We demonstrate that the problem is NP-hard and we propose three heuristic algorithms to address it. While optimizing individual trips, algorithm executions also update relevant city-wide state variables, managing queues at the different points of interest, which are then used to assess cumulative urban impact. Experiments were conducted as simulations across multiple scenarios with varying numbers of tourists and different arrival patterns. The results compared favorably with a baseline method and demonstrated that well-engineered micro-level trip decisions can positively influence city-wide outcomes.

Keywords: Tourist Trip Design · Combinatorial Optimization · Heuristic Algorithms · Urban Sustainability · Recommendation System.

1 Introduction

Mass tourism has become a major concern for the sustainable conservation of cultural and natural heritage sites around the world [14]. An excessive concentration of visitors reduces the overall quality of the tourist experience, while amplifying environmental and social pressures. Long queues at popular attractions increase the site's energy consumption due to the intensive use of air conditioning, lighting, toilets, and other support infrastructure. High visitor density leads to increased waste production, noise pollution, and wear and tear on pavements, stairs, and historical materials. This results in the need for more frequent maintenance, with a consequent increase in costs and operational requirements. In addition, overcrowding compromises the accessibility and functionality of exhibition areas, pathways, and main points of interest, making visitor flow management more complex. Ultimately, these factors reduce visitor comfort and intensify the environmental impact on urban areas.

To address these challenges, in this work we introduce a recommendation system for tourist trip planning. The recommendation task is modeled as a constrained combinatorial optimization problem, introducing a novel variant of the Tourist Trip Design Problem (TTDP) [18], which we term the Tourist Trip Design Problem with Emerging Urban Benefit (TTDPwEUB). A tourist requests for a trip suggestion, combined with available data on Points of Interest (PoIs), constitutes an instance of the TTDPwEUB. Since the TTDPwEUB is NP-hard, we propose three constructive heuristic algorithms. Each algorithm execution not only generates a personalized trip for the tourist but also updates system-level variables, managing the queues forming at each PoI. These queue variables are then used to evaluate the cumulative urban benefit over a period of visits (e.g., one day) for the proposed itineraries. Therefore, the main objective is to maximize overall tourist satisfaction while minimizing queue lengths at the city's PoIs over a period of visits, even though the algorithms operate locally for a single tourist request.

The approach is applied to data collected from the city of Perugia, a small urban context in central Italy characterized by significant cultural and touristic relevance [12, 15]. For each tourist request, the algorithm computes an individualized itinerary, while the overall system performance is evaluated through global metrics that integrate both aggregate tourist satisfaction and total waiting times across all PoIs within a single day of visits. To validate the proposed system, extensive experimental simulations are conducted under multiple scenarios, with varying numbers of tourists and different arrival patterns

The paper is organized as follows. Related work is reviewed in Section 2. The TTDPwEUB problem and the Perugia case study are presented in Section 3. The proposed heuristic algorithms are described in Section 4. Experimental simulations are presented and analyzed in Section 5, while Section 6 concludes the paper and outlines future research directions.

2 Related work

A natural basis for designing tourist routing models is the Orienteering Problem (OP), a fundamental model for finding routes among points of interest (PoIs) and for formulating problems such as the TTDP.

OP, introduced by [9], is a combinatorial problem on a graph where, given a certain amount of time available, it is required to select and visit a subset of vertices in order to maximize the total score collected on them, without being obliged to visit all the vertices of the graph. For a comprehensive overview of classical and recent variants of the OP, we refer the reader to [22] and [10].

Among the most effective algorithms, however, we note that for classical OP, [19] propose an adaptive large neighborhood search (ALNS) that achieves several of the best known results on standard benchmark instances. Another approach closely related to this work is the Memetic-GRASP hybrid by [13], which combines GRASP-based construction, local search, and evolutionary operators to achieve highly competitive performance. Finally, one of the approaches that deviates from the classical framework, but is still highly competitive, is the memetic algorithm by [20], which integrates evolutionary search with advanced local improvement mechanisms and further consolidates OP as a key reference model for profit-maximizing routing problems.

An extension of OP is the Team Orienteering Problem (TOP), first introduced by [5]. The TOP extends the OP to multiple routes managed in parallel, each subject to its own time or distance budget, with the goal of assigning customers (or points of interest) to different tours in order to maximize the total collected score. Over the years, several effective heuristic frameworks have been proposed for solving the TOP, including a metaheuristic approach developed by [2] or the guided local search approach by [23] and the path-relinking strategy by [21], which further contributed to establishing the TOP as a key modeling and algorithmic reference for multi-route, score maximization problems.

Starting from this foundation, the Tourist Trip Design Problem (TTDP) emerges as a natural evolution of OP and TOP towards realistic itinerary planning in real and urban contexts. [8] provide a comprehensive survey of algorithmic approaches and methodological advances for TTDP variants, while [24] refine the problem through advanced exact and hybrid schemes. More recently, the relevance of personalization has motivated contributions specifically devoted to the generation of diverse high-quality itineraries, as investigated in [16], further strengthening the TTDP as a flexible framework for tourist-oriented decision support.

Another variant for the TTDP is represented by sustainability-oriented formulations. For example in [17] environmental criteria are explicitly integrated into itinerary planning, as in the multi-objective fuzzy models promoting low-carbon itineraries. Another work is the Green TTDP with Time Windows for urban tours by [6]. [11] introduce the sustainable group trip planning problem combining economic and environmental objectives via adaptive large neighborhood search. In line with this ecological perspective, we introduce a realistic TTDP problem inspired by the city of Perugia, where tourists are assumed to travel exclusively on foot. The itineraries are optimized not only to reduce the environmental impact of urban mobility, but also to alleviate pressure on cultural sites by explicitly mitigating queues and overcrowding at the most popular points of interest.

