eCO-friendly urban Multi-modal route PlAnning Services for mobile uSers

FP7 - Information and Communication Technologies

Grant Agreement No: 288094
Collaborative Project
Project start: 1 November 2011, Duration: 36 months

D3.1 – New Prospects in Multi-modal Route Planning Problems

Workpackage:	WP3 - Algorithms for Multimodal Human Mobility
Due date of deliverable:	30 April 2012
Actual submission date:	30 April 2012
Responsible Partner:	CTI
Contributing Partners:	CTI, ETHZ, KIT, TomTom

Nature: ☒ Report ☐ Prototype ☐ Demonstrator ☐ Other

Dissemination Level:
☒ PU: Public
☐ PP: Restricted to other programme participants (including the Commission Services)
☐ RE: Restricted to a group specified by the consortium (including the Commission Services)
☐ CO: Confidential, only for members of the consortium (including the Commission Services)

Keyword List: Multi-modal, Multi-criteria, Route planning, Label-constrained shortest paths, Algorithm engineering, Public transport, Timetable information systems, Shortest Path, Uncertainty, Robustness, Delay Prediction, Timetable optimization, Tourist Trip Design Problem, Orienteering Problem, Time Window, Time Dependent Team Orienteering.

The eCOMPASS project (www.ecompass-project.eu) is funded by the European Commission, Information Society and Media Directorate General, Unit G4-ICT for Transport, under the FP7 Programme.

The eCOMPASS Consortium

Computer Technology Institute & Press 'Diophantus' (CTI) (coordinator), Greece

Centre for Research and Technology Hellas (CERTH), Greece

Eidgenössische Technische Hochschule Zürich (ETHZ), Switzerland

Karlsruher Institut fuer Technologie (KIT), Germany

TOMTOM INTERNATIONAL BV (TOMTOM), Netherlands

PTV PLANUNG TRANSPORT VERKEHR AG. (PTV), Germany

Document history			
Version	Date	Status	Modifications made by
0.3	05.03.2012	ToC	Damianos Gavalas, CTI Yann Disser, ETHZ Julian Dibbelt, KIT
0.7	14.03.2012	First Draft	Damianos Gavalas, CTI
0.8	03.04.2012	Second Draft	Yann Disser, ETHZ
0.9	06.04.2012	Third Draft	Julian Dibbelt, KIT
1.0	10.04.2012	Sent to internal reviewers	Damianos Gavalas, CTI
1.1	25.04.2012	Reviewers comments incorporated; sent to PQB	Damianos Gavalas, CTI
1.2	27.04.2012	PQBs comments incorporated	Damianos Gavalas, CTI
1.3	30.04.2012	Final (approved by PQB, sent to the Project Officer)	Christos Zaroliagis, CTI

Deliverable manager

- Damianos Gavalas, CTI

List of Contributors

- Katerina Böhmova, ETHZ
- Julian Dibbelt, KIT
- Yann Disser, ETHZ
- Damianos Gavalas, CTI
- Kalliopi (Lina) Giannakopoulou, CTI
- Felix König, TomTom
- Charalambos Konstantopoulos, CTI
- Georgia Mali, CTI
- Konstantinos Mastakas, CTI
- Grammati Pantziou, CTI
- Ioannis Tasoulas, CTI
- Christos Zaroliagis, CTI

List of Evaluators

- Dionisis Kehagias, CERTH
- Peter Widmayer, ETHZ

Summary

The purpose of the present deliverable is to present recent advances, their possible limitations, and new prospects for algorithmic techniques and methodologies concerning the problems related to multi-modal route planning in urban areas. The focus is on state-of-art approaches that take into account both timetabled public transportation services (such as bus, tram or metro) and non-timetabled modes of transportation (such as walking, biking, private car navigation or taxi), accommodate uncertainty of traffic data and arrival delays and consider a variety of optimality criteria such as earliest arrival, travel-time, number of transfers or price. The deliverable also addresses route planning needs of tourists visiting a destination, deriving personalized recommendations for daily sightseeing tours.

Contents

1 Introduction

The aim of this deliverable is to present recent advances, their possible limitations, and new prospects for algorithmic techniques and methodologies concerning the problems related to multi-modal route planning in urban areas, which are studied in WP3.

1.1 Objectives of WP3

WP3 addresses the objective of reducing the environmental impact of human mobility in urban spaces through laying the theoretical ground for implementing intelligent web and mobile services that promote and encourage the use of urban public transportation. Namely, it is based on the valid hypothesis of the environmental friendliness of using public transportation instead of private vehicles. Along this line, WP3 investigates public transportation-based human mobility optimization problems, initiating and exploring new modeling and algorithmic perspectives, which will serve as a feeder for other WPs. From a high abstraction level, in WP3 -along with WP2- we distil and generalize the inherent mathematical and algorithmic ideas arising in the overall project. As such, WP3 identifies new developments, evaluates their potential and feeds generalizations back to all partners. WP3 comprises three main threads of research, with increasing scientific challenge, innovation, and risk:

- Development of models and efficient algorithmic approaches to the problem of multi-modal point-to-point route-planning of humans in urban environments, targeting both server-based back-ends as well as mobile devices.
- Development of new models that incorporate stochasticity and adapt methods from machine learning that yield multi-modal, urban routes with higher accuracy and reliability.
- Development of models and algorithmic solutions for context-aware multi-modal daily route-planning problems for tourists visiting multiple points of interests (tourist trip design problems, TTDP).

WP3 addresses challenging questions regarding the tradeoffs between effectiveness (the deviation from the optimal solution) and efficiency (the complexity algorithmic approaches). The common goal for all route planning optimization approaches dealt with in WP3 is to ensure low complexity on the derived solutions so as to be applicable to online and mobile applications.

1.2 Scope of this deliverable

The present deliverable, D3.1, is the first part of the work in WP3, comprising the output of Task 3.1. The main objective of D3.1 is to document the specific fundamental structure of the aforementioned route planning optimization problems and establish the ground for the development of efficient algorithmic solutions. We investigate the theoretical foundations of route planning and present recent advances in the field with respect to approaches which deal with independent and complementary parts of several of our problems. The deliverable also highlights the limitations of existing methods and suggests new prospects for algorithmic techniques and methodologies concerning the problems related to multi-modal route planning.

The focus of the deliverable is on state-of-art approaches in computing journeys over transportation networks (considering both timetabled and non-timetabled modes of transportation); several optimization criteria are considered, while accommodating uncertainty of public transportations and looking for robust solutions, possibly with meaningful alternatives. Recent approaches with relevance to the TTDP are also examined, focusing on problem models that best capture a multitude of realistic user constraints, while also investigating several TTDP variants.

1.3 Structure of the Document

Section 2, gives an overview of the state-of-the-art in multi-modal route planning. In multimodal route planning the objective is to present a user one or several alternative journeys between two arbitrary locations of his/her choice. These journeys are to be computed on a combined transportation network of both timetabled public transport services (such as bus, tram or metro) and non-timetabled modes of transportation (such as walking, biking, private car navigation or taxi). In order to be useful these journeys have to be optimal or close to optimal. Common optimality criteria are earliest arrival, travel-time, number of transfers or price. Furthermore, the user should be able to choose preferred modes of transportation or otherwise constrain the modes of transportation considered during computation (e.g., because the user does not own a bike or does not like to walk very far).

In Section 3, we give a summary of literature dealing with uncertainty in multimodal networks. Most prominently, we consider route planning in public transportation networks. We also give an overview over the closely related fields of delay prediction and timetable optimization. For each problem we outline different approaches for capturing robustness of a route with respect to stochastic disturbances; and for finding optimum robust routes in this regard.

In Section 4 our focus is on algorithmic approaches that address the route planning requirements of tourists that visit an urban destination for a number of days. The main objective is to maximize the tourist satisfaction (i.e., account for including visits to points of interest (POIs) mostly interesting to the user within her daily tours) while taking into account a multitude of parameters (e.g., distances among POIs, visiting time required for each POI, POIs visiting days/hours, entrance fees, weather conditions) and respecting the time available for sightseeing in daily basis. Tourists are assumed to use all modes of transport available at the tourist destination, including public transportation, walking and bicycle.

Finally, in Section 5 we highlight new prospects in multimodal route planning research. We refer to all problems dealt with in Sections 2-4, discussing promising research directions on the course of the WP3 of eCOMPASS.

2 Multi-Modal Route Planning

2.1 Motivation

Multi-modal route planning considers the problem of computing good journeys through a combined network of both road and public transport networks. Depending on the scenario these journeys can be any sequence of modes of transportation, such as walking, biking, driving a private car, taking the taxi, taking a bus or tram, a train or a fast long-distance train. Multi-modal route planning has important applications in online journey planning systems (where several alternative journeys are presented to the user) while also providing input to vehicle routing problems (where the "distance" between a large number of locations needs to be known) and others.

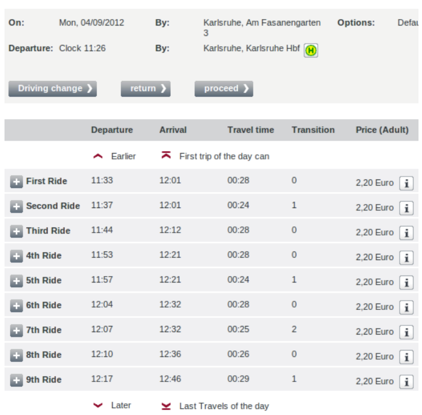
Work package WP3 of the eCOMPASS project aims to solve several questions arising in urban multi-modal scenarios, most importantly the computation of environmentally friendly journeys. These scenarios will involve several layers of transportation as described above. Criteria for good routes are the minimization of emissions, earliest arrival time, total travel-time, number of transfers or price. Other criteria could be the minimization or maximization of time spend in a certain mode of transport (e. g. less walking, less driving, more time spend in public transport reading a good book). Because different users weight different criteria differently, several journeys should be computed and presented to the user for evaluation. Also, the user should be able to guide the use of modes of transportation in these routes (e.g. no car!, no bike!). Section 2 is outlined as follows: In subsection 2.1, we give an extended motivation. In subsection 2.2, we discuss current commercial timetable information systems and journey planners. In Section 2.3, we review recent research in the field of multi-modal route planning. In subsection 2.3.1 we detail recent developments in unimodal route planning for public transport networks (for unimodal route planning for road networks see Deliverable D2.1). In subsection 2.3.2 we outline different techniques that extend known algorithmic results from unimodal route planning to route planning in multi-modal combined networks composed of several timetabled and non-timetabled modes of transportation. Finally, subsection 2.3.3 discusses the weaknesses identified in existing approaches.

2.2 Web Tools for Multi-Modal Route Planning

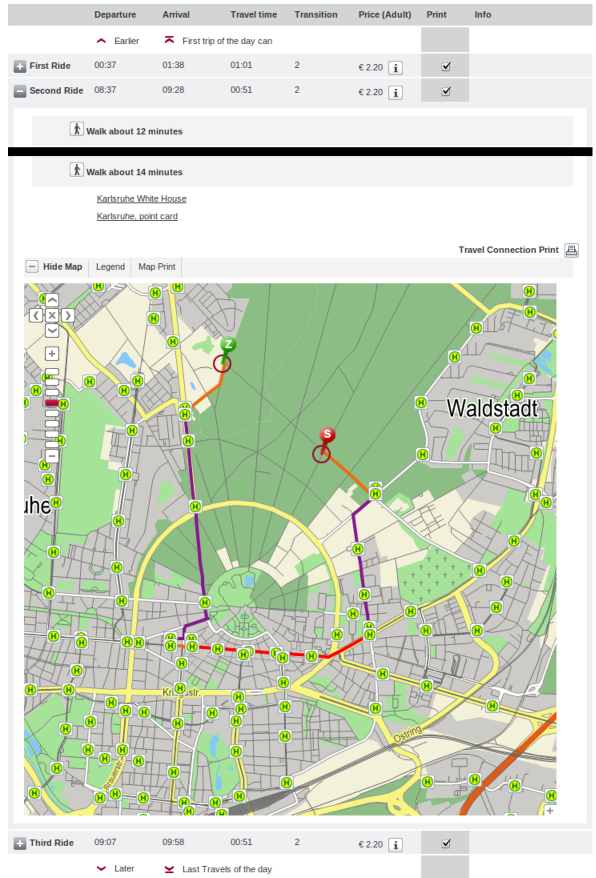
Web-based timetable information and route planning services have become quite common in recent years. Many local public transportation operators have their offering but also larger organization such as Deutsche Bahn (the German national train operator) or Google offer transportation route planning. In this section, we review three such services: kvv.de of the public transportation provider in Karlsruhe, Germany, youdrive.gr in Greece as well as Google Transit.

A typical result of kvv.de can be seen in Figure 1(a). Input to the query are the day, the departure (or arrival) time, the source stop or street address and the destination. The result shows several alternative journeys departing in an expandable time window. For every journey travel time, number of transfers and the price is shown. Only journeys optimal in either departure time, travel time or transfers are presented (Pareto-optimal *multi-criteria time-range* query, see Section 2.3.1 for details). Routing on kvv.de considers real-time delay data of vehicles (where available). Computation is fast but eco-friendliness is neither evaluated nor an optimization criterion. Also, kvv.de is lacking in the multi-modal integration: it does not offer several alternatives based on different modes of transportation. Park-and-ride queries are not available and kvv.de cannot suggest to use rental bikes (Call-a-bike, a service by Deutsche Bahn, has several stations in Karlsruhe). Furthermore, multi-modal routing derived by a heuristic as can be seen in Figure 1(b): it would be faster, cheaper and emission-free to move by foot however this journey is not recommended.

As for the service offered by youdrive.org Figure 2 and 3 show the result of two queries. Travel-time, number of transfers and emissions are shown. Other than for kvv.de the query does not ask for a departure time and the result does not show several journeys within a time window. Rather, the routing seems to be frequency-based. This might be a good a idea for very frequent connections



(a) Query from Am Fasanengarten 5 to Main Station, Karlsruhe, Germany.

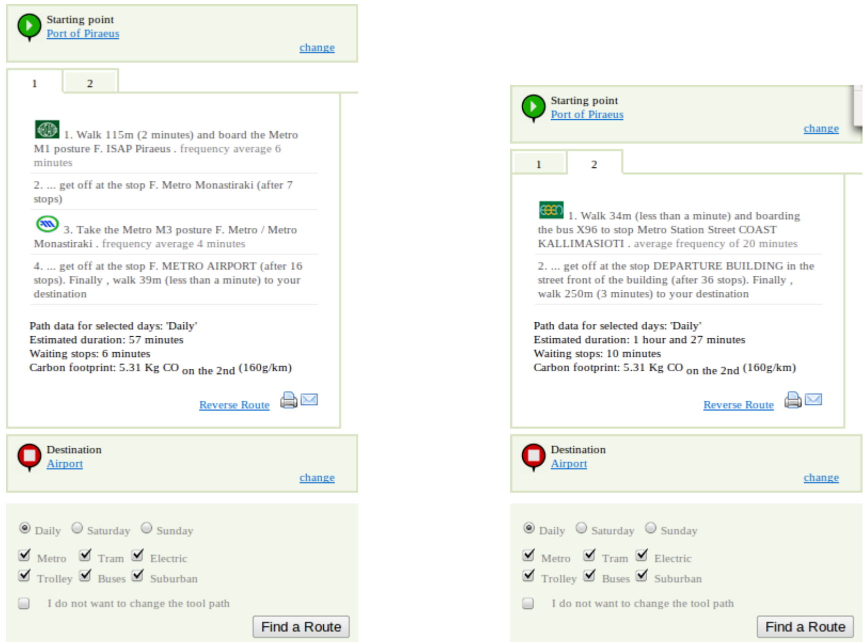


(b) Non-optimal multi-modal results: Last journey of the day at 00:37. First journey of the day at 9:28. Walking would be faster and possible all night.

