Published in IET Software
Received on 3rd September 2011
Revised on 23rd December 2011
doi: 10.1049/iet-sen.2011.0156

ISSN 1751-8806

Web application for recommending personalised mobile tourist routes

D. Gavalas[1] *M. Kenteris*[1] *C. Konstantopoulos*[2] *G. Pantziou*[3]
[1]*Department of Cultural Technology and Communication, University of the Aegean, Mytilene, Greece*
[2]*Department of Informatics, University of Piraeus, Piraeus, Greece*
[3]*Department of Informatics, Technological Educational Institution of Athens, Athens, Greece*
E-mail: dgavalas@aegean.gr

Abstract: This study deals with the problem of deriving personalised recommendations for daily sightseeing itineraries for tourists visiting any destination. The authors' approach considers selected places of interest that a traveller would potentially wish to visit and derives a near-optimal itinerary for each day of visit; the places of potential interest are selected based on stated or implied user preferences. The authors' method enables the planning of customised daily personalised tourist itineraries considering user preferences, time available for visiting sights on a daily basis, opening days of sights and average visiting times for these sights. Herein, the authors propose a heuristic solution to this problem addressed to both web and mobile web users. Evaluation and simulation results verify the competence of the authors' approach against an alternative method.

1 Introduction

Tourists who visit a destination for one or multiple days are unlikely to visit every tourist sight; rather, tourists are faced with the dilemma of which points of interest (POIs) would be more interesting for them to visit. These choices are normally based on information gathered by tourists via internet, magazines, printed tourist guides etc. After deciding on which sights to visit, tourists have to decide on which route to take, that is, the order in which to visit each POI, with respect to the visiting time required for each POI, the POI's visiting days/hours and the time available for sightseeing on a daily basis.

Tourists encounter many problems following this procedure. The information contained in printed guide books is often outdated (e.g. the opening times of some museums might have changed or some other memorial sites might be closed because of maintenance works etc.), the weather conditions might be prohibitive during one of the visiting days to visit an important POI etc. [1]. The selection of the most important and interesting POIs for visiting also requires fusion of information typically provided from separate often non-credible sources. Usually, tourists are satisfied if a fairly attractive or feasible route is derived, yet, they cannot know of any alternative routes that would potentially be better to follow. Some tourist guides do acknowledge such problems and try to propose more generalised tourist routes to a city or an area. Of course, these routes are designed to satisfy the likes of the majority of its readers, but not those with specialised interests, needs or constraints [2].

Mobile tourist guides may be used as tools to offer solution to these types of problems [3–6]. Based on a list of personal interests, up-to-date information for the sight and information about the visit (e.g. date of arrival and departure, accommodation address etc.), a mobile guide can suggest near-optimal and feasible routes that include visits to a series of sights, as well as recommending the order of each sight's visit along the route [7]. Generalised tourist routes do not take into consideration the context of the user for example. the starting or ending point of the user, the available time the user affords, the current time, predicted weather conditions while on journey etc. Taking into account the parameters of context and location awareness brings forward a challenge for the design of appropriate tourist routes [8]. Kramer *et al.* [9] analysed the interests in the profiles of each tourist and concluded that they particularly varied from each other. This conclusion supports the argumentation for deriving personalised, instead of generalised, tourist routes.

Given a list of sights of some tourist destination in which a user-tourist would potentially be interested in visiting, the problem involves deriving the order in which the tourist should visit the selected POIs, for each day the tourist stays at that destination. We term this problem as the 'tourist itinerary design problem' (TIDP). Interestingly, the TIDP presents similarities to problems that have arisen in the past in the field of operational research; such problems are based on the mathematical theory of graphs (graph theory) and comprise variations of the well-known travelling salesman problem (TSP).

For instance, the team orienteering problem (TOP) appoints an initial and final point as well as N points for visiting, where each point is associated with a 'score' or 'profit'. Given a particular time margin for each of the M team members, the TOP determines M routes (from the initial to the end point) via a subset of N points, aiming at maximising the overall profit of visited points [10]. The TOP cannot be solved in polynomial time (NP-complete) [11], hence heuristics deriving near-optimal solutions are the only realistic way to tackle such problems, especially when considering online applications. TOP can be thought of as a starting point to model TIDP whereby the M team members are reduced to the number of days available for the tourist to stay and the profit of a sight signifies the potential interest (or degree of satisfaction) of a particular tourist visiting the POI within a given time span available for sightseeing daily (therefore TOP considers the time spent while visiting each POI as well as the time needed to travel from one POI to another).

Nevertheless, TOP does not take into consideration the POIs' visiting days and hours. Therein, the resemblance of TIDP with another operational research problem (travelling salesman problem with time windows, TSPTW) [12] comes forward. TSPTW concerns the minimum cost path for a vehicle which visits a set of nodes. Each node must be visited only once and the visit must be carried out inside an allowed time interval (time window). The correlation of time windows with the POIs visiting days/hours is obvious. However, TSPTW involves planning of only one route (i.e. not M, as many as days available to the tourist to visit POIs), while it requires the vehicle to visit the whole set of nodes. A generalisation of TOP and TSPTW is referred to as team orienting problem with time windows (TOPTW) [13] and considers multiple vehicles (i.e. itineraries) that should visit a subset of nodes, each within its allowed time window.

The main contribution of this paper lies in modelling and investigating a generalisation of TOPTW through introducing a novel heuristic that provides near-optimal solutions to TIDP: the daily tourist itinerary planning (DailyTRIP). It is noted that some preliminary ideas of our technique have also been presented in [14].

The remaining of this article is organised as follows: related work is discussed in Section 2. The modelling, design and implementation of DailyTRIP are presented in Sections 3–5, respectively. Section 6 compares the approaches taken by DailyTRIP against a relevant algorithmic solution. Section 7 discusses simulation results whereas Section 8 draws conclusions and grounds for future work.

2 Related work

The issue of personalised tourist itineraries has not been looked at in the electronic and mobile tourism literature, with the exception of the algorithms proposed in [11, 15]. In Souffriau *et al.* [15], proposed a heuristic solution for the orienteering problem, that is, they only consider a single tourist itinerary. The algorithm presented in [11] deals with TOPTW and also takes into account the time needed to visit a sight; hence, it is directly comparable with our work.

Other relevant research projects with respect to tourist itineraries have been reported in [16, 17]. In [16], a method for deriving a single multi-modal tourist itinerary is proposed considering a variety of constraints and following a genetic algorithm-based approach. However, derived route recommendations are not personalised as user preferences about specific types of sites are not taken into account. P-Tour [18], dynamic tour guide (DTG) [19] and City Trip Planner [20] address these shortcomings; however, they derive routes day-by-day. The City Trip Planner currently offers the most advanced route planning functionalities considering several user constraints. Google city tours application [21] represents another interesting approach along the same line suggesting multiple daily itineraries through the familiar Google maps interface. Yet, the suggested itineraries are not personalised. Furthermore, city tours implementations are only provided through a web interface and have not been tested on mobiles; hence, they lack location-based and context-aware features [16, 17]. On the other hand, P-Tour and DTG have been implemented on mobiles, but can only deal with a small number of POIs (i.e. their scalability is questionable).

