A Delay-Robust Touristic Plan Recommendation Using Real-World Public Transportation Information

Victor A. Arrascue Ayala
*Department of Computer Science
University of Freiburg
Georges-Köhler-Allee 051, 79110
Freiburg, DE
arrascue@cs.uni-freiburg.de

Kemal Cagin Gülsen
*Department of Computer Science
University of Freiburg
Georges-Köhler-Allee 051, 79110
Freiburg, DE
guelsenk@cs.uni-freiburg.de

Marco Muñiz
+Department of Computer Science
Aalborg University
Fredrik Bajers Vej 5, DK-9220 Aalborg
East, DK
muniz@cs.aau.dk

Anas Alzogbi
*Department of Computer Science
University of Freiburg
Georges-Köhler-Allee 051, 79110
Freiburg, DE
alzoghba@cs.uni-freiburg.de

Michael Färber
*Department of Computer Science
University of Freiburg
Georges-Köhler-Allee 051, 79110
Freiburg, DE
michael.faerber@cs.uni-freiburg.de

Georg Lausen
*Department of Computer Science
University of Freiburg
Georges-Köhler-Allee 051, 79110
Freiburg, DE
lausen@cs.uni-freiburg.de

ABSTRACT

Tourism Recommender Systems (TRS) assist tourists in designing a plan for a soon-to-be visited city, which consists of a selection of relevant points-of-interest (POI), the order in which they will be visited, the start and end time of the visits, etc. These tools filter POIs based on the tourist's preferences and take into account time constraints, like the desired duration of the plan, or the POI's opening or closing times. However, being able to provide tourists with an additional travel plan which explains how to reach those POIs using public transportation is a feature in which TRSs come short. Existing solutions try to solve the problem in a simplified way and do not model all possible events involved in using public transportation, such as combining transfer times and trips, changing vehicles, or dealing with delays of transportation units. We therefore propose three novel approaches to generate visit plans and their corresponding travel plans, namely SILS, TRILS and PHILS, which overcome these weaknesses. These approaches generate visit plans by iteratively adjusting them according to the traveling information and differ in the way the adjustment is done. Our experiments on a real-world POI dataset and public transportation information of the city of Izmir show that our approaches outperform the state-of-the-art in terms of quality of recommendations. Moreover, they are also able to provide both visit and travel plans in real-time and are robust in case of delays. To the best of our knowledge previous approaches have not been able to achieve this level of practicality.

KEYWORDS

Tourism Recommender Systems, Tourist Trip Design Problem, Time-dependent Orienteering Problem with Time Windows, Iterated Local Search, Route Planner, Delays.

1 INTRODUCTION

Visiting touristic attractions, walking through historical places, or trying local food are among the main activities tourists undertake when they visit a new city or country. Given that the number of attractions is typically large and that tourists are also restricted to time and/or money budgets, they need to optimize and sometimes compromise on the selection of relevant attractions. In addition to this, they also have to figure out if the place is reachable and eventually find a feasible way to reach the location using public transportation. This process can be very cumbersome. Therefore, Tourism Recommender Systems (TRS) have been developed for assisting tourists in planning their trips.

In the literature the problem of generating a visit plan (no public transportation involved) is known as the *tourist trip design problem (TTDP)* [10], while the attractions are referred to as points-of-interest (POIs). This problem is formally defined as follows: given a set of POIs $p_1, ..., p_n$, each POI is associated with some *profit* s_i, which reflects the user's affinity towards this POI. The goal is to find $V = (p_i, p_j, ..., p_l, p_k)$, a sequence of POIs to be visited in a given order (i.e. a *visit plan*) which maximizes the collected profits taking into account the following time constraints: (1) the user's specified time budget, (2) the availability of POIs (e.g. opening hours), and (3) the travel time required to move from one POI to the next one.

In this paper we aim at additionally integrating public transportation information to assist the user in traveling from one attraction to the next one. This feature is indeed perceived by tourists as one of the most useful functionalities [6]. However, the biggest challenge when tackling the TTDP problem including also public transportation information is the increased level of difficulty of constraint (3), i.e. the fact that travel times can significantly vary depending on the situation, e.g. the timetables, the time required to reach the closest station by foot, etc. This effect is known as *time-dependency*. Formally, in addition to the visit plan V, we would like to generate $T = (t_{(i,j)}, ..., t_{(l,k)})$, the sequence of travel plans, each indicating how to move from one attraction to the next one, e.g. from p_i to p_j, according to V using public transportation. Examples of those instructions could be how to reach the departure station from an attraction by foot, at what time to leave the attraction, in which station to get out from a bus/tram, when to change service type, etc. Moreover, both plans V and T should be provided in real-time (time response below 5 secs).

Surprisingly, integrating public transportation information is still quite unexplored [10]. This is likely due to the associated computational challenges and the need for real-time data. Note that a TRS of this kind needs to consider (i) the public transportation system's status itself (e.g. buses and trams may have delays) as well as (ii) the point in time when the user departs to the next attraction[1] (e.g. if a bus cannot be reached any more, the user needs to wait, which stretches the travel time and which might lead to the fact that a planned attraction cannot be visited any more). The TTDP problem with public transportation becomes even more complicated when considering changing buses etc.

Existing solutions for TTDP, which integrate public transportation information (and therefore, with the time-dependency constraint) such as [6, 8, 11, 25] try to solve the problem in a simplified and therefore not realistic way. They can be grouped as follows: (1) Approaches that get rid of the time-dependency by considering a time-independent approximation of the problem, e.g. by using average travel times. However, trip plans computed with average travel times differ in practice significantly with respect to solutions with real travel times [25]. (2) Approaches that pre-compute all travel times for all possible pairs of POIs and times. However, pre-computing all possible travel times, considering all possible combinations of POIs and times is clearly feasible only for small transportation networks or small cities [6]. (3) Approaches that pre-compute travel times exploiting regularities in the schedules. However, the assumption of periodic service schedules does not hold in realistic urban transportation networks [11]. (4) Approaches that sacrifice some route planning aspects, e.g. the multi-modal feature which allows us to model different types of transportation services, or transfers which allows us to model changes between transportation services in a route.

It is important to note that incorporating real-time transportation information of a city and generating a travel plan T with detailed instructions to move along POIs, even in case of delays, is not provided by any approach so far, to the best of our knowledge. We therefore propose such an approach which generates a sequence of attractions to visit, together with a travel plan with concrete instructions for the user. In total, we make the following contributions:

(1) We combine i) an approach for generating a sequence of attractions to be visited, using the time constraints regarding attractions and total trip time (i.e. solving the TTDP problem and using the time-dependency constraint for public transportation) with ii) an approach for generating a realistic, delay-aware travel plan. We employ three different strategies (sections 5.1 to 5.3). The travel plan generation is designed to be realistic, as it uses the real-time transportation system's information (departure times with potential delays) and the current place and time of the user. Furthermore, the real-time computation constraint is considered.

(2) We evaluate our three approaches extensively with a large real-world data set, namely the POIs and the public transportation information of Izmir, Turkey. This data set contains 75 POIs, and a large transportation network which consists of approx. 8K stations and 26K bus runs per day. Our evaluation results show that the designed strategies outperform the state-of-the-art in terms of quality of recommendations.

