The Tourist Trip Design Problem with POI Categories via an Expectation-Maximization Based Method

Costas Panagiotakis[1], Evangelia Daskalaki[2], Harris Papadakis[2] and Paraskevi Fragopoulou[2]

[1]Department of Management Science and Technology, Hellenic Mediterranean University, 72100, Agios Nikolaos, Greece
[2]Department of Electrical and Computer Engineering, Hellenic Mediterranean University, 71004, Heraklion, Greece

Abstract
In this work, we propose an efficient deterministic method based on Expectation-Maximization (EM) to solve the challenging problem of the tourist trip design or Personalized Itinerary Recommendation (PIR) with POI categories. PIR aims to recommend a personalized tour that consists of a sequence of Points of Interest (POIs), which maximizes user satisfaction and adheres to user time budget constraints. Additionally, POIs are divided into categories, so that the tourist is able to provide minimum/maximum limits on the number of POIs belonging to each category. This framework mainly focuses on the POIs sequence selection problem exploiting the personalized POI recommendations provided by a recommender system. The proposed method sequentially solves the PIR problem by providing in each step the POI that is expected to maximize a suitable objective function, taking into account user satisfaction, user time budget, POIs opening hours, POIs category constraints and spatial constraints (e.g. start and end point, POIs locations, etc). The proposed system has been also applied in a version with multiple collaborating instances that improves the exploration of the search space and increases the score of the objective function. The proposed system is also integrated with a complete tourist trip design system. Experimental results and comparisons to existing methods on a large number of synthetic and real datasets demonstrate the high performance, robustness and the computational efficiency of the proposed system.

Keywords
Itinerary recommendation, Trip planning, Orienteering problem, Recommender systems

1. Introduction

Recommender Systems predict the preferences of users for specific items, based on an analysis of previous user preferences [1, 2]. They have become increasingly popular in assisting users in decision making problems. A large number of different techniques appear in the literature for Recommender Systems which can be classified into two main categories namely, *Collaborative Filtering* and *Content-based*. Collaborative filtering uses only the preferences (e.g. ratings) of

RecSys Workshop on Recommenders in Tourism (RecTour 2022), September 22th, 2022, co-located with the 16th ACM Conference on Recommender Systems, Seattle, WA, USA
cpanag@hmu.gr (C. Panagiotakis); eva@ics.forth.gr (E. Daskalaki); adanar@hmu.gr (H. Papadakis); fragopou@ics.forth.gr (P. Fragopoulou)
https://sites.google.com/site/costaspanagiotakis/ (C. Panagiotakis); http://users.ics.forth.gr/~eva/ (E. Daskalaki); https://www.ics.forth.gr/person/Fragopoulou/Paraskevi (P. Fragopoulou)
0000-0003-3680-7087 (C. Panagiotakis); 0000-0003-3056-1311 (E. Daskalaki); 0000-0002-5751-1923 (H. Papadakis); 0000-0002-7134-9029 (P. Fragopoulou)

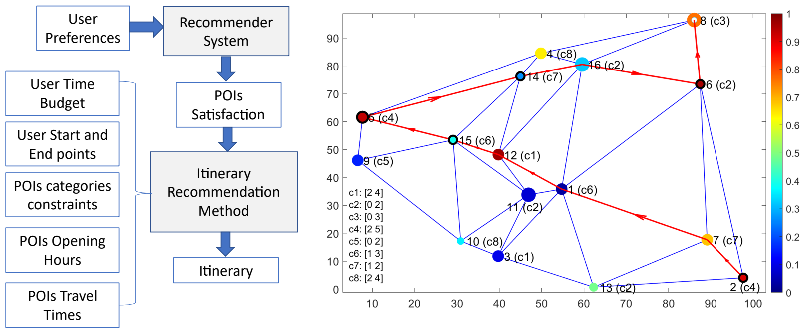
users for items [3]. Content-based systems suggest items whose content is similar to items that have been evaluated by a user [4]. Approaches that use a combination of these two main categories have also been proposed [5]. Recommender systems have been successfully applied on a variety of entities such as e-shop items, web pages, news feeds, social networks, articles, movies, music, hotels, television shows, books, restaurants, friends, etc.

Recommender systems have been also successfully applied to an important and complex task related to tourists that concerns the planning and scheduling of tour itineraries, which comprise a sequence of Points-of-Interest (POIs) based on the unique preferences of each tourist [6]. The complex task of tour itinerary recommendation may also incorporate, apart from user preferences, various real-life constraints such as limited time for touring, traffic conditions, spatial heterogeneity of POIs, POIs opening hours, weather conditions, group travel, POI popularity, queuing times, pricing and crowdedness. The selection of the most valuable POIs is not trivial due to the aforementioned constrains and parameters as well as the personalized satisfaction criteria and limitations of each tourist.

The tourist trip design problem or personalized itinerary recommendation (PIR) problem is an extension of the orienteering problem applied to tourism. In the orienteering problem, a set of vertices is given, each with a score. The goal is to determine a path, limited in length, that visits some vertices and maximises the sum of the collected scores [7]. The tourist trip design problem consists in selecting a subset of locations to visit from among a larger set while maximizing the benefit (user satisfaction) for the tourist. The benefit is given by the sum of the rewards collected at each location visited with constraints such as budget, POIs opening hours (i.e. time windows at the locations), start and end points, starting time and maximum trip duration [6, 8].

Therefore, in this work, we study the PIR problem. In order to concentrate on the POIs sequence selection problem, we assume that the gained user satisfaction per visited POI is provided to the system (e.g. predicted by a Recommender System based on user preferences). So, the main goal of this work is to provide a sequence of POIs that maximize user satisfaction under several given constraints such as user time budget, user defined POI categories, POI opening hours and spatial constraints (e.g. start and end user points, POIs locations, etc). An advantage of the proposed method is that it can be easily adapted to the user preference of POIs which is provided as an input to the proposed method, while other systems try to predict them based on historical data of user itineraries. Additionally, although the tourist is interested in visiting as many POIs as possible according to her/his preferences, she/he may wish to avoid visiting too many POIs that have similar characteristics (e.g., restaurants, art galleries). Similarly to [9, 8], this is implemented in the proposed system by enforced limits on the minimum and maximum number of POIs that the tourist can visit from each category. Furthermore, the proposed system offers the possibility to the tourist to select a set of mandatory POIs that should be included in her/his itinerary, by selecting appropriate values on the user defined category constraints. The schema of the proposed system architecture is outlined in Fig. 1(a).