3 DailyTRIP modelling

DailyTRIP modelling involves the definition and the description of the user model, visit model and the sight (POI) model (see Fig. 1) taking into consideration parameters/constraints like those listed below:

1. User model:

   - device (e.g. screen resolution, available storage space, processing power etc.);
   - language of content, localisation;
   - personal 'demographic' data (e.g. age, educational level);
   - interests (explicit declaration or implicitly collected);
   - disability (e.g. blind, deaf, kinetic disability);
   - budget threshold willing to spend on sightseeing.

2. Visit model:

   - geographical location of accommodation;
   - period of stay (arrival and departure date);
   - time constraints (e.g. available time each day to tour, number and duration of desirable breaks etc.);
   - means of travel (e.g. walking, driving, bus, metro etc.).

3. Sight (POI) model:

   - category (e.g. museum, archaeological site, monument etc.);
   - available multimedia resources (collection of texts, video, audio etc. localised in different languages;
   - geographical position (coordinates);
   - weight or 'objective' importance (e.g. the Acropolis of Athens is thought to be 'objectively' more important of the Coin Museum of Athens, hence the Acropolis is assigned a larger weight);
   - average duration of visit (e.g. the Archaeological Museum of Athens typically takes longer to visit than

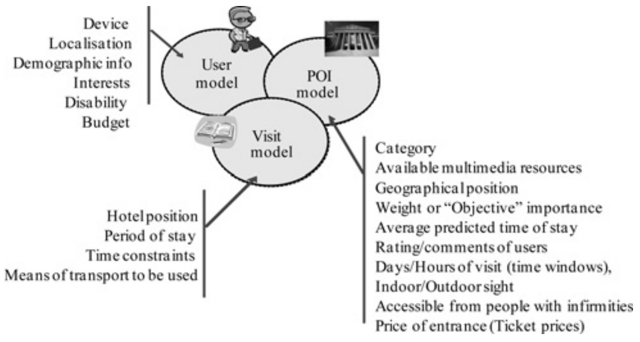

Device
Localisation
Demographic info
Interests
Disability
Budget
User model
POI model
Visit model
Hotel position
Period of stay
Time constraints
Means of transport to be used
Category
Available multimedia resources
Geographical position
Weight or "Objective" importance
Average predicted time of stay
Rating/comments of users
Days/Hours of visit (time windows),
Indoor/Outdoor sight
Accessible from people with infirmities
Price of entrance (Ticket prices)

Fig. 1 *Description of user, visit and sight models in TIDP*

the city's Coin Museum because of size difference and the nature of exhibition);
- rating/comments of users;
- opening days/hours (time windows), which could be provided by the web service of an administrative body or the Ministry of Culture;
- whether it is a indoor/outdoor site;
- whether it is a accessible from people with disabilities;
- admission price (ticket prices).

Notably, the above-stated parameters/constraints are not exhaustive. From those parameters the elements listed below may be easily derived:

- The topological distance (or Manhattan distance) among the POIs and also among the accommodation and the POIs, based on their geographical coordinates and the local map.
- The number of routes that must be generated are based upon the period of stay of the user at the tourist destination.
- The anticipated duration of visit of a user at a POI derives from the average duration and the user's potential interest (concluded by examining the user's profile).
- The ability to visit open air sites in a particular day during the user's visit for example. outdoor sites are not recommended to visit during a rainy day (meteorological forecasts can be retrieved from an Internet web service).

The problem's definition also includes the 'profit' of a POI, calculated as a weighted function of the objective and subjective importance of each POI (subjectivity refers to the users' individual preferences). Our algorithmic solution maximises the overall profit, that is, it enables the construction of personalised routes which include the most important (for each tourist) sights under specific constraints (opening hours, weather conditions, time available for sightseeing). The most crucial constraint in seeking sound algorithmic solutions is the daily time limit T which a tourist wishes to spend on visiting sights; the overall daily route duration (i.e. the sum of visiting times plus the overall time spent moving from a POI to another which is a function of the topological distance) should be kept below T.

4 DailyTRIP: a heuristic for deriving near-optimal personalised daily tourist itineraries

4.1 Problem statement

The TIDP problem involves a complete graph $G = (V, E)$, $|V| = n$, where each node i, $i = 0, \ldots, n-1$, in V corresponds to a POI and each edge (i, j) in E corresponds to the shortest path (in terms of Manhattan distance $d_{i,j}$) linking individual POIs i and j.

Each POI $i \in V$ is associated with a weight w_i which denotes the 'objective' importance of the POI and a profit value p_i, which reflects the importance of that POI for a particular user and depends on their personal preferences. Each POI i is also associated with a set of days $D_c(i)$ when visiting is not feasible (e.g. Mondays and during some bank holidays) and the anticipated visit duration of the user at the POI $t_v(i)$; similar to the profit, $t_v(i)$ also depends on the user's personal preferences (for instance, someone interested in archaeology is expected to take longer to visit an archaeological museum than others).

The cost of each edge (i, j) $c_{i,j}$, namely the cost of visiting j after visiting i, is a weighted function of travelling time from i to j $t_{i,j}$ (the latter depends on the Manhattan distance $d_{i,j}$ between i and j and the means of travel), the profit of the arriving node p_j and the duration of visit at the arriving node $t_v(j)$: $C_{i,j} = a_1 t_{i,j} - a_2 p_j + a_2 t_v(j)$, where a_1, a_2 and a_3 are weight coefficients. This formula signifies that having visited node i, node j will be visited next, if the latter is of relatively high profit and takes short to arrive and visit. Notably, $C_{i,j} \neq C_{j,i}$.

Travellers typically plan to visit the area for a set of days, D. Users also define a starting and ending time for their daily itineraries, T_{start} and T_{end}, which denote what time they prefer to depart from the starting point S and arrive at the end point (destination) E. Hence, a daily time budget devoted to visiting sights may be easily calculated: $T = T_{\text{end}} - T_{\text{start}}$. Without loss of generality, we assume that the starting and end points of the $|D|$ daily itineraries coincide, that is, $S \equiv E$ (typically these will coincide with the user's accommodation H).

Summarising, the objective of DailyTRIP is to derive $|D|$ itineraries I_i that maximise the overall profit $\sum_{i=1}^{|D|}\sum_{j=1}^{|I_i|} p_j$, ensuring that the time needed to complete each itinerary does not exceed the user-defined daily time budget T, that is,. $T(I_i) \leq T$.

4.2 DailyTRIP algorithm flow

4.2.1 DailyTRIP comprises the following execution phases:

Phase 1: Definition of the problem's model: The first phase first involves the definition of problem's space, that is, the nodes of G, the nodes' weight w_i and the travelling time $t_{i,j}$ that denotes the time needed to travel between node pairs (see Fig. 2*a*); notably, $t_{i,j} \neq t_{j,i}$, since the route $i \rightarrow j$ differs from the route $j \rightarrow i$ because of considering one-way roads. Taking into consideration personalisation issues (e.g. in a simplified scenario, user preferences upon POIs' categories), the cost matrix (i.e. the cost values $c_{i,j}$ associated with the two-directional edges) as well as the nodes' profit p_i and visit duration $t_v(i)$ with respect to a specified user are also computed.