Our approach is applicable to tourist plan recommendation in any city or location in which data about the attractions, their locations, and availability times are available, as its real-time public transportation system information.

The rest of the paper is organized as follows: First we give an overview of the TTDP and the approaches to solve that problem in Related Work (section 2). Then we present the Task Description (section 3) and the models our approaches are based on. We explain the basic concepts of the state-of-the-art method to solve the TTDP, the Iterated Local Search (ILS), in section 4. Section 5 describes our main contribution, three approaches built on top of ILS, i.e. SILS, TRILS, and PHILS, to produce feasible and realistic trip and travel plans. The experiments are shown in section 6. Finally, conclusions and future work are presented in section 7.

2 RELATED WORK

The tourist trip design problem (TTDP) is the problem of generating a sequence of the most relevant POIs to visit without violating user restrictions such as time budget. Although the TTDP has been widely investigated, a gap remains between theoretically solving TTDPs and applying them in practice [26]. One example is the incorporation of public transportation information into the trip plan. In fact, both creating travel plans for POIs and generating travel plans incorporating public transportation are separately considered to be hard to solve and have their own challenges in terms of modeling and computation. In addition to this, the solutions have to be computed in real-time.

One of the earlier works which addresses the TTDP problem is the one by Tumas and Ricci [20], which provides a set of ranked routes (using public transportation) fixing a start and end location. The route itself might lead through famous POIs. While this involves the user in selecting her preferences, the route selection remains the central concept.

However, most of the solutions seen in the literature consist of modeling the problem as extensions of the *Orienteering problem (OP)*. The OP is a combination of two classical problems: the Knapsack Problem and the Traveling Salesman problem. This problem can be defined formally as a graph in which nodes represent POIs and the edges a feasible travel route from one POI to another one. Each node has a "profit 'score" (henceforth simply "profit") that the user can collect by visiting the POI and the edges are weighted with travel times. The OP and its variations cannot be solved in polynomial time and therefore most of the solutions proposed are based on meta-heuristic algorithms which provide near-optimal solutions [10]. Among the existing OP extensions, the best modeler of the usage of public transportation is the *Time-dependent Team Orienteering Problem with Time Windows (TDTOPTW)* [14, 23]. The term *time-dependency* (TD) reflects the fact that the travel time to

[1] In this paper, we assume that the user does not want to leave the attractions earlier than planed, as this might stress her and as it would make the problem hardly manageable.

move from one POI to the other depends on the departure time, e.g. on the schedule of buses, trams, etc. The "*Team*" (T) extension allows one to model trip plans for multiple days. Moreover, the *Time Windows* (TW) represent the fact that visits are limited by opening and closing times of the POI. Algorithms based on Iterated Local Search (ILS) heuristics are considered the state-of-the-art to solve OP and their extensions [14]. An ILS approach for solving extensions of OP typically consists of two operations: INSERT, which inserts a POI into the solution, and SHAKE, which removes a POI to escape from local optima. The most popular version of ILS is probably the one proposed by Vansteenwegen [21].

State-of-the-art for TD(T)OPTW. Garcia et al. [6] were the first ones who tried to address the TDTOPTW problem using real public transportation data from the city of San Sebastian. Their ILS metaheuristic is built upon that of Vansteenwegen [21]. To deal with time-dependency they implement two approaches. The first one consists of using average travel times in their ILS. Since average travel times are not always accurate, they propose an approach to adjust the plan according to real travel times in case this is infeasible, i.e. if the time windows of the nodes included in the solution are violated [11]. This might lead to the removal of some of the attractions, reducing then the overall profit. The second approach is based on a fast local evaluation of the possible insertions. They design three variants: 1) direct public transportation without transfers; 2) an approach based on a pre-calculation which considers transfers; 3) an approach in which transfers are modeled as direct connections. The first two variants fulfill the real-time response requirement. However, they do not realistically model public transportation, because in large networks like cities, it is not always possible to reach a place without transfers, nor are the schedules always regular. To deal with the time-dependency Gavalas et al. extend in [11] a previous cluster-based meta-heuristic approach called CSCRoutes to deal with TDTOPTW. This results into two new approaches, TDCSCRoutes and SlackCSCRoutes. These do not make any assumptions about periodic schedules. In a subsequent work [8] they integrate multimodality into the routing logic as well as other features, such as the possibility of incorporating lunch breaks and the support of arbitrary start and end itinerary locations. Travel instructions are also included. The approach is validated with metropolitan transit network information and real POIs from Athens and Berlin. Verbeeck et al. [25] extend a previous approach [24] to solve the TDTOPTW problem. They use an ant colony system (ACS) based algorithm which constructs several independent solutions. The road network data used consists of data sent by taxis, commercial vehicles, and private cars, rather than public transportation so it is very unlikely that they model transfers. Moreover, they consider some kind of regularities by dividing a day into k slots. Then the set of time-dependent travel times is calculated by repeatedly using a modified version of Dijkstra's algorithm with a departure time equal to the start of a time slot.

Further extensions. The TDTOPTW is not the only extension of TOPTW which tries to model more realistic scenarios. For example POIs are typically treated as points. However, in pratice these might represent large areas such as market areas or neighborhoods in which tourists might require a walking route. Therefore, Gavalas et al. [7] extend the TOPTW problem to incorporate scenic walking routes for exploring tourist destinations and call the new model MTOPTW. Vansteenwegen et al. [22] extend TOPTW by adding constraints that allow POIs to have multiple time windows, e.g. windows which differ on different days. The first extension which aims at generating plans for groups of tourists was proposed by Sylejmani et al. [19]. They extend the Multi Constraint TOPTW (MCTOPTW) to MC-Multiple-TOPTW (MCMTOPTW) to model the multiple trips and tours for tourist groups. Other approaches model congestions or other events which might affect the travel times between nodes. Sometimes these events are difficult or even impossible to predict in a deterministic way [14]. These models are therefore called Stochastic OPTW (SOPTW) and are related to vehicle routing models [15].

Other approaches to solve TTDP. While the most popular version of ILS is the one proposed by Vansteenwegen [21], other variants have been proposed [12, 17]. In [12] they extend the ILS with further operations, namely SWAP, INSERT, ACCEPTANCE CRITERION, and TIMELIMIT.

Completely different strategies have been suggested as well, in addition to these ILS heuristics. In [9] another near-optimal heuristic approach called DailyTRIP was proposed. Other examples include a Tabu Search meta-heuristic [18], fireworks algorithm [5], and simulated annealing [13, 16]. Bitonto et al. [4] model the problem of building itineraries for tourists addressing a Constraint Satisfaction Problem (CSP) by means of the transitive closure. The work of Wörndl [26] is particularly interesting, since they try to solve the whole problem with a modified version of Dijkstra's algorithm, which not only finds shortest paths but also solves TTDP. To do so they maximize the quotient of entertainment divided by distance for each subpath (entertainment is the sum of the scores of all venues along the path). An extended version called constraint-based takes into account the time and budget constraints for the route. None of these approaches directly deals with public transportation.