Figure 1 depicts an instance of a personalized tour itinerary, where the user starts at point 2 and ends at point 8. In this example, a solution of the proposed framework is plotted with red color, that consists of five POIs (2, 15, 5, 14, 6) which provide high user satisfaction. The category of each POI is shown in parenthesis, while the minimum and maximum limits per category are depicted on the bottom left of Fig. 1(b). It holds that six out of eight user defined POI category



(a) (b)

	POI	Time-Start	Time-End	Duration
1	2	10.00	10.00	0
2	2	10.00	10.30	30
3	15	10.48	11.18	30
4	5	11.23	12.23	60
5	14	12.32	13.02	30
6	6	13.11	13.41	30
7	8	13.46	13.46	0

(c)

Figure 1: (a) The schema of the proposed framework. **(b),(c)** An example of a personalized itinerary on a 2D map with 16 POIs. The proposed personalized itinerary consists of three visited POIs (2, 15, 5, 14, 6). The tour starts at point 2 at 10:00 and should end at point 8 before 14:00 (user time budget: 10:00-14:00). **(b)** A map of 16 POIs, where each POI is drawn by a circle. The size and the color of a POI correspond to the duration of the visit and the gained user satisfaction, respectively. The category of each POI is shown in parenthesis. The itinerary is indicated with a red line. **(c)** The timetable of the personalized itinerary.

constraints (c2,c3,c4,c5,c6 and c7) are also satisfied. The size and the color of a POI correspond to the duration of the visit and the gained user satisfaction, respectively. Additionally, all graph edges are assigned a travelling time. According to the proposed timetable (see Fig. 1(c)), the tour start at 10:00 and ends at 13:46. Therefore, it holds that the selected tour itinerary passes from POIs that provide high user satisfaction, while it respects the user time budget (10:00 to 14:00).

The main contribution of this work concerns the problem formulation of PIR with POI categories based on the maximization of an appropriate objective function, as well as a proposed high performance and computationally efficient deterministic method. Another significant contribution of the proposed system is its ability to control the trade off between the exploration of the search space and its computational cost by changing the number of collaborating system instances. For each visited POI, the proposed objective function takes into account the user satisfaction, POI's visit duration and its category constraints, as well as the number of already visited POIs, in order to get higher values as the number of POIs increases. In our work, the gained user satisfaction is also related to the POI visit duration making more realistic the proposed objective function. Moreover, we show the high performance and computational efficiency of the proposed framework that is based on an EM schema. Another contribution of the proposed method concerns its applicability, as it can be easily combined with any recommender system (see Figure 1(a) and the the integration of the proposed system to a tourist trip design system of Section 4.3). An additional contribution of this work concerns the creation of a large synthetic public dataset used to test the Personalized Itinerary Recommendation systems under different values of the problem parameters.

The remainder of this paper is organized as follows: Section 2 reviews the related work for the itinerary recommendation problem. In Section 3, we present the main problem formulation of itinerary recommendation that we study in this paper. Section 4 describes the proposed itinerary recommendation method. Section 5 describes the experimental setup along with the obtained results. Finally, conclusions and future research directions are provided in Section 6.

2. Related work

In recent years, with the popularity and explosive growth of location-based social networks and smartphones, demographics, user preferences, and space-time information and ratings of itineraries for POIs visited by tourists, are easily collected providing rich datasets that can be used to infer user interests. This huge information collection has been exploited by PIR systems, thus improving the quality of personalized recommendations. Therefore, a large number of different techniques appear in the literature for itinerary recommendation [6, 10]. Many prior studies formulated the itinerary recommendation as a variant of the Orienteering Problem (OP) [7] or the Travelling Salesman Problem (TSP) [11, 12] with multiple constraints, and subsequently solved them using optimization techniques to obtain the recommended itineraries [13, 14]. These methods generally failed to incorporate personalization into itineraries of individual users. In personalization-based approaches, the main challenges are:

1. implicitly inferring the interest preferences of tourists and
2. incorporating these interests as part of the recommended tour itinerary [6].

TripBuilder [15], is an unsupervised framework for planning personalized sightseeing tours in cities based on categorized POIs from Wikipedia and albums of geo-referenced photos from Flickr. It aims to plan a tour comprising POIs that maximize tourists' personal interests while adhering to a specific visiting time budget. The PersTour algorithm [16] considers both POIs' popularity and user preferences to recommend suitable POIs to visit and the amount of time to spend at each POI. PersTour personalize a POI's visit duration based on the relative interest of individual users, instead of relying on the average visit duration of a POI for all users. PersTour introduces two adaptive weighting methods to automatically determine the emphasis on a POI's popularity and the user interest preferences.

The method proposed in [17] recommends emotionally pleasing tours in a city. To quantify the extent to which urban locations are pleasant, data from a crowd-sourcing platform have been used. The construction of the best itinerary from source to destination is performed in four steps:

1. Identify M shortest paths between source and destination using Eppstein's algorithm.
2. Compute the average rank for all locations in each of the first m ($m < M$) paths. At each exploration, the path with the lowest (best) average rank is stored.
3. Terminate when the average rank drops below a threshold.
4. Select the path with the best rank found.

In [18] a Genetic Algorithm (GA) has been proposed to provide a travel plan consisting of a set of high-ranked tourist attractions and restaurants with respect to several constraints. GA uses natural selection and genetic principles to solve the optimization problem of itinerary recommendation. It uses multistage processing such as initialization, selection, crossover, and mutation to generate and refine the candidate solution.

AGAM [19] is another genetic algorithm with crossover and mutation probabilities for this problem. In this algorithm, different weights are allocated to every factor to generate a PIR for better results that meets many kinds of tourists' preferences. In the performance evaluation section, the experimental results of the proposed method are compared to [17] and [18]. UTP [20] recommends interesting locations in the itineraries that similar tourists have traveled to before, based on a collaborative filtering algorithm with time preferences. The DCC-PersIRE method [10] solves the PIR problem by integrating user-POI visits, POI textual information and POI categories in order to predict user interests and duration of visits. Finally, an iterative local search based algorithm has been proposed to solve the PIR problem. In a more recent work, the PWP algorithm [21] recommends multiple itineraries based on the interests of visitors, the popularity and the cost of itineraries. PWP effectively optimizes interest, popularity and cost during the selection of each itinerary using the NSGA-II approach via genetic operators. An itinerary list is generated by comparing local and global tourists.

Recently, extra practical tourism constraints have been included in the tourist trip design problem such as mandatory visits, limits on the number of locations of each category, as well as in which order selected locations are visited [8]. In [8], four methods are proposed based on the branch-and-check approach to solve the classical itinerary problem with extra practical tourism constraints and POIs categories. The master problem selects a subset of locations, verifying all except time-related constraints. These locations define candidate solutions to the master problem. For each candidate solution, the sub-problem checks whether a feasible trip can be built using the given locations.

Group itinerary recommendation methods provide itineraries with a balance between group preferences and the given temporal and spatial constraints. AMT-IRE [22] is designed to schedule visits to POIs of interest based on the overall group preferences provided in the form of a sequence with time constraints. The proposed AMT model jointly calculates group member preferences and overall group preferences via the attention mechanism. The predicted overall group preferences are used in a variant of the orienteering problem and an iterated local search-based algorithm recommends group itineraries. Another group itinerary recommendation methods is proposed in [23], that receives a set of must-visit and preferred points of interest from each tourist and forms multi-day tours that cover all must-visit points. The proposed framework attempts to maximize fairness among group members. The problem of next POI recommendation considers the sequential information of users' check-ins in addition to users' preferences. In [24], the Spatio-Temporal Gated Network has been proposed to model personalized sequential patterns for users' long and short term preferences in the next POI recommendation. In [25], the proposed system, that is also based on a neural network architecture, has been applied to recommend the next personalized travel destinations to airlines' customers.