Figure 1: kvv.de Route Planning Web Service. Results show departure, arrival and travel-time as well as number of transfers (transition) and price for several journeys within a time range.

but not in case of the second query as seen in Figure 3: having an average frequency of service of 34 minutes makes it really likely that the presented journey does not correctly reflect realistic connection behavior, likely causing very long waiting times. Also, youdrive.gr cannot account for real-time delay information. While youdrive.gr offers alternative journeys based on the number of transfers, typically these are only two. As with kvv.de, the multi-modal integration is very limited and restricted to short foot walks to and from the source/destination station. Though youdrive.gr shows an estimate of the emissions caused by the journey it is unclear whether the journey is actually optimized for low emissions or whether the journey is optimized for travel time and the emissions were just calculated for the resulting journey. Suspiciously, all alternative results of the same query share the same emission estimate despite using very different connections. It seems that emission are not optimized for and the calculation of emissions is heuristic at best.

Google Transit (transit.google.com or maps.google.com) is available for a growing number of cities around the world. As with kvv.de, queries account for departure (or arrival) time and day. Results show several alternatives in a time window based on travel time and number of transfers. Queries are very fast but cannot account for real-time delay data (they are or were based on very heavy preprocessing [29]). Environmental impact is neither shown nor optimized. The same applies



(a) 1st journey. Duration: 57 min, Emission: 5.31 kg CO_2.

(b) 2nd journey. Duration: 87 min, Emission: 5.31 kg CO_2.

Figure 2: youdrive.gr Route Planning Web Service. Query from Port Piraeus to Athens Airport, Greece. Computation takes about 7 seconds on average over five trials. Results show expected waiting, average service frequencies, estimated duration and carbon footprint.

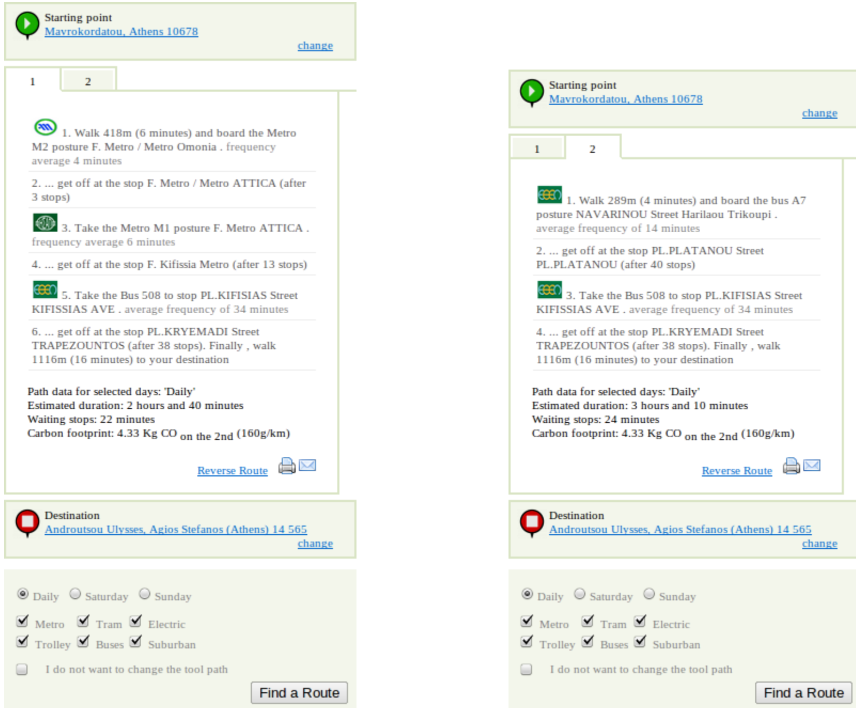
to fare costs. Rental bike stations (e. g. Barclays Cycle Hire, London) are not included.

In eCOMPASS our aim is to integrate all these features discussed so far into one single urban multi-modal eco-friendly route planning service prototype. In order to do so we have to investigate further the frontier of known algorithmic research, which we will discuss next.

2.3 State-of-the-Art Algorithms

In recent years research in the field of route planning has quickly evolved. Speed-up techniques achieve very fast queries on huge networks with provably optimal results. This was enabled by a two-step approach: in a *preprocessing* step specific information about the network is computed ahead of time and the routing graph annotated accordingly. The *query*—an extension of Dijkstra's Algorithm [70]— then uses the annotated data to achieve enormous query performance. On continental-sized road networks where Dijkstra's algorithm would take several seconds per query speedup techniques enable queries in less than one millisecond (250 ns for the currently fastest time-independent road technique [6]). It can be shown that the effectiveness of preprocessing is based on structural properties of road networks [7]. The structure of public transportation networks on the other hand is fundamentally different [30, 28].

Multi-modal routing requires solving the respective unimodal problems. Hence, in the following we will first give an overview of recent algorithmic results in route planning for public transportation networks. See Deliverable D2.1 for details on routing in road networks. Then we will discuss first results on extensions of unimodal road and transit routing to multi-modal scenarios and the new challenges that arise in this context.



(a) 1st journey. Duration: 160 min, Emission: 4.33 kg CO_2.

(b) 2nd journey. Duration: 190 min, Emission: 4.33 kg CO_2.

Figure 3: youdrive.gr Route Planning Webservice. Query from Mavrokordatou to Androutsou Ulysses, Agios Stefano, Athens, Greece. Computation takes about 22 seconds on average. Results show expected waiting, average service frequencies, estimated duration and carbon footprint.

2.3.1 Route Planning for Public Transportation Networks

The most important property that distinguishes public transportation networks from road networks is the fact that the underlying timetable of a transportation network introduces a specific form of time-dependency that needs to be accounted for in algorithmic approaches.

In the following we will discuss the most common approaches to model the time-dependency arising in timetables. Then, we will discuss common types of queries arising in route planning for public transportation networks. We conclude by an overview of recent algorithmic results in route planning for timetable networks.

Modeling of Timetables A *timetable* consists of a set of *stops* (also called *stations*), a set of *trips* (often called *trains*), a set of (elementary) *connections*, and a set of *transfers* (or *foot-paths*). A stop is a distinct location in the network where a traveler can get on or off a bus, train, etc., such as bus stops, train stations or train platforms (depending on the level of granularity the model supplies). A trip is the specific alternating sequence of stops that a vehicle serves and connections that connect these stops. Each connection corresponds to a departure time at its departure stop and an arrival time at its arrival stop. A trip must be consistent in time, i. e. each connection of the trip must depart later in time than it's preceding connection and must arrive earlier than it's succeeding connection. It is a common to cluster the trips into a set of *routes* such that all trips of a route visit exactly the same sequence of stops. Note that this concept of routes should not be confused with the output of a route planning algorithm (called route or journey).

For the purpose of journey planning routing algorithms the timetable of the network is commonly

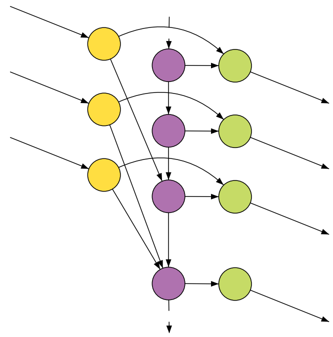


Figure 4: A station as in the realistic time-expanded model with minimum transfer time. Arrival nodes are yellow, departure nodes green, and transfer nodes purple.

transformed into a graph (as for road networks). There, a shortest journey corresponds to a shortest path in the graph which can be found by graph-based algorithms such as Dijkstra's Algorithm. Two approaches to model the time-dependency (introduced by scheduled departure and arrival events) of the timetable are common: time-expanded and time-dependent. See [147] for an overview.

In the *time-expanded* approach time is "rolled out": Each station is represented by a number of *arrival, transfer and departure nodes*, see Figure 4. Each departure node is labeled with the departure time of corresponding connections departing at this station. Each arrival node is labeled with the arrival time of corresponding connections arriving at this station. Transfer nodes are used to represent minimum transfer times and waiting at the station. There are exactly as many transfer nodes as departure nodes at each station and each transfer node is labeled by the time of its corresponding departure node. There are four types of arcs: Arrival and departure node of the same trip are connected within each station, weighted by the time the respective vehicle stays in the station waiting for passengers to get off and on. Departure node and arrival node that belong to the same connection are connected by a *travel arc* weighted by the travel-time of that connection. Each arrival node is also connected to a transfer node of length equal to the minimum transfer time. Each transfer node is connected to its departure node by an arc of zero length representing getting on a train. Within in each station, all transfer nodes are connected by *waiting arcs* that represent waiting on the next good connection.

The time-expanded timetable graph is directed and acyclic. Basic Dijkstra can be used to solve the shortest path problem. Yet, it is less commonly used in recent years, partly because the concept of trips and routes is badly represented in the expansion. See [65] for more details.

In the basic *time-dependent* model [40] for each station there exists a *station node* with elementary connections between station nodes. Each edge has a length *function*. These length functions are time-dependent, which means that the value of the function changes depending on the time t that this edge is used, representing the travel time of departing connections and the waiting time on these connections, see Figure 5.

The model of the graph also incorporates transfers within a station. The basic model is extended in [147] by firstly defining *routes* as a partition of T such that two connections are in the same route if and only if they share equal stations and properties. Using this partition, a station is represented by several *route nodes* in addition to its station node. There is one route node for each arrival or departure to and from the station. The route nodes are linked to their station node and to route nodes of adjacent stations. The links to their station nodes incorporate the transfer cost. Only edges leaving a route node towards the station node have a transfer cost. Edges from a station node to a route node have transfer cost equal to 0. This modeling can be seen in Figure 6.

In addition to transfer edges, constant-time foot paths between stations in geographic proximity

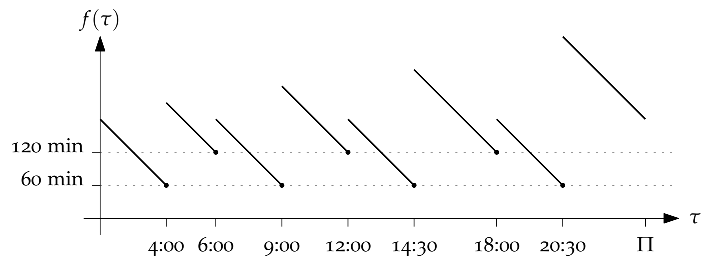


Figure 5: A time-dependent rail edge function representing seven departures: four fast trains with a travel-time of 60 minutes to the next stop, three slow trains with a travel-time of 120 minutes.

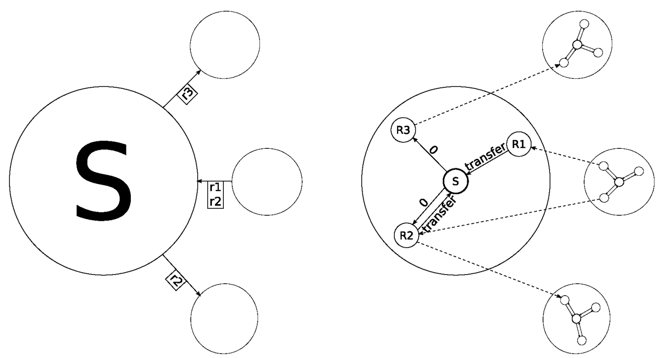


Figure 6: Extension of a simple time-dependent graph (left) to support transfers. The timetable has three routes r_1, r_2, r_3 so that the extended station (right) has three route nodes.

are incorporated in the model. These foot connections are linking special *foot nodes* that are added to the station graph. Finally special transfers that may depend on a sequence of trains may be incorporated by using a special transfer link between route nodes which is to be used only after arriving with a certain node. Such model can be seen in Figure 7.

On time-dependent models an extension of Dijkstra's Algorithm is used [40]. When ever it settles a route node and relaxes the edges towards the next route node it needs to evaluate the attached edge function. The result of that evaluation is the necessary waiting time plus the travel-time to arrive at the tail node fastest.

Besides these two general approaches, several models have been published that are specifically designed for certain scenarios. See [67] for an example on flight networks.

In the most recent work on routing in transportation networks the authors discard the transformation of the timetable into a graph altogether [68]. Instead they propose algorithms that directly work on the sets of stops, trips and routes. See below for a review.

Types of Queries Before turning to algorithms we shortly review common problem definitions arising in the field of public transportation routing. The most basic problem for timetable networks with respect to journey planing is the *Earliest Arrival Problem*: For each (s, t, τ)-query a journey through the network is computed that departs at source stop s after time τ and arrives at the destination stop t the earliest (of all journeys departing at s after τ). Note that this journey does not necessarily minimize travel-time. Still, it is useful to users already on the go.

More useful to users planning ahead are profile and time-range queries (s, t, τ_1, τ_2). They return

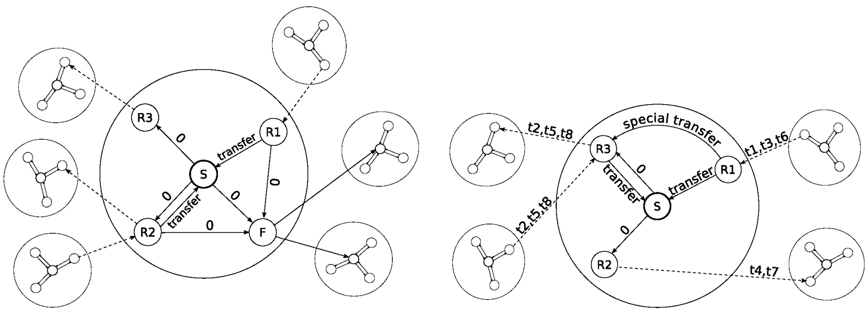


Figure 7: (a) Illustration of a station with two foot-edges in the time-dependent model (b) Modifications to the graph for a station with a special transfer from train t_1 to train t_2

for the time interval $[\tau_1, \tau_2]$ all Pareto-optimal journeys with corresponding departure and arrival time (τ_{dep}, τ_{arr}). Such a journey is Pareto-optimal if no other journey exists that departs later and arrives earlier. From the output of such queries the traveler can choose the journey that best fits her needs, e. g. the one journey that departs late enough but minimizes overall time spent traveling. See [64] for details.

But travel and arrival time are not the only criteria of interest when planning journeys in public transportation networks. Equally important are the number of transfers that have to be made on a specific journey and the price of that journey. Often, for each time-optimal journey there exists a journey with considerable less *transfers* that does not take much longer. Equally, slower journeys can be considerably less expensive than fast journeys. Because the exact trade-off between these three criteria is user-specific one cannot simply try to optimize a mixed goal function as a weighted sum of the criteria. Instead, a *Multi-Criteria Pareto-*(s, t, τ)-query computes all Pareto-optimal journeys starting from s to t departing after τ. In this context a journey is Pareto-optimal if it is not dominated by any other journey. Given two journeys p and q, we say that p *dominates* q if and only if there is at least one criterion for which p has a better value than q and there is no criterion for which p has a worse value than q. Sometimes it is sufficient to return the lexicographically first solution of that Pareto-set, see [147] for details.