3 TASK DESCRIPTION

3.1 Overall Task (TTDP-TI)

Figure. 1 illustrates the desired result, i.e. plans, V and T. In order to explain the underlying models used in our approach we need to provide some fundamental definitions. The POIs part of the visit plan V are selected among the set of available POIs $P = \{p_1, p_2, ..., p_n\}$. Each POI has a time window $[o_i, c_i]$ with *opening time* o_i and *closing time* c_i. Therefore, visits should take place within the time window. There is a variable for each POI p_i which models if the user visits it or not. Let v_i be this variable: $v_i = 1$ if p_i is visited and included in the plan V, 0 otherwise. Let t_i^v be the time the tourist u spends at p_i, if $v_i = 1$. If the visit takes place, this time is fixed, i.e. each POI has a recommended visit time and we assume for simplicity this cannot be shortened or extended. Let t_i^s be the start time of a visit at p_i. Then, the ending time of a visit is simply given by $t_i^e = t_i^s + t_i^v$. The time required to travel from p_i to p_j is $w_{(i,j)}^t$. This times depends on the departure time t from p_i. When a tourist visits a POI p_i she collects the *profit* s_i. Furthermore, let $x_{(i,j)}$ be a variable which models the fact that u visits p_j after she visited p_i. If this occurs, then $x_{(i,j)} = 1$, and 0 otherwise. The overall visit time for a plan should not exceed the tourist's total available time. Let t_{max} be the time budget of u. The goal is to generate a visit

plan V together with a travel plan T (which includes the detailed instructions tailored for public transportation). The time-dependency itself is modeled within $w^t_{(i,j)}$. Therefore, we call this problem the *tourist trip design problem with travel instructions (TTDP-TI)*. This can be divided into two subtasks: (1) the generation of a visit plan V taking into account the time-dependency, and (2) providing the travel instructions. Each of these subtasks will be explained in the following subsections.

3.2 Part 1: TTDP

The TTDP part of our problem is modeled as the time-dependent orienteering problem with time windows (TDOPTW). The goal is to produce a visit plan V, i.e. a route r which goes through some of the available POIs, thereby collecting as much profit as possible (*objective function*). For this problem one POI is required as the starting point and one as the ending point of the trip. Typically p_1 and p_n are picked as start and end locations [14, 23]. The problem is subject to a given set of constraints:

(1) A tour starts and ends at two fixed POIs $\in P$.
(2) A tour is a path which connects all visited POIs $\in V$. Each POI included in V should be visited at most once.
(3) The waiting time before a visit at a POI starts is limited by a constant M.
$t^e_i + w^t_{(i,j)} - t^s_j \leq M(1 - x_{(i,j)}), \quad (i, j = 1, ..., n).$
(4) The total time spent for the trip which includes the visit times, travel times and eventually the waiting times before a visit takes place should be less than the specified time budget t_{max}.
(5) The visits take place within the POI's time window.

In the same way as OP, TDOPTW can be formulated in two possible ways: as a graph or as integer linear programming problem. We refer to Vansteenwegen et al. [21] for a linear programming formulation of this problem.

The sequence $V = (p_i, ..., p_k)$ is reconstructed from the variables $x_{(l,m)}$, each of which represents a consecutive visit. In addition to V the visitation times for the each POI, $((t^s_i, t^e_i), ..., (t^s_k, t^e_k))$ are returned. It is important here to notice that getting the travel plans T (i.e. how to reach POIs) are beyond the scope of TDOPTW. They are generated by a separate algorithm, if needed. Most of the solutions relax the travel time $w^t_{(i,j)}$ by removing the time-dependency: $w_{(i,j)}$. Therefore, when the plan is adjusted according to real travel times, this might become infeasible (see section 5).

One important aspect of the TTDP is to determine the importance of a POI for a user u, i.e. to estimate the profit s_i the user could get by visiting the POI. This can be done in different ways.

3.3 Part 2: Route planning for public transportation

In order to obtain the travel instructions T we make use of a route planner in real-time. In fact, designing approaches which provide travel plans has been one of the main concerns of route planning. Not only the time-dependency, but also the real-time requirement are aspects which have been widely studied in this field. To fulfill the real-time requirement graphs are the most recurrent models in public transportation [2].

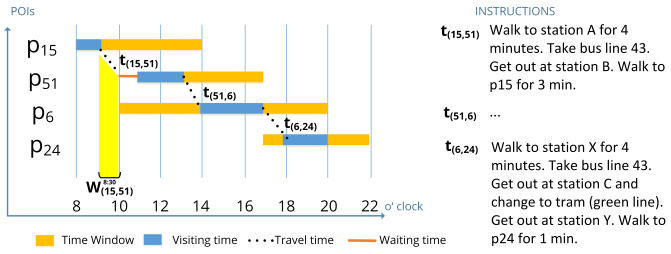

Figure 1: Example for solving the TTDP-IT problem

Our route planner has two key features: the time-dependent model (TDM) together with a state-of-the-art speed-up technique called *transfer patterns* [1]. The route planner is able to provide the path of minimum cost between two locations in the order of milliseconds. Note that without a route planner with these characteristics it would not have been possible to couple the TTDP approach with the travel information. This model supports walking (between stations or to change vehicle), transfers, waiting times at stations, etc. Delays of transportation units are also supported, although the optimality of the solution is no longer guaranteed. However, empirical studies show that transfer patterns are a very robust technique in the presence of delays and leads to suboptimal solutions in less than 3% of the cases in large networks [3].

4 ITERATED LOCAL SEARCH

We now come back to the TTDP problem, which is modeled in our case as a TDOPTW. In this regard the Iterated Local Search (ILS) heuristic is the state-of-the-art to solve TDOPTW. Our three approaches, SILS, TRILS, and PHILS are built upon ILS as conceived by Vansteenwegen [21] and García [6]. Our approaches will be presented in section 5 with a special focus on the novelty aspect. In this section, we will explain the basic concepts underlying the ILS algorithm, which are necessary to explain our approaches. The goal of an ILS algorithm is to return a near-optimal visit plan $V = (p_i, ..., p_k)$. Solutions are built by applying iteratively two operations: An *insertion step*, which inserts a POI into the solution, and a *shake step*, which removes a POI from it. The purpose of the latter is to escape from local optima. The stop condition of ILS requires that a solution is not improved for a certain number of iterations. The time-dependency plays a key role when inserting a new POI into the solution, because the exact time in which the visit starts depends on the departure from the previous POI. In the following, we explain these two steps and the ILS algorithm that combines them.

Insertion Step. In the insertion step, a new visit (POI) is added to the tour. The POI with the highest $Ratio_i$ is always the one picked for the insertion and we calculate $Ratio_i$ as follows:

$$Ratio_i = (s_i)^2 / Shift_i \tag{1}$$

where s_i is the profit, and $Shift_i$ is the extra time added to the duration of the trip plan if p_i is added to it. To compute the extra time, not only the visiting time of p_i is taken into account but also the travel time from the previous POI and to the following POI in the sequence. Since time is limited by the time budget t_{max}, every

time we insert a new POI it is checked whether existing visits still fit in their corresponding POI's time window.