3. Problem definition

In this section, we set the scene of the various aspects of the problem that this paper addresses, and simultaneously we present the stepping-stones where our developments are based on. We start below by defining the personalised itinerary recommendation problem.

First, we define preliminaries concerning the input of our approach. We assume a graph

Symbols	Definitions
$P = \{p_1, ..., p_n\}$	The set of n POIs in the given Map
p_1/p_n	The starting/ending locations
st	The starting time of tour
B	The time budget
T	Traveling times matrix ($n \times n$) of the pair-wise distances for all pairs POIs
C	Set of POIs categories
N_g^{min}/N_g^{max}	The minimum/maximum preferable number of POIs belonging to category g
d_i	Visit duration of POI p_i
o_i	Opening time window of POI p_i
s_i	Gained user satisfaction per hour by visiting POI p_i
at_i / dt_i	The arrival/departure time at POI p_i
c	The itinerary that consists of triples (p_i, at_i, dt_i)
$v(c) \subseteq c$	The set of visited POIs of itinerary c
$F(c)$	Objective function
$\overline{F}(c)$	Expected value of the objective function F(c)

Table 1
Summary of the notation used throughout this work.

(e.g. city map) with n POIs $P = \{p_1, ..., p_n\}$. Let T be the traveling time matrix ($n \times n$) of the pair-wise distances for all POI[1] pairs. Let C be the set of POI categories (e.g. restaurant, museum, beach, shops etc.). For each category $g \in C$, the minimum (N_g^{min}) and the maximum (N_g^{max}) preferable number of POIs belonging to category g, according to user preferences is also given. Additionally, for each POI p_i the visit duration d_i and the opening time window o_i is known. Without loss of generality, we can assume that p_1 and p_n are the given starting and ending locations (POIs) of the tour. Hereafter, for simplicity reasons, we will assume that $p_1 \neq p_n$, however, our method is able to work even if the starting and ending locations coincide ($p_1 = p_n$).

According to the problem definition, the user gives the starting time of the tour itinerary st and the time budget (duration) B of the tour. This means that the tour itinerary should end at $st + B$ or earlier. s_i defines the gained user satisfaction per hour by the visit of POI p_i. In our framework, s_i is computed offline e.g. by a recommender system based on user preferences or other features (travel, history, etc.) as depicted in Figure 1(a).

In the definition of itinerary c, we have included the visited POIs as well as the corresponding temporal information. Therefore, an itinerary c is defined by a sequence of triples, where each triple (p_i, at_i, dt_i) is comprised by the visited POI p_i with the corresponding arrival at_i and departure times dt_i. The case that a user can pass from a POI without visiting it, is also supported, which means that $at_i = dt_i$. Thus, we denote by $v(c)$, the sequence of visited triples (p_i, at_i, dt_i) of itinerary c, for which it holds that $dt_i > at_i$. Therefore, according to the itinerary recommendation problem definition, itinerary c should meet the following constrains:

1. $\forall p_i : p_i \in v(c)$ it holds that $[at_i, dt_i] \subseteq o_i$. This means that the corresponding POI should be open during the visit.
2. $\forall p_i : p_i \in v(c), c \geq 2$ it holds that the arrival time at_i is given by $at_i = dt_{i-1} + T_{p_{i-1},p_i}$.

[1] T can be computed by applied Johnson's algorithm on the graph of POIs in $O(n^2)$, under the assumption that the number of graph edges is $O(n)$ which is usually true for city maps.

3. The itinerary should start at POI p_1, meaning that the first triple of c should be $c(1) = (p_1, at_1, dt_1)$.
4. $at_1 = st$, meaning that the tour itinerary starting time is the same with the arrival time at p_1.
5. The itinerary should end at p_n POI, meaning that the last triple of c should be the $c(|c|) = (p_n, at_n, dt_n)$.
6. $dt_n \leq st + B$, meaning that the tour itinerary ends at time $st + B$ or earlier.

Additionally, the number of categories $g \in C$ satisfying the following condition should be maximized:

$$N_g^{min} \leq \sum_{cat(c(i))=g} 1 \leq N_g^{max}, \tag{1}$$

where $cat(c(i))$ is the category of $c(i)$. Table 1 summarizes the notation used throughout this work.

3.1. Evaluating an itinerary

Solving the itinerary recommendation problem amounts to finding the legal (i.e. satisfying the pre-mentioned problem constrains) itinerary c^* that maximizes an appropriately defined objective function F. In notation,

$$c^* = argmax_{c \in LS} \;\; F(c), \tag{2}$$

where LS is the set of legal itineraries according to the problem constrains.

In order to assess this itinerary, we propose an objective function F that has the following properties:

- The main goal of the objective function is to achieve the highest user satisfaction, while respecting the given problem constraints.
- For each category ($g \in C$), we take into account the corresponding constraint (see Eq. 1), so that the largest number of constraints satisfied, the more preferable an itinerary c.
- For each POI (p_i) of c, we take into account the corresponding gained user satisfaction per hour that is multiplied by the visit duration $dt_i - at_i$. Intuitively, the larger gained satisfaction, the more preferable the itinerary c.
- The number of visited points $|v(c)|$ also increases the value of the objective function.
- The value of the objective function for legal itineraries is non-negative.
- The value of the objective function for non legal itineraries is set to $-\infty$.

The aforementioned properties are well captured by the objective function $F(c) \leq 1$ defined as follows:

$$f_g(c) = \begin{cases} \frac{\sum_{cat(c(i))=g} 1}{N_g^{min}}, & \text{if } \sum_{cat(c(i))=g} 1 < N_g^{min} \\ 1, & \text{if } N_g^{min} \leq \sum_{cat(c(i))=g} 1 \leq N_g^{max} \\ \frac{N_g^{max}}{\sum_{cat(c(i))=g} 1}, & \text{if } \sum_{cat(c(i))=g} 1 > N_g^{max} \end{cases} \tag{3}$$

$$Fc(c) = \sum_{g \in C} f_g(c) \tag{4}$$

$$Fs(c) = \frac{(1 + log(|v(c)|)) \cdot \sum_{(p_i, at_i, dt_i) \in c} s_i \cdot (dt_i - at_i)}{(1 + log(n)) \cdot B} \tag{5}$$

$$F(c) = \begin{cases} \frac{Fc(c)+Fs(c)}{|C|+1} & \text{if } c \in LS \\ -\infty & \text{if } c \notin LS \end{cases} \tag{6}$$

When c is a legal itinerary, $F(c)$ is defined by the sum of $fs(c)$ and $fc(c)$.