Phase 2: Reduction of the problem's space: The initial set of sights around the tourist destination is sorted in decreasing order of profit p, where the value of p mainly depends on its category (i.e. whether the POI is museum or an architectural monument) and the user's preference upon this category. To reduce the computational effort required to reach valid solutions (i.e. to reduce the problem's space) we discard:

- nodes (POIs) with profit p_i smaller than a threshold value $p_{\min}$;
- POIs located too far from the origin point H, that is, every node v for which $t_{H,v} > t_{\max}$, where $t_{\max}$ is an upper time limit (see Fig. 2*b*).

An alternative approach would be to exclude the relatively low-profit POIs located far from H, that is, exclude every POI i for which $a_1{}^* p_i - a_2{}^* d_{i,H}$, where a_1 and a_2 are weight coefficients and t a threshold value.

Phase 3: Selection of first daily itinerary nodes: DailyTRIP determines the $|D|$ POIs that will be the first to include in the $|D|$ daily itineraries I_i, where $i = 1..\,|D|$. We select the set of $|D|$ nodes $\{N_i\}$, where $i = 1..\,|D|$, located furthest apart from one another, that is, those for which the minimum distance from one another is the maximum among any other permutation of $|D|$ nodes. For instance, in the example topology of Fig. 2*c*, assuming that $|D| = 3$, we select the nodes i, j and k that $\max\min_{i,j,k}\{d_{i,j}, d_{i,k}, d_{j,k}\}$. Then, the $|D|$ daily itineraries are initialised, each incorporating one of those nodes $I_i = \{N_i\}$, $\forall i = 1..|D|$. The philosophy behind

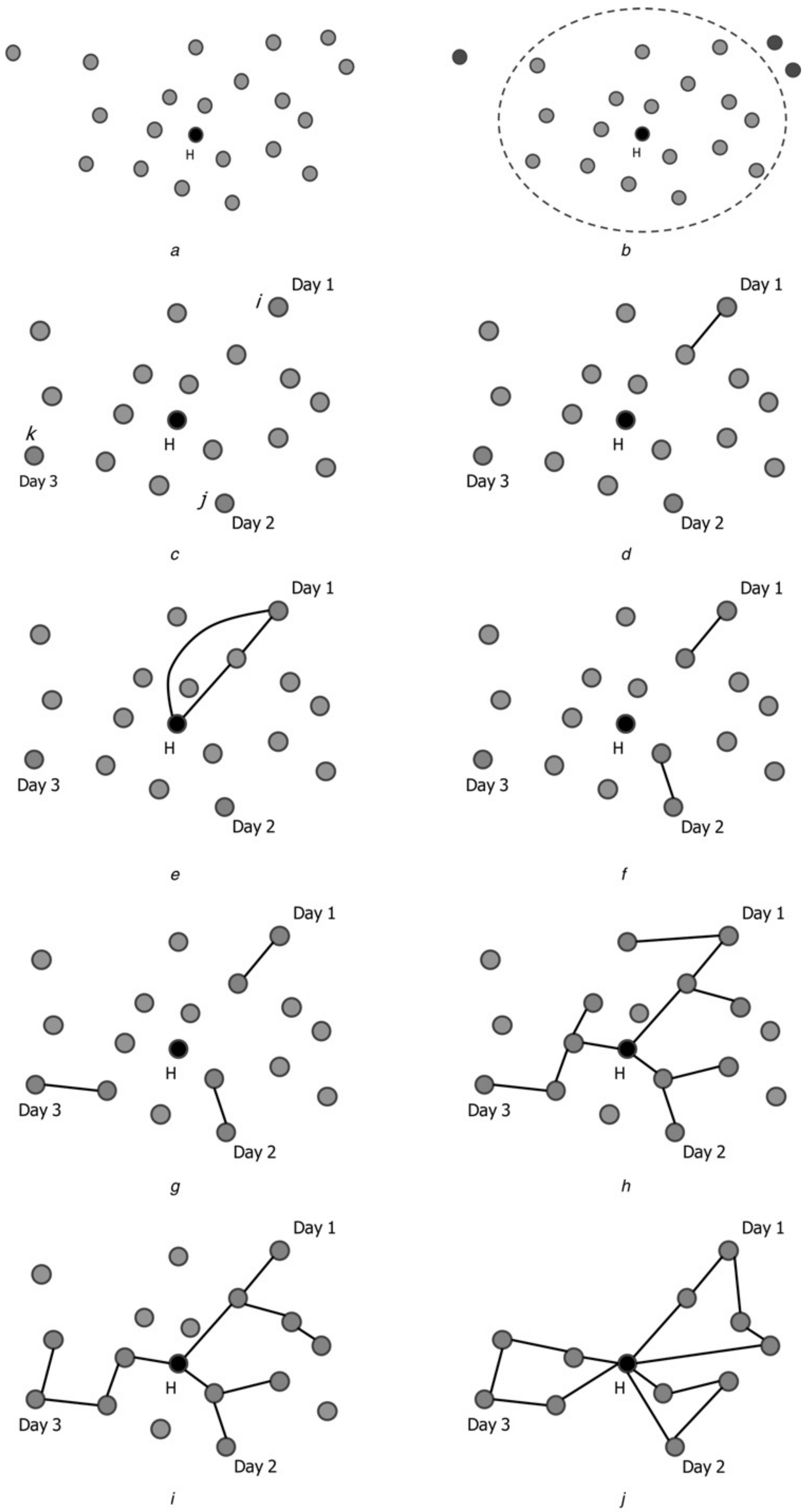


Fig. 2 *Execution phases of DailyTRIP*

this approach is to achieve a geographical 'segmentation' of the tourist destination in separate zones, as it makes sense, for example, for a tourist who visits a city for 2 days, to focus on POIs located on the northern part of the city on his/her first day and on POIs located on the southern part on his/her second day.

Phase 4: Construction of itinerary trees: On each of the following algorithm's steps, itineraries I_i are considered interchangeably incorporating a new node N not yet included in any of the I_i. In particular, for each I_i, the candidate node N with the minimum connection cost $C_{j,N}$ to any of the nodes $j \in I_i$ joins I_i (through accepting the $j \to N$ edge), given that the daily time budget T condition is not violated for this itinerary (see Fig. 2*d*). Notably, as the candidate node N may be connected to any of the I_i nodes (i.e. not necessarily to the edge nodes of the itinerary), I_i grows as a tree structure rather than a multipoint line. The time T_i corresponding to the completion of the itinerary I_i is calculated first by temporarily connecting H with the I_i node nearest to H, then converting the I_i itinerary tree to a multipoint line (through a post-order tree traversal) and finally calculating $T(I_i) = t_{H,1} + \Sigma_{k=1}^{|I_i|}[(t]_v(k) + d_{k,k+1})$. Namely, $I_i = I_i \cup N$, if $T(I_i) \leq T$. This is illustrated in Fig. 2*e*.