Recent Algorithmic Results In the following, we review the current state of the art in the case of routing with multiple criteria in time-dependent transportation networks. In this problem the goal is to find the shortest path from a source location to a destination location, optimizing two or more criteria such as the total travel time, the travel cost or the number of transfers in case of railway networks. Moreover, the problem is time-dependent which means that the values of the criteria over every edge of the network is not constant but changes over time. In the following, we review a study of the problem of finding all Pareto-optimal solutions in a multi-criteria setting of the shortest path problem in time-dependent graphs presented in [71]. Then, we review two speed up techniques for time-dependent point-to-point shortest path problems with fully dynamic updates in a multi-criteria setting, namely SUBITO and k-flags, presented in [32]. We conclude this section by a review of a very recent study of a non-graph-based, very efficient approach to the multi-criteria query problem in transportation networks.

Multi-criteria Shortest Paths in Time-Dependent Train Networks The problem of finding all Pareto-optimal solutions in a multi-criteria setting of the shortest path problem in time-dependent graphs resulting from train networks has been studied in [71]. This problem has applications in timetable information systems for train schedules where there may be multiple criteria for optimization, such as travel time, number of transfers and others.

In this work, a timetable consists of a set T of trains, a set S of stations and a set E of elementary connections. An elementary connection is a connection between two adjacent train stations without intermediate stops. Such a connection is defined by the departure and arrival stations, the arrival and departure times at each station and several properties like the train class, number, and traffic days. Each train $tr \in T$ is a list of connections. A *train connection* is a sequence of elementary connections consistent with the sequence of arrival and departure stations.

In order to find Pareto-optimal paths with respect to given criteria and a given a query, an extension of Dijkstra's algorithm should be applied. Each node keeps multi-dimensional labels containing an entry for each criterion and a reference to its predecessor on the path. The node keeps a list of such labels that represent paths that are not dominated by any other label on the node. During an iteration of the algorithm, the lexicographically minimal label is extracted from the queue, and the node associated with it is processed. The process starts with scanning all outgoing edges and updating the labels on the neighboring nodes provided an update is feasible. All newly computed labels of the neighboring nodes are then compared to the list of labels of the initial node, and are inserted in the queue only if they are not dominated by any other label. On the other hand, labels on the initial node that are dominated by a newly computed label are discarded.

A good measure of performance of the proposed algorithm is the number of labels created during a search. A secondary measure is the number of labels that are inserted into the queue. In order to optimize the search several techniques are proposed which are discussed below.

A way to optimize the process is to compare a newly computed label with labels that we already have in order to eliminate the need of using them if they are already dominated. This technique depends on the order that the labels are processed. If a nearly non-dominated path is discovered before others, then the remaining dominated paths can be immediately discarded. In order to find the first results early on, a goal directed search is used. When inserting a node to the priority queue, a *potential* is added to its distance in order to give prioritize it over nodes with less potential. The potential used in this case is the lower bound of the travel time between the node and the target. The lower bound is initially computed for each node on an auxiliary static time-independent graph which contains the edges with the minimum constant travel times, without time-dependency. Then it is added to the distance of each node when inserted into the queue during the query phase. This way, some nearly non-dominated paths can be found earlier than before, and dominated paths can be discarded earlier resulting in less label computations and insertions into the queue.

Furthermore, the algorithm is modified to avoid the propagation of labels back to the node which they originated from, a phenomenon that is called *"hopping"*. In such cases the labels are immediately dominated. The search is modified to forbid "hopping".

Last but not least, since all connections from station nodes to route nodes have a cost of zero for all criteria, and because they are explored immediately after the extraction of the lexicographically minimum label from the queue, when processed they are also lexicographically smaller than any other label. Inserting them into the queue means extracting them again in the immediate next steps. Therefore, they are not inserted into the queue but are kept in a temporary list which is separately processed right before the next iteration of the algorithm.

Fully Dynamic Speed-Up Techniques for Multi-criteria Shortest Path Searches in Time-Dependent Networks Multiple speed-up techniques have been proposed for the routing problem in static networks, such as on a road map, or on a published schedule in public transport. However, this approach of routing is not realistic since unforeseeable events may change the underlying graph structure in unexpected ways (for example, a traffic jam on a road network). Moreover, such changes usually have a cascading effect, as in railway networks where an initial train delay may cause secondary delays on other trains that have to wait. In such cases, a fully dynamic approach is needed, which should be able to efficiently incorporate all changes occurring randomly on the network. In this section, we review the approach taken in [32] that proposes two new speed-up techniques for time-dependent point-to-point shortest path problems.

The main focus of [32] is routing on railway networks using timetable information. The proposed approach assumes that the travel time on each time-dependent arc can be lower-bounded by some positive value which is called the *minimum-theoretical travel time*. Above this lower bound, the travel time can be freely modified by dynamically changing delays. An unavailability of an arc is modeled by setting its travel time to $+\infty$.

A timetable $TT := (Z, S, C)$ consists of a tuple of sets where Z is the set of trains, S is the set of stations and C is a set of connections such that if $c \in C$ then $c = (z, s, s', t_d, t_a)$ where z is the train of the connection, s and s' the station endpoints of the connection and t_d, t_a the respective departure and arrival times. In addition, the *station graph* $G = (V, A)$ consists of the station nodes and the respective connections between them. Finally, the *station route graph* consists of ordered sequences of connections using the same train.

Nodes and arcs are augmented by arrival and departure events where $dep_u := (time, train, route)$ represents exactly one departure at a certain time using a certain train on a certain route and $arr_u := (time, train, route)$ is defined accordingly. Moreover, arcs are related to departure events dep_υ and arrival events arr_u to define the event that *one reaches vertex u at arr_u if he departs from υ with departure event dep_υ and traverse arc α on route r*. Staying at any vertex is limited to minimum and maximum staying times $minstay(arr_\upsilon, dep_\upsilon)$ and $maxstay(arr_\upsilon, dep_\upsilon)$, respectively. All the above define the event-dependent graph consisting of event-dependent vertices V_E and arcs A_E.

For railway networks the weight function $ontrip := (eatt, transfers)$ is used where $eatt$ is the earliest possible arrival time at a vertex or arc and $transfers$ is the number of transfers between trains. This weight function is applied both to arcs and vertices. Given a path P_E between two event-dependent vertices υ, u, the cost of P_E is defined as $ontrip(P_E) := \sum_{\upsilon_e \in V_E} ontrip(\upsilon_e) + \sum_{\alpha e \in A_E} ontrip(\alpha_e)$. Pareto-optimal paths can be computed by a Dijkstra-like labeling algorithm as described in Subsection 1.

On the above definitions of an event-dependent graph and considering a dynamic timetable two speed-up techniques, namely *SUBITO* and *k-Flags*, are proposed in [32]. The goal of *SUBITO* is to find all on-trip $s - t$ paths arriving at most max_{time} after the earliest possible arrival time. In the first step, the algorithm determines all minimum theoretical travel times from s to all vertices $\upsilon \in V$. In a next step, it determines the earliest arrival time at t with respect to the earliest start time from s. Then, an upper bound c_{max} is computed on the maximum deviation from the earliest arrival time accepted. Finally, a query is executed, running a labeling algorithm to compute Pareto optimal paths, pruning edges that lead to arrival times greater than the upper bound c_{max}.

The idea behind *k-Flags* is closely related to a classical speed-up technique named *Arc-flags*. A path $P_{u,\upsilon}$ is called c_i-optimal, $c_i \in \mathbb{N}_+ \cup \{0\}$, $c_1 < ... < c_k$ if its weight is at most c_i times the smallest minimal theoretical travel time between u and υ. At first, the graph is partitioned into *regions* V^j. Then, using flags on edges, only a subgraph $G^i_{V^j}$, $i \in \{1, ..., k\}$ of the original event-dependent graph G_r is used during each query. The subgraph $G^i_{V^j}$ is induced by all arcs (u, υ) for which there exists a c_i-optimal path between u and t. Therefore, $G^i_{V^j}$ contains only a small subset of the actual arcs. Hence, given a query and an upper bound on the travel time c_i, the algorithm only considers the subgraph $G^i_{V^j}$ containing all c_i-optimal paths, reducing considerably the search space of the query.

Algorithms for Itinerary Planning in Multi-modal Transportation Networks A recent work that investigates the journey planning problem (the authors call it itinerary planning) in a transportation network combined from several transport networks (bus, tram, ...) appeared in [187]. The itinerary planning problem in an urban public transport system constitutes a common routing and scheduling decision faced by travelers. A major decision that arises for the traveler relates to the selection of the itinerary that complies with his/her preferences and requirements. The objective of the study in [187] is to solve the above problem determining the itinerary that lexicographically optimizes a set of criteria (i.e, total travel time, number of transfers, and total walking and waiting time) while departing from the origin and arriving at the destination within specified time windows. Thus, the itinerary planning problem is expressed as a shortest path problem in a multi-modal time-schedule network with time windows and time-dependent travel times. In this work, the term multi-modal is used to refer to combined public transportation systems with fixed time schedules (e. g. metro, tram, bus, ...) unlike other work that use the term multi-modal to refer to a combined network containing several modes of transportation with and without timetables (e. g. combined taxi and train networks). The main result of [187] is a new formulation which results in an enhancement of the existing trip models. This is achieved by simultaneously taking into account the time-dependent travel times and the multiple traveling criteria. Moreover, it expands on a special case of the main problem where there is an additional constraint of visiting an intermediate stop (e.g., sightseeing) within a given time window.

Prior to defining the problem, we give a description of public transportation network considered in Zografos et al.'s work. The stops of the urban public transport network is the set of nodes (vertices) N of the multi-modal graph. Each node υ_i is assigned a departure time $\tau^s_{\upsilon_i}$ from the node for each service $s \in S$. The set of departure times from node u_i for each service s is demoted by $ST^s_{u_i}$. A sequence of nodes of a certain service s comprises a route R_s. The arcs of the graph is the set of arcs formed by the routes. The travel time on an arc depends on the departure time τ from the source node u_i and is denoted by $t^\tau_s(u_i, u_{i+1})$. The graph is then defined as $G(N, A \cup A')$ where A' are additional arcs representing foot paths between nodes.

An itinerary is a sequence of arcs with respect to departure times from each node and the service used to traverse them: $p^\tau(u_0, u_n) = \{[(u_0, u_1); \tau_{u_0}; s_0], [(u_1, u_2); \tau_{u_1}; s_1]; ...; [(u_{n-1}, u_n); \tau_{u_{n-1}}; s_{n-1}]\}$.

The *elementary* itinerary planning problem then tries to find the optimal itinerary according to given criteria for total travel time, number of transfers and total transfer time (walking and waiting time) with respect to the following constraints:

1. No departure from a node u_{i+1} can occur before the respective arrival to node u_{i+1} from the preceding node u_i.
2. The departure time from the origin node u_0 must occur within a given time window $[d^e_{u_0}, d^l_{u_0}]$.
3. The arrival time to the destination node u_n must occur within a given time window $[a^e_{u_n}, a^l_{u_n}]$.

One approach to dealing with the problem is to find all non-dominated itineraries but this is too complex and inefficient. Another approach is to arbitrarily rank the criteria and then select itineraries with smaller lexicographic order over others. The latter ensures that only the best single solution is presented to the user.

An interesting extension of the *elementary* itinerary planning problem is the incorporation of an additional mandatory visit to an intermediate node. This is formulated as additional constraints to the initial problem:

4. The arrival time to an intermediate node u_q must occur within a given time window $[a^e_{u_q}, a^l_{u_q}]$.
5. The mandatory visit at u_q must last at least t_{u_q}.
6. The departure time from an intermediate node u_q must occur within a given time window $[d^e_{u_q}, d^l_{u_q}]$.

The basic idea behind the proposed algorithm for the solution of the *elementary* itinerary planning problem is to move backwards in time, starting at the destination node u_n at arrival time $a^l_{u_n}$ and gradually trying to find lexicographically minimal itineraries from all other nodes (including the origin node u_0) to u_n for any possible start time $\tau \in [d^e_{u_0}, a^l_{u_n}]$. During an iteration of the algorithm where the lexicographically minimal itinerary from an intermediate node u_j to u_n has been already found and a new arc (u_i, u_j) is considered for inclusion, the lexicographically minimal itinerary from u_i can be determined by comparing:

1. the new itinerary including arc (u_i, u_j) joined with the lexicographically minimal itinerary from u_j to u_n.

2. the itinerary derived by adding one unit of waiting time at the existing lexicographically minimal itinerary from u_i to u_n.

3. any other itinerary that might be produced by joining a walking arc from u_i to a node u_q with an existing lexicographically minimal itinerary.

Having found a solution to the *elementary* itinerary planning problem, the solution can be easily extended to solve the itinerary planning problem in which there is a mandatory waiting time at a single intermediate node. The problem is solved by splitting it, for any intermediate node with a mandatory visit u_q, into two separate basic itinerary problems between (u_0, u_q) and (u_q, u_n). Of course, the solution should respect arrival and departure times at u_q.

Solving the multi-criteria time-dependent routing and scheduling problem in a multi-modal fixed schedule network A closely related to [187] but slightly different problem has been studied in [12]. The main difference of the problem studied in [12] w.r.t. that in [187] is that even though departing from the origin node and arriving to the destination node must occur inside specified time windows, arriving and departing from each intermediate node has to occur inside a certain, specified time window too. There is also a difference in the solution approach for this problem in comparison with [187]. In [187] the objective is to present a single lexicographically minimal itinerary between a pair of nodes. The objective in [12] is to determine the *entire* set of non-dominated solutions to the multi-criteria shortest path problem using intermediate stops in a combined time-dependent network of several public transportation services.

The multi-criteria itinerary planning problem involves the determination of the non-dominated itineraries from an origin node u_0 to a destination node u_{m+1} for all possible departure times that pass through the sequence of intermediate nodes $\{u_1, u_2, ..., u_m\}$. The departure time from u_0 is bounded by $[edt_{u_0}, ldt_{u_0}]$ where edt_{u_0} is the earliest departure time from u_0 and ldt_{u_0} is the latest departure time from u_0. Accordingly, the arrival time at the destination node u_{m+1} is bounded by the time window $[eat_{u_{m+1}}, lat_{u_{m+1}}]$. Furthermore the arrival at any intermediate node u_k is bounded by the time window $[eat_{u_k}, lat_{u_k}]$ while the duration of the visit in u_k is constant. Figure 8 shows the evolution of an itinerary in time.

A feasible itinerary consists of a sequence of elementary itineraries connecting (u_k, u_{k+1}) for $k = 0, ..., m$. Therefore, the problem can be decomposed into a sequence of elementary multi-criteria time-dependent shortest path problems with a time window. Thus, it is sufficient to solve $m+1$ elementary itinerary planning problems between the visiting nodes. The problem then reads as follows: "Given the non-dominated itineraries from node u_{k+1} to u_{m+1} for every possible departure time from node u_{k+1} determine the corresponding set of non-dominated itineraries from u_k to u_{m+1} through u_{k+1} for every possible departure time from u_k".