In this way, the proposed framework promotes a more sustainable and balanced use of local resources, contributing both to higher visitor satisfaction and to reduced stress on the city.

3 The Tourist Trip Design Problem with Emerging Urban Benefit

The TTDPwEUB pursues global urban objectives — such as minimizing overall waiting times and maximizing tourist satisfaction across an entire day of visits — while operating at the level of individual tourists. Although these global metrics arise from system-wide behaviour, an algorithm for the TTDPwEUB works locally: it handles each tourist request independently and returns an optimized trip tailored to that tourist. To ensure consistency between local decisions and global objectives, the problem maintains suitably defined system-wide state variables, allowing the effects of previously planned trips to influence the planning of subsequent ones.

Therefore, the formal definition of the TTDPwEUB provided in Section 3.1 focuses on optimizing a single trip based on city's data and tourist's preferences. Analyses of city-wide metrics, which consider multiple tourists over a full day of visits, are presented in the experimental section of this paper. These analyses are based on simulations generated using the instance generator described in Section 3.2.

3.1 Formal definition

An instance of the TTDPwEUB is defined in terms of a city and a tourist. For clarity, we organize the presentation into three parts: city variables, tourist variables and preferences, and solution space.

City variables.
The city consists of N points of interest (PoIs) $\mathcal{V} = \{1, \ldots, N\}$. PoIs are classified into M categories $\mathcal{C} = \{1, \ldots, M\}$ and each PoI $i \in \mathcal{V}$ belongs to a unique category, denoted by CA_i. For each category $j \in \mathcal{C}$, we denote by $\mathcal{V}_j$ the set of all PoIs belonging to category j.

Within $\mathcal{C}$, two categories play special roles: the category of *starting points* (category 1), and the category of *restaurants* (category 2). PoIs in $\mathcal{V}_1$ are visited exclusively as the starting and ending points of trips, while PoIs in $\mathcal{V}_2$ require special handling due to specific constraints, as will be detailed later.

Each PoI i, except the starting point, can be visited at most once during a trip. The visit lasts for exactly DU_i minutes and must occur within the PoI's opening hours, from OP_i to CL_i.

Certain PoIs, such as parks or viewpoints, have no restriction and can be visited at any time within their opening interval $[OP_i, CL_i]$, without limits on the number of simultaneous visitors. In contrast, other PoIs, such as museums and historical buildings, require advance ticketing and are subject to visit restrictions. We denote by $\mathcal{R}$ the set of all such PoIs. For each PoI $i \in \mathcal{R}$, its opening interval $[OP_i, CL_i]$ is divided into $NT_i = \lfloor (CL_i - OP_i)/DU_i \rfloor$ turns of length DU_i minutes. Visits can only start at the beginning of a turn, and each turn can accommodate at most MC_i tourists, who must reserve their tickets in advance.

The number of ticket already reserved for turn $h \in \{0, \ldots, NT_i - 1\}$ is denoted by $RV_{i,h}$. Clearly, $RV_{i,h} \leq MC_i$.

Each PoI i can be reached from any other PoI j, and the travel time between them is denoted by $TT_{i,j}$.

Tourist variables and preferences.
A tourist who wants to visit the city expresses her preferences by assigning a profit $pr_i \in [0, 100]$ to each PoI i. She then selects a PoI sp of category 1 to serve as both the starting and ending point of the trip, specifies the start time of the trip st, and sets the maximum duration md of the trip. Moreover she can express three types of soft constraints as follows.

1. *Time Window Preference.* For each PoI i, the tourist can specify a preferred visiting interval $[ep_i, lp_i]$. We denote by $\mathcal{J}_1$ the set of PoIs for which the tourist has expressed this type of constraint.
2. *Minimum Visit Preference.* The tourist can specify a minimum number of visits mv_j for each PoI category j. We denote by $\mathcal{J}_2$ the set of categories for which the tourist has expressed this type of constraint.
3. *Lunch Preference.* The tourist may require a visit to a restaurant, preferably within a specified time window $[ep, lp]$. This preference is represented by the binary variable rs, with $rs = 1$ indicating that it has been expressed by the tourist.

It is worth noting that, while a tourist can express multiple time window or minimum visit constraints, the restaurant visit constraint can be specified at most once per tourist.

Solution space.
We can now formally specify how a solution to the TTDPwEUB is represented.

Definition 1 *A solution of the TTDPwEUB problem is a finite sequence of PoIs*

$$\boldsymbol{p} = \langle p_0, p_1, \ldots, p_\ell, p_{\ell+1} \rangle$$

such that: (i) $p_0 = p_{\ell+1} = sp$, (ii) no PoI appears more than once in the sub-sequence $\langle p_1, \ldots, p_\ell \rangle$, (iii) at most one restaurant is included in $\boldsymbol{p}$, and (iv) the sequence is temporally feasible.

To formalize the temporal feasibility of a solution $\boldsymbol{p}$, we consider, for each PoI i appearing in the sequence, i.e., $i = p_h$ for some $h = 1, \ldots, \ell + 1$, the following times:

- t_i^{arr} is the time at which the tourist arrives at PoI i,
- t_i^{vis} is the time at which the tourist begins the visit to i,
- t_i^{dep} is the time at which the tourist finishes to visit i and departs from i.

The arrival time t_i^{arr} satisfies the equation $t_i^{arr} = t_j^{dep} + TT_{j,i}$, where $j = p_{h-1}$ is the previous PoI in the sequence. Note that when $i = p_1$, we set $t_j^{dep} = st$ and $t_i^{arr} = st + TT_{sp,i}$.