Shake Step. The shake step removes at least one visit from the given tour. The purpose of this step is to escape local optima. The soon-to-be-removed POI is selected in a random fashion by means of variables which rotate over the whole visit plan. The visits scheduled after the removed POI are shifted towards the beginning to avoid unnecessary waiting times. Some of the visits may not be shifted because of the time window constraint. The visits after those non-shiftable visits remain unchanged.

For further details about the used variables and how these are updated in both the INSERT and SHAKE steps, we refer the reader to [21].

ILS: combining both steps. A pseudo-code showing the key steps of the ILS can be found in [21]. The search for a solution is performed until no better solution is found for 150 iterations[2]. At the beginning visits are inserted one by one using $Ratio_i$ until it is not possible to add more of them. This plan is then stored as the current best solution, using the variable *BestSolution*. Another variable *NoImprovementCounter* keeps track of the number of iterations in which no improvement could be achieved with respect to the current best solution. In the next iterations another produced visit plan might be compared with the last best solution found and if this is beaten, *BestSolution* is updated and *NoImprovementCounter* is reset. After checking the new solution, the shake step is applied.

The insertion operation deals directly with the time-dependency, because the travel time from the previous POI has to be considered in order to place the inserted POI at the right position in time. This might lead to the false belief that this information can be directly requested from the route planner every time an insertion is carried out. However, in practice too many visits are inserted and therefore the number of requests sent to the route planner would be too high to keep the whole computation of ILS operating in real-time.

Therefore, works like that of García et al. [6] either use pre-computed travel times, e.g. average travel times, or leave some events, such as transfers (changing buses in a trip), aside. We adopt instead a different strategy, which is outlined in section 5.

5 OUR APPROACHES

A method to produce a visit plan V based on pre-computed values might differ from the real travel times provided by a route planner. Therefore, a visit plan might have to be adjusted, i.e. the travel time between POIs has to be corrected. This might cause an increase in the waiting time or a shift of the visit times. If the time window of any of the POIs included in the solution is violated, then the plan is said to be *infeasible* [11] and it requires a repair strategy. García et al. [6] propose a simple repair strategy: if the real travel time is larger than the average one, some visits have to start later. If this causes a visit to become infeasible the POI is removed from the route. This means that in the best case the profit remains the same. Otherwise some profit is lost.

We therefore designed three strategies which not only provide feasible visit plans but also avoid sacrificing profit. The novelty aspect lies in the fact that we dynamically adjust the visit plan according to real travel times obtained from the route planner. Our heuristics are similar to those of Vansteenwegen [21] and García [6] in the sense that an optimal solution is first computed using average travel times for efficiency. However, while searching for an optimal solution the adjustment takes place by shifting some visits forwards or backwards in time, repairing the plan if required, and then letting the ILS continue with the search. This process allows our ILS-based heuristics to potentially find a different near-optimal solution, while keeping the plan realistic.

[2]This value has been empirically found in the experiments of Vansteenwegen and is a good trade-off between the execution time and the quality of the solution. In that setting the results of the ILS metaheuristic deviate from the optimal solution by only 1.8% using only 1 second of computation [17].

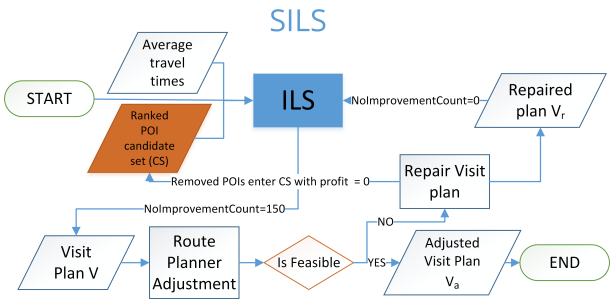

Figure 2: SILS Logic

5.1 Strict ILS (SILS)

The logic of SILS is illustrated in Figure 2. The intuition behind this heuristic is as follows: When *NoImprovementCounter=150* the found visit plan is adjusted with respect to the real travel times returned by the route planner. If after this adjustment the visit plan is feasible, this is returned as the solution (V_a).

Otherwise, the plan is repaired by removing the POIs in which the time window constraints are violated. The repaired solution V_r is then given to the ILS heuristic, which continues the search for an optimal solution from the plan. In addition, the removed POI(s) are penalized and relocated among other POI candidates with a profit $s_i = 0$. This basically disables the POI, which cannot be selected anymore ($Ratio_i = 0$). Moreover, the *NoImprovementCounter* is set to 0.

The process is repeated until a feasible visit plan is returned. Note that this approach can fail if all POIs are disabled. However, this never occurred in our experiments.

5.2 Time-relaxed ILS (TRILS)

The logic of TRILS is illustrated in Figure 3. Excluding one or more POIs from the ILS computation might not help in situations where the average travel time is a bad estimator of the actual travel time. This is the case when the timetable is not very regular or when the variance of the travel times in a day is too large.

Therefore, this approach rather gradually increases the estimated average travel times every time a solution is infeasible by multiplying it with a constant ($step = 0.05$). In this way the estimated travel times can be greater than the real travel times at some point and therefore a feasible plan can be returned. The downside of this strategy is that it might exclude more compact solutions with a higher profit.

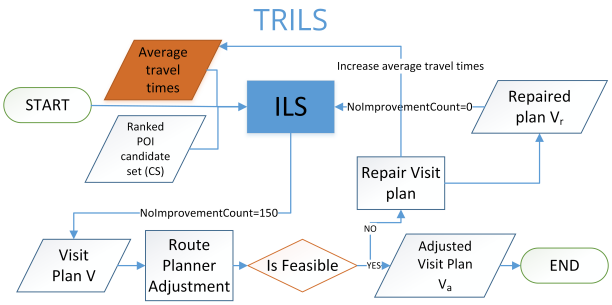

Figure 3: TRILS Logic represented as flowchart

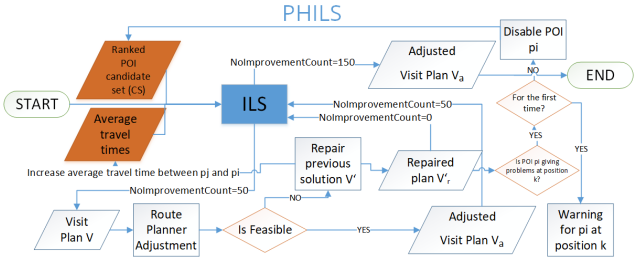

Figure 4: PHILS Logic

As in SILS, TRILS checks the validity of the solution when *NoImprovementCounter=150*. The visit plan is validated against the real time travels returned by the route planner. If after the adjustment, the visit plan is feasible, this is returned as the solution (V_a). Otherwise, the plan is repaired by removing the POIs in which the visits violate the time window constraints. In addition to this, the average travel times are increased by a constant. The first time an infeasible solution is returned, the average travel times are multiplied by 1.05. The second time by 1.10, and so on. The ILS heuristic continues the computation using the repaired plan V_r. It is important to notice that although the profits of the removed POIs remain unchanged, the insert operation might exclude them because of the increased average travel time, which potentially reduces the number of time slots into which these can be potentially inserted.