In order to solve the multi-criteria *elementary* itinerary planning problem a set of labels is defined for each node which keeps all non-dominated itineraries from each node to the destination node. Non-dominated itineraries can be systematically generated through the identification of other non-dominated itineraries which constitute them. At the initialization step, a vector of labels is constructed for the destination node i_n for every step in time between the arrival time window $[eat_{i_n}, latt_{i_n}]$ where eat_{i_n} is the earliest arrival time for node i_n and $latt_{i_n}$ is the respective latest arrival time for this node. A sequence of iterations is then performed, moving backwards in time, gradually discovering non-dominated itineraries from all nodes to the destination node. The outcome of each iteration is the determination of all non-dominated vectors of labels for each point in time. During an iteration, and considering an arc joining nodes i_r, i_{r+1} where all non-dominated vectors of labels have been discovered for node i_{r+1}, the new *candidate* itineraries emerge by joining the arc with the existing itineraries in i_{r+1}. The newly constructed itineraries are then

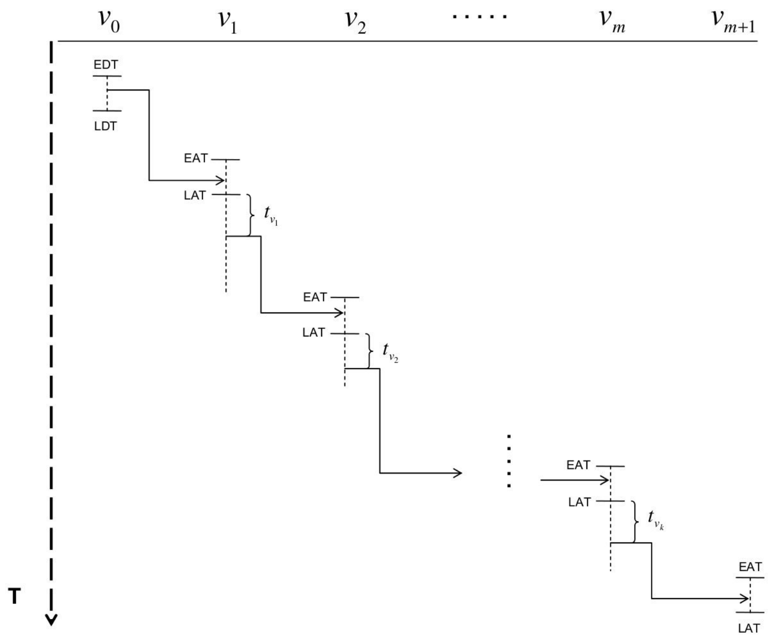


Figure 8: Graphical display of an itinerary evolving in time.

tested for domination and if they are found non-dominated, they are appended to the respective non-dominated labels of i_r.

Finally, using the above algorithm to solve the elementary itinerary planning problem, the non-dominated itineraries between each visited node can be computed. These solution can then be used to compute all non-dominated itineraries between the origin node and the destination node in the multi-criteria itinerary planning problem.

Round-Based Public Transportation Routing The common approach to timetable routing as discussed above is to model the timetable as a graph (time-expanded or time-dependent) in which a fastest journey can be found by means of a graph-based shortest path algorithm such as Dijkstra's Algorithm [70] and its variants.

A very recent publication challenges this approach [68]. The authors of that paper observe that operating on a graph hinders the exploitation of transportation network specific properties. Instead they give a dynamic program that solely operates on the basic elements of a transportation network: the stops, the routes and the trips that serve each single route (for the definition of routes see the above modeling section). The basic algorithm (RAPTOR) computes bi-criteria Pareto-optimal journeys minimizing arrival time and number of transfers. It only uses simple data structures and unlike Dijkstra's Algorithm it exhibits excellent data locality enabling very cache-efficient implementations. Also unlike Dijkstra's Algorithm it is easy to parallelize. Through several extensions of the basic algorithm the authors can solve multi-criteria queries with additional criteria such as fare zones (McRAPTOR) and time-range/profile queries (rRAPTOR) that give Pareto-optimal journeys for the whole day.

The basic variant of RAPTOR operates in rounds: before the first round the source stop p_s is marked with the earliest departure time τ as given in the input. During the first round all routes incident to the source stop are traversed. For each such route the earliest feasible trip of that route is determined and subsequent stations on that route are marked with the respective arrival time of the trip. After traversion of said routes foot-paths are expanded from each newly marked stop and stops reachable by foot are marked accordingly. The arrival at each stop p is saved in a bi-criteria multilabel $\tau(p)$. After the first round, all stations reachable by one trip are marked with their respective earliest arrival time. The following invariant holds: if after $k-1$ rounds, the first k entries in $\tau(p) = (\tau_0(p), \tau_1(p), \ldots, \tau_{k-1}(p), \ldots)$ are correct, i. e. entry $\tau_i(p)$ represent the earliest arrival time at stop p after at most i trips, then round k correctly computes $\tau_k(p)$ for all p. Note that the entries of the multilabel $\tau(p)$ are computed with decreasing earliest arrival times. Let $\tau^*(p)$ denote the earliest known arrival time. Each round operates on several stages (as described for the first round): for every stop marked in the last round the respective route is scanned: at the earliest marked stop the first reachable trip is determined (taking into account minimum transfer time). Using that trip the route is traversed and for each subsequent stop p_j served by the trip the earliest arrival time $\tau_k(p_j)$ is updated accordingly but only if it is better than $\tau^*(p)$ (*local pruning*). Also, if for some j, $\tau_k(p_j)$ is earlier than the arrival time of the trip, the current trip is to be updated accordingly. Stops whose arrival time is greater than the best known arrival time $\tau^*(p_t)$ at destination stop p_t are not marked either (*target pruning*). At the end of each round, local foot-paths are expanded as described above. Overall, in each round a route is scanned at most once. Also, unlike Dijkstra's Algorithm no priority queue is required. Two routes that have no stops in common can be scanned in parallel.

The more-criteria extension (McRAPTOR) is based on the same ideas as the graph-based multi-label label-correcting algorithms discussed before. Instead of multilabels each stop stores bags of multilabels representing the Pareto-optimal solutions found so far. For details, seed [68]. The time-range extension (rRAPTOR) is based on ideas from a graph-based technique published in [64]. The last trip at p_s within the time-range is relaxed first. Subsequent trips employ *self-pruning*: if a trip cannot improve on label $\tau_k(p)$, i. e. a later departure at p_s has arrived non-later at p, the current trip cannot be Pareto-optimal and can safely prune itself.

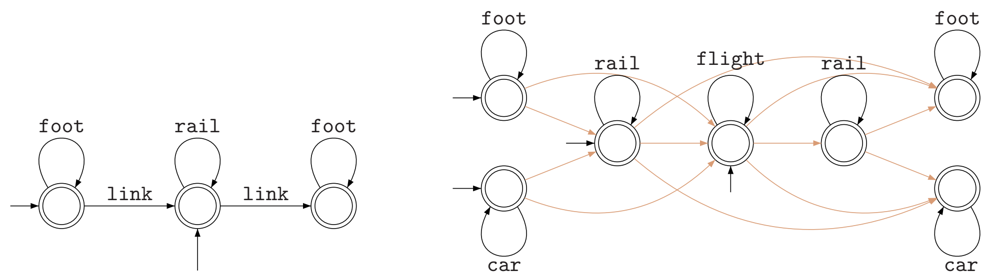
The authors evaluate the performance of their algorithms on several urban transportation networks, most notably on the London network of subways, buses, trams and Dockland Light Rail. They show that all variants of RAPTOR achieve unprecedented speed compared to other fully dynamic techniques. Unlike other fast techniques such as [29], RAPTOR needs no preprocessing and allows for easy handling of delays or route changes.

2.3.2 Route Planning for Multi-Modal Networks

In subsection 2.3.1, we reviewed current algorithmic approaches to journey planning in public transportation networks where the goal is to compute one or several journeys from source stop to destination stop. Sometimes these approaches are called multi-modal by their respective authors (cp. [187]) because the considered networks are a combination of several timetable layers such as metro, bus, tram, trolley, fast trains, long-distance trains, regional trains, etc., plus a limited number of foot paths between stops modeling short transfers.

In this subsection we review techniques that compute door-to-door journeys, that is, the considered transportation network is a combination of both public transportation (timetable) and road (no timetable) networks. This is of particular interest to eCOMPASS as such techniques are applicable to park-and-ride route planning scenarios as well as to a fully realistic assessment of transportation alternatives in an urban environment: is it better to take the car, to walk to the nearest station and then transit, to take the bike to a station further a way, to walk the whole way, to use a rental bike (e. g. Barclays Cycle Hire in London, Call a Bike in Germany) at the destination or in-between, or to leave the car at a park-and-ride station?

Besides computing journeys door-to-door, the algorithms discussed in this subsection (unlike those surveyed in subsection 2.3.1) actively handle the mode of transportation. This is so that constraints can be put on the modes of transportation allowed in a given context but also in order to extend known preprocessing techniques to the multi-modal setting (i. e. in order to apply preprocessing differently based on the transportation mode of a subnetwork).



(a) `foot-and-rail`. (b) `hierarchical`.

Figure 9: Two example automata. In the bottom figure, light edges are labeled as `link`.

Multi-Modal Route Planning: The Basic Approach All the reviewed papers have the following basic approach in common: First, the individual subnetworks are modeled as graphs as appropriate: For *road networks*, nodes model intersections and edges depict street segments. For *public transit networks*, one of the models as described in subsection 2.3.1 is chosen. Then, to obtain an integrated *multi-modal* network, the node and edge sets of each individual network are merged. To enable transfers between modes of transportation *link edges* (or *transfer edges*) between the subgraphs are added. Choosing these links is very critical in terms of the considered scenario, it directly reflects where e. g. a car can be parked in a park-and-ride scenario: only at certain stops? At every stop? At certain public parking lots? At every street? Also, choosing these links has direct influence on how hard the considered scenario is to preprocess (more densely interconnected networks are usually harder to solve fast).

Since the naïve approach of using Dijkstra's algorithm on the combined network does not incorporate modal constraints, the reviewed techniques consider the Label Constrained Shortest Path Problem (LCSPP) [27]: each edge in the combined graph has a label $\mathrm{lbl}(e)$ assigned to it. The goal is to compute a shortest s-t-path P where the word $w(P)$ formed by concatenating the edge labels along P is element of a language L, a query input.

Modeling a sequence of constraints is done by specifying L. Typically, *regular languages* of the following form suffice. The alphabet Σ consists of the available transport modes. In the corresponding NFA $\mathcal{A}_L$, states depict one or more transport modes. To model traveling within one transport mode, we require $(q, \sigma, q) \in \delta$ for those transport modes $\sigma \in \Sigma$ that q represents. Moreover, to allow transfers between different modes of transport, states $q, q' \in Q$, $q \neq q'$ are connected by `link` labels, i. e., $(q, \texttt{link}, q) \in \delta$. Finally, states are marked as initial/final if its modes of transport can be used at the beginning/end of the journey. Example automata are shown in Figure 9.

In general, LCSPP is solvable in polynomial time, if L is context-free. For the constraints used here, a generalization of Dijsktra's algorithm works [27].

Accelerating Multi-Modal Route Planning by Access-Nodes The first multi-modal speedup technique was published in [66]. It can answer multi-modal earliest arrival queries within milliseconds. The authors observe that the largest part of the combined multi-modal network is contrived of road nodes and edges. They argue that (in many applications) the road network should only be taken in the beginning and the end of a journey. In such scenarios, a multi-modal journey can be computed in two stages: First, the distance from the source/destination location to/from all relevant public transit stops is computed. Then, a multi-source-multi-target public transit routing is performed with these stops initialized in the priority queue accordingly. The authors call the public transit stops relevant for a road node its *Access-Nodes* (as they provide access to the public transportation network). They observe that a stop is relevant for a road node u if throughout the day this stop is part of at least one shortest journey originating at u. Having these access-nodes of every road node precomputed, the first stage of the query only involves a couple of table lookups, basically skipping the shortest-path computation within the road network. This enables very large speedups compared to a multi-modal generalization of Dijkstra's algorithm that has to look at a large part of the comparably large road network for an average query.

In order to compute the access-nodes of each node a one-to-all multi-modal profile search is needed, but this turns out to be too costly. Instead the authors propose an over-approximation of the set of access-nodes that involves backward searches from each stop. Furthermore, in order to save time on preprocessing the authors propose to first contract large parts of the road network (see Deliverable D2.1 on Contraction in road networks). Access-nodes are only computed for road nodes in the core (the uncontracted part). If a query source or destination lies in the component (the contracted part of the road graph), a local road shortest-path search must be performed until all paths are covered by core nodes. Then, access-nodes of these core nodes can be used to initialize the public transit search of stage 2.

Using these techniques, Access-Node Routing (ANR) can achieve random earliest arrival queries on an intercontinental combination of road and flight networks within milliseconds at the cost of preprocessing of several hours. It should be noted though that the choice of applicable label constraints is limited by the restriction that modes of transportation involving road (i. e. walking, biking, car driving) can only appear in the beginning and end of the journey. Also, only single-criteria earliest arrival queries were considered in this work. Furthermore, the road network is considered time-independent, i. e. no historical knowledge about rush hours is included. Unfortunately, delays on the public transportation subnetwork might change which transit stops appear as access-nodes for the road subnetwork, triggering an expensive rerun of the preprocessing step. Last but not least, the preprocessing is restricted to the chosen constraint automaton. This can be a real problem if the goal is to show several alternative journeys with respect of modes of transportation used.

UniALT for Regular Language Constraint Shortest Paths on a Multi-Modal Transportation Network It has been known since [144] that ALT [93] can easily be applied to multi-modal networks. ALT is a speedup technique that combines the A* heuristic [146] with distance bounds via preprocessed landmarks and the triangle inequality (hence the name). Since the fastest mode of transportation between two locations is always a feasible bound no matter what restrictions are applied by the constraint language, ALT can directly be used on a multi-modal network. Unfortunately, the speedup of ALT correlates with the tightness of bounds, rendering this approach mute. In [112] the authors propose an extension of ALT that they call state-dependent ALT (SDALT). For each state of the considered constraint automaton different landmark distances are calculated. This allows the query algorithm to use different bounds depending on the state the query is currently in, making the bounds tighter overall. For example, when the constraint automaton prohibits use of the flight network, landmark distance become longer, correctly reflecting the fact that journeys may take longer.

Using this approach, SDALT can achieve mild speedups of earliest arrival queries when compared to a multi-modal generalization of Dijkstra's algorithm. Yet, the preprocessing is much less intensive that in [66] and a wider class of constraint automata can be used (also with road use in the middle of a journey). As before, the preprocessing is restricted to the chosen constraint automaton, and the authors only consider single-criteria earliest arrival queries.

User-Constrained Multi-Modal Route Planning Both ANR and SDALT predetermine the label constraints during preprocessing. The goal of a very recent publication is to make the multi-modal preprocessing independent of the label constraint automaton used, allowing for sequence constraints on the modes of transportation to be chosen by the user [69]. To this end, the authors propose a variant of Contraction Hierarchies [88] (see Deliverable D2.1 for details) tailored to multi-modal networks: nodes incident to link edges might not be contracted. That way the preprocessing does not need to make any assumptions about the query-time constraint automaton other than that travel within a mode of transportation is unconstrained. After the preprocessing several components (contracted graphs) are connected via one single core (uncontracted road and timetable nodes). The query of UCCH works like this: if either source or target lie in the component a basic CH (bi-directional, "upward") query is applied. Once in the core, the query switches to a multi-modal generalization of Dijkstra's algorithm that observes label constraints. Since the core is much smaller than the original graph large speed-ups of up to four orders of magnitude can be achieved after careful optimization of data structures and layout. Yet, the achievable speedup deteriorates when tested instances become increasingly interconnected (i. e. transfers between modes of transportation are possible at a larger fraction of nodes). Still, UCCH can preprocess denser instances than ANR.