For a PoI $i \in \mathcal{R}$, i.e., a PoI subject to restrictions, the visit start time t_i^{vis} is computed as follows. First, we determine the set

$$\mathcal{A}_i = \{h \in \{0, \ldots, NT_i - 1\} \,:\, OP_i + h \cdot DU_i \geq t_i^{arr} \text{ and } RV_{i,h} < MC_i\},$$

which contains all visit turns of PoI i that start at or after t_i^{arr} and still have at least one available ticket. If $\mathcal{A}_i = \emptyset$, it is not possible to visit i at time t_i^{arr} or later, and the solution is therefore infeasible. Otherwise, the visit start time is set as $t_i^{vis} = OP_i + h \cdot DU_i$, where $h = \min \mathcal{A}_i$. Therefore, the visit starts at the first available turn.

Note that, upon arriving at PoI i, the tourist waits for $t_i^{vis} - t_i^{arr}$ minutes before beginning the visit. For PoIs without visit restrictions, i.e., all those PoIs $i \notin \mathcal{R}$, t_i^{vis} coincides with t_i^{arr}. Finally, the departure time is calculated as $t_i^{dep} = t_i^{vis} + DU_i$.

Therefore, we can now define the conditions under which a solution is temporally feasible, as well as the objective function to be maximized.

Definition 2 *A solution* $\boldsymbol{p}$ *is temporally feasible if (i)* $\mathcal{A}_i \neq \emptyset$ *for all PoIs* i *in* $\boldsymbol{p}$ *such that* $i \in \mathcal{R}$*, (ii) for every PoI* i *in* $\boldsymbol{p}$ *the constraints*

$$OP_i \leq t_i^{vis} < t_i^{dep} \leq CL_i \tag{1}$$

hold, and (iii) the trip ends within the allowed duration, i.e., when $t_{sp}^{arr} \leq st + md$.

It is important to remark that in a feasible solution, for each visited PoI i, the condition (1) also holds when $i \in \mathcal{R}$, and that all the times t_i^{arr}, t_i^{vis} and t_i^{dep} are within the bounds $[st, st + md]$.

Definition 3 *A solution* $\boldsymbol{p} = \langle p_0, p_1, \ldots, p_{\ell+1} \rangle$ *is optimal if it maximizes the following objective function*

$$sat(\boldsymbol{p}) = \sum_{i \in \boldsymbol{p}} pr_i - \alpha \sum_{i \in \mathcal{J}_1} \mathbf{P}_1(i) - \beta \sum_{j \in \mathcal{J}_2} \mathbf{P}_2(j) - \gamma \mathbf{P}_3 \tag{2}$$

where: $\mathbf{P}_1, \mathbf{P}_2$*, and* $\mathbf{P}_3$ *penalize violations of the corresponding type-1, type-2, and type-3 soft constraints, while the positive constants* α, β*, and* γ *scale the penalties so that they are expressed in comparable profit units.*

The first penalty function is defined as $\mathbf{P}_1(i) = \max(ep_i - t_i^{vis}, 0) + \max(t_i^{dep} - lp_i, 0)$ and penalizes cases in which the visit begins before ep_i or ends after lp_i.

The second penalty function is $\mathbf{P}_2(j) = \max(mv_j - nv_j, 0)$, where nv_j denotes the number of PoIs in $\boldsymbol{p}$ belonging to category j. This penalty measures how many visits are missing to respect the minimum requirement mv_j.

Finally, the third term $\mathbf{P}_3$: is 0 when $rs = 0$; takes a large value $NL = 50$ when $rs = 1$ but no restaurant is included in the solution; and it becomes

$(\max\{er - t_h^{vis}, 0\} + \max\{t_h^{dep} - lr, 0\})/2$ when $rs = 1$ and h is the unique restaurant in $\boldsymbol{p}$.

The constants α and γ are set to 1, while β is set to 25. The rationale behind this choice is that type-2 constraints, which impose a minimum number of visits to PoIs of a given category, represent more critical tourist preferences and are therefore penalized by 25 points per unit of violation. In contrast, type-1 and type-3 constraints correspond to temporal preferences, and each minute of violation results in a penalty of one point of satisfaction.

3.2 Instance generation

In this section, we describe how we generate the TTDPwEUB instances used in our experiments. We begin by constructing the city data for our study case: starting from OpenStreetMap data for Perugia, we select a realistic set of PoIs, determine which of them are reachable on foot, and compute walking distances and times between all locations. Then, we generate the tourists data: for each tourist, we randomly sample a starting PoI, assign a start time, set maximum visit duration, and define individual preferences.

City data generation.
City-related data are derived using OpenStreetMap information for the municipality of Perugia. Initially, we identify the city center and build a circular buffer with a fixed radius of 5 km around this point. This buffer defines the study area, and all PoIs in the instance are located within it.

PoIs are extracted from OpenStreetMap using the *osmnx* package [4], by selecting all entities tagged with `tourism` and belonging to the categories *museum*, *gallery*, *attraction*, *viewpoint*, and *restaurant* within the buffer. In the resulting instance, each location is assigned an integer category label: (1) *start_point*, (2) *restaurant*, (3) *museum*, (4) *gallery*, (5) *attraction*, (6) *viewpoint*.

To ensure that the selected points are actually reachable on foot, we construct a walk travel graph on the study area. Then, we calculate the minimum distances with respect to the node corresponding to the city center, and we discard all PoIs that would require a walking distance of more than 5 km.