Hence, on each step itineraries I_i grow interchangeably (see Figs. 2*f* and *g*), typically approaching the start/end point H, until no further insertion is feasible. Upon completion, each itinerary is connected to the 'hotel' node H, that is, the edge $j \to N$ is accepted, where j is the itinerary's node nearest to H (see Fig. 2*h*).

It is noted that the acceptance of candidate nodes also depends on the corresponding POIs' scheduled visiting days. In particular, for each joining node i that may not be visited during the days $D_c(i)$, the 'excluded' days of the itinerary I joined by i is adapted excluding those days: $D_c(I) = \cup_{i=1}^{|I|} D_c(i)$, signifying that during those days the itinerary is not feasible either. Apparently, a POI i may join an itinerary I if the intersection of their valid days (i.e. those when visiting is feasible) is not null and also this intersection includes at least one of the D days of visit, namely if $D_v(I) \cap D_v(i) \cap D \neq \phi$.

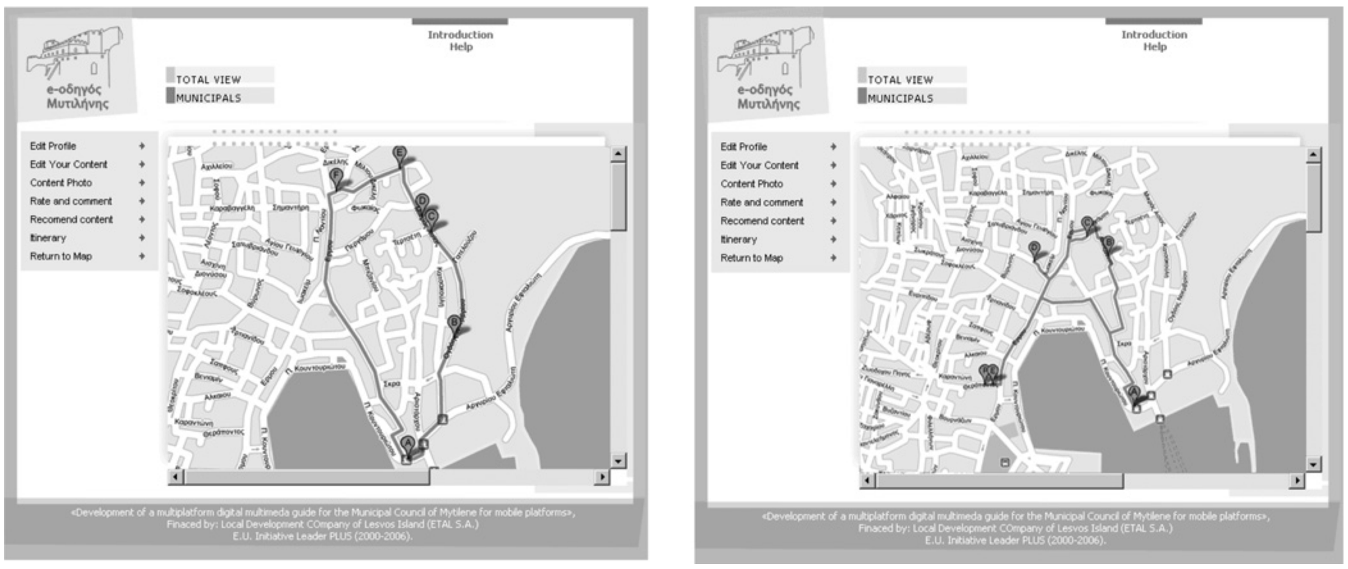
Phase 5: Rearrangement of itinerary trees: Phase 5 is optional and aims at improving the solutions derived in the previous phase, that is,. either increasing the overall profit or maintaining the same profit while reducing the itinerary completion time $T(I)$ (see Fig. 2*i*). Improved solutions are searched for every itinerary by: (a) substituting each itinerary tree node by any node not included in any itinerary at the end of the previous phase, and (b) by swapping nodes included on different itineraries. In any case, the new itinerary solutions should satisfy the daily time budget constraint.

Phase 6: Traversal of itinerary trees: Notably, the outcome of the previous phases is not a set of itineraries, but rather a set of itinerary trees. Hence, the last phase of DailyTRIP involves the conversion of the $|D|$ trees to multipoint lines I_i through executing a heuristic TSP algorithm [22] upon each itinerary tree (see Fig. 2*j*).

5 Implementation details of DailyTRIP

DailyTRIP has been developed using JSP/MySQL web technologies and Google Maps as the main user interface. The user first provides some personal demographic data and preferences upon tourist content items, that is, the user may state preference in visiting museums, archaeological sites, monuments etc. Further, the user points the location H of his accommodation, the period of visit, the hours available for visit, the means of transport and the radius around the hotel he is willing to move in order to visit a POI. In the mobile application case, the user is provided the option to receive recommendations for itineraries that originate at the current location (instead of the hotel in which the user is staying). The user is then shown a list of the initially selected POIs based on his preferences, which he is allowed to modify adding/ removing POIs.

The algorithm filters the POIs left out from the problem's space (because of their distance from the user's accommodation, their incompatibility with the user's preferences or their intentional removal by the user) and populates the travelling time matrix $t_{i,j}$ for the remaining nodes through first computing the distance matrix entries and considering the average expected velocity v of the selected means of transport. Distances among pairs of nodes are found by means of using the shortest-route



«Development of a multiplatform digital multimedia guide for the Municipal Council of Mytilene for mobile platforms», Financed by: Local Development COmpany of Lesvos Island (ETAL S.A.) E.U. Initiative Leader PLUS (2000-2006).

Fig. 3 *Output of DailyTRIP for two daily itineraries on Google Maps (point 'A' denotes the user's accommodation location, that is, the start/ end point of the two itineraries)*