Both ANR and User-Constrained Contraction Hierarchies (UCCH) achieve very fast queries but UCCH needs less preprocessing effort. Both SDALT and UCCH can work on a wider class of label constraints but UCCH does not predetermine the used constraint during preprocessing. On the downside, UCCH regards the road network as time-independent, i. e. rush hour cannot be modeled, and the preprocessing is still too expensive to warrant integration of delay data from real-time road traffic reports. But unlike ANR, UCCH can easily integrate delay data for public transportation networks. All three techniques, ANR, SDALT, and UCCH only compute earliest arrival queries.

Object Modeling and Path Computation for Multimodal Travel Systems Path planning in an urban public transport network can be extended to support fast inter-city route calculation [34]. Data preprocessing is performed to reduce search space when querying from a location in e. g. city A to another in city B. The algorithm is based on viable path construction [132].

The viable path from an origin to a destination is the path that respects a set of constraints:

- Modal transfers from public transport to private vehicles are not allowed (i. e. use of private vehicles is only allowed at the beginning of a journey).
- Maximum number of modal transfers are defined as a user-chosen constraint.
- Paths consist of only one consecutive sequence of public transportation mode, or only one consecutive sequence of private vehicle mode of transportation with initial mode.

The execution of the algorithm determines the k-best shortest paths based on the number of modal transfers given by the user and respecting the scheduled departures associated with each transit station. The time complexity is $\mathcal{O}(|E|^2|V|)$, where $|E|$ denotes the number of arcs and $|V|$ the number of nodes.

Transfer Graph Approach for Multimodal Transport Problems Published in [98], this work concerns a route guidance method in a multimodal transportation network. The main idea is to divide the multimodal shortest path problem into several sub-problems, calculating sets of shortest paths for each sub-problem, in order to handle multimodal routing in a more clear and efficient manner. The publication is part of a work package of the Carlink (Wireless Traffic Service Platform for Linking Cars) project. The multimodal routes are computed in such a way that all existing unimodal transportation networks are handled separately. A shortest path algorithm is performed on each individual unimodal network. The final operation produces a graph that contains all computed networks with the connections between them. A shortest path algorithm is performed on the basis of this precomputation in order to calculate shortest paths on the multimodal network.

Design and Assessment of an Online Passenger Information System for Integrated Multi-modal Trip Planning An online passenger information system for delivering personalized multi-modal trip planning services was developed in [188], based on the algorithm in [187]. This system, called *ENOSIS* provides both urban and interurban multi-modal trip planning services and real-time travel information through a single point of access. These services are provided through many communication channels such as the Internet, mobile phones and information kiosks. ENOSIS has been developed, applied and evaluated for supporting travel decisions in Greece even though it also aims to be applied beyond the borders of the European Union. ENOSIS is focused on developing a service for planning multi-modal journeys by taking into consideration multiple criteria and incorporating complex scheduling constraints (for example time windows on departure and arrival) defined by the users. Moreover, ENOSIS is complemented by an online booking service, a journey alert system (reminders, information on the next transport service) and tourist information center.

The main focus of [188] is the assessment of ENOSIS' system performance and an evaluation of the service. Therefore, the assessment is considering the following categories:

- technical performance
- user acceptance
- socio-economic impacts

In order to make an assessment of the technical performance of ENOSIS, computational time thresholds were defined by users which were then collated with the actual average computational time. The actual average computational times never exceeded the user-defined threshold. Therefore, they were acceptable and provided satisfactory services to the users.

For assessing the user acceptance of the system, a survey was conducted and a questionnaire was provided to potential users to fill in. A general conclusion from the survey was that no major defect was identified with respect to the coverage of the qualitative operational requirements of the system.

Finally, for the financial assessment of the system, two categories of stakeholders were considered, the actual travelers and the travel information providers. The effectiveness of ENOSIS impacts on the end users is assessed through measuring their willingness to pay. This revealed that travelers are willing to pay up to 0.3 €per use. The cost effectiveness evaluation for travel information providers was based on the comparative assessment of the ENOSIS system with the existing system of the Port of Heraklion in Crete, Greece. The results implied that ENOSIS is more cost-effective as compared against the existing passenger information system.

2.3.3 Discussion – Weaknesses of Existing Approaches

Public transit routing has traditionally been much slower than route planning for road networks [28] but the gap has been closed to some extend lately [68]. It seems that fast routing on combined networks of public transportation and road should be within reach. Yet, from our review of state-of-the-art algorithms it can be seen that none of the current approaches deals with all scenarios and challenges relevant in the context of eCOMPASS. Each of the reviewed techniques falls short on at least one of the following issues:

- Pareto-optimal multi-criteria door-to-door queries. The reviewed techniques can either answer station-to-station multi-criteria queries or door-to-door earliest arrival queries.
- Time range (profile) door-to-door queries. The reviewed techniques can either answer station-to-station time range queries or door-to-door earliest arrival queries.
- Incorporation of historic knowledge about delays and/or live delay data of both transportation and road networks. The reviewed techniques either do not account for road travel (besides short precalculated foot-paths) or they cannot handle delays and traffic incidents on either road or transit—nor can they handle knowledge about rush hours on road networks (known as time-dependent road routing, see Deliverable D2.1).

- Close integration of several modes of transportation. Most of the reviewed works do not look at scenarios involving bikes (either private or via rental stations). None of the reviewed works look at scenarios requiring the integration of park-and-ride.
- Evaluation of and optimization for ecological friendliness. None of the techniques reviewed can account for this criterion. One of the commercial tools (youdrive.gr) tries to but the results are of questionable nature.
- Reasonably fast response time. While a single user might be willing to wait several seconds on a response, a route planning web service serving hundreds of users at once needs faster algorithms than that. Also, in order to solve multi-modal tourist trip design problems (see Section 4) a large amount of multi-modal shortest path distances needs to be computed also making fast multi-modal route planning algorithms a necessity.

3 Computing shortest paths under uncertainty

3.1 Motivation

It is a big challenge for urban areas to popularize public transportation over the use of private vehicles. Public transportation can offer significant improvements both for transportation efficiency and for the environmental footprint of modern metropolises. However, in order to convince people to take a train or a bus instead of their private cars, public transportation networks need to offer competitive advantages to the individual user. The main way how to appeal to the user lies in offering attractive connections that feature reasonable durations, prices and an acceptable level of discomfort. Most of all, connections need to be reliable, even if minor disturbances occur.

To highlight the importance of robustness, think of a commuter that decides to try a train connection instead of using her private car. If the connection is reliable, the additional comfort might be appealing to the user and convince her to stick with public transportation in the future. If, on the other hand, the commuter misses a connecting train even once and is delayed by more than an hour as a result, there will be a high chance that she abandons public transportation altogether. It is therefore imperative to provide reliable connections, that offer alternatives in case connections cannot be reached.

The example illustrates a major difficulty when dealing with uncertainty in multimodal networks. While small delays might not affect the travel time of an individual mode much, they can propagate and cause significant secondary delays down the road, e.g., when trains wait for each other. Multimodal networks also exhibit other unique features that require particular techniques, like non-additive cost functions of connections, caused by special tariff zones that govern prices.

In the following, we consider important problems in multimodal transportation networks. Our focus is on techniques that can deal with uncertainty of the input and deliver solutions that are reasonably good for every likely scenario. We conduct a literature review to expose the different approaches that have been taken in this direction.

We are particularly interested in the route planning problem in uncertain multimodal networks, where the goal is to compute a path that maximizes utility while being reliable. Our summary starts with research concerning this problem. We then move on to delay prediction, which is a useful tool for assessing the reliability of a route. Finally, we consider the question of how to design a robust network in the first place. The goal is to minimize the propagation of disruptions and to offer robust routes or good alternatives, while maximizing the expected utility for the user.

3.2 Multimodal Route Planning under Uncertainty

Route planning in multimodal networks mostly refers to the problem of finding an optimal connection in a public transportation network. From the user's perspective, there might be many different criteria that influence the decision on which route is best. A connection should foremost be quick, but the price of a ticket and the comfort of the ride are important factors as well. A common way to deal with this is to estimate the user's valuation of the different criteria by a utility function, and to optimize this utility using a classical shortest path algorithm. Another option is to compute all Pareto optima and let the user decide which route is most favorable.

The situation becomes much more involved once we introduce uncertainties. Most prominently, uncertainties arise in the form of unforeseeable delays that can arise in a multitude of ways. Also, weather conditions and fluctuations in the number of travelers can, apart from causing delays, influence the valuation of routes by the user. There is no general solution how to cope with such uncertainties in multimodal networks.

In the following, we summarize a variety of different models and approaches that aim at defining and delivering good, reliable routes in the presence of uncertainty.

Time-Dependent Uncertainty

The central feature of multimodal networks lies in the time dependency of the travel times along edges. Consider, for example, a bus connection between two stations. Assume that the bus leaves every 20 minutes and the trip takes 15 minutes. The length of the corresponding edge varies between 15 minutes and 35 minutes, depending on how long we have to wait for the bus.

In realistic environments there is a certain probability for delays. A straight-forward way of modeling this is to let the edge lengths to be governed by probability distributions. While in deterministic networks we are often interested in finding a shortest path, in stochastic networks it makes sense to consider paths of minimum *expected* length. If the distributions for the arrival times of the edges are independent, such a path can be found by Dijkstra's classical shortest path algorithm: By linearity of expectation, we can work with the expected lengths of the edges instead of their distributions.

If time-dependency is introduced into a stochastic network, arrival times of edges generally depend on each other. In this case, there is no single best path to reach a vertex anymore. Consider a situation where we need to get to the train station in order to take a long-distance train. There are two ways to reach the station: by taking the metro at 10:10 which arrives more or less reliably at 10:25, or by taking the taxi at 10:15 which arrives sometime between 10:20 and 10:40 due to changing traffic conditions. Should we prefer metro or taxi? The answer depends on our onward journey. If our train departs at 10:30, taking the metro will give us a high probability of catching the train, while taking the taxi has a significant probability of missing it. If our train departs at 10:24 on the other hand, taking the taxi might give us a better probability of catching the train. The distribution of our arrival time at the final destination clearly depends on how we plan to get to the train station.

Dijkstra's classical shortest path algorithm relies on the *optimality principle:* Combining *any* best path from A to B with *any* best path from B to C results in a best path from A to C via B. As demonstrated in the example above, this principle cannot be carried over to time-dependent stochastic networks.

Hall [99] was the first to propose an algorithm for finding a path of shortest expected length in a time-dependent stochastic network. Instead of maintaining the single best path to each node as in the classical setting, his algorithm maintains all paths leading to each node. He also argues that a better way of dealing with uncertainty is to devise an algorithm that is allowed to adapt online when delays are encountered. The paper proposes such a modification to the original algorithm, and investigates both solutions computationally, using a simple model for a scheduled bus service.

Kaufman and Smith [107] devise a consistency condition for deterministic time-dependent networks, under which Dijkstra's algorithm is guaranteed to work. We use $c_e(t)$ to denote the length of edge e at time t. The consistency condition requires that, for any $t_1 \leq t_2$, we have $t_1 + c_e(t_1) \leq t_2 + c_e(t_2)$. In other words, taking an edge earlier might take longer, but will not result in a later arrival. Kaufman and Smith also propose an improvement to Hall's algorithm that uses upper and lower bounds on the expected length of the best path. With these bounds, the improved algorithm can discard some paths early on and does not need to store all paths leading to each node.

Wellman et al. [181] adapt the consistency condition of Kaufman and Smith to the stochastic case. The straight-forward extension of the condition to expectations would be to require $\mathbb{E}[t_1 + c_e(t_1)] \leq \mathbb{E}[t_2 + c_e(t_2)]$ if $\mathbb{E}[t_1] \leq \mathbb{E}[t_2]$. This criterion does not make sense, as shown in our example above, assuming that the train leaves at 10:24: If we use the metro, our expected arrival time at the station is earlier than if we use the taxi; however, for the expected arrival at the final destination it is better to use the taxi. Instead, the authors propose the condition $\Pr(t_1 + c_e(t_1) \leq t) \geq \Pr(t_2 + c_e(t_2) \leq t)$ for all times t and $t_1 \leq t_2$. This condition allows for dominance pruning of paths that cannot possibly lead to the best solution. Wellman et al. use this concept to further improve the algorithm of Kaufman and Smith. They also provide an adaptive version of their algorithm and compare both experimentally to the earlier algorithms, using the bus model introduced by Hall.

The approaches we outlined so far concentrate on finding a path of shortest expected length. Miller et al. [137] suggest other optimization criteria instead. They propose an algorithm that maintains all Pareto optimal paths to each node, i.e., all paths that have an advantage of any kind over all others. In the end, one path can be chosen using various selection rules. Miller et al. consider selection rules based on minimum expected length, minimum variance, maximum probability to be shorter than a fixed threshold, maximum probability of being shortest, and minimum probability of being longest. In addition, they show how to determine the probability distribution of the length of the shortest path.

Wu et al. [182] study the applicability of the above approaches (among others) to scheduled transportation networks in the presence of delays. Their survey suggests either one of the algorithms in [137, 181], depending on the objective at hand.

Timetable Information with Delays

The timetable information problem is the problem of answering shortest path queries in realtime with respect to data from realistic public transportation networks. Usually, travel time is not the only criterion for selecting routes, and other criteria like the number of transfers or the cost of a connection have to be minimized at the same time. Uncertainty plays a crucial role in realistic approaches to public transportation and is usually present in the form of possible delays along edges or at vertices. There are various approaches in the literature on how to deal with delays. We give an overview over the most prominent ones.

Disser et al. [71] are the first to implement a system for optimizing multiple criteria in a realistic time-dependent train network. They use a simple approach to ensure robustness by introducing a measure of “transfer safety” as an additional criterion besides travel time and the number of transfers. In this approach, the safety of a transfer depends solely on the buffer time at a station, i.e., the extra time available to make up for possible delays. This approach delegates control over the robustness of a chosen solution: The user is presented with a selection of Pareto-optimal results and can apply her own valuation of robustness when deciding for one of them. Disser et al. evaluate their method experimentally on the train timetable of Germany.

Another approach for overcoming uncertainty in public transportation is to try to predict future delays given current delay data. MŎ3ccller-Hannemann and Schnee [140] develop a system that uses a *dependency graph* for capturing the correlation between primary and secondary delays. Using this graph, the system tries to predict future (secondary) delays that will be caused by current (primary) ones. Berger et al. [31] advance a step further and develop a stochastic model applied delay propagation. Their model includes waiting policies for when trains should wait for each other, driving time profiles along edges, and catch-up potential for winning back lost time along certain edges. Both papers provide an empirical analysis of performance and reliability using the train timetable of Germany.

Goerigk et al. [92] consider a scenario-based approach to robustness. They consider two definitions of robustness for a route. Strict robustness requires the route to be valid in all possible scenarios, i.e., it cannot rely on transfers that are not available in all scenarios. Light robustness bounds the overall length of the route to be not more than a factor longer than the optimum, but requires that the number of unsafe transfers is minimum. In order to find routes that are strictly robust, it is necessary to determine which transfers are always safe and which are not. Goerigk et al. show that this problem is already NP-hard, and propose to restrict the number of allowed transfers further. They provide an efficient algorithm for identifying this smaller set of transfers and use it as a basis for a routing algorithm. The paper also conducts an empirical evaluation of how much worse travel times become when strict/light robustness is required, using the high-speed-train timetable of Germany. Strict robustness is observed to be expensive, while light robustness seems to represent a better compromise.