Each PoI i is assigned values for the variables introduced in Section 3.1, i.e., the visit duration DU_i, an opening interval $[OP_i, CL_i]$, the maximum effective capacity MC_i, a coefficient about the daily average number of visits $\mu_i^{(p)}$, and the number of service turns NT_i, determined based on capacity and the width of the time window. The starting PoIs (e.g., parking lots or central squares) are modeled as always-available locations with zero visit duration.

Finally, we construct the matrix of traveling times $TT_{i,j}$ between every pair of PoIs i and j.

Tourists data generation.
The data of a tourist are generated using the following procedure.

The starting point sp is randomly chosen among the all PoIs in category 1.

The start time st is sampled from a mixture of three Normal distributions $\sum_{i=1}^{3} w_i \mathcal{N}(\mu_i, \sigma_i^2)$, where: the means are set as $\mu_1 = 9am$, $\mu_2 = 11am$, and $\mu_3 = 2pm$, corresponding to three peak times; the standard deviation is set to $\sigma_i = 30$ minutes for $i = 1, 2, 3$; while the weights are $w_1 = 4/9$, $w_2 = 3/9$, and $w_3 = 2/9$.

The maximum duration md is chosen as follows. We first define a set of candidate visit end times $ET = \{1pm, 3pm, 6pm, 7pm, 8pm\}$. Considering the start time st, we select a subset of admissible end times $ET(st) \subseteq ET$, and then draw one value $et \in ET(st)$ uniformly at random. Specifically, if $st < 10am$ we set $ET(st) = ET$; if $10am \leq st < 3pm$ we set $ET(st) = \{7pm, 8pm\}$; otherwise we set $ET(st) = \{6pm, 7pm, 8pm\}$ and keep only those end times satisfying $et \geq st$. The maximum duration is then obtained as the time difference between the chosen end time and the start time, i.e., $md = et - st$.

The profit (tourist preference) pr_i for PoI i is sampled from the Normal distribution $\mathcal{N}(\mu_i^{(p)}, 0.1)$. Then, all tourist profits are normalized to sum to 100.

Since preliminary experiments have confirmed the intuition that a solution is usually formed by a small number of PoIs, we have decided to generate at most one tourist constraint for each type. In fact, a larger number of constraints would have produced many over-constrained instances, with small or null chances of satisfying all the soft constraints.

We introduce a soft constraint of type-1 (time window preference) with probability 0.5. If the constraint is present, a PoI i is randomly selected among the 25% PoIs with the largest profit and the time window $[ep_i, lp_i]$ is randomly chosen within the intersection of the maximum time window of the tourist $[st, st + md]$ and the time window $[OP_i, CL_i]$ when i is open. Clearly, $\mathcal{J}_1 = \{i\}$.

Also the second constraint (minimum visit preference) is introduced with probability 0.5. If the constraint is present, a category j is selected among $\mathcal{C} \setminus \{1, 2\}$ with a probability proportional to the size of the category. Therefore, $\mathcal{J}_2 = \{j\}$. The minimum number of PoIs mv_j is set to 2.

The type-3 constraint (lunch preference) is introduced with probability 0.9, but only if the maximum time interval for the trip $[st, st + md]$ contains the usual time range for the lunch. A suitable time interval for the lunch is randomly generated by uniformly selecting a one-hour time slot between $12pm$ and $15pm$.

4 The constructive algorithms

The search space of the TTDPwEUB problem can be characterized as follow. Let $\overline{\mathcal{V}}$ denote the set $\mathcal{V} \setminus (\mathcal{V}_1 \cup \mathcal{V}_2)$ and n its size. Let $\mathcal{P}$ be the set of finite sequences of elements of $\overline{\mathcal{V}}$ without repetitions. Hence, any element of $\mathcal{P}$ has length less or equal than n. A solution of the problem $\boldsymbol{p}$ is obtained from an element $\boldsymbol{p}' \in \mathcal{P}$ by adding at the beginning and at the end of $\boldsymbol{p}'$ the starting point sp.

The number of possible solutions is then

$$|\mathcal{P}| = \sum_{\ell=0}^{n} \binom{n}{\ell} \ell! = \sum_{\ell=0}^{n} \frac{n!}{(n-\ell)!} \tag{3}$$

It is easy to see that the quantity in (3) is larger than $n!$, implying that the search space of the TTDPwEUB problem is substantially larger than that of classical permutation problems, such as the Permutation Flowshop Scheduling Problem or the Quadratic Assignment Problem.

In the following sections, we present three heuristic algorithms for the TTDP-wEUB, all based on the same underlying concept, as well as a random algorithm that will serve as a baseline for our experiments.

4.1 Greedy Construction Heuristic

The greedy constructive heuristic GREEDY iteratively builds a solution, adding one PoI at a time to the sequence $\boldsymbol{p}$, which is initialized to $\langle sp \rangle$.

We denote by $\mathcal{NV}$ the set of all PoIs which can be visited at the current iteration. Initially, $\mathcal{NV} = \{sp\} \cup (\mathcal{V} \setminus \mathcal{V}_1)$, i.e., all PoIs not belonging to category 1. Note that sp is included in $\mathcal{NV}$ to allow the trip to end at the starting point. Moreover, when $rs = 0$, also the restaurants (category 2) have to be removed from $\mathcal{NV}$, which then becomes $\mathcal{NV} = \{sp\} \cup (\mathcal{V} \setminus (\mathcal{V}_1 \cup \mathcal{V}_2))$.

At each iteration, the first step is to determine the set $\mathcal{F}$ of PoIs which can be safely added to the sequence $\boldsymbol{p}$. A PoI i is included in $\mathcal{F}$ if all the following conditions are satisfied.