- $Fs(c)$ sums the gained user satisfaction multiplied by the corresponding visit duration (see Eq. 5). The term $\frac{(1+log(|v(c)|))}{1+log(n)} \leq 1$ is used to slightly increase the value of the objective function as the the number of visited points $|v(c)|$ increases.
- $Fc(c)$ captures the satisfied categories' constraints by counting the number of categories satisfying the corresponding constraints (second branch of $f_g(c)$). For those categories that don't satisfy the categories' constraints, it holds that
  - $\sum_{cat(c(i))=g} 1 < N_g^{min}$ (number of categories with POIs less than N_g^{min}) or
  - $\sum_{cat(c(i))=g} 1 > N_g^{max}$ (number of categories with POIs more than N_g^{max}),

  two terms are included. Both terms increase as $\sum_{cat(c(i))=g} 1$ approaches the corresponding category limit N_g^{min} or N_g^{max} (see Eq. 3).

In our implementation, an itinerary that satisfies more POIs categories constraints ($Fc(c)$) is more preferable, even if it yields less gained user satisfaction ($Fs(c)$). Therefore, $F(c)$ is influenced by the function $Fc(c)$ rather than the $Fc(c)$. This is also verified by the fact that $Fc(c) \leq |C|$ and $Fs(c) \leq 1$.

According to the proposed methodology, the following function $\overline{F}(c)$ that expresses the expected value of the objective function $F(c)$ is maximized. $\overline{F}(c)$ is defined by the sum the of the excepted values of $Fc(c)$ and $Fs(c)$ ($\overline{Fc}(c) + \overline{Fs}(c)$, see Eq. 12). $\overline{Fs}(c)$ gives the expected value of $Fs(c)$ under the assumption that the value of $Fs(c)$ linearly increases with the duration of itinerary c (see Eq. 10).

$\overline{Fc}(c)$ gives the expected value of $Fc(c)$. The expected value of $Fc(c)$ is only computed by the categories $g \in C$ that satisfy constraint $\sum_{cat(c(i))=g} 1 < N_g^{min}$ (term $C_1(c)$ of Eq. 7), since extra included POIs from these categories, are only expected to increase the value of $Fc(.)$. The expected value of $C_1(c)$ is computed under the assumption that it linearly increases with the duration of itinerary c respecting the upper limit: $C_1(c) \leq C_1^{max}(c)$ (see Eq. 10). $C_1^{max}(c)$ is equal to the number of categories that satisfy the constraint $\sum_{cat(c(i))=g} 1 < N_g^{min}$ (see Eq.9).

$$C_1(c) = \sum_{g \in C : \sum_{cat(c(i))=g} < N_g^{min}} f_g(c) \tag{7}$$

$$C_1^{max}(c) = \sum_{g \in C : \sum_{cat(c(i))=g} < N_g^{min}} 1 \tag{8}$$

$$\overline{Fc}(c) = Fc(c) - \frac{C_1(c)}{|C|+1} + \frac{min\{C_1^{max}(c), \phi \cdot C_1(c)\}}{|C|+1} \tag{9}$$

$$\overline{Fs}(c) = \phi \cdot fs(c) \tag{10}$$

$$\phi = \frac{B}{dt_n - at_1} \tag{11}$$

$$\overline{F}(c) = \overline{Fc}(c) + \overline{Fs}(c) \tag{12}$$

In this formulation, the total duration of itinerary c is given by the difference $dt_n - at_1$.

The computational cost of the exhaustive method for determining the optimal itinerary by maximizing the objective function defined in Eq. 6 is $O(n \cdot (n-2)! \cdot 2^{n-2}) = O((n-1)! \cdot 2^n)$, since there exist 2^{n-2} different itineraries in a map of $n-2$ POIs (assuming the first and last POIs are given). Factor $(n-2)!$ results from the number of permutations, since the order of POIs should also be considered. The evaluation of the objective function that costs $O(|v(c)|) = O(n)$, is included in term $(n-1)!$. This is too costly. Hereafter, we capitalize on the properties and the structure of the problem to propose an algorithm that provides an almost optimal solution in polynomial time.

4. Personalized itinerary recommendation

4.1. PIREM algorithm

Based on the problem formulation and constraints presented in Section 3, we now present *PIREM* algorithm, for solving the problem of PIR based on EM.

The PIREM algorithm: In this work, we propose an iterative optimization method, described in Algorithm 1, that sub-optimally solves the problem in $O(n \cdot B^3)$, where n denotes the number of given POIs, under the assumption that itinerary length increases linearly with B, as shown in our experimental results. According to the proposed method, the itinerary recommendation problem is solved by sequentially adding the most suitable unvisited POI in the current itinerary, the one that maximizes the expected value of the objective function, as this is defined in Eq. (12). Due to the proposed EM based method, a short in time duration itinerary is more promising and it is preferred to be selected as optimal itinerary to be extended, than a long in time duration one with similar values on the objective function.

The input to the proposed method is variables $P, st, B, T, C, d_i, o_i, s_i, i \in \{1, ..., n\}, N_g^{min}, N_g^{max}, g \in C$ as described in Section 3. The goal of the proposed method is to compute a solution for the PIR problem that is denoted by c^* in Algorithm 1. The first four lines of Algorithm 1 is the initialization phase. Variable nc that counts the number of changes in each main iteration of the method is set to zero, as well as the current optimal value of the objective function $\overline{FB}$. The set S of the indexes of visited POI is initialized to the empty set, while the first triplet of c^* is set equal to $\{(p_1, st, st)\}$, according to the problem definition.

In the main loop of the proposed *PIREM* method (lines 5-26 of Algorithm 1), we get the set of the indexes of all unvisited POIs U (line 6 of algorithm 1), that will be used to find the next visited POI. Additionally, we set $nc = 0$. In the computation of U, we ignore the ending POI p_n ($U = \{1, ..., n-1\} - S$), since this is definitely inserted after the last visited POI c^* ($c^*(|c|)$) at the

input : $P, st, B, T, C, d_i, o_i, s_i, i \in \{1, ..., n\}, N_g^{min}, N_g^{max}, g \in C$.
output: c^*.

1 $nc = 0$
2 $S = \emptyset$
3 $\overline{FB} = 0$
4 $c^* = \{(p_1, st, st)\}$
5 **repeat**
6 $\quad U = \{1, ..., n-1\} - S$
7 $\quad nc = 0$
8 $\quad$ **foreach** $k \in U$ **do**
9 $\quad\quad$ **for** $m = 1$ **to** $|c^*| + 1$ **do**
10 $\quad\quad\quad c = c^*.addPOI(p_k, m)$
11 $\quad\quad\quad$ **if** c *is valid* **then**
12 $\quad\quad\quad\quad \overline{f} = \overline{F}(c)$
13 $\quad\quad\quad\quad$ **if** $\overline{f} > \overline{FB}$ **then**
14 $\quad\quad\quad\quad\quad \overline{FB} = \overline{f}$
15 $\quad\quad\quad\quad\quad cb = c$
16 $\quad\quad\quad\quad\quad k^* = k$
17 $\quad\quad\quad\quad\quad nc = nc + 1$
18 $\quad\quad\quad\quad$ **end**
19 $\quad\quad\quad$ **end**
20 $\quad\quad$ **end**
21 $\quad$ **end**
22 $\quad$ **if** $nc > 0$ **then**
23 $\quad\quad c^* = cb$
24 $\quad\quad S = S \cup k^*$
25 $\quad$ **end**
26 **until** $nc \neq 0 \vee U = \emptyset$;
27 $(p_j, at_j, dt_j) = c^*(|c|)$
28 **if** $dt_j + T(j, n) + d_n \leq B + st$ **then**
29 $\quad c^* = c^* \cup \{(p_n, dt_j + T(j, n), dt_j + T(j, n) + d_n)\}$
30 **else**
31 $\quad c^* = c^* \cup$
32 $\quad\quad \{(p_n, dt_j + T(j, n), dt_j + T(j, n))\}$
33 **end**