The process of increasing the average travel times is repeated until eventually a feasible solution is found. The approach fails if the average travel times are increased 5 times. However, this never occurred in our experiments.

5.3 Precise Hybrid ILS (PHILS)

The logic of PHILS is shown in Figure 4. This approach combines the ideas of the two previous approaches in a more fine-grained fashion. First, this approach tries to validate the current best solution when *NoImprovementCounter=50*, i.e. at an earlier stage of the solution computation. Let V be the best known plan at this point of calculation. If the alignment with the route planner does not cause the visit plan to be infeasible, then the ILS continues the search for a better solution. If no better solution is found until *NoImprovementCounter=150* then the visit plan V_a, which is already adjusted, is returned as the solution.

In contrast, if the adjustment causes the plan V to be infeasible three measures are taken: (1) if p_i is the POI in $V = \{..., p_j, p_i, ...\}$ in which the time window constraint is violated, then the average travel time between p_j and p_i is increased. Note that this correction is done for only a single pair of POIs and not for all possible pairs as in TRILS. (2) Instead of letting the ILS continue the search from the repaired plan V_r, the previous best known solution is retrieved. Let V' be this solution. (3) If V' contains p_i, then this is removed from it and this solution is used as the new starting point. The intuition is that it is better to correct the solution which lead to the infeasible plan, rather than the plan itself. Then, the ID of p_i and the position k in the sequence at which it failed are stored. If p_i disrupts another solution at the same position, p_i is permanently removed from the candidate set. Note that p_i is disabled when the failure is produced at the same position k in the sequence and not simply when it occurs, as in SILS. The ILS continues the search for a better solution based on the repaired plan V'_r. The process is then repeated until a feasible visit plan is produced.

5.4 Travel instructions

To generate the travel instructions T required to move from one POI to the next one in the sequence, the route planner can simply store the travel plans (including the travel instructions) of the last adjusted visit plan V. In fact, the last-adjusted plan is also the final returned solution in all three strategies.

To summarize. The three approaches are able to produce feasible solutions, in contrast to approaches based on ILS and average travel times. On the one hand, in the case of SILS and TRILS, the solution is adjusted according to the real travel times before it is returned as the final solution. If the adjusted plan is infeasible, some corrections are being made in the selection strategy, i.e. disabling an out-of-window POI, or increasing the average travel times, respectively. A new optimal solution is then searched iteratively on top of that repaired plan. On the other hand, PHILS combines both strategies in a more fine-grained manner. The travel instructions are provided by the route planner without the need for further calculations.

6 EXPERIMENTS

The ultimate goal of Recommender Systems is to maximize the user satisfaction. Solutions which model the TTDP as TDOPTW or similar extensions assume that profit is a good measure of the user's satisfaction. Therefore, they are compared based on their collected profit under the same constraints. We follow the same line of evaluation. An additional aspect we assess is whether our approaches are able to provide a plan in real-time.

6.1 Set up

Our POI dataset for Izmir consists of 75 POIs in total. POI information like the labels used to build both the user and POI profiles are obtained from Foursquare[3]. The POIs are distributed in 4 Regions (see figure 5): 36 are located within the city center; 17 POIs in the north; 8 POIs east; 11 POIs south-west and 3 POIs outside of the city

[3]https://developer.foursquare.com/

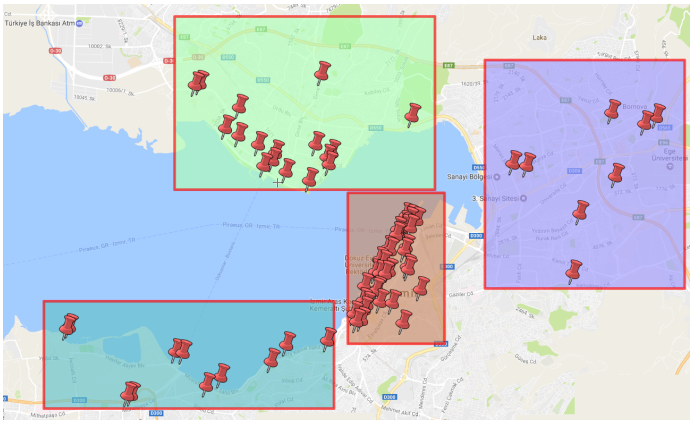

Figure 5: Distribution of POIs in Izmir

(not shown in the map). The General Directorate of ESHOT[4], the public bus transportation corporation of the Municipality of Izmir, Turkey kindly provided us with the public transportation data upon request. All POIs can be reached using the public transportation network. The network consists of 7788 stations and 333 working bus lines operating both ways, which results in 25849 bus runs in a single day.

In order evaluate the performance of the different approaches we first build visit plan requests. Using two of the 75 available POIs as the starting and ending points of the tour, we produce $75 \times 75 = 5625$ requests for visit plans. Note that this also includes the case in which the same POI is used as both starting and ending point. This simply means that a route would start and end at the same position, but it might still contain an arbitrary number of POIs to visit based on the user's specified time budget. The time budget is set as either 4, 6 or 8 hours. Starting times of the visit plan are either 10:00 or 12:00. This gives us 6 time-spans, 10:00 to 14:00 (4 hours), 10:00 to 16:00 (6 hours), 10:00 to 18:00 (8 hours), 12:00 to 16:00 (4 hours), etc. In addition, we also consider 5 different user profiles. To summarize, we have $5 \times 5625 \times 6 = 168,750$ requests for visit plans. The following approaches are compared:

(1) AvgILS. This is the approach implemented by García et al. [6]. Their ILS approach is computed using average travel times. Note that this approach is validated against our route planner to assess how many infeasible plans are produced. The evaluation scores for infeasible plans are counted as 0.

(2) RepAvgILS. AvgILS combined with the repairing method proposed by García et al. [6] which is described in section 5.

Our three designed approaches, **(3) SILS**, **(4) TRILS**, and **(5) PHILS**, which dynamically adjust the visit plan according to the real travel times provided by a route planner.

Each of the approaches is evaluated under two different circumstances: **Without Delays (ND)**, i.e. the case in which all bus units run perfectly on time according to their timetable, and **With Delays (D)**, in which delays are simulated at the level of single transportation units. Note that we assume that the time of request is the start of each time-span, i.e. either 10:00 or 12:00. The delays are therefore introduced before the time of request.

The evaluated metrics are shown at the end of the results table. All experiments were conducted on a single machine with 40 GB of RAM and a 64 bit Intel Xeon E5-2640, 2.5 GHz processor. The route planner ran for the entire duration of the experiments.

6.2 Interpretation of results

Tables 1 to 5 show the results of our experiments for each approach without and with delays in the transportation network. Each table's cell condenses the scores obtained for all requests and the five considered profiles in the given timespan. Moreover, for each timespan and metric the best scores achieved are shown in red.

Without delays (left tables). SILS produced 0.07% more profit than RepAvgILS, while TRILS and PHILS obtain 0.05% and 0.06%, respectively. A first observation is that the differences between overall total scores (TS) increase the more infeasible plans are produced (VTW) by AvgILS. The reason is that our approaches produce only feasible plans and therefore manage to gain additional profit in these cases (infeasible solutions contribute zero profit to the TS). This is also reflected in the columns TRS and ARS, the total and average repaired scores for those infeasible solutions. For the infeasible plans the improvement with respect to RepAvgILS is as follows: SILS achieves 6.76% improvement whereas PHILS 6.06%, and TRILS 5.12%.