(i) $i \in \mathcal{NV}$.
(ii) If $i \in \mathcal{R}$, the set $\mathcal{A}_i$ must be non-empty, ensuring that i can be visited. On the other hand, if $i \notin \mathcal{R}$, the condition in Equation (1) must be satisfied.
(iii) $t_i^{dep} + TT_{i,sp} \leq st + md$, ensuring that the tourist can return to sp within the available time.

Condition (iii) also ensures that the visit to i concludes before $st+md$. Moreover, $\mathcal{F}$ cannot be empty, because sp always satisfies the conditions (i)-(iii).

For each PoI $i \in \mathcal{F}$, a score s_i is computed as

$$s_i = \frac{pr_i - \alpha\Delta_1^{(i)} - \beta\Delta_2^{(i)} - \gamma\Delta_3^{(i)}}{\left(t_i^{dep} - t_{i'}^{dep}\right) + (TT_{i,sp} - TT_{i',sp})}, \tag{4}$$

where i' is the last element of $\boldsymbol{p}$.

The numerator of Equation (4) is the difference of the objective function computed on the sequence $\boldsymbol{p} + i$, obtained by adding i to $\boldsymbol{p}$, and on the current sequence $\boldsymbol{p}$, i.e., $sat(\boldsymbol{p} + i) - sat(\boldsymbol{p})$.

The terms $\Delta_1^{(i)}, \Delta_2^{(i)}$ and $\Delta_3^{(i)}$ denote the algebraic contributions of adding i with respect to the satisfaction of the soft constraints of type 1, 2, and 3, respectively. If $i \notin \mathcal{J}_1$ (i.e., i is not subject to constraint 1), then $\Delta_1^{(i)} = 0$, otherwise $\Delta_1^{(i)} = \mathbf{P}_1(i)$. If $CA_i \notin \mathcal{J}_2$ (i.e., i is not subject to constraint 2) or if the number of already visited PoIs in category CA_i is at least mv_{CA_i}, then $\Delta_2^{(i)} = 0$, otherwise $\Delta_2^{(i)} = -1$, since adding i to current solution reduces $\mathbf{P}_2(CA_i)$ by 1. Finally, if $CA_i \neq 2$ (i.e., i is not a restaurant), then $\Delta_3^{(i)} = 0$, otherwise, $\Delta_3^{(i)} = (\max\{er - t_i^{vis}, 0\} + \max\{t_i^{dep} - lr, 0\})/2 - NL$, which takes into account the satisfaction of the temporal constraint on lunch time and the fact that the penalty NL is no longer valid. In fact, a restaurant i is taken into account only if the tourist needs to have lunch during the trip. By adding i, this request is fulfilled.

The denominator of Equation (4) represents the time spent traveling from PoI i' to PoI i, taking into account that the trip must eventually return to the starting point sp. The rationale behind this choice is that s_i represents the amount of satisfaction gained per unit of time spent, under the hypothesis that PoI i is added to the trip.

The iteration continues by selecting the PoI $i^\star \in \mathcal{F}$ with the highest score appending it to $\boldsymbol{p}$. $i^\star$ is then removed from $\mathcal{NV}$ to prevent duplicates in $\boldsymbol{p}$. Moreover, if $i^\star$ is a restaurant, all the restaurants are removed from $\mathcal{NV}$ to avoid that the tourist goes again in a restaurant.

The loop terminates with success as soon as $i^\star = sp$. In this way, the trip is complete. It is easy to prove that this algorithm always produces a feasible solution because, by construction, all the properties (i)-(iv) of Definition 1 are satisfied.

4.2 Random Construction Heuristic

The random constructive heuristic RANDOM serves as a baseline method. It works in a similar way as GREEDY, except that the PoI to be added to $\boldsymbol{p}$ at each iteration is randomly selected from $\mathcal{F}$ with a uniform distribution.

4.3 Roulette Wheel Construction Heuristic

The GREEDY heuristic is deterministic and may occasionally make myopic decisions which can lead to solutions with a low value of the objective function sat.

To add some randomness to GREEDY and try to overcome its limitation, the Roulette Wheel Construction Heuristic ROULETTE randomly selects at each iteration a PoI i from $\mathcal{F}$ with a probability proportional to $\bar{s}_i$. The adjusted score $\bar{s}_i$ is computed as

$$\bar{s}_i = s_i - \min_j s_j + \epsilon$$

where ϵ is a small value. This transformation is necessary because the scores s_i can be negative. The role of ϵ is simply to avoid zero probability.

Since ROULETTE is stochastic, it is applied $N_{tries} = 100$ times and the best solution (i.e., the solution with highest value of sat) is taken as its result.

4.4 Hybrid Construction Heuristic

The Hybrid Construction Heuristic HYBRID combines GREEDY and ROULETTE. At each iteration, with a small probability (set to 0.1 in our experiments), the

Table 1: Median waiting time and satisfaction with IQR for each algorithm and number of tourists.

		Total Waiting Time		Satisfaction	
# Tourists	Algorithms	Median	IQR	Median	IQR
1000	Greedy	▽ 134.00	±4.25	▽ 13.75	±2.52
	Hybrid	**124.00**	±2.50	**19.09**	±0.25
	Random	125.00	±7.50	▽ -36.96	±0.59
	Roulette	125.00	±2.25	18.45	±0.25
2000	Greedy	▽ 134.00	±2.50	▽ 13.83	±0.47
	Hybrid	118.00	±2.50	**17.76**	±0.26
	Random	▽ 121.50	±4.50	▽ -36.34	±0.77
	Roulette	**117.50**	±3.50	17.10	±0.18
3000	Greedy	▽ 133.00	±3.00	▽ 11.46	±1.62
	Hybrid	108.50	±2.00	**15.68**	±0.25
	Random	**107.00**	±1.25	▽ -35.73	±1.07
	Roulette	▽ 109.00	±3.50	15.44	±0.19

PoI i with the largest score s_i in $\mathcal{F}$ is selected, as in GREEDY, otherwise i is chosen as in ROULETTE algorithm.