Quite a different approach to robustness is taken by Bósing [41]. Her approach is targeted at mobile users and tries to exploit the computational power of mobile devices to reduce the load on centralized systems. In this scenario, the idea is, instead of requiring the central server to compute a new shortest route whenever new delay data is available, to ask the server to provide a smallest possible subnetwork that contains all likely shortest paths. When the shortest path needs to be updated due to delays, the mobile device can simply incorporate the new delay data into the subnetwork and perform the shortest path computation itself. Bósing considers three variants of the problem of finding the smallest relevant subnetwork. If a discrete set of scenarios is given, the problem is shown to be NP-hard. If scenarios are described by giving intervals for the lengths of the edges, the problem is shown to be in coNP and inapproximable to a factor better than $m^{1-\varepsilon}$, where m is the number of edges of the network and $\varepsilon > 0$. If there is a bound on the number of edges with actual length longer than the lowest possible length, the problem is NP-hard and inapproximable to a factor better than m. Together with these results Bósing provides a best-possible approximation algorithm that achieves a factor of $m/(l+1)$ for any $l \in \mathbb{N}$.

Robust Routes using Park and Ride

The eCOMPASS project has a special interest in approaches for incorporating park and ride options into route computation. Park and ride allows users to use both their private vehicles as well as public transportation. Designated park and ride locations are used for transfers between the corresponding networks. Finding attractive routes that make use of park and ride can help to reduce congestion and thus air pollution in city centers.

Li et al. [126] incorporate park and ride into their trip planning system. They use real-time delay data to predict arrival times using a model designed from historical delay data. The authors conduct an empirical study with data from the South Bay area of San Francisco.

Ambrosino and Sciomachen [11] describe a system specifically tailored to users that plan on utilizing park and ride for their trip. The computed routes always make use of park and ride and of at most one other transfer within the public transportation network. The system compares different possible park and ride locations using criteria like “connectivity”, “accessibility” and “expected delay”. Ambrosino and Sciomachen use time-dependent data and optimize a single criterion which is composed of time, cost, and a measure of discomfort for the user. They apply their system to real data from the region around Genova and evaluate the robustness of the system using stochastic noise.

Route Selection with Machine Learning

Realistic datasets for route planning can be massive. Many practical systems use complex heuristic speed-up techniques to quickly filter out relevant parts of the data. Another approach is to use techniques from machine learning to automatically find important structure in the data. An advantage of these techniques is that they are inherently robust to noise and even often incorporate noise as an integral part of the method. The following are examples for machine learning techniques applied to route planning in multimodal transportation networks.

Qu and Chen [148] view route planning in the multimodal setting as a multi-criteria decision problem. They train a neural network with feed-forward learning to solve this decision problem, assuming that the utility valuation is available as a single function. The authors conduct an empirical analysis based on a multimodal transportation problem for goods. They consider 6 criteria and 15 sub-criteria that contribute to the utility function.

Abbaspour and Samadzadegan [4] use an evolutionary algorithm for route planning with minimizing travel time as the only objective. They consider a time-dependent multimodal network with buses, subways and the possibility to walk. Routes are encoded as chromosomes and suitable mutations and crossovers are defined. The authors apply the evolutionary algorithm using geodata and timetable information from Tehran. The approach yields accurate results in most cases.

Information Routing

A very different routing problem in public transportation networks arises if we see vehicles not as a means of transportation, but as a part of a *delay tolerant network* for communication. Research in this area has focused on bus networks, assuming that buses can communicate with each other at close range. The usual question in this context is how to route information between nodes using buses as relays. The goal is to minimize latency while keeping the required memory of each bus as low as possible. An additional difficulty comes from the fact that due to delays it is unclear if and where pairs of buses meet. The problem has been studied in general dynamic graphs, but we focus on approaches that exploit the structure of the underlying public transportation network.

Doehring et al. [72] estimate the contact probability of two buses at intersections and along edges of the network. They propose an algorithm that sends data along one or more paths with the lowest expected latency. The authors use real timetable data and delay information for an empirical study involving disturbances and complete breakdowns of buses. Their results suggest that the algorithm outperforms its counterparts for general graphs, even when 20% of the buses are behind schedule.

Sede et al. [157] consider a setting with many potential delays. They propose to consider contact probabilities between bus lines instead of individual buses. Their algorithm computes a graph that encodes contact probabilities for bus lines and uses this graph for the routing of information. An empirical analysis for the city of Shanghai suggests that dealing with bus lines as a whole copes better with a network that features a lot of delays.

Ahmed and Kanhere [8] use historical data to determine which buses meet regularly. They use this information to compute clusters of buses that form reliable subnetworks. An algorithm is proposed for routing information based on these clusters. The method is evaluated experimentally using timetable data from Seattle.

3.3 Delay Prediction

Primary delays in a multimodal network are delays caused by external disruptions of the timetable, e.g., due to weather conditions or technical problems. Since the different elements of a public transportation network (trains, buses, airplanes, etc.) are interconnected through sharing edges and nodes, primary delays can propagate and cause *secondary delays* in other parts of the network. Delay prediction refers to the problem of estimating these secondary delays, given a set of primary delays. Solving this problem accurately requires detailed knowledge of the network architecture and precise data regarding primary delays.

Good predictions of future delays can help significantly in planning routes that are less likely to be affected by delays. Also, delay prediction can be used as a tool to assess the robustness of a network/timetable and can help to minimize delay propagation in the design process. Finally, delay prediction can be used when solving the delay management problem, where we look for policies that govern under what circumstances vehicles should wait for each other.

Most work on delay prediction has been done for long-distance railway networks, and a few other studies consider airplanes. We will first summarize work done for airplanes and then move on to delay prediction for railways.

Delay Prediction for Airplanes

Traditionally, airlines have counted on airplanes being always on time. With increasing demand and the tendency to try to use airplanes to their full capacities, schedules have become tighter and late arrivals are no exception anymore. Delay prediction is an important tool to deal with these delays both while planning and when they actually occur.

Wang et al. [178] develop a model for the repeated propagation of delays of aircraft. They make use of historical data to extract distributions of throughput and arrival delays at each airport. The authors distinguish *fixed* times that can hardly be influenced (e.g., turnaround times for airplanes at the airport) from *variable* times (e.g., flight durations, waiting times, etc.) that can be controlled and influenced by external disturbances. They apply the model to real national-airport-system data.

Xu et al. [183] use Bayesian networks to estimate the delay propagation between airports. The focus is on capturing how delays at one airport influence delays at others. The authors apply their model to a selection of important American airports. Their results provide better predictions than using a naive approach based on linear regression.

Delay Prediction for Railways

In urban transportation networks delay essentially does not propagate, since vehicles usually do not wait for each other. The situation is much different for inter-city railway networks. A lot of research has been done in this area, in particular for dense railway networks in Europe.

Approaches for delay prediction in railways typically employ a model consisting of arrival and departure events and of "processes" that connect the events. Processes are influenced by running times, transfer times, precedence relations, etc. The corresponding graph is often referred to as *stochastic event graph*. We herein provide a summary over various works on delay prediction.

Carey and Kwieciński [45] investigate the interdependence of trains along a single track, supposing that trains cannot overtake one another. They try to approximate the impact of a delayed train on the following trains by using a stochastic approach. A simple formula is derived that can be applied to consecutive trains repeatedly, providing an estimate of delay propagation. The authors compare tuning the parameters of the model using regression techniques and using other heuristics. They assess their results using a stochastic simulation of the interaction of trains on a single track.

In a later paper, Carey and Kwieciński [46] extend their previous model to encompass a network instead of a single track. The condition of trains not overtaking each other is carried over by fixing the order in which trains arrive at each station. The model is assessed using a simulation with stochastic delays that keep the order of arrivals intact.

Meester and Muns [136] consider the model of Carey and Kwieciński and present an improved approximation of delay propagation. The improvement is due to more detailed considerations relying on the assumption that the order of arrivals at each station is fixed. The authors model deviations from scheduled times via random variables and conduct a stochastic analysis.

Higgins and Kozan [101] develop an analytical model for delay prediction. Their model takes into account primary delays, as well as secondary delays caused by trains waiting for each other and by trains delaying other trains on the same track. They derive a system of delay equations that can be solved by an iterative refinement algorithm. They apply the model to the suburban rail network of Queensland.

Huisman and Boucherie [102] study the prediction of delays caused by trains of different speeds on a common track. They develop a stochastic model for the inter-dependencies of running times and delays. The approach is applied to sections of the Dutch railway network.

Yuan [185] discusses various types of dependencies between the delays of trains within a single station. The paper develops a stochastic model for delay propagation by dealing with trains of different tracks separately. This model is applied to a Dutch railway station.

Van der Meer et al. [170] use statistical analysis of historical data to predict delays in the near future. Their model accounts for conflicts along tracks and waiting policies of individual trains. The goal is to design a system that can give a reasonably accurate prediction quickly for realistic networks. The authors use data mining to extract parameters for dependencies of running times and delays. Their system is applied to an important corridor of the Dutch railway network.

Goverde [96] uses an algebraic model for the propagation of delays in periodic timetables. He provides an efficient algorithm to compute the delays according to this model. The algorithm groups arrival and departure events together into buckets of similar times. The buckets are then dealt with in order of increasing times, propagating delays where appropriate.

Móller-Hannemann et al. [31, 140] devise an efficient way to quickly propagate a large set of primary delays. Their goal is to provide robust solutions for real-time timetable information queries (for more details cf. subsection 3.2).

3.4 Timetable Optimization

In the two previous sections we focused on approaches for planning routes and predicting delay given a stochastic time-dependent multimodal network. We now go one step back and consider the question of how to design the network in the first place. The goal primarily is to come up with a timetable for multimodal transportation on a given network that will lead to quick and robust routes for the users. On the one hand, we would like the timetable to allow for quick connections if there are no delays. On the other hand, we also try to provide good alternative routes if transfers become infeasible and to reduce delay propagation as much as possible. While other criteria can play a role, like the cost for providing sufficiently many vehicles to implement the schedule, optimizing travel time and robustness is crucial in search of a timetable that satisfies users.

In the literature, two main variants of the problem are studied: the design of *periodic* (*cyclic*) and of *aperiodic* timetables. In the first variant we aim for a timetable that is repetitive and easy to remember (e.g., trains that leave each hour, buses that leave every 10 minutes, etc.). In the aperiodic setting, we do not impose such a requirement.

A practical timetable needs to account for possible disturbances that may occur due to accidents, weather conditions, etc. Disturbances usually translate into delays and the timetable should be ready to cope with a certain amount of such delays without losing its overall structure. For users to keep trusting the timetable (and keep on using it), they should be able to reach their destination in a time that does not differ too much from the expected travel time, even if a reasonable number of disturbances occur. Generally, a certain level of robustness can be achieved by overestimating travel times along edges a bit or by using longer transfer times. Of course, these techniques have to be applied carefully, since they prolong travel times even in the absense of disturbances.

We give an overview over different methods that have been used to generate timetables with a certain level of robustness. We start by presenting various approaches of different natures that were applied to the problem. We then go on to more systematic ways of formalizing a specific kind of robustness and optimizing timetables with respect to it.

Various Methods for Timetabling

Lan et al. [120] look for a robust flight schedule where delay propagation and the number of missed connections are minimized. They reformulate the problem as a mixed integer linear program with stochastically generated inputs. A solution to this program is supposed to reduce delay propagation. They also reschedule departure times within a short time window in order to reduce the chances of passengers missing their connecting flights.

Kroon et al. [115] deal with the problem of cyclic railway timetabling, minimizing the average delay of trains. Their approach takes external stochastic disturbances into account and tries to reallocate time supplements (difference between planned running time along an edge and shortest possible running time) and buffer times at stations. Kroon et al. first study how to generate a timetable for a single train under external stochastic disturbances, assigning time supplements in order to minimize the average delay at each station. The model is successively generalized first to multiple trains meeting in one station and then to general railway infrastructures. The authors consider a given timetable and try to modify it with the goal of achieving a fair compromise between reliability and efficiency (travel time, cost, resource occupancy, etc.). In practice, time supplements are usually chosen to be proportional to the travel time along an edge. The authors start by generating such a timetable and then perform several modifications, comparing alternatives by simulating stochastic disturbances. An experimental study suggests that larger time supplements should generally be assigned to the beginning of a line.

Schöbel and Kratz [156] consider timetabling as a bicriteria optimization problem, adding robustness as the second criterion besides average travel time. They characterize robustness by the maximum delay for which all connections are maintained, i.e., for which no transfer is missed. The proposed approach is to generate Pareto-optimal timetables that allow to select a compromise according to the expected amount of delays. The authors show that the bicriteria optimization problem has the same time complexity as the classical case with a single optimization criterion.

Marin et al. [134] combine timetable optimization with network design. Instead of assuming the underlying network to be fixed, the authors try to design a network together with a periodic timetable. Their goal is to achieve a certain level of robustness without degrading connection times in case no delays occur. Marin et al. consider a solution to be robust towards delays if passengers can choose alternative routes that are not much longer than their ideal connection. Solving the combined problem is difficult, and the authors suggest to solve the network design and the timetable optimization parts separately, combining the results afterwards. For both problems an integer linear problem is formulated. The program for designing the network does not at first take robustness into account. Instead, the initial network design is made more robust iteratively by adding more and more constraints a new network is then designed (ensuring that the new constraints are fulfilled) subsequently optimizing the timetable. This process is repeated until a limit for the available budget is reached.

Defining Robustness

We start this subsection by summarizing two classical approaches for dealing with uncertainty in optimization problems. These approaches form the basis for the methods described later on.

Stochastic programming assumes knowledge of the probability distributions that govern the uncertainty in the input. One variant of stochastic programming is *chance constrained programming*, where the goal is to generate a solution that satisfies all constraints with high probability, i.e., that is feasible in most scenarios. In *two-stage stochastic programming* the set of variables is divided into variables that need to remain fixed for every scenario and variables that can be adjusted once the actual data is revealed. The goal of the first stage is to assign the fixed variables such that the expected outcome of the second stage is the best possible. There are ways to convert the two-stage problem into a (large) one-stage problem. The cost of the overall solution is composed of the cost for the solution in the first stage, plus the expected cost in the second stage.

Robust optimization uses a deterministic objective function (in contrast to stochastic programming). The most common notion of robustness is *strict robustness*, which requires the solution to be feasible in all possible scenarios. While any restriction of the scenarios is desirable, strict robustness does not rely on previous knowledge about the distribution of the data.

Both stochastic programming and robust optimization have their advantages and shortcomings. Stochastic programming requires more knowledge about the data, but is more flexible regarding the optimization criteria. Robust optimization is quite restrictive, but often much easier to achieve computationally. Recent approaches have tried to combine the advantages of both methods. This lead to the concepts of *light robustness* and *recoverable robustness*. We summarize efforts that have been undertaken to apply these concepts in timetable optimization.

Light Robustness. In practice, strict robustness is an overly conservative requirement that leads to a deterioration of the quality of the solution to the optimization problem. Fischetti and Monaci [80] introduce light robustness as a relaxation of strict robustness that allows some slack in robustness while demanding a certain quality of the solution. The idea is to start with a formulation similar to the classical one for strict robustness and to introduce an additional constraint that requires the solution to have some minimum quality. This generally leads to an infeasible program. The solution proposed by Fischetti and Monaci is to use a two-stage approach as in stochastic programming by introducing slack variables that relax the requirement on robustness. The objective of the program is to find a feasible solution that minimizes this slack. The idea is that the initially strictly robust solution does not lose too much of its robustness in the process. The authors provide an empirical analysis for timetable optimization.