Algorithm 1: The proposed iterative optimization method solving the PIR problem based on EM (*PIREM*).

end of the method (lines 27-32 of algorithm 1). This loop terminates when no changes take place in the main loop or the set U is empty.

Subsequently, in the second loop (lines 8-21 of Algorithm 1), we evaluate whether the insertion of each unvisited POI p_k, $k \in U$ at the position m of the current optimal itinerary c^* (see the third for loop of line 9) is legal according to the problem constraints and whether it improves the current optimal value of $\overline{FB}$ (expected value of the objective function). If both of the following statements are true, it means that the insertion the insertion of p_k at position m of c is valid (see lines 10-11 of Algorithm 1):

1. All visited POIs of c are opened.
2. Tour c ends at time $st + B$ or earlier.

Finally, we check if the insertion of p_k improves the current optimal value $\overline{FB}$ and we update $\overline{FB}$ (see lines 13-18). The current optimal itinerary c^* and set S are updated in this loop (see lines 22-25), so that the most suitable POI is inserted at the most suitable position of c^*.

4.2. M-PIREM algorithm

The resulting solution of *PIREM* may land on a local minima of the objective function due to the sequential optimization. Thus, we propose an extended version with multiple (*M*) collaborating instances of *PIREM*, called *M-PIREM* to improve the *PIREM* solution via a better exploration of the search space. Parameter *M* controls the trade off between search space exploration and computational cost, by changing the number of collaborating instances. Hereafter, the *M-PIREM* algorithm is presented.

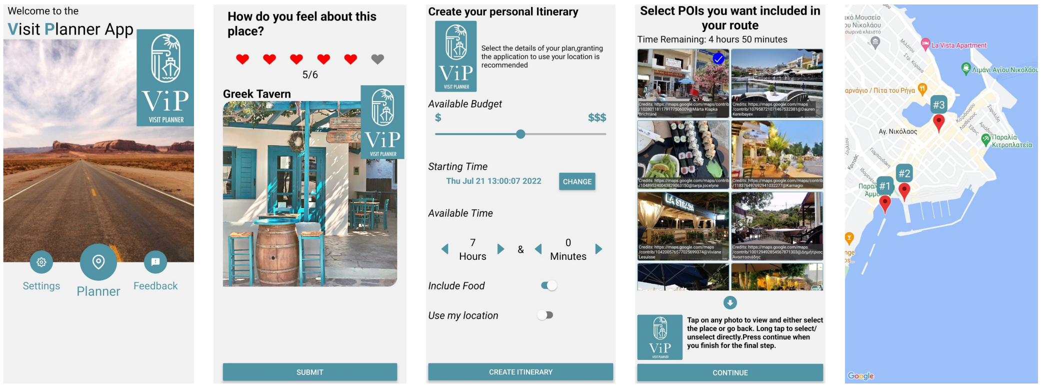
1. Let *H* be the set of *M* collaborating itineraries (instances of *PIREM*). In the initialization step, the set of visited POIs of the *M* itinerraries *H*, $H\{i\}, i \in \{1, ..., M\}$ is set to the empty set, while the first triplet of each itinerary is set equal to $\{(p_1, st, st)\}$, according to the problem definition.
2. In the main loop, we derive itinerary c^- from set *H* with the lowest expected value $\overline{FB}$ ($c^- = argmin_{c \in H} FB(c)$). Then, we apply an iteration of the main loop of the *PIREM* method, to get an new valid itinerary (c^+) that does not exist in *H*[2].
3. Subsequently, set *H* is updated by replacing itinerary (c^-), having the lowest expected value, by the new one (c^+). This process is repeated until the expected value of the objective function cannot be further improved.

Improving the lowest expected value (steps 2 and 3) increases the diversity of the possible solutions in *H*, in order to better avoid local minima. The computational cost of this method is $O(M \cdot n \cdot B^3)$. It holds that the solutions provided by *M-PIREM* are better or equivalent to the corresponding solutions of *PIREM*. In our experiments, we used $M = 32$.

4.3. Integration with a tourist trip design system

The proposed system is integrated with the Visit Planner App (Fig. 2(a)) that is a complete tourist trip design system (mobile app). According to Visit Planner App, the tourist gives her/his preferences by providing ratings on several POIs (Fig. 2(b)), so that a recommender system ([1]) is able to predict her/his preferences on the whole set of POIs. Then, the tourist provides some parameters for the trip e.g. starting time, budget etc. (see Fig. 2(c)) and the system will be able to create a trip according to the proposed objective function. For an even better user satisfaction, the visitor is able to change/select the POIs to be included in the itinerary among a list of the top-20 highest personally recommended POIs (Fig. 2(d)), while the proposed system will provide the best route that passes from the selected POIs, taking also account PIR constraints (e.g. opening hours, etc.) (Fig. 2(e)). The current beta version of Visit Planner App has been applied on the Municipality of Agios Nikolaos, Crete. It is available online at Google Play (https://play.google.com/store/apps/details?id=com.netmechanics.vip).

[2]If the case itinerary c^- is not a new itinerary, that does not exist in set *H*, we select the itinerary from *H* with the second lowest expected value $\overline{FB}$ and so on.



(a) (b) (c) (d) (e)

Figure 2: Screenshots of the Visit Planner App that is based on the proposed system.

5. Experimental evaluation

In this section, we describe the experiments we conducted using several frameworks and datasets.

5.1. Synthetic and Real Datasets

For our experiments, we have created 1024 different experimental setups on 64 synthetic datasets (16 different experiments per dataset) to study the performance and computational efficiency of the proposed methods and other methods from the literature under several problem parameters. Our intention is to provide a high number of random experimental setups that are realistic concerning the default parameter values in order to be able to fairly compare all methods under almost real conditions, different scenarios and scales. Hereafter, we present the proposed experimental parameter settings.