Also, the heuristic HYBRID is repeated $N_{tries} = 100$ times and the best solution in terms of satisfaction is returned.

5 Experiments

The experiments were carried out with a C++ implementation of the solvers. We considered three problem sizes with 1000, 2000, and 3000 tourists and the four different heuristic algorithms (GREEDY, ROULETTE, HYBRID, RANDOM).For each combination of number of tourists and algorithm, 15 independent simulations were executed. Instances data were generated as described in Section 3.2, and each simulation produces statistics on the city's receptivity, individual tourist visits, and waiting times. In total, 180 experiments were carried out (3 problem sizes $\times$ 4 algorithms $\times$ 15 simulations).

Table 1 shows, for each combination of problem size and algorithm, median and inter-quartile range (IQR) of the waiting time and degree of satisfaction. Values in bold indicate the best medians in each row. The results are statistically validated using the Mann Whitney U test, applied to compare each algorithm against the best-performing algorithm for the problem. A confidence threshold of 0.05 is considered. Therefore, the algorithms obtaining significantly worse results are marked with the symbol $\triangledown$.

All the three algorithms outperform the baseline method Random, which has high waiting times and yields strongly negative satisfaction across all settings, indicating that tourist preferences are largely ignored. Overall, Hybrid offers the most robust trade-off: for 1000 tourists it achieves both the lowest median waiting time (124 minutes) and the highest satisfaction (19.09), and for 2000 and 3000 tourists it consistently yields the highest satisfaction (17.76 and 15.68) with waiting times very close to the best ones. Roulette is competitive in terms of waiting time, though slightly worse in terms of satisfaction, while Greedy produces consistently larger waiting times and smaller values for the satisfaction.

Table 2 shows the overall average results of all simulations, displaying the performance of the four algorithms as the number of tourists increases. Here, *Len*

Table 2: Average trip metrics and penalty terms for the four algorithms on simulations with 1000, 2000, and 3000 tourists.

# Tourists	Algorithm	Len.	Wait/PoI	Sat.	Ob. Profit	Viol.1	Viol.2	Viol.3
1000	Greedy	6.72	21.89	5.38	18.05	0.07	0.15	12.45
	Hybrid	7.10	17.60	11.57	17.79	0.04	0.13	6.05
	Random	4.46	33.39	-41.20	-1.85	11.11	0.50	27.74
	Roulette	6.98	18.21	10.56	16.58	0.03	0.14	5.85
2000	Greedy	6.44	23.32	3.94	15.06	0.03	0.19	10.90
	Hybrid	7.03	16.89	8.97	15.85	0.08	0.18	6.62
	Random	4.56	32.63	-40.50	-1.80	11.36	0.53	26.81
	Roulette	6.94	17.42	8.15	15.00	0.06	0.18	6.61
3000	Greedy	6.18	24.61	0.76	12.32	0.05	0.25	11.26
	Hybrid	6.86	16.38	3.72	13.54	0.12	0.23	9.47
	Random	4.32	31.38	-39.72	-3.11	10.71	0.57	25.33
	Roulette	6.77	16.77	3.07	12.81	0.10	0.24	9.40

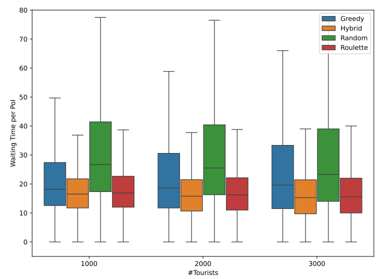
is the average trip length, and *Wait/PoI* is the average waiting time per visited PoI. The *Sat* column indicates the average tourist satisfaction, i.e., the value returned by the objective function in Equation (2). The following columns also show the average values for each one of the objective function terms: *Ob. Profit* is the average of observed profits for visited PoIs, while *Viol.1*, *Viol.2* and *Viol.3* correspond to the values of the three penalty terms .

From Table 2, it emerges that Greedy, Hybrid, and Roulette produce trips that are broadly comparable in terms of average length and waiting times. However, Hybrid and Roulette achieve higher average overall satisfaction while still maintaining very low levels of constraint violation. Greedy remains competitive, but performs slightly worse than these two, especially with respect to overall satisfaction. By contrast, the Random algorithm stands out negatively: it visits fewer points of interest on average, produces longer waiting times on the visited PoIs, yields negative overall satisfaction values, and results in a substantially higher number of constraint violations. The smaller value of the Total Waiting Time obtained by Random on Table 1 can be explained by considering that this algorithm produces considerably shorter solutions and hence the sum of all the time waited by the tourist is smaller, even though at each PoI she waits for a longer amount of time.

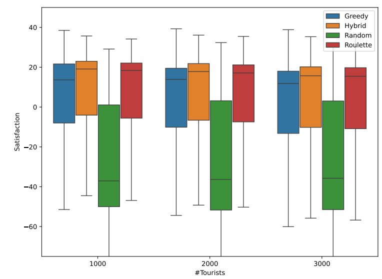
Globally, the table clearly shows that guided approaches such as Hybrid and Roulette produce more balanced and higher quality trips, while a purely random construction leads to decidedly unsatisfactory solutions, thus promoting our proposed algorithms.