Fischetti et al. [81] propose different methods of designing a robust timetable and evaluate their results on real-world data. They consider two-stage optimization techniques to compute a robust aperiodic timetable. The first stage consists in finding a solution without taking robustness into account at all. Fischetti et al. use four different ways to make this initial timetable robust. Their first method is based on stochastic programming and is expensive to compute, but yields good results. Two other methods achieve relatively good results by using a lighter form of the stochastic programming approach. The authors also consider light robustness as explained before.

Liebchen et al. [130] aim at finding delay resistant periodic timetables. Their approach uses an extension of light robustness to compute an initial timetable. The final solution is obtained by tuning the tradeoff between buffering for possible delays and decreasing travel time. To compare the quality of different solutions, the authors solve the *delay management* problem for different delay scenarios, i.e., they determine which trains have to wait for feeder trains and which do not. Once the solution to the delay management problem is known, it can be used to measure the impact that the delays have on the travel times.

Recoverable Robustness. Recoverable robustness allows routes to become infeasible for some scenarios, as long as there are good alternatives to recover, i.e., there have to be other routes that remain feasible and are not much worse. Liebchen et al. [129] introduce the notion of recoverable robustness. They require the set of scenarios against which the solution should be robust to be provided as input and find a solution that can be adapted to each scenario with limited effort. The authors use a method similar to two-stage stochastic optimization. Their algorithm has to come up with a solution that is *always* recoverable and it therefore does not need knowledge of how the scenarios are distributed exactly. The authors apply their method to timetabling and delay management and assess the price of robustness, i.e., the deterioration of the travel time that results from requiring recoverable robustness. Liebchen et al. [128] discuss recoverable robustness more generally. They show that recoverable robustness admits some desirable properties, combining the flexibility of stochastic programming with the tractability of robust optimization. The authors apply recoverable robustness to linear programs and provide an efficient algorithm tailored for a certain kind of disturbances. They apply this technique to periodic and aperiodic timetabling, optimizing the timetable and the recovery strategy at the same time.

Cicerone et al. [59] extend the concept of recoverable robustness to *dynamic recoverable robustness,* where disruptions appear one after the other, each requiring an immediate recovery. They apply the concept to timetabling (as well as delay management) and develop different dynamic algorithms. The algorithms are analyzed with respect to how much worse the objective function value can become compared to the deterministic problem.

3.5 Discussion - Weaknesses of existing approaches

There are various uncertainties that passengers have to face when planning a route in a public transportation network, most being in the form of possible delays that may interrupt a route unexpectedly. On the other hand, passengers are not limited to a specific means of transport, and may respond to interruptions by switching modes if this appears as a better alternative. For example, if a bus is stuck in a traffic jam, passengers can choose to get off and walk to the nearest train/tram station instead.

Various disturbances of the network may affect modes differently and cause delays for some of them only: metros are unaffected by weather and traffic jams, buses are unaffected by power outages, etc. Even if a vehicle is not directly affected by a disturbance, it might still have to wait for another delayed one, leading to delay propagation. Missing a connection can be a serious problem, especially for long-distance connections that run with low frequency. Waiting too often, on the other hand, can lead to excessive delay propagation and harm the integrity of the entire timetable. A robust route planning system should take delay propagation into account as well as the possibility of missing a scheduled connection. The goal should be to propose routes which are either sufficiently stable towards delays, or offer alternative connections that the users can switch to if necessary.i

Existing approaches to robust route planning have several important shortcomings. Each of the approaches we summarized before (see Section 3) is limited in at least one of the following ways.

- Many techniques require knowledge of the exact distributions governing possible disturbances in the input data. In a realistic setting it is hardly possible to obtain such information. The usual workaround is to create a simplified models to capture reality. This needs expert knowledge of the network and carries the risk of introducing artificial features.

- Methods that do not assume knowledge of the input distribution generally restrict themselves to a set of representative scenarios. To generate such a scenario set, again, expert knowledge about the setting is necessary. Since the distribution of the scenarios is not known, the usual notion of a robust solution requires the solution to be reasonably good in all scenarios. This tends to generate solutions that are overly conservative and impractical.

- There were some attempts to combine stochastic approaches with scenariobased optimization in an effort to overcome the disadvantages of both. These methods usually make use of heuristics to relax the robustness requirement on the solution. The parameters of the heuristics need to be tweaked manually, which, once again, requires expert knowledge and makes it impossible to offer formal guarantees about the level of robustness of solutions.

- Virtually all approaches produce results that are based on historical data and are relatively good with respect to this data. However, there is no way to provide feedback concerning the expected quality of the solution for a unknown future instance.

4 Multimodal tourist trip design problems

4.1 Motivation

Tourists that visit a destination for one or multiple days are unlikely to visit every tourist site; rather, tourists are dealt with the dilemma of which points of interest (POIs) would be more interesting to visit. These choices are normally based on information gathered by tourists via the Internet, magazines, printed tourist guides, etc. After deciding of which sights to visit, tourists have to decide on which route to take, i.e. 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 in daily basis.

Tourists encounter many problems following this pathway. The information contained in printed guide books is often outdated (e.g. the opening times of some museums might have changed, some sites might be inaccessible due to maintenance works, etc), the weather conditions might be prohibitive on certain dates to visit an important POI, etc [74]. 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 potentially more interesting or appropriate to follow. Some tourist guides do acknowledge such problems and try to propose more generalized 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 specialized interests, needs or constraints [55].

Mobile tourist guides may be used as tools to offer solution to such types of problems [56], [111], [110], [133]. 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 [176]. Generalized tourist routes do not take into consideration the context of the user e.g. 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 context and location information represents a major challenge for the design of appropriate tourist routes [116]. Kramer et al. [114] analyzed the interests in the profiles of each tourist and concluded that they particularly varied from each other. This conclusion supports the argumentation for deriving personalized instead of generalized 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 length or duration of stay at that destination. This problem is termed as the 'tourist trip design problem' (TTDP) [176]. Interestingly, the TTDP presents similarities to problems which have arisen in the past in the field of operational research and comprise variations of the travelling salesman problem (TSP).

The design of daily tourist itineraries (typically starting and ending at the tourist's accommodation location) which take into account the tourists' personal profile and preferences represents a significant challenge that will be addressed in the context of eCOMPASS. Hence, our objective will be to recommend public transportation routes (a separate route for each day of stay) along various sights aiming at maximizing the overall 'satisfaction' of the tourists (i.e. include those sights they would potentially be most interested in visiting). This service is expected to assist the care-free movement of visitors and tourists and also promote the cultural 'reserve' of any urban destination, thereby attracting more tourists.

In the context of eCOMPASS, the modeling of TTDP is approached considering (see Figure 10):

- a set of candidate POIs, each associated with a number of attributes (e.g. type, location, popularity, opening days/hours, etc)
- a number of user-defined preferences and restrictions (e.g. tourist interests, period of visit, accommodation location, daily time allowance for sightseeing, etc)

- multi-modal routing among POIs, i.e. tourists are assumed to use all modes of transport available at the tourist destination, including public transportation, walking and bicycle.
- end users are assumed to be either web or mobile users; in addition to location, several contextual parameters may be taken into account in recommending sub-optimal itineraries to mobile users (e.g. day/time, weather conditions, traffic conditions, etc).

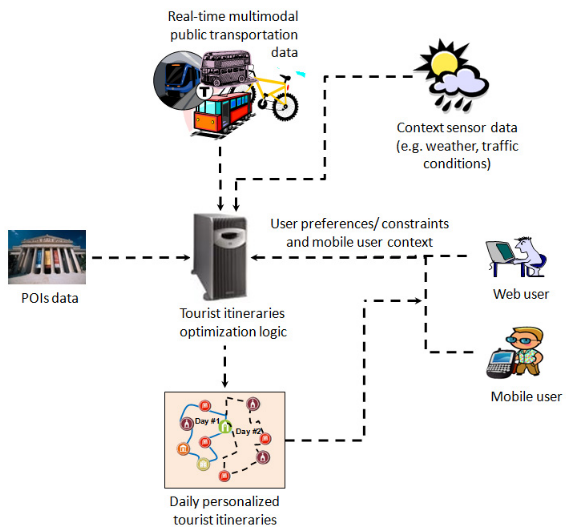


Figure 10: User groups, input data and recommended itineraries in TTDP.

The remainder of this section is structured as follows: Subsection 4.2 presents the detailed modeling of the TTDP, while Subsection 4.3 reviews the state-of-the art with respect to commercial tools relevant to TTDP. Subsection 4.4 introduces optimization problems relevant to the TTDP. Exact, approximation and heuristic algorithmic approaches proposed with regards to those optimization problems along with their limitations are discussed in Subsection 4.5.

4.2 TTDP Modeling

TTDP modeling involves the definition and the description of the user model, visit model and the sight (POI) model [87] (see Figure 11 taking into consideration parameters/constraints like those listed below:

- User Model:
  - device (e.g. screen resolution, available storage space, processing power, etc);
  - language of content, localization;
  - personal 'demographic' data (e.g. age, educational level);
  - interests (explicit declaration or implicitly deduced);
  - disabilities (e.g. blind, deaf, kinetic disability);
  - overall budget that may be spend on sightseeing.
- 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).
- start/end point of each tour (e.g. hotel location).

- Site (POI) Model:
  - category (e.g. museum, archaeological site, monument, etc);
  - available multimedia resources (collection of texts, video, audio, etc, localized 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 the city's Coin Museum due to 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 an indoor/outdoor site;
  - whether it is accessible from people with disabilities;
  - admission price (ticket prices).

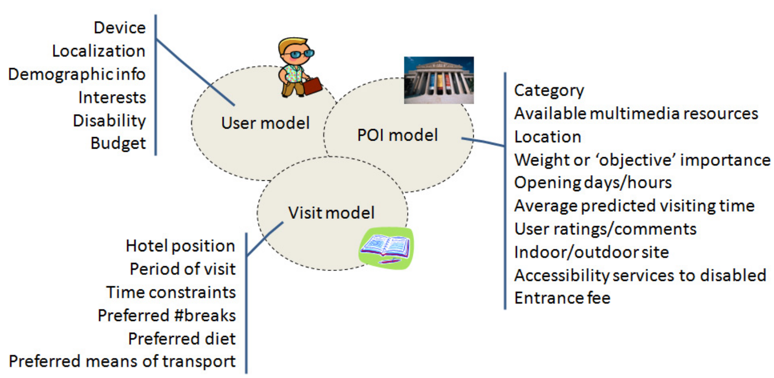


Figure 11: Description of user, visit and sight models in TTDP.

Notably, the above stated parameters/constraints are not exhaustive. From those parameters the below listed elements 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 same holds for calculating the POIs' profits: the profit derives by adjusting the POI's weight based on user-defined interests on specific POI categories.

- The ability to visit open air sites in a particular day during the user's visit, e.g. outdoor sites are not recommended to visit during a rainy day (meteorological forecasts can be retrieved from a 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 maximizes the overall profit, i.e. it enables the construction of personalized 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.

4.3 Online and commercial tools relevant to TTDP

A number of web and mobile applications have recently incorporated tourist route recommendations within their core functionality. Among them, we herein focus on those found most relevant to eCOMPASS context.

Google city tours has been the first web tool known to derive multiple daily tourist tours via city attractions, visualized through the familiar Google maps interface. However, the project has been discontinued by Google [166].

The City Trip Planner [1], [175] is a publicly available web tool, which offered more advanced route planning functionality incorporating several user-defined parameters within its recommendation logic (days of visit, daily time budget, preferences upon POI categories, start/end point), while also allowing the user to manually edit the derived routes, e.g. adding/removing POIs (see Figure 12). Yet, the tool does not take into account POIs opening/closing hours. Furthermore, City Trip Planner is provided through a web interface and has not yet been ported to mobiles; hence it lacks location and context-aware features.

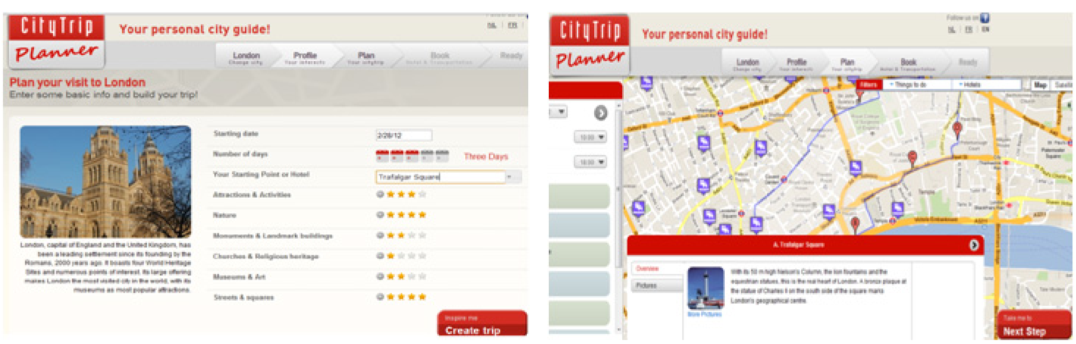


Figure 12: Screenshots of the City Trip Planner web application.

mtrip [2] (see Figure 13) is a commercial mobile application that represents a more recent development, known to work on Android, iPhone and iPad devices. mtrip generates personalized itineraries for selected travel destinations (based on user preferences upon POI categories, popularity and days/hours of operation of POIs, chosen visiting pace/intensity), navigation and use of augmented reality to offer enhanced views of physical spots. The routes are computed offline, hence no context-aware features are offered.

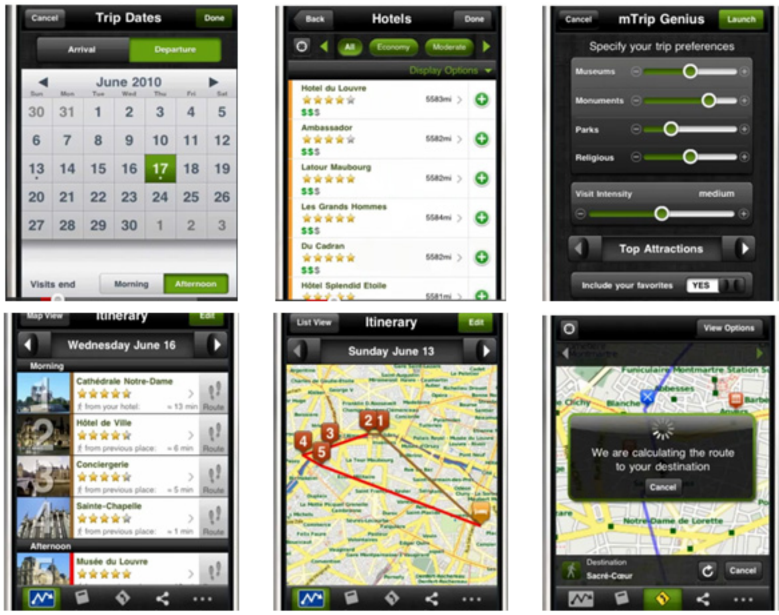


Figure 13: Screenshots taken from the mtrip iPhone application.