Each of the 64 synthetic datasets is generated by adding n POIs at random positions on a 2D-map, where $n \in \{32, 64, 96, 128\}$. The roads (edges) of each map are generated as follows, we sequentially connect the closest POIs according to the following rule: An edge is created if the distance between its middle point and the rest edges exceeds a predefined threshold in order not to create edges that are very close to each other. This is also almost true on real maps. In order to create the 64 synthetic datasets, we have created 16 maps for every value of n, following the aforementioned procedure. Subsequently, we set the parameters for each POI p_i of each synthetic dataset. Parameters d_i and o_i are selected randomly from $\{0.25, 0.5, 0.75, 1\}$ and {[9:00, 24:00], [12:00, 21:00] , [9:00, 14:00], [14:00, 24:00], [9:00, 14:00] ∪ [17:00, 21:00]}, respectively. For each synthetic dataset, we create 16 different experimental setups by randomly selecting the starting and ending locations of the tour from the available POIs. For each setup, we set the starting time of tour at 9:00 (st =9:00), while the time budget B is randomly selected from $\{5, 6, 7, 8, 9\}$. The value of parameter s_i is randomly selected in $[0, 1]$. An example of the synthetic datasets with $n = 16$ is illustrated in Fig. 1.

Additionally, in order to test our method with real data, we used three real datasets from Vienna, Budapest and Delhi cities presented in [16]. Vienna, Budapest and Delhi datasets comprise a set of users and their visits to $n = 28$, $n = 38$ and $n = 23$ POIs, respectively. The real datasets comprise a set of users and their visits to POIs naturally clustered into 6-8 different POI categories based on geo-tagged YFCC100M Flickr photos. For the real dataset, we create 256 different experimental setups following the same procedure applied on the synthetic datasets (see previous paragraph). The value of parameter s_i of a POI is given by the ratio of the POI visits according to the data provided by [16].

In synthetic datasets, we used 8 different POI categories, thus, the category of each POI of a synthetic dataset is randomly selected from a predefined set of eight values. Additionally, we used four different strategies for the N_g^{min} and N_g^{max} parameter setting. Therefore, the experimental setups of each synthetic and real dataset, are equally divided into the following four classes determined by parameters N_g^{min} and N_g^{max} capturing different realistic conditions, and extended the two different setups (Tight and Flexible) proposed in [9].

1. *Tight*: For each $g \in C$, it holds that N_g^{min} is randomly selected from the set $\{0, 1, 2\}$ and $N_g^{max} = N_g^{min}$. This class simulates tight cases, where the user want to visit a specific number of POIs according to their categories.
2. *Semi-flexible*: For each $g \in C$, it holds that N_g^{min} is randomly selected from the set $\{0, 1, 2\}$ and $N_g^{max} = max\{1, N_g^{min}\}$. This class simulates semi-flexible cases, where the user want to visit specific number of POIs according to their categories with the flexibility that each category can be visited $N_g^{max} \geq 1$ and $0 \leq N_g^{max} - N_g^{min} \leq 1$ times.
3. *Flexible*: For each $g \in C$, it holds that N_g^{min} is randomly selected from the set $\{0, 1, 2\}$ and $N_g^{max} = N_g^{min} + r$, where r is randomly selected from the set $\{1, 2, 3\}$. This class simulates flexible cases, where it holds that $1 \leq N_g^{max} - N_g^{min} \leq 3$.
4. *No Categories*: $N_g^{min} = 0$ and $N_g^{max} = \infty$. This class simulates cases without POIs categories constraints.

5.2. Baseline algorithms

In our experiments, we have included the proposed methods *PIREM* and *M-PIREM* as described in Section 4. In order to show the importance of the EM criterion, we have implemented a variant of the proposed *PIREM* method that maximizes the value of objective function $F(c)$ instead of $\overline{F}(c)$. This variant is called *PIRM*.

Moreover, to evaluate the performance of the proposed methods, we compared it against the following PIR methods [17, 18]. Both methods are described in Section 2. Hereafter, the method proposed in [17] that is based on shortest paths is called *SPM* and the genetic algorithm proposed in [18] is called *GA*. Both methods have been modified to maximize the proposed objective function $F(c)$.

The itineraries provided by the aforementioned methods are evaluated according to the objective function $F(c)$ that measures the quality (user satisfaction and POIs categories constraints satisfaction) of the recommended itineraries according to the problem definition (see Section 3). Moreover, we have evaluated the computational efficiency of the various methods by measuring

	n				Class of POI category constraints				
Method	32	64	96	128	Tight	Semi-flexible	Flexible	No Categories	Average
M-PIREM	**0.889**	**0.910**	**0.910**	**0.908**	**0.891**	**0.899**	**0.897**	**0.929**	**0.904**
PIREM	0.870	0.890	0.888	0.886	0.865	0.872	0.870	0.926	0.883
PIRM	0.792	0.769	0.734	0.732	0.697	0.711	0.703	0.916	0.757
SPM	0.760	0.790	0.800	0.804	0.719	0.756	0.761	0.920	0.789
GA	0.862	0.877	0.870	0.862	0.815	0.866	0.869	0.921	0.868

Table 2

The average values of objective function F for the five methods on the synthetic datasets for different values of n and class of POIs categories constraints.

		Precision					Objective Function				
Dataset	Method	Tight	Semi flexible	Flexible	No Cat.	Average	Tight	Semi flexible	Flexible	No Cat.	Average
Vienna	M-PIREM	**0.828**	**0.922**	**0.891**	**0.969**	**0.902**	**0.664**	**0.701**	**0.708**	**0.917**	**0.748**
	PIREM	0.188	0.172	0.188	0.500	0.262	0.637	0.669	0.682	0.916	0.726
	PIRM	0.156	0.141	0.078	0.016	0.098	0.596	0.650	0.643	0.913	0.700
	SPM	0.016	0.094	0.078	0.016	0.051	0.565	0.606	0.598	0.907	0.669
	GA	0.172	0.109	0.125	0.016	0.105	0.638	0.674	0.690	0.913	0.729
Budapest	M-PIREM	**0.875**	**0.938**	**0.891**	**1.000**	**0.926**	**0.710**	**0.740**	**0.736**	**0.916**	**0.776**
	PIREM	0.219	0.094	0.234	0.422	0.242	0.708	0.726	0.726	0.916	0.769
	PIRM	0.094	0.000	0.000	0.000	0.023	0.588	0.628	0.628	0.908	0.688
	SPM	0.047	0.000	0.016	0.031	0.023	0.606	0.648	0.637	0.909	0.700
	GA	0.078	0.063	0.031	0.016	0.047	0.683	0.722	0.723	0.912	0.760
Delhi	M-PIREM	**0.844**	**0.922**	**0.844**	**0.984**	**0.898**	**0.602**	**0.618**	**0.602**	**0.892**	**0.679**
	PIREM	0.109	0.297	0.141	0.344	0.223	0.595	0.607	0.587	0.892	0.670
	PIRM	0.203	0.125	0.078	0.219	0.156	0.533	0.531	0.519	0.892	0.619
	SPM	0.328	0.344	0.375	0.313	0.340	0.588	0.594	0.579	0.892	0.663
	GA	0.313	0.391	0.406	0.219	0.332	0.587	0.607	0.592	0.892	0.669

Table 3

The average values of Precision (left part) and objective function F (right part) for the five methods on the Vienna, Budapest and Delhi datasets for different values of class of POIs categories constraints.