For a more comprehensive analysis, Figures 1 present the boxplots of the main indicators, highlighting the variability of the solutions produced by the different algorithms.

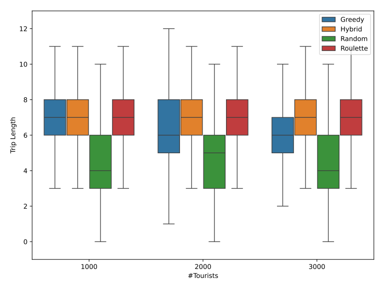
Fig. 1a compares the distributions of the waiting times across the different algorithm, showing that Hybrid and Roulette generally yield lower and more stable waiting times than Greedy, while Random attains comparable medians but with higher dispersion. Overall, waiting times slightly increase as the number of tourists grows, due both to higher congestion in the system and to the limited accommodation capacity of the city.



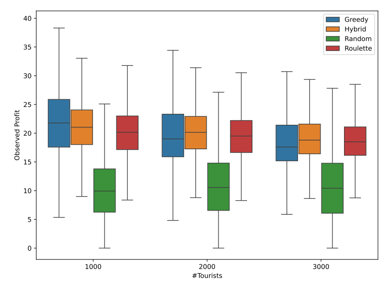
(a) Waiting Time per PoI



(b) Satisfaction



(c) Trip Length



(d) Observed Profits

Fig. 1: Summary of the experimental results

Fig. 1b shows the distribution of tourist satisfaction values for each algorithm. We can see that Hybrid and Roulette consistently achieve the highest and most stable satisfaction values, with positive medians in all scenarios, while Greedy achieves lower but still positive values. On the contrary, Random is clearly the worst algorithm for tourist satisfaction, with distributions concentrated on strongly negative satisfaction. There is a slight decrease in satisfaction for all algorithms as the number of tourists increases, reflecting the greater difficulty of satisfying time preferences in more congested contexts.

Fig. 1c shows the distribution of trip length, measured as the number of PoIs actually visited, for the four algorithms. In this way, the figure highlights which algorithms tend to construct longer or shorter itineraries and how stable these choices are, clarifying how intensively the city is explored in each configuration.

In Fig. 1d the distribution of the observed profit obtained by the four algorithms (Greedy, Hybrid, Random, Roulette) is reported. Overall, the figure integrates the previous metrics, showing the effectiveness with which each strategy exploits the opportunities available under different demand conditions.

6 Conclusion and Future Work

In this work, we introduced the Tourist Trip Design Problem with Emerging Urban Benefit (TTDPwEUB), a new variant of the Tourist Trip Design Problem that explicitly combines the optimization of individual itineraries with city-level sustainability issues.

Each tourist request is handled locally, but the solver maintains global state variables describing queues and capacities at points of interest, so that micro-level decisions about individual trips contribute to macro-level goals such as reducing congestion and improving the overall visitor experience. We have proposed a set of algorithms, namely GREEDY, ROULETTE, and HYBRID, complemented by a baseline method which generates random (but feasible) solutions.

Experimental analysis, based on simulations with different numbers of tourists, showed that guided heuristics can effectively balance individual satisfaction and urban sustainability. In particular, HYBRID and ROULETTE strategies consistently achieve higher satisfaction levels and lower constraint violation levels than the greedy constructive heuristic, while maintaining travel times and waiting times at competitive levels. The random baseline, on the other hand, leads to shorter or comparable waiting times only at the cost of frequent violations of user-generated preferences, resulting in unsatisfaction and poorly structured itineraries. As the number of tourists increased, we observed a moderate increase in waiting times and a slight decrease in satisfaction, reflecting the combined effect of increased congestion and the city's limited capacity. The results support the idea that well-designed recommendation policies can mitigate some of the problems of mass tourism while preserving the quality of the visitor experience.

The proposed model also offers several benefits from an urban sustainability perspective: it recommends restaurants without biases, promoting economic equity and supporting local businesses; considers PoIs that are reachable on foot; and tries to minimize overall waiting times.

Future work will investigate more advanced optimization techniques and richer variants of the problem. From an algorithmic perspective, a natural direction is to design meta-heuristic approaches for the problem, such as Iterated Local Search methods or Evolutionary Algorithms (such as e.g. [1, 3, 7]).

From the point of view of problem modeling, the current formulation could be extended to handle dynamic re-optimization of trips during the day (e.g., in response to unexpected interruptions or congestion) and more refined profiling of visitors based on their interests. This would allow, for example, tourists to be directed to PoIs more consistent with their inclinations. In addition, tourist flows could be organized according to season and different time slots, integrating predictive models to anticipate congestion at points of interest in critical areas of the city.

Acknowledgments. Valentino Santucci has been partially supported from the following research funds/projects: "Università per Stranieri di Perugia – Finanziamento Dipartimentale alla Ricerca – FDR 2023-2026", and "Università per Stranieri di Perugia – Progettazione e sviluppo di strumenti digitali per la formazione a distanza".

Marco Baioletti has been partially supported by the SERICS project (PE00000014) under the MUR National Recovery and Resilience Plan funded by the European Union-NextGenerationEU.