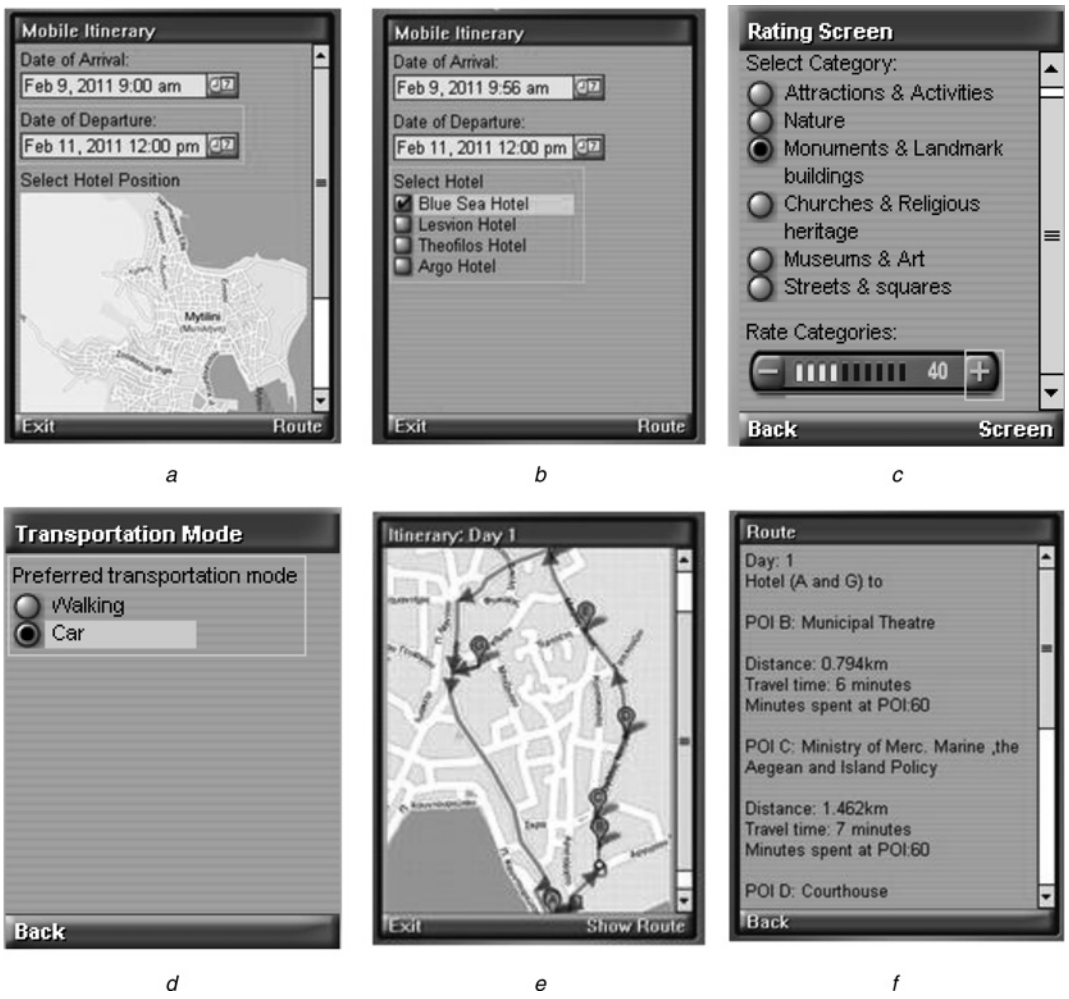
functionality of the Google Maps API [23] which refers to Manhattan distances and takes into account one-way roads.

Our implementation is based on the following assumptions: (a) each daily itinerary starts and ends at the same node, which typically coincides with the user's accommodation; (b) among all possible routes between a pair of nodes we only consider the shortest route in terms of length, although this might not be shortest in time; (c) the user is assumed to move with constant velocity regardless of the traversed edge or the time of day (admittedly, this is a valid assumption only for tourists walking around a city); and (d) the POIs are assumed to be open for visiting during the hours available to the tourist for sightseeing.

As stated in Section 4, Phase 6 of DailyTRIP involves the conversion of the $|D|$ trees to multipoint lines through a heuristic TSP algorithm. The latter is based on the Lin–Kernighan heuristic [22] and was implemented in Java. The Lin–Kernighan heuristic is considered as one of the most efficient algorithms for deriving near-optimal solutions for the symmetric TSP.

The output of DailyTRIP is sketched on a Google Maps interface, with each itinerary drawn on separate screen and the order of visiting POIs denoted by the alphabetical order of characters representing POIs (see Fig. 3). The maps derived by the web application are then converted to static images using the Google Static Maps API [24] in order to display on mobile phone screens. The mobile application (see Fig. 4) prompts the user for specifying the period of stay at the destination, determining the start/end location of the itineraries (that may be the current location of the user), rating content categories and pointing the preferred transportation mode (either walking or private vehicle). Evidently, the algorithm is straightforward to customise, enabling users to effortlessly prescribe their personal preferences. Certainly, users declaring different preferences will be recommended different tourist daily routes. The derived routes are then displayed upon a static map, accompanied by a textual description.

It is noted that mobile users may update their itineraries at any time, if the deviate from the originally calculated itineraries. Those updates take into account contextual parameters, such as the remaining time budget and the sites already visited by the user. Currently, we are working on an intelligent mechanism that will trigger automated itinerary updates when significant deviations from the original schedules are observed.


Mobile Itinerary
Date of Arrival:
Feb 9, 2011 9:00 am
Date of Departure:
Feb 11, 2011 12:00 pm
Select Hotel Position
Exit Route

Mobile Itinerary
Date of Arrival:
Feb 9, 2011 9:56 am
Date of Departure:
Feb 11, 2011 12:00 pm
Select Hotel
[✓ Blue Sea Hotel]
Lesvion Hotel
Theofilos Hotel
Argo Hotel
Exit Route

Rating Screen
Select Category:
Attractions & Activities
Nature
[✓ Monuments & Landmark buildings]
Churches & Religious heritage
Museums & Art
Streets & squares
Rate Categories:
40
Back Screen

Transportation Mode
Preferred transportation mode
Walking
[✓ Car]
Back

Itinerary: Day 1
Exit Show Route

Route
Day: 1
Hotel (A and G) to

POI B: Municipal Theatre

Distance: 0.794km
Travel time: 6 minutes
Minutes spent at POI:60

POI C: Ministry of Merc. Marine ,the Aegean and Island Policy

Distance: 1.462km
Travel time: 7 minutes
Minutes spent at POI:60

POI D: Courthouse
Back

Fig. 4 *Screens of the DailyTRIP mobile web application*

a Accommodations' selection through a map interface
b Accommodations' selection through a set of predefined choices
c Rating of POI categories (0–100 scale, with step 20)
d Selection of preferred transportation mode
e Visualisation of a daily itinerary
f Textual description of a daily itinerary

6 Qualitative comparison of DailyTRIP against Iterated Local Search (ILS)

ILS [11] is so far the only known TOPTW algorithm explicitly developed to derive solutions for the TIDP. Hence, ILS serves as a benchmark against which we compared the results of DailyTRIP. This section compares the approaches taken by ILS and DailyTRIP algorithms. ILS comprises two separate steps:

1. An insertion step adds, one by one, new visits to a tour; before an extra visit is inserted in a tour, it is verified that all visits scheduled after the insertion place still satisfy their time window (i.e. all visits are completed within POIs' opening hours) and the overall daily time budget is not exceeded. Among feasible insertion points, the most appropriate, that is, the one with minimum time overhead (for travelling and visiting) is calculated. Finally, the visit that maximises a 'profit to time cost' ratio is inserted into the route.
2. A shake step is performed to escape local optima: one or more visits are removed from each tour and the insertion step is repeated until no further insertion is feasible. If the new solution found improves the so far best found, the former is kept as the new best found solution. The shake and insert process is repeated up to 150 times, if no improved solution is derived.