With delays (right tables). Interestingly, all approaches generate trip plans with higher scores because the number of infeasible trip plans decreases. The reason for this is that delays are simulated for randomly picked units under independent assumptions. Therefore when a user travels to a POI more options, namely the delayed units, are available to reach it, which might reduce the travel time. PHILS performs the best, with an improvement in the overall score of 0.05% wrt. RepAvgILS. This is followed by SILS (0.04%) and TRILS (0.035%). For the infeasible plans the improvement with respect to RepAvgILS is as follows: PHILS achieves a 6.9% improvement whereas SILS 6.6%, and TRILS 4.95%. This shows the robustness of our approaches in the presence of delays.

As expected RepAvgILS was the fastest ILS-based heuristic due to its simple repairing strategy, whereas PHILS required the longest time (26.12% slower) to find the final solution. However, the execution times of both SILS and TRILS, approx. 1.2% and 1.5% slower, respectively, are very close to that of RepAvgILS. In any case our approaches are able to generate a plan in under 15ms (on average) which is a nice achievement considering this includes the adjustment time and interaction with the route planner.

Conclusion. SILS, TRILS and PHILS manage to produce only feasible plans thanks to the novel interactive aspect, i.e. the two-way flow of information between the core approach, which assembles the solution, and the route planner. Our results also show that more infeasible plans are produced when the time budget is large or the visit ends late (visits are closer to the closing times of all POIs).

7 CONCLUSION AND FUTURE WORK

We presented, SILS, TRILS and PHILS, three novel approaches to solve the *tourist trip design problem with travel instructions* (TTDP-TI). This problem is an extension of the TTDP which, in addition

[4]General Directorate of ESHOT. http://www.eshot.gov.tr

	AvgILS without delays											AvgILS with delays										
	TS	AS	StD	TRS	ARS	ARP	StD	AET	StD	VTW	VTB	TS	AS	StD	TRS	ARS	ARP	StD	AET	StD	VTW	VTB
10:00-14:00	1429877,3	50,84	8,57	-	-	6,52	1,24	5,72	2,24	2	12908	**1429979,6**	**50,84**	8,57	-	-	6,52	1,24	5,82	2,12	0	12629
10:00-16:00	1893186,9	67,31	7,98	-	-	8,73	1,24	9,64	2,45	97	6292	1897425,1	67,47	7,98	-	-	8,73	1,24	9,3	2,58	46	6010
10:00-18:00	2335471,4	83,04	7,72	-	-	10,81	1,23	13,65	3,84	330	3793	2345451,9	83,39	7,73	-	-	10,81	1,23	13,58	3,88	209	3657
12:00-16:00	1428277,3	50,78	8,76	-	-	6,39	1,2	5,73	1,66	108	12675	1430924,7	50,88	8,76	-	-	6,39	1,2	5,76	1,68	62	12453
12:00-18:00	1872254,5	66,57	7,81	-	-	8,56	1,16	9,23	2,29	352	6011	1881889,1	66,91	7,8	-	-	8,56	1,16	9,25	2,3	224	5852
12:00-20:00	2268541,3	80,66	7,34	-	-	10,48	1,17	12,38	3,44	689	3651	2284818,9	81,24	7,34	-	-	10,48	1,17	11,5	3,34	507	3523

Table 1: Results of the AvgILS experiments

	RepAvgILS without delays											RepAvgILS with delays										
	TS	AS	StD	TRS	ARS	ARP	StD	AET	StD	VTW	VTB	TS	AS	StD	TRS	ARS	ARP	StD	AET	StD	VTW	VTB
10:00-14:00	**1429960,6**	**50,84**	8,57	**83,3**	**41,65**	6,52	1,24	5,72	2,24	0	12910	**1429979,6**	**50,84**	8,57	0	0	6,52	1,24	5,82	2,12	0	12629
10:00-16:00	**1901084,4**	**67,59**	7,98	**7897,5**	**81,42**	8,73	1,24	9,61	2,49	0	6323	**1901546**	**67,61**	7,97	**4120,9**	**89,58**	8,73	1,24	9,29	2,6	0	6024
10:00-18:00	2363974,4	84,05	7,74	28503	86,37	10,8	1,23	13,51	4,03	0	3837	2365012,7	84,09	7,71	19560,8	93,59	10,8	1,23	13,49	4,01	0	3664
12:00-16:00	1433242,4	50,96	8,77	4965,1	45,97	6,38	1,2	5,71	1,67	0	12727	1433662	50,97	8,76	2737,3	44,15	6,38	1,2	5,76	1,68	0	12476
12:00-18:00	1895734,8	67,4	7,81	23480,3	66,71	8,55	1,16	9,13	2,42	0	6096	1896849,3	67,44	7,8	14960,2	66,79	8,56	1,15	9,13	2,43	0	5911
12:00-20:00	2324103,1	82,63	7,44	55561,8	80,64	10,46	1,18	12,13	3,73	0	3714	2325649,1	82,69	7,38	40830,2	80,53	10,47	1,17	12,07	3,71	0	3578

Table 2: Results of the RepAvgILS experiments

	SILS without delays											SILS with delays										
	TS	AS	StD	TRS	ARS	ARP	StD	AET	StD	VTW	VTB	TS	AS	StD	TRS	ARS	ARP	StD	AET	StD	VTW	VTB
10:00-14:00	1429926,3	50,84	8,57	49	24,5	6,52	1,24	5,72	2,24	0	12908	**1429979,6**	**50,84**	8,57	0	0	6,52	1,24	5,82	2,12	0	12629
10:00-16:00	1900967,6	67,59	7,98	7780,7	80,21	8,73	1,24	9,65	2,45	0	6320	1901444,9	67,61	7,98	4019,8	87,39	8,73	1,24	9,3	2,58	0	6020
10:00-18:00	**2365608,7**	**84,11**	7,71	**30137,3**	**91,33**	10,8	1,23	13,68	3,86	0	3883	2366033,7	84,13	7,7	20581,8	98,48	10,81	1,23	13,61	3,89	0	3677
12:00-16:00	**1433759,7**	**50,98**	8,76	**5482,4**	**50,76**	6,39	1,2	5,73	1,66	0	12747	**1433910,5**	**50,98**	8,76	**2985,8**	**48,16**	6,39	1,2	5,87	1,69	0	12487
12:00-18:00	**1897730,1**	**67,47**	7,79	**25475,6**	**72,37**	8,56	1,15	9,26	2,3	0	6135	**1898078,5**	**67,49**	7,79	**16189,4**	**72,27**	8,56	1,15	9,26	2,38	0	5907
12:00-20:00	2328254	82,78	7,32	59712,7	86,67	10,47	1,17	12,46	3,53	0	3793	2328678,3	82,8	7,31	43859,4	86,51	10,48	1,17	12,35	3,53	0	3619