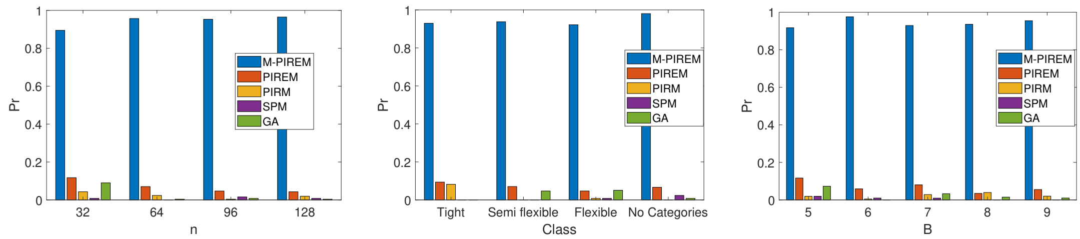
their execution times. All the analysis has been done using MATLAB 2020a on an Intel i7 core 3.20GHz with 32 GB RAM[3].

5.3. Comparisons

Table 2 presents the average values of objective function F for the five methods presented in Section 5.2 on the synthetic datasets for various values of n and class of POI category constraints. It holds that the proposed *M-PIREM* method clearly outperforms all methods under any map size and class of POI category constraints. *PIREM* also shows high performance results since it outperforms all other methods under any map size and class of POI category constraints. Next, it appears that good performance results are obtained by *GA*. Low performance results are obtained by *SPM* and *PIRM*. The methods' ranking obtained for each value of n and class of POIs categories constraints agree with the average results of the last column of Table 2.

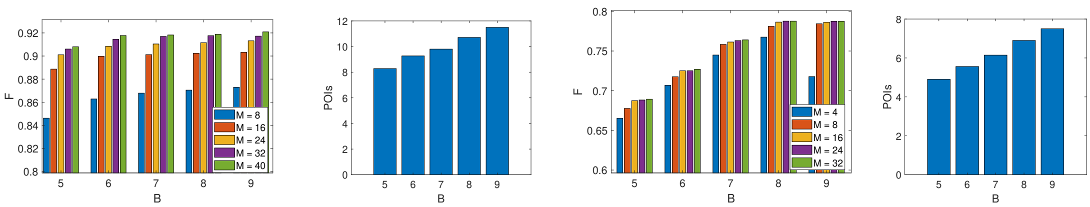
Figure 3 shows the average precision (pr) of each method on the synthetic datasets for different values of n (Fig. 3(a)), POI category constraints classes (Fig. 3(b)) and time budget B (Fig. 3(c)). For each method the precision is computed by the percentage of datasets for which the method yields the best itinerary according to the objective function $F(c)$ criterion over all methods. The

[3]The code implementing the proposed methods along with the synthetic datasets are publicly available online: https://sites.google.com/site/costaspanagiotakis/research/pirem.



(a) (b) (c)

Figure 3: The average Precision of each method on the synthetic datasets for different values of **(a)** n **(b)** POI category constraints classes, **(c)** time budget B.



(a) Synthetic datasets (b) Synthetic datasets (c) Vienna dataset (d) Vienna dataset

Figure 4: **(a), (c)** The average values of objective function F for the *M-PIREM* method under different values of M for the **(a)** synthetic and **(c)** Vienna dataset. **(b),(d)** The average number of POIs for different values of time budget B for the *M-PIREM* method (M = 32) for the **(b)** synthetic and **(d)** Vienna dataset.

results of Fig. 3, concerning the ranking of the methods, agree in most cases with the results of Table 2. Figure 3 shows that the proposed *M-PIREM* method clearly outperforms all the methods, having average precision more than 89.5% under any value of n, POI category constraint class and time budget. The average precision of *M-PIREM* and *PIREM* for all datasets is 94.3% and 6.9%, respectively. The average precision of any other method is less than 3%. According to the *pr* criterion, *M-PIREM* clearly outperforms all the other methods.

Table 3 presents the average values of precision *Pr* (left part) and objective function F (right part) for the five methods on the three real datasets (Vienna, Budapest and Delhi) for different values of class of POIs categories constraints. The results agree with the corresponding results on the synthetic datasets (see Table 2 and Fig. 3). It holds that the proposed *M-PIREM* method clearly outperforms all methods under dataset, criterion and class of POI category constraints. In some cases, *GA* slightly outperforms *PIREM*, because it exhaustively searches the solution space for simple problem instances (low values of n). Additionally, the outperformance of the proposed method *M-PIREM* compared to the rest systems slightly increases for more complex problem instances under real (see Budapest dataset on Table 3) and synthetic datasets (see $n = 128$ on Table 2).

5.4. Evaluation of the proposed methods

The importance of the EM criterion of the proposed schema is clearly shown in our experiments (see Fig. 3, Table 2 and 3). In all experiments performed, *M-PIREM* and *PIREM* clearly outperform the *PIRM* method. According to the results of Table 2, it holds that on average, the user satisfaction, as measured by the proposed objective function, is about 17% higher when the *EM* criterion is used.

Figure 4 presents several results for the proposed top performing method *M-PIREM* on the synthetic and Vienna datasets. Vienna dataset is selected since it better represents the real datasets according to dataset size (complexity) criterion. In Figs. 4(a) and 4(c), the average values of objective function F for the *M-PIREM* method under different values of M are depicted on the synthetic and Vienna datasets, respectively. As expected, the higher the value of M, the higher the obtained user satisfaction. For the synthetic datasets, when $M > 32$, the improvement of the obtained solutions is not so critical, thus the selection of parameter $M = 32$ offers a good balance for the trade off between search space exploration and computational cost, which increases linearly with M. The Vienna dataset is less complex ($n = 28$), thus when $M > 24$, the *M-PIREM* provides equivalent solutions. Figures 4(b) and 4(d) present the average number of POIs for different values of time budget B for the *M-PIREM* method (M = 32) for Vienna and synthetic datasets, respectively. As expected, the itinerary length increases linearly with B. For the synthetic datasets, the average itinerary length over all synthetic datasets is 9.9 with standard deviation $\sigma = 2.1$. For the less complex Vienna dataset, the average itinerary length over the 256 experiments is 6.1 with standard deviation $\sigma = 1.2$.

5.5. Computational efficiency

Due to the low computational cost of the proposed methods ($O(n \cdot B^3)$ and $O(M \cdot n \cdot B^3)$), they have higher computational efficiency compared to *SPM* and *GA*. Over all synthetic datasets, it holds that on average *M-PIREM* is about 5.5 and 140 times faster than *SPM* and *GA*, respectively. Concerning *PIREM*, it appears to be about 322 and 8120 times faster than *SPM* and *GA*, respectively. The average execution time over all synthetic datasets of *M-PIREM* with M = 32 is 3.2 sec. The corresponding average execution time of *PIREM* is only 0.06 sec.