Overall, ILS represents a fair compromise in terms of speed against deriving routes of reasonable quality. On the other hand, DailyTRIP takes a different approach in search of improved solution, without sacrificing the algorithm's capacity to satisfy real-time application requirements. The differences between the two approaches are summarised in the following:

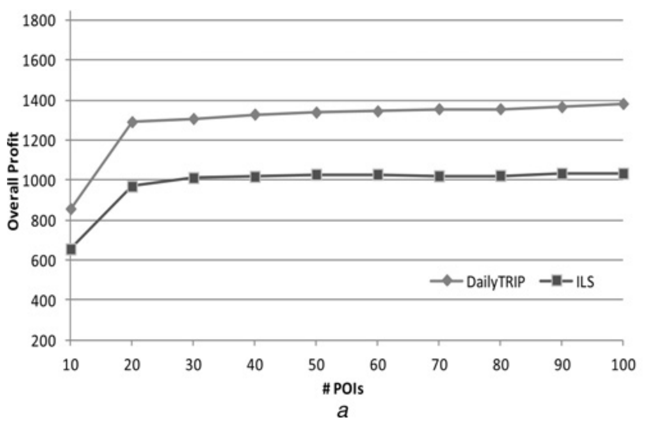
- On its first (insertion) step ILS rules out candidate nodes with high profit value as long as they are relatively time expensive to reach (from nodes already included in routes). This is also the case even when whole groups of high profit nodes are located within a restricted area of the plane, but far from the current route instance. In case the route instance gradually grows and converges towards the high-profit nodes, those may be no longer feasible to insert (because of time constraints). On the other hand, DailyTRIP creates routes that cover a broader area of the topology (since initial routes' nodes are far from the hotel site and from each other), which makes it more likely to approach and include a high-profit node.
- DailyTRIP explicitly takes into account both travelling and visiting time into its objective function (i.e. it employs a multi-criteria objective). That promotes the side-objective of designing itineraries with reduced total time cost. In contrast, ILS exclusively considers collected profit and regards travelling and visiting times only as constraints in the insertion step.
- DailyTRIP considers node insertions interchangeably among constructed itineraries, wheras ILS first finalises each itinerary before proceeding to the next one. In several topology scenarios, this may lead ILS to incorporating nodes to the wrong itinerary, while those nodes could find a better itinerary match (i.e. yield decreased route completion times) if connected to another itinerary. A side-effect is that ILS itineraries are unbalanced, as 'important' POIs are absorbed into the first routes.
- DailyTRIP swaps nodes included on different itineraries on its rearrangement step (Phase 5); this often leads to route instances that may accommodate additional nodes, thereby increasing the overall collected profit. In contrast, ILS does not consider such restructuring.

7 Simulation results

Simulation results have been executed on a Java-based simulation tool, which takes into account most of the TIDP parameters discussed in Section 4 and includes a visual interface for displaying the step-by-step output of the investigated algorithms. The simulator allows easy specification of simulation parameters and graphically illustrates the output of the DailyTRIP and ILS algorithm, while also recording their respective overall itinerary length, visit order of POIs and respective travel times (for the same topologies and TIDP parameter values). It also takes into account a number of constraints, for instance, the attributes a_1, a_2, a_3 that determine the weight of travelling time, visiting time and profit of POIs in the calculation of the cost matrix (see Section 4.1), the number of days to visit, the minimum/maximum weight to assign to each POI and the minimum/maximum time to visit a POI (stay time). The assignment of weight and stay time for each POI follows a uniform distribution. The network topology (i.e. POIs' coordinates) is randomly generated, given the dimensions of the city field, while the start/end point's position is explicitly set by the user.

Unless otherwise stated, in all our simulation tests, the weight of each POI is a random number between 20 and 100, the time a user can spend at a POI ranges from 5 to 90 min, while daily itineraries start at 09:00 and end at 18:00. In order to denote a high importance with regard to distance, the distance coefficient has been set to 20% (0, 2), whereas the profit coefficient was set to 80% (0, 8) to increase the importance of profit in the POIs selection process. As a result, the visiting time coefficient was set to 0. Since ILS takes into account time windows (i.e. opening hours) of POIS, and DailyTRIP only considers opening days, we have set ILS opening hours from 09:00 to 18:00 to allow a direct comparison between the two algorithms. The simulation results presented herein have been averaged over ten simulation runs (i.e. for ten different network topologies) to ensure statistical validity. Simulation tests have been performed on a single-core P4 2.8 GHz with PC 2 GB RAM.

Fig. 5*a* illustrates the overall profit for the DailyTRIP in comparison with ILS as the number of POIs increase. For this simulation we assumed 2 days of stay (i.e. two itineraries). The overall profit of DailyTRIP is shown to improve the solutions obtained from ILS, mainly because of ILS tendency to overlook 'important' POIs located relatively far from the constructed routes' instances (see relevant discussion in Section 6). It should be stressed that the total itinerary profit of DailyTRIP very much depends on the value of profit coefficient, that is decreasing the profit coefficient results in decreased total profit (it signifies that the profit of POIs to visit becomes less important) and vice versa. Both algorithms' curves increase more mildly when crossing a threshold of around 20 POIs. This is because no more POIs may be accommodated within the two itineraries (typically a maximum of ∼10 POIs can fit per itinerary). However, larger topologies create opportunities to substitute some POIs with others having



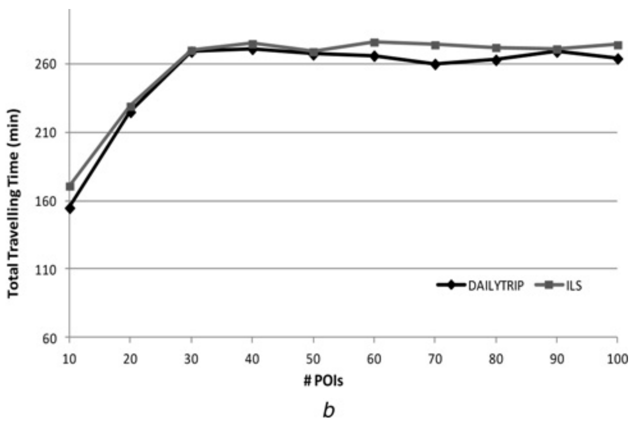


Fig. 5 *Comparison of the overall profit and total travel time*

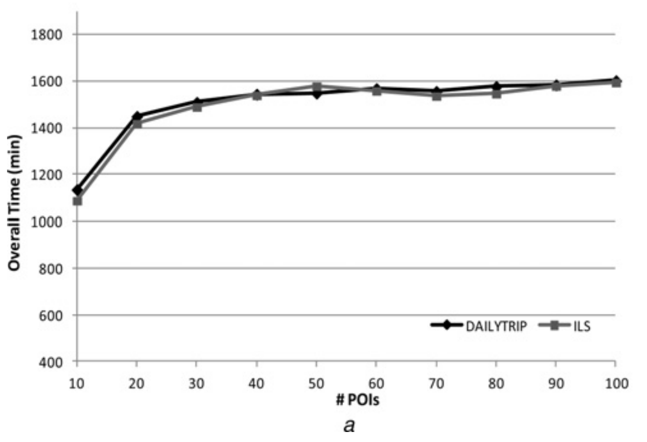
a Comparison of the overall profit of DailyTRIP against ILS for two itineraries
b Comparison of the total travel time derived from DailyTRIP and ILS for three itineraries

larger profit values, hence, larger topologies yield incremental overall profit values.