Table 3: Results of the SILS experiments

	TRILS without delays											TRILS with delays										
	TS	AS	StD	TRS	ARS	ARP	StD	AET	StD	VTW	VTB	TS	AS	StD	TRS	ARS	ARP	StD	AET	StD	VTW	VTB
10:00-14:00	1429926,3	50,84	8,57	49	24,5	6,52	1,24	5,72	2,24	0	12908	**1429979,6**	**50,84**	8,57	0	0	6,52	1,24	5,82	2,12	0	12629
10:00-16:00	1901007,6	67,59	7,98	7820,7	80,63	8,73	1,24	9,65	2,46	0	6322	1901460,7	67,61	7,98	4035,6	87,73	8,73	1,24	9,3	2,58	0	6018
10:00-18:00	2365043,9	84,09	7,72	29572,5	89,61	10,8	1,23	13,7	3,99	0	3915	2365623,1	84,11	7,71	20171,2	96,51	10,81	1,23	13,63	4,04	0	3709
12:00-16:00	1433735,6	50,98	8,76	5458,3	50,54	6,38	1,2	5,73	1,66	0	12756	1433906,3	50,98	8,76	2981,6	48,09	6,39	1,2	5,76	1,69	0	12496
12:00-18:00	1897355,7	67,46	7,79	25101,2	71,31	8,56	1,15	9,28	2,44	0	6160	1897846,3	67,48	7,79	15957,2	71,24	8,56	1,15	9,27	2,41	0	5941
12:00-20:00	2327203,1	82,74	7,34	58661,8	85,14	10,47	1,17	12,57	4,29	0	3896	2327957,7	82,77	7,32	43138,8	85,09	10,47	1,17	12,52	4,25	0	3702

Table 4: Results of the TRILS experiments

	PHILS without delays											PHILS with delays										
	TS	AS	StD	TRS	ARS	ARP	StD	AET	StD	VTW	VTB	TS	AS	StD	TRS	ARS	ARP	StD	AET	StD	VTW	VTB
10:00-14:00	1429925,5	50,84	8,57	48,2	24,1	6,52	1,24	7,82	2,41	0	12910	**1429979,6**	**50,84**	8,57	0	0	6,52	1,24	8,03	2,42	0	12629
10:00-16:00	1900914,5	67,59	7,99	7727,6	79,66	8,73	1,24	12,07	2,95	0	6300	1901497,8	67,61	7,98	4087,64	88,86	8,73	1,24	12,02	2,77	0	6011
10:00-18:00	2365429,1	84,1	7,72	29957,7	90,78	10,8	1,23	16,59	4,44	0	3858	**2366359,4**	**84,14**	7,7	**20852,84**	**99,77**	10,81	1,23	16,87	4,42	0	3678
12:00-16:00	1432980,6	50,95	8,78	4703,3	43,5	6,38	1,21	7,8	1,94	0	12705	1433667,3	50,97	8,77	2742,6	44,23	6,38	1,2	7,99	2,12	0	12471
12:00-18:00	1897202,8	67,46	7,79	24948,3	70,87	8,56	1,15	11,21	2,97	0	6081	1897777,2	67,48	7,79	15888,1	70,92	8,56	1,16	11,2	2,87	0	5885
12:00-20:00	**2328961,9**	**82,81**	7,31	**60416,39**	**87,68**	10,48	1,17	14,9	4,31	0	3763	**2329146**	**82,81**	7,31	**44322,9**	**87,42**	10,48	1,17	14,46	4,32	0	3598

Table 5: Results of the PHILS experiments

METRICS: **Total Score (TS)**. Overall score for all visit plans requests. **Average Score (AS)**. The average score obtained, i.e. $\frac{TS}{5625\times5}$. **Total Repair Score (TRS)**. Overall score for the repaired visit plans. **Average Repair Score (ARS)**. The average score obtained from the repaired visit plans, i.e. $\frac{TRS}{VTW_{AvgILS}}$. **Avg. number of Recommended POIs (ARP).** Number of recommended POIs on average for the feasible visit plans. **Avg. Execution time (AET)**. The execution time in milliseconds to return a solution. Standard deviation from AS, ARP and AET are provided next to each corresponding column. **Number of plans with at least one violation of a POI's time window (VTW)**. Number of requests which have at least one visiting time out of the POI time window. **Number of plans which violate the time budget (VTB)**. After adjusting the plan using the route planner, some visits are shifted back and forth leading to violations of the time budge constraint t_{max}.

to producing a visit plan (i.e. the sequence of POIs to visit in a city), also produces a travel plan with instructions on how to reach those attractions using public transportation. The novelty of these approaches lies in the way the visit plan is dynamically adjusted according to real travel times. While average travel times are still used in the computation for efficiency, the solution is eventually adjusted with the information provided by a route planner. If the adjustment leads to an infeasible plan each approach takes different countermeasures to repair it. The search for an optimal solution continues on top of the repaired plan. This makes it possible to not only return feasible plans (without violations of the POI's time windows) without sacrificing profit, but also to return both visit and travel times in real-time. Moreover, our state-of-the-art route planner is able to model a large variety of events related to public transportation, such as walking times (between stations, to a station to take a bus, etc.), changing vehicles (transfers), but especially to model delays of transportation units. To the best of our knowledge, no previous approach was able to provide travel instructions for the visit plans at this level of realism.

As we showed in our experiments, SILS, TRILS and PHILS are at comparable performance levels in terms of collected profit, while all approaches manage to outperform ILS-heuristics based on estimated travel times even after a repair. Moreover even PHILS, which required the longest execution time, is able to produce plans in real-time.

In the future we would like to extend our approach by modeling further constraints and events. In addition we would like to focus more on the personalization aspect of the TRS. Finally, our approaches could be used to model dynamic changes, too, thanks to the ability to deliver fast plans. If the user deviates from the visit plan, e.g. if she enjoys staying at one place and prolongs the visit, a new plan has to be recomputed together with the travel plans. Our approaches seem to be a good fit for this problem too.

REFERENCES

[1] Hannah Bast, Erik Carlsson, Arno Eigenwillig, Robert Geisberger, Chris Harrelson, Veselin Raychev, and Fabien Viger. 2010. Fast Routing in Very Large Public Transportation Networks Using Transfer Patterns. In *Proceedings of the 18th Annual European Conference on Algorithms: Part I (ESA'10)*. Springer-Verlag, Berlin, Heidelberg, 12. http://dl.acm.org/citation.cfm?id=1888935.1888969

[2] Hannah Bast, Daniel Delling, Andrew Goldberg, Matthias Müller-Hannemann, Thomas Pajor, Peter Sanders, Dorothea Wagner, and Renato F. Werneck. 2016. *Route Planning in Transportation Networks*. Springer International Publishing, Cham, Switzerland. http://dx.doi.org/10.1007/978-3-319-49487-6_2

[3] Hannah Bast, Jonas Sternisko, and Sabine Storandt. 2013. Delay-robustness of transfer patterns in public transportation route planning. In *ATMOS-13th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems-2013*, Vol. 33. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