6. Conclusions

In this work, the challenging problem of PIR with POI categories has been solved by a successive selection of POIs approach based on EM. We focus on the POIs sequence selection problem exploiting the personalized POI recommendations provided by a recommender system. More specifically, we propose the *PIREM* method that sequentially selects unvisited POIs taking into account user interests, user time budget and POI opening hours, spatial constraints and POIs categories. The proposed *M-PIREM* method with multiple collaborating instances improves the obtained results of *PIREM*. The number of instances parameter M of the *M-PIREM* method balances the trade of the search space exploration and the computational cost.

The proposed system has been successfully integrated with the Visit Planner App, a complete tourist trip design system. Additionally, it has been successfully applied on synthetic and real datasets providing high performance results by maximizing user satisfaction, respecting user defined POIs categories constraints and adhering to user time budget. We have performed 1024 experiments on 64 different synthetic maps of POIs and 256 experiments on three real datasets, where the problem parameters (user time budget, number of POIs, POIs opening hours and spatial constraints, POIs categories, etc.) vary. All the experiments demonstrate the proposed framework outperforms various state-of-the-art baselines on solution quality as well as computational efficiency. In the future work, we focus on the further development and the evaluation of the Visit Planner App. Additionally, we will study the problem of group itinerary recommendation [22], that can be defined as an extension of the PIR according to the proposed problem formulation.

Acknowledgments

This research has been co-financed by the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH - CREATE - INNOVATE B cycle (project code: T2EDK-03135).

References

[1] H. Papadakis, C. Panagiotakis, P. Fragopoulou, SCoR: A synthetic coordinate based system for recommendations, Expert Systems with Applications 79 (2017) 8–19.

[2] C. Panagiotakis, H. Papadakis, A. Papagrigoriou, P. Fragopoulou, Improving recommender systems via a dual training error based correction approach, Expert Systems with Applications 183 (2021) 115386.

[3] M. Elahi, F. Ricci, N. Rubens, A survey of active learning in collaborative filtering recommender systems, Computer Science Review 20 (2016) 29–50.

[4] L. Boratto, S. Carta, G. Fenu, R. Saia, Semantics-aware content-based recommender systems: Design and architecture guidelines, Neurocomputing 254 (2017) 79–85.

[5] M. E. B. H. Kbaier, H. Masri, S. Krichen, A personalized hybrid tourism recommender system, in: 2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA), Ieee, 2017, pp. 244–250.

[6] K. H. Lim, J. Chan, S. Karunasekera, C. Leckie, Tour recommendation and trip planning using location-based social media: a survey, Knowledge and Information Systems 60 (2019) 1247–1275. URL: https://doi.org/10.1007/s10115-018-1297-4. doi:10.1007/s10115-018-1297-4.

[7] P. Vansteenwegen, W. Souffriau, D. Van Oudheusden, The orienteering problem: A survey, European Journal of Operational Research 209 (2011) 1–10.

[8] D. M. Vu, Y. Kergosien, J. E. Mendoza, P. Desport, Branch-and-check approaches for the tourist trip design problem with rich constraints, Computers and Operations Research 138 (2022) 105566. URL: https://doi.org/10.1016/j.cor.2021.105566.

[9] A. Expósito, S. Mancini, J. Brito, J. A. Moreno, A fuzzy GRASP for the tourist trip design with clustered POIs, Expert Systems with Applications 127 (2019) 210–227. doi:10.1016/j.eswa.2019.03.004.

[10] L. Chen, L. Zhang, S. Cao, Z. Wu, J. Cao, Personalized itinerary recommendation: Deep and collaborative learning with textual information, Expert Systems with Applications 144 (2020). doi:10.1016/j.eswa.2019.113070.
[11] S. Markaki, C. Panagiotakis, D. Lasthiotaki, Image sorting via a reduction in travelling salesman problem, IET Image Processing 14 (2020) 31–39.
[12] G. Gutin, A. P. Punnen, The traveling salesman problem and its variations, volume 12, Springer Science & Business Media, 2006.
[13] P. Vansteenwegen, W. Souffriau, G. V. Berghe, D. Van Oudheusden, Iterated local search for the team orienteering problem with time windows, Computers & Operations Research 36 (2009) 3281–3290.
[14] M. De Choudhury, M. Feldman, S. Amer-Yahia, N. Golbandi, R. Lempel, C. Yu, Automatic construction of travel itineraries using social breadcrumbs, in: Proceedings of the 21st ACM Conference on Hypertext and Hypermedia, 2010, pp. 35–44.
[15] I. R. Brilhante, J. A. Macedo, F. M. Nardini, R. Perego, C. Renso, On planning sightseeing tours with tripbuilder, Information Processing & Management 51 (2015) 1–15.
[16] K. H. Lim, J. Chan, C. Leckie, S. Karunasekera, Personalized trip recommendation for tourists based on user interests, points of interest visit durations and visit recency, Knowledge and Information Systems 54 (2018) 375–406. doi:10.1007/s10115-017-1056-y.
[17] D. Quercia, R. Schifanella, L. M. Aiello, The shortest path to happiness: Recommending beautiful, quiet, and happy routes in the city, in: Proceedings of the 25th ACM conference on Hypertext and social media, 2014, pp. 116–125.
[18] B. S. Wibowo, M. Handayani, A genetic algorithm for generating travel itinerary recommendation with restaurant selection, in: 2018 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), IEEE, 2018, pp. 427–431.
[19] P. Yochum, L. Chang, T. Gu, M. Zhu, An Adaptive Genetic Algorithm for Personalized Itinerary Planning, IEEE Access 8 (2020) 88147–88157. doi:10.1109/ACCESS.2020.2990916.
[20] Y. L. Hsueh, H. M. Huang, Personalized itinerary recommendation with time constraints using GPS datasets, Knowledge and Information Systems 60 (2019) 523–544. doi:10.1007/s10115-018-1217-7.
[21] J. L. Sarkar, A. Majumder, A new point-of-interest approach based on multi-itinerary recommendation engine, Expert Systems with Applications 181 (2021) 115026.
[22] L. Chen, J. Cao, H. Chen, W. Liang, H. Tao, G. Zhu, Attentive multi-task learning for group itinerary recommendation, Knowledge and Information Systems 63 (2021) 1687–1716.
[23] M. Kargar, Z. Lin, A socially motivating and environmentally friendly tour recommendation framework for tourist groups, Expert Systems with Applications 180 (2021) 115083. URL: https://doi.org/10.1016/j.eswa.2021.115083. doi:10.1016/j.eswa.2021.115083.
[24] P. Zhao, A. Luo, Y. Liu, F. Zhuang, J. Xu, Z. Li, V. S. Sheng, X. Zhou, Where to go next: A spatio-temporal gated network for next poi recommendation, IEEE Transactions on Knowledge and Data Engineering (2020).
[25] A. Dadoun, R. Troncy, M. Defoin-Platel, G. Solano, Predicting your next trip: A knowledge graph-based multi-task learning approach for travel destination recommendation, in: 2021 Workshop on Recommenders in Tourism, RecTour 2021, 2021, pp. 23–38.