Fig. 5*b* shows the total travel time for 3-day tours (i.e. three itineraries) as a function of the problem space (i.e. number of POIs). Clearly, as the number of POIs increase (i.e. itineraries are prolonged) so does the total travel time, until a 30 POIs threshold is crossed. The DailyTRIP yields slightly improved results, which is mainly because of the execution of the TSP algorithm on its last phase ensuring near-optimality of each daily itinerary in terms of length. Another reason is that ILS only considers POIs' profit as its only optimisation criterion, whereas DailyTRIP takes a multi-objective approach, considering both profit and topological distance (i.e. travelling time). However, the total travelling time of DailyTRIP very much depends on the value of distance coefficient, that is decreasing the distance coefficient results in increased total itinerary length (it signifies that the time spent by the user for travelling is less important) and vice versa. That represents an interesting trade-off among profit and travelling time.

Fig. 6*a* compares DailyTRIP against ILS in terms of their respective overall itinerary time. That sums up the overall time spent for travelling (i.e. moving from one POI to another) and visiting POIs. As expected, the overall time yield for both algorithms converges to 1620 min (three daily itineraries, each 9 h long). Although the travelling time of DailyTRIP is slightly shorter (see Fig. 5*b*), the opposite is true for the overall itinerary time. This is due to the effective DailyTRIP rearrangement phase which allows accommodating a larger number of POIs into derived itineraries and, therefore increases the total visiting time. Fig. 6*b* compares the total itineraries length of DailyTRIP and ILS (that is the sum of individual daily itinerary lengths). Evidently, the illustrated distance values are proportional to the travelling times of Fig. 5*b*. It is noted that on those tests we assume walking transportation mode, with an average walking speed of 4.5 km/h.

Fig. 7*a* portrays the variance among overall daily profits (i.e. the sum of profits of POIs visited on each day). Herein we assume four daily itineraries (i.e. staying 4 days at the destination). Small variance values denote fairly balanced itinerary profit values over the four itineraries whereas larger variance values denote that some itineraries are much more 'profitable' than others. Notably, DailyTRIP yields smaller variance values in comparison with ILS for small-scale topologies. This is because DailyTRIP considers the four itineraries interchangeably while inserting new nodes, hence derived itineraries comprise approximately the same number of POIs (± 1), ensuring a balanced overall itinerary profits. On the other hand, ILS first completes an itinerary



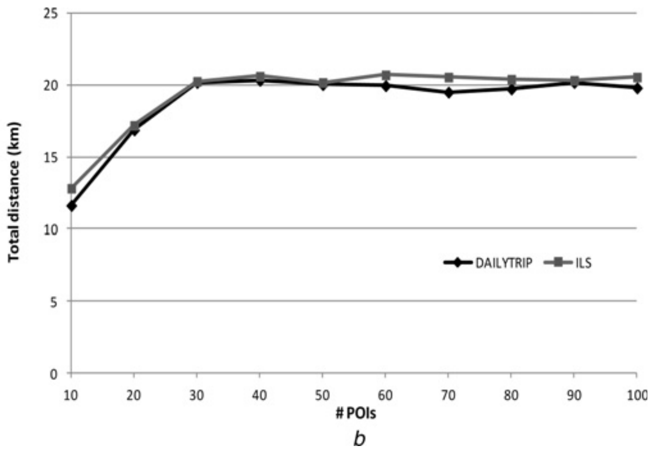


Fig. 6 *Comparison of the overall time and total distance*

a Comparison of the overall time derived from DailyTRIP and ILS algorithms for three itineraries
b Comparison of the total distance for three itineraries

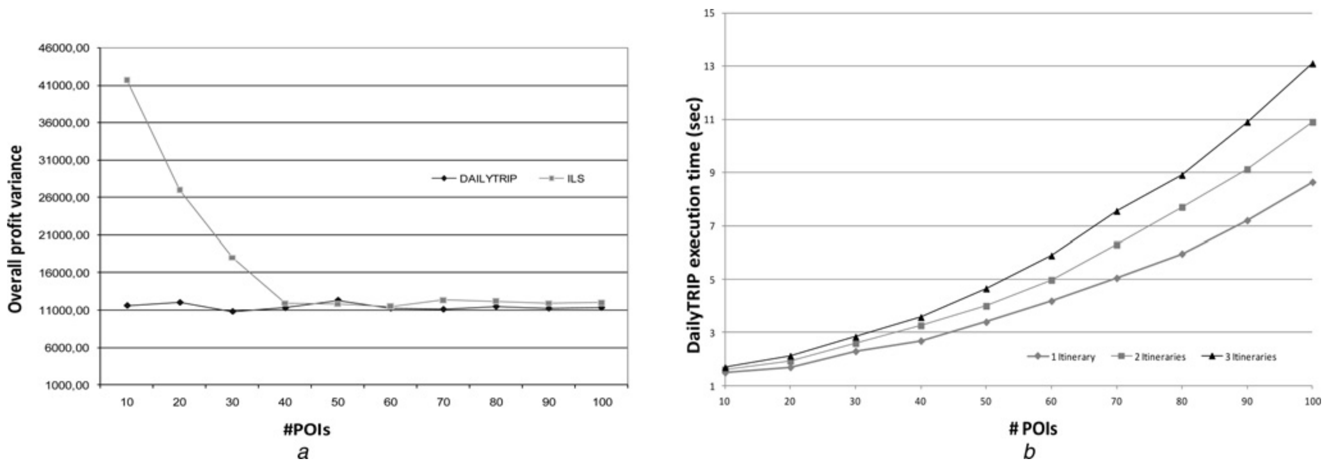


Fig. 7 *Variance and execution time*

a Variance of the individual itineraries' overall profit per day for both algorithms (four daily itineraries)
b Execution time of DailyTRIP for a varying number of POIs and itineraries

before proceeding to the next one, which implies that in scenarios with few POIs and relatively large number of itineraries, ILS derives some 'full' and some 'empty' itineraries (hence, large variance values). In other words, DailyTRIP distributes visited POIs over all available days of stay, whereas ILS derives 'overloaded' days in the beginning and relatively 'relaxed' days at the end of the staying time. Notably, as the number of POIs increase, ILS obtains improved distributions of POIs into individual itineraries (i.e. its itineraries are gradually 'filled' with POIs) which, in turn, decrease variance values.

Last, Fig. 7*b* illustrates the execution time of DailyTRIP for an increasing number of POIs. Despite the time increment as the problem space increases, DailyTRIP appears suitable for online usage in realistic scenarios. In particular, the algorithm requires less than 2 s for topologies spanning up to 25 nodes and less than 4.5 s for topologies up to 50 POIs, which represents a reasonable number of POIs for a medium-to-large-scale city. Clearly, performance also depends on the number of derived itineraries: the TSP algorithm is executed once per itinerary, while the number of swaps tried between nodes included on different itineraries increases proportionally. It is noted that we currently work on speedup techniques to further improve DailyTRIP performance. That involves the implementation of a faster technique to identify the initial itinerary nodes (Phase 3) as well as a more efficient TSP algorithm (appropriate pre-calculation techniques are also investigated); it also involves restrictions on nodes swapped in Phase 5 so that only nodes located in proximity (i.e. those that are more likely to derive improved solutions) are exchanged.