[4] Pierpaolo Di Bitonto, Francesco Di Tria, Maria Laterza, Teresa Roselli, Veronica Rossano, and Filippo Tangorra. 2010. Automated Generation of Itineraries in Recommender Systems for Tourism. In *Current Trends in Web Engineering - 10th International Conference on Web Engineering, ICWE 2010 Workshops, Vienna, Austria, July 2010, Revised Selected Papers*. https://doi.org/10.1007/978-3-642-16985-4_48

[5] H. Ding, L. Ke, and Z. Geng. 2016. Route planning in a new tourist recommender system: A fireworks algorithm based approach. In *2016 IEEE Congress on Evolutionary Computation (CEC)*. https://doi.org/10.1109/CEC.2016.7744300

[6] Ander García, Pieter Vansteenwegen, Olatz Arbelaitz, Wouter Souffriau, and María Teresa Linaza. 2013. Integrating public transportation in personalised electronic tourist guides. *Computers & OR* 40, 3 (2013). https://doi.org/10.1016/j.cor.2011.03.020

[7] Damianos Gavalas, Vlasios Kasapakis, Charalampos Konstantopoulos, Grammati Pantziou, and Nikolaos Vathis. 2017. Scenic route planning for tourists. *Personal and Ubiquitous Computing* 21, 1 (2017). https://doi.org/10.1007/s00779-016-0971-3

[8] Damianos Gavalas, Vlasios Kasapakis, Charalampos Konstantopoulos, Grammati E. Pantziou, Nikolaos Vathis, and Christos D. Zaroliagis. 2015. The eCOMPASS multimodal tourist tour planner. *Expert Syst. Appl.* 42, 21 (2015), 7303–7316. https://doi.org/10.1016/j.eswa.2015.05.046

[9] Damianos Gavalas, Michael Kenteris, Charalampos Konstantopoulos, and Grammati E. Pantziou. 2012. Web application for recommending personalised mobile tourist routes. *IET Software* 6, 4 (2012). https://doi.org/10.1049/iet-sen.2011.0156

[10] Damianos Gavalas, Charalampos Konstantopoulos, Konstantinos Mastakas, and Grammati E. Pantziou. 2014. Mobile recommender systems in tourism. *J. Network and Computer Applications* 39 (2014). https://doi.org/10.1016/j.jnca.2013.04.006

[11] Damianos Gavalas, Charalampos Konstantopoulos, Konstantinos Mastakas, Grammati E. Pantziou, and Nikolaos Vathis. 2014. Efficient Heuristics for the Time Dependent Team Orienteering Problem with Time Windows. In *Applied Algorithms - First International Conference, ICAA 2014, Kolkata, India, 2014. Proceedings*. https://doi.org/10.1007/978-3-319-04126-1_13

[12] Aldy Gunawan, Hoong Chuin Lau, and Kun Lu. 2015. An Iterated Local Search Algorithm for Solving the Orienteering Problem with Time Windows. In *Evolutionary Computation in Combinatorial Optimization - 15th European Conference, EvoCOP 2015, Copenhagen, Denmark, 2015, Proceedings*. https://doi.org/10.1007/978-3-319-16468-7_6

[13] Aldy Gunawan, Hoong Chuin Lau, and Kun Lu. 2015. SAILS: hybrid algorithm for the team orienteering problem with time windows. In *Proceedings of the 7th Multidisciplinary International Scheduling Conference (MISTA)*.

[14] Aldy Gunawan, Hoong Chuin Lau, and Pieter Vansteenwegen. 2016. Orienteering Problem: A survey of recent variants, solution approaches and applications. *European Journal of Operational Research* 255, 2 (2016). https://doi.org/10.1016/j.ejor.2016.04.059

[15] Gilbert Laporte, François V. Louveaux, and Hélène Mercure. 1992. The Vehicle Routing Problem with Stochastic Travel Times. *Transportation Science* 26, 3 (1992), 161–170. https://doi.org/10.1287/trsc.26.3.161

[16] Shih-Wei Lin and Vincent F. Yu. 2012. A simulated annealing heuristic for the team orienteering problem with time windows. *European Journal of Operational Research* 217, 1 (2012). https://doi.org/10.1016/j.ejor.2011.08.024

[17] Wouter Souffriau, Joris Maervoet, Pieter Vansteenwegen, Greet Vanden Berghe, and Dirk Van Oudheusden. 2009. A Mobile Tourist Decision Support System for Small Footprint Devices. In *Bio-Inspired Systems: Computational and Ambient Intelligence, 10th International Work-Conference on Artificial Neural Networks, IWANN 2009, Salamanca, Spain, 2009. Proceedings, Part I*. https://doi.org/10.1007/978-3-642-02478-8_156

[18] Kadri Sylejmani, Jürgen Dorn, and Nysret Musliu. 2012. A Tabu Search approach for Multi Constrained Team Orienteering Problem and its application in touristic trip planning. In *12th International Conference on Hybrid Intelligent Systems, HIS 2012, Pune, India, 2012*. https://doi.org/10.1109/HIS.2012.6421351

[19] Kadri Sylejmani, Jürgen Dorn, and Nysret Musliu. 2017. Planning the trip itinerary for tourist groups. *Information Technology & Tourism* (2017). https://doi.org/10.1007/s40558-017-0080-9

[20] Gytis Tumas and Francesco Ricci. 2009. Personalized Mobile City Transport Advisory System. In *Information and Communication Technologies in Tourism, ENTER 2009, Proceedings of the International Conference in Amsterdam, The Netherlands, 2009*. https://doi.org/10.1007/978-3-211-93971-0_15

[21] Pieter Vansteenwegen, Wouter Souffriau, Greet Vanden Berghe, and Dirk Van Oudheusden. 2009. Iterated local search for the team orienteering problem with time windows. *Computers & OR* 36, 12 (2009). https://doi.org/10.1016/j.cor.2009.03.008

[22] Pieter Vansteenwegen, Wouter Souffriau, Greet Vanden Berghe, and Dirk Van Oudheusden. 2011. The City Trip Planner: An expert system for tourists. *Expert Syst. Appl.* 38, 6 (2011). https://doi.org/10.1016/j.eswa.2010.11.085

[23] Pieter Vansteenwegen, Wouter Souffriau, and Dirk Van Oudheusden. 2011. The orienteering problem: A survey. *European Journal of Operational Research* 209, 1 (2011), 1–10. https://doi.org/10.1016/j.ejor.2010.03.045

[24] C. Verbeeck, Kenneth Sörensen, El-Houssaine Aghezzaf, and Pieter Vansteenwegen. 2014. A fast solution method for the time-dependent orienteering problem. *European Journal of Operational Research* 236, 2 (2014). https://doi.org/10.1016/j.ejor.2013.11.038

[25] Cédric Verbeeck, Pieter Vansteenwegen, and El-Houssaine Aghezzaf. 2017. The time-dependent orienteering problem with time windows: a fast ant colony system. *Annals of Operations Research* (2017). https://doi.org/10.1007/s10479-017-2409-3

[26] Wolfgang Wörndl, Alexander Hefele, and Daniel Herzog. 2017. Recommending a sequence of interesting places for tourist trips. *J. of IT & Tourism* 17, 1 (2017). https://doi.org/10.1007/s40558-017-0076-5