References

1. Agresta, A., Baioletti, M., Biscarini, C., Milani, A., Santucci, V.: Evolutionary algorithms for roughness coefficient estimation in river flow analyses. In: International Conference on the Applications of Evolutionary Computation (Part of EvoStar). pp. 795–811. Springer (2021)
2. Archetti, C., Hertz, A., Speranza, M.G.: Metaheuristics for the team orienteering problem. Journal of Heuristics **13**(1), 49–76 (Feb 2007). https://doi.org/10.1007/s10732-006-9004-0
3. Baioletti, M., Milani, A., Santucci, V.: Variable neighborhood algebraic differential evolution: An application to the linear ordering problem with cumulative costs. Information Sciences **507**, 37–52 (2020)
4. Boeing, G.: Modeling and analyzing urban networks and amenities with osmnx. Geographical Analysis **57**(4), 567–577 (2025). https://doi.org/https://doi.org/10.1111/gean.70009
5. Chao, I.M., Golden, B.L., Wasil, E.A.: The team orienteering problem. European Journal of Operational Research **88**(3), 464–474 (1996). https://doi.org/10.1016/0377-2217(94)00289-4
6. De Maio, A., Musmanno, R., Skrame, A.: The green tourist trip design problem with time windows: A model application on a urban scale. pp. 62–70 (01 2023). https://doi.org/10.5220/0011669500003396
7. Di Bari, G., Baioletti, M., Santucci, V.: An experimental evaluation of the algebraic differential evolution algorithm on the single row facility layout problem. In: Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion. pp. 1678–1684 (2020)
8. Gavalas, D., Konstantopoulos, C., Mastakas, K., Pantziou, G.: A survey on algorithmic approaches for solving tourist trip design problems. Journal of Heuristics **20**(3), 291–328 (2014). https://doi.org/10.1007/s10732-014-9242-5
9. Golden, B., Levy, L., Vohra, R.: The orienteering problem. Nav Res Logist **34**, 307–318 (06 1987). https://doi.org/10.1002/1520-6750(198706)34:3<307::AID-NAV3220340302>3.0.CO;2-D
10. Gunawan, A., Lau, H.C., Vansteenwegen, P.: Orienteering problem: A survey of recent variants, solution approaches and applications. European Journal of Operational Research **255**(2), 315–332 (2016). https://doi.org/10.1016/j.ejor.2016.04.059
11. Kolaee, M.H., Jabbarzadeh, A., e hashem, S.M.J.M.A.: Sustainable group tourist trip planning: An adaptive large neighborhood search algorithm. Expert Systems with Applications **237**, 121375 (2024). https://doi.org/10.1016/j.eswa.2023.121375
12. Liberotti, R., Paolocci, M.: Perugia, city walls and green areas: Possible interactions between heritage and public space restoration. Sustainability **17**(15), 6663 (2025)
13. Marinakis, Y., Politis, M., Marinaki, M., Matsatsinis, N.: A memetic-grasp algorithm for the solution of the orienteering problem. In: Le Thi, H.A., Pham Dinh, T., Nguyen, N.T. (eds.) Modelling, Computation and Optimization in Information Systems and Management Sciences. pp. 105–116. Springer International Publishing, Cham (2015). https://doi.org/10.1007/978-3-319-18167-7_10
14. Mihalic, T.: Conceptualising overtourism: A sustainability approach. Annals of Tourism Research **84**, 103025 (2020). https://doi.org/10.1016/j.annals.2020.103025
15. Murgante, B., Patimisco, L., Annunziata, A.: Developing a 15-minute city: A comparative study of four italian cities-cagliari, perugia, pisa, and trieste. Cities **146**, 104765 (2024)

16. Pérez-Cañedo, B., Novoa-Hernández, P., Pelta, D.A.: On the generation of diverse solution sets in a tourist trip design problem - an experimental study. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion. p. 579–582. GECCO '25 Companion, Association for Computing Machinery, New York, NY, USA (2025). https://doi.org/10.1145/3712255.3726656
17. Ruiz-Meza, J., Brito, J., Montoya-Torres, J.: Multi-objective fuzzy tourist trip design problem with heterogeneous preferences and sustainable itineraries. Sustainability **13**, 9771 (08 2021). https://doi.org/10.3390/su13179771
18. Ruiz-Meza, J., Montoya-Torres, J.R.: A systematic literature review for the tourist trip design problem: Extensions, solution techniques and future research lines. Operations Research Perspectives **9**, 100228 (2022). https://doi.org/10.1016/j.orp.2022.100228
19. Santini, A.: An adaptive large neighbourhood search algorithm for the orienteering problem. Expert Systems with Applications **123**, 154–167 (2019). https://doi.org/10.1016/j.eswa.2018.12.050
20. Santucci, V., Baioletti, M.: A memetic approach for the orienteering problem. In: Cicirelli, F., Guerrieri, A., Pizzuti, C., Socievole, A., Spezzano, G., Vinci, A. (eds.) Artificial Life and Evolutionary Computation. pp. 38–48. Springer International Publishing, Cham (2020)
21. Souffriau, W., Vansteenwegen, P., Vanden Berghe, G., Van Oudheusden, D.: A path relinking approach for the team orienteering problem. Computers & Operations Research **37**(11), 1853–1859 (2010). https://doi.org/10.1016/j.cor.2009.05.002
22. Vansteenwegen, P., Souffriau, W., Oudheusden, D.V.: The orienteering problem: A survey. European Journal of Operational Research **209**(1), 1–10 (2011). https://doi.org/10.1016/j.ejor.2010.03.045
23. Vansteenwegen, P., Souffriau, W., Vanden Berghe, G.: A guided local search metaheuristic for the team orienteering problem. European Journal of Operational Research **196**, 118–127 (07 2009). https://doi.org/10.1016/j.ejor.2008.02.037
24. Vu, D.M., Kergosien, Y., Mendoza, J.E., Desport, P.: Branch-and-check approaches for the tourist trip design problem with rich constraints. Computers & Operations Research **138**, 105566 (2022). https://doi.org/10.1016/j.cor.2021.105566