8 Conclusions and future research

This paper introduced DailyTRIP, a heuristic approach for deriving personalised recommendations of daily tourist itineraries for tourists visiting any tourist destination. The algorithm targets both web and mobile users, with the latter offered location-aware recommendations. DailyTRIP considers selected POIs that a traveller would potentially like to visit and derives a near-optimal itinerary for the traveller for each day of visit. Our approach takes into account user preferences, time available for visiting sights on a daily basis, opening days of sites and average visiting times for these sites. The objective of DailyTRIP is to maximise the overall profit associated with visited POIs (where individual profits are calculated as a function of the POIs' 'objective' importance and the user's potential interest for the POI) while not violating the daily time budget for sightseeing. Our algorithm has been implemented and proved suitable for online applications (i.e. real-time design of itineraries), whereas simulation results validated its performance gain over ILS algorithm.

Our future research will focus on variations of DailyTRIP algorithm that will incorporate additional TIDP problem parameters and constraints, for example, weather conditions during travel, financial budget (for transport and POIs admission charges) etc. We will also investigate the use of a combination of transport modalities, for example, walking and bus service, taking into account various aspects of alternative transport services (e.g. walking time to the nearest metro station, time-dependent metro service frequencies etc.). Currently, we are working on an intelligent mechanism that will trigger automated itinerary updates when significant deviations from the original schedules are observed. Methods for fast, automated itinerary updates will also be considered, either for mobile users that have deviated from the suggested itinerary or because of sudden weather changes, unexpected public transportation schedule changes etc. Last, the mobile application will incorporate various contextual parameters, such as time of day, predicted queuing delay, parking availability at each POI etc.

9 Acknowledgment

This work was supported by the EU FP7/2007-2013 (DG INFSO.G4-ICT for Transport) under grant agreement no. 288094 (project eCOMPASS).

10 References

1 Dunlop, M., Ptasinski, P., Morrison, A., McCallum, S., Risbey, C., Stewart, F.: 'Design and development of Taeneb city guide – from paper maps and guidebooks to electronic guides'. Proc. Int. Conf. on Information and Communication Technologies in Tourism (ENTER'2004), 2004, pp. 58–64
2 Cheverst, K., Davies, N., Mitchell, K., Friday, A., Efstratiou, C.: 'Developing a context-aware electronic tourist guide: some issues and

experiences'. Proc. SIGCHI Conf. on Human Factors in Computing Systems, 2000, pp. 17–24

3 Cheverst, K., Mitchell, K., Davies, N.: 'The role of adaptive hypermedia in a context-aware tourist GUIDE', *Commun. ACM*, 2002, **45**, (5), pp. 47–51

4 Kenteris, M., Gavalas, D., Economou, D.: 'An innovative mobile electronic tourist guide application', *Pers. Ubiquit. Comput.*, 2009, **13**, (2), pp. 103–118

5 Kenteris, M., Gavalas, D., Economou, D.: 'Electronic mobile guides: a survey', *Pers. Ubiquit. Comput.*, 2011, **15**, (1), pp. 97–111

6 Malaka, R., Zipf, A.: 'DEEP MAP – challenging it research in the framework of a tourist information system'. Proc. Int. Conf. on Information and Communication Technologies in Tourism (ENTER'2000), 2000

7 Vansteenwegen, P., Van Oudheusden, D.: 'The mobile tourist guide: an or opportunity', *Oper. Res. Insight*, 2007, **20**, (3), pp. 21–27

8 Krüger, A., Malaka, R.: 'Artificial intelligence goes mobile', *Appl. Artif. Intell.*, 2004, **18**, pp. 469–476

9 Kramer, R., Modsching, M., ten Hagen, K.: 'A city guide agent creating and adapting individual sightseeing tours based on field trial results', *Int. J. Comput. Intell. Res.*, 2006, **2**, (2), pp. 191–206

10 Chao, I.-M., Golden, B.L., Wasil, E.A.: 'The team orienteering problem', *Eur. J. Oper. Res.*, 1996, **88**, (3), pp. 475–489

11 Vansteenwegen, P., Souffriau, W., Vanden Berghe, G., Van Oudheusden, D.D.: 'Iterated local search for the team orienteering problem with time windows', *Comput. Oper. Res.*, 2009, **36**, (12), pp. 3281–3290

12 Dumas, Y., Desrosiers, J., Gelinas, E., Solomon, M.M.: 'An optimal algorithm for the traveling salesman problem with time windows', *Oper. Res.*, 1995, **43**, (2), pp. 367–371

13 Vansteenwegen, P.: 'Planning in tourism and public transportation'. PhD thesis, Katholieke Universiteit Leuven, 2008

14 Kenteris, M., Gavalas, D., Pantziou, G., Konstantopoulos, C.: 'Near-optimal personalized daily itineraries for a mobile tourist guide'. Proc. 15th IEEE Symp. on Computers and Communications (ISCC'2010), 2010, pp. 862–864

15 Souffriau, W., Maervoet, J., Vansteenwegen, P., Vanden Berghe, G., Van Oudheusden, D.: 'A mobile tourist decision support system for small footprint devices'. Proc. Tenth Int. Workshop on Artificial Neural Networks (IWANN'2009), 2009, pp. 1248–12 55

16 Abbaspour, R.A., Samadzadegan, F.: 'Itinerary planning in multi-modal urban transportation network', *J. Appl. Sci.*, 2009, **9**, (10), pp. 1898–1906

17 Garcia, A., Linaza, M.T., Arbelaitz, O., Vansteenwegen, P.: 'Intelligent routing system for a personalised electronic tourist guide'. Proc. Int. Conf. on Information and Communication Technologies in Tourism 2009 (ENTER'2009), 2009, pp. 185–197

18 Maruyama, A., Shibata, N., Murata, Y., Yasumoto, K., Ito, M.: 'P-tour: a personal navigation system for tourism'. Proc. 11th World Congress on ITS, 2004, pp. 18–21

19 Hagen, K., Kramer, R., Hermkes, M., Schumann, B., Mueller, P.: 'Semantic matching and heuristic search for a dynamic tour guide'. Proc. Int. Conf. on Information and Communication Technologies in Tourism, 2005, pp. 149–159

20 City Trip Planner: http://www.citytripplanner.com/, accessed 30 January 2011

21 Google city tours: http://citytours.googlelabs.com/, accessed 30 January 2011

22 Helsgaun, K.: 'An effective implementation of the Lin–Kernighan traveling salesman Heuristic', *Eur. J. Oper. Res.*, 2000, **126**, (1), pp. 106–130

23 Google Maps. Google Maps API Concepts http://code.google.com/apis/maps/documentation/, accessed 30 January 2011

24 Google Static Maps API http://code.google.com/apis/maps/documentation/staticmaps/, accessed 30 January 2011