Neither City Trip Planner nor mtrip take into account public transportation transfers (i.e. they both exclusively consider walking trails). Even more so, features like consideration of breaks (either for food, coffee or rest), budget allowance for POIs entrance fees or recommendations of walks along architecturally or historically-valued trails are missing.

4.4 Optimization problems relevant to TTDP

The TTDP describes requirements that should be satisfied by applications providing tourist route planning functionality; those applications are expected to derive daily, ordered visits to POIs, while respecting user constraints and POIs attributes. High quality TTDP solutions should feature POI recommendations that match user preferences (thereby maximizing user satisfaction) and near-optimal feasible route scheduling.

The algorithmic and operational research literature include many route planning problem modeling approaches, effectively simplified versions of the TTDP. One of the simplest problems that may serve as a basic model for TTDP is the orienteering problem (OP) [168] (see Figure 14(a)). The OP (also known as the selective travelling salesman person problem [122] and maximum collection problem [42]) is based on the orienteering game, in which several locations with an associated profit have to be visited within a given time limit. Each location may be visited only once, while the aim is to maximize the overall profit collected on a single tour. The OP clearly relates to the TTDP: the OP locations are POIs associated with a profit (i.e. user satisfaction) and the goal is to maximize the profit collected within a given time budget (time allowed for sightseeing in a single day).

Extensions of the OP have been successfully applied to model the TTDP. The team orienteering problem (TOP) [48] (see Figure 14(b)) extends the OP considering multiple routes (i.e. daily tourist itineraries). The TOP with time windows (TOPTW) (see Figure 15) considers visits to locations within a predefined time window (this allows modeling opening and closing hours of POIs). The time-dependent TOPTW (TDTOPTW) considers time dependency in the estimation of time required to move from one location to another (this is suitable for modeling multi-modal transports

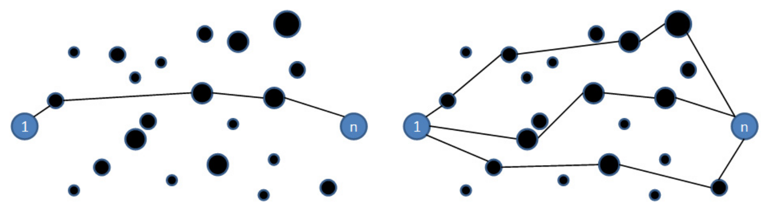


Figure 14: (a) OP illustration; (b) TOP illustration. Circles' radius denote nodes' profit.

among POIs). Several further generalizations exist that allow even more detailed modeling of the TTDP, e.g. taking into account multiple user constraints (MCTOPTW) such as the overall budget that may be spent for POI entrance fees.

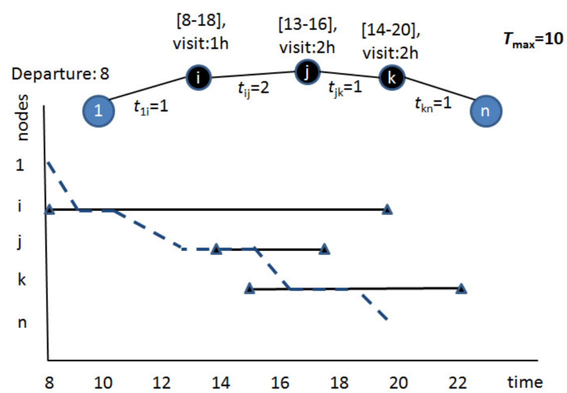


Figure 15: OPTW illustration (dashed lines denote the sheduled route, while triangles opening/closing times)

A non-exhaustive illustration of the optimization problems with relevance to the TTDP as referred to in the literature is given in Figure 16. In the next section we survey exact, approximate and heuristic approaches on those problems. It is noted that particular emphasis is given to problems highly relevant to TTDP (e.g. TOPTW and TDTOPTW).

4.5 Algorithmic approaches on problems relevant to TTDP

4.5.1 Orienteering Problem (OP)

The Orienteering Problem (OP) was introduced by Tsiligirides [168] named after a sport game called orienteering. Other names used for OP are Selective Traveling Salesperson Problem (STSP) [122], Maximum Collection Problem (MCP) [106] and Bank Robber Problem [20]. The problems that we are going to deal with in eCOMPASS are most closest modeled with variants emanating from OP. OP can be formulated as follows: Let $G = (V, E)$ be an edge-weighted graph with profits (rewards or scores) on its vertices. Given a starting node s, a terminal vertex t and a positive time limit (budget) B, the goal is to find a path from s to t (or tour if $s = t$) with length at most B such that the total profit of the visited vertices is maximized.

OP can be formulated as an integer problem as follows [172]: Let N be the number of vertices labelled by $1, 2, \ldots, N$ where $s = 1$ and $t = N$, p_i be the profit of visiting node i and c_{ij} be cost of traveling from i to j. For every path from 1 to N, if node i is followed by node j we set the variable x_{ij} equal to 1 or equal to 0 otherwise. Finally, u_i denotes the place of node i in the path. With

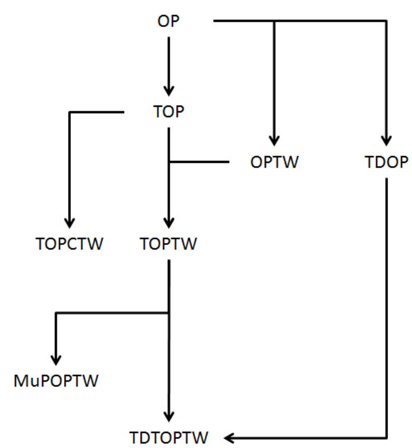


Figure 16: Optimization problems relevant to the TTDP (arrows denote problem extensions/generalizations).

this notation we have the following relations:

$$\max \sum_{i=2}^{N-1} \sum_{j=2}^{N} p_i x_{ij}, \tag{1}$$

s.t.

$$\sum_{j=2}^{N} x_{1j} = \sum_{i=1}^{N-1} x_{iN} = 1, \tag{2}$$

$$\sum_{i=1}^{N-1} x_{ir} = \sum_{j=2}^{N} x_{rj} \leq 1, \text{ for all } r = 2, \ldots, N-1, \tag{3}$$

$$\sum_{i=1}^{N-1} \sum_{j=2}^{N} c_{ij} x_{ij} \leq B, \tag{4}$$

$$2 \leq u_i \leq N, \text{ for all } i = 1, 2, \ldots, N, \tag{5}$$

$$u_i - u_j + 1 \leq (N-1)(1 - x_{ij}), \text{ for all } i, j = 2, \ldots, N, \tag{6}$$

$$x_{ij} \in \{0, 1\}, \text{ for all } i, j = 1, \ldots, N. \tag{7}$$

The objective function (1) is to maximize the total profit of visited vertices. Constraint (2) ensures that the path starts at node 1 and ends at node N. Constraint (3) ensures that the path starting at node 1 and ending at node N is connected and each node is visited at most once. Constraint (4 ensures that the path meets the time budget. Finally, constraints (5) and (6) ensure that there are no closed subtours.

The most important variants of OP considered so far depend on whether the graph is undirected or directed (undirected OP or directed OP), whether the vertices have different or the same profits (weighted OP or unweighted OP), whether there is no fixed terminal node but only a fixed starting node called root (rooted OP) or whether there are not fixed end points at all (unrooted OP) and their combinations. OP is harder than rooted OP, which in turn is harder than unrooted OP, since algorithms for rooted OP can be used to solve unrooted OP by considering each node of the graph as the root. Furthermore, OP can be used to solve rooted OP by considering as the starting node the root and each node of the graph as the finishing node.

OP is NP-hard (e.g. see [95], [122]). Hence, exact solutions for OP are only feasible for graphs with a small number of vertices. Some of the exact algorithms proposed for the OP are based on branch-and-bound [122, 150] and branch-and-cut [89, 79]. There exist a number of approximation algorithms for the above variants of OP, however with high complexity. Note that rooted OP is APX-hard (e.g. see [37], where it is proved that rooted OP is NP-hard to approximate to within a factor of $\frac{1481}{1480}$).

Some helpful remarks concerning the approximability of certain OP variants are the following:

- In the approximation algorithms for the OP, the input graph can be restricted to graphs having nodes with unit profit since Korula [113, Lemma 2.6] proved that an a-approximation algorithm for OP with unit profits yields an $a(1+O(1))$-approximation algorithm for weighted OP. The basic idea is to use a standard scaling technique to adjust the weights into integers from 1 to n^2, where n is the number of vertices, and then to transform the graph to a new graph with at most n^3 vertices having unit profits. A solution with the above approximation is derived for the weighted OP by applying an a-approximation algorithm on the newly transformed graph.

- An approach for approximating the unrooted OP in undirected graphs comes from approximation algorithms for the k-TSP problem (find a tour of minimal length while visiting at least k nodes). The basic idea is to break such a tour into pieces bounded by B and then pick the one with the largest profit (for more details, see [23]).

- Usually, the approximation algorithms for OP have highest complexity in directed graphs than in undirected graphs (e.g. see [50]).

The fundamental idea to approximate the rooted OP in undirected graphs was presented by Blum et al. in [37]. They use, as an intermediate step, the solution of the min-excess $(s-t)$ path problem (find a minimum-excess path connecting fixed nodes s and t[1] that visits at least k nodes or collecting at least k profit). The basic idea is to guess[2] the profit OPT of the optimal solution of the rooted OP and try for every node to compute the min-excess path from the root to this node that collects at least a fixed fraction of OPT, until a path is found that has length at most B. In this work they obtain a 4-approximation algorithm for rooted OP in undirected graphs by using a $(2+\epsilon)$-approximation to the min-excess $(s-t)$ path problem. In fact, most subsequent approximation algorithms (e.g. see [50]) use the min-excess path problem as intermediate step.

Later, Bansal et al. [26] give a 3-approximation algorithm for OP in metric spaces. In their approach they show that a $(2+\epsilon)$-approximation to the min-excess $(s-t)$ path problem can be used to obtain a 3-approximation for OP and hence improving the previous result by Blum et al [37], [36].

Chekuri and Pal [52] give an $O(\log OPT)$-approximation algorithm for solving the OP in directed graphs. In their formulation of OP, called submodular OP, the total profit of the vertices visited is not necesserily the sum of the profit of each node but has the submodular property, i.e., for subsets A, B of the set of vertices the total weight f satisfies the inequality: $f(A \cup B) \leq f(A) + f(B) - f(A \cap B)$.

Chen et al. [53] present a PTAS for the rooted OP in $\mathbb{R}^d$, where every location has unit profit. In order to create the PTAS an approximation algorithm is presented for the $k-$TSP in $\mathbb{R}^d$ based on Mitchell's [138] approximation algorithm for the $k-$TSP and Arora's work on the same problem[21]. Chekuri et al. [50], following Blum et al. [37], give approximation algorithms for the OP (in directed and undirected graphs). In particular, they give a $(2+\epsilon)$ approximation algorithm for the undirected OP with running time $n^{O(1/\epsilon)}$ and $O(\log^2 OPT)$ approximation algorithm for directed OP, where OPT denotes the number of vertices in an optimal solution. They follow Blum et al. and focus on the k-stroll problem (find a minimum length s-t path that visits at least k vertices) and give

[1] excess of an $s-t$ path is the difference of the path length from the shortest $s-t$ path.

[2] i.e., try exhaustive search

bi-criteria approximations for k-stroll in directed and undirected graphs with respect to the path length and the number of vertices visited.

Nagarajan and Ravi [142] give a $O(\frac{\log^2 n}{\log \log n})$-approximation algorithm for OP in directed graphs, (improving the previously best known result by Chekuri and Pal [52]), by approximately solving a number of problem in the the following order: from minimum ratio ATSP to directed k-path problem, then to the minimum excess problem and finally to OP in directed graphs. One of the first works for approximating a rooted OP variant is that of Arkin et al. [20] that gives an $(2 + \epsilon)$-approximation algorithm for OP restricted to points in the 2-dimensional plane.

For practical applications, many researchers propose heuristics to tackle the OP based on different solution approaches. Some characteristic methods are the following:

Tsiligirides [168] presents two algorithms for OP. A stochastic algorithm based on Monte-Carlo techniques that constructs a large number of routes and picks the one with the maximum profit and a deterministic heuristic algorithm, that partitions the geographic area into concentric circles and restricts the allowed routes into the sectors defined by the circles.

In [95] a center-of-gravity heuristic for the OP is presented where the solution tour is constructed by the cheapest insertion procedure according to a combined measure for node selection. Golden et al. in [94] improve the center-of-gravity heuristic by rewarding nodes associated with above-average tours while penalizing those associated with below-average tours.

In [149] Ramesh et al. propose a four-phase heuristic. After choosing the best solution from iterations over a set of three phases (node insertion, edge exchange and node deletion), a fourth phase is entered, where one attempts to insert unvisited nodes into the tour.

In [179] the authors apply a neural network approach to solve the OP. They derive an energy function and learning algorithm for a modified, continuous Hopfield neural network.

Chao et al. [47] propose a heuristic algorithm for OP that proceeds as follows. Initially, the set of nodes is partitioned in a greedy way into paths each with length bounded by B and the current solution is the path with the most profit. Then an iterative method is employed. At each iteration a local search procedure is applied to improve the current solution. However, if a better solution is not found, a solution with slightly less profit is accepted. At the end of the iteration a perturbation move is applied and a number of nodes (that depends on the current iteration) with the smallest ratio of profit to insertion cost are removed from the solution.

In [90] a tabu search heuristic for the unrooted OP is presented. The algorithm iteratively inserts clusters of nodes in the current tour or removes a chain of nodes. Compared to the previous approaches, this method reduces the chance to get trapped in a local optimum. Tests performed by the authors on randomly generated instances with up to 300 vertices show that the algorithm yields near-optimal solutions.

4.5.2 Orienteering Problem with Time Windows (OPTW)

In OP with Time Windows (OPTW) each vertex of the graph G can be visited only within one or more specific time intervals (windows) which may be different for each vertex. Vansteenwegen et al. [172] argue that time windows significantly affect the nature of OP and its respective algorithmic approaches. For instance, reducing the travel time by reordering scheduled visits, is no longer appropriate due to the time windows. Actually, it has been proved that OPTW is NP-hard even on the line [169].

Righini et al. [153] give two exact dynamic programming algorithms for OPTW. The first algorithm uses bidirectional search and the label of each node u used in the algorithm is a binary vectors representing the nodes included in the path ending at u. In the second method, the state space relaxation (SSR) [58] is applied, where the label now is only an integer denoting the number of visits along the path. Since now a node may be visited more than once due to the reduced information kept at each label, the authors correct this by applying the decremental SSR (DSSR) method [152] which is an iterative algorithm optimally solving the relaxed problem with the additional constraint that a specific set of nodes cannot be visited more than once.

Kantor and Rosenwein [105]proposed two heuristics for solving the OPTW. The first, the insertion heuristic, incrementally builds the solution and at each step it selects the node with the highest ratio of profit over insertion cost as the next node to be inserted in the path. The second heuristic, the tree heuristic, is employed when the time windows constraints are tight and the numbers of nodes of input graph are relatively few. By a depth first search exploration of the input graph, it maintains a number of partial solutions simultaneously and repeatedly inserts new nodes in these partially constructed paths as long as the attempted insertion satisfies the problem constraints as well as some heuristic criteria showing the potential of this insertion for producing a final good solution.

Also a number of approximation algorithms have been proposed for OPTW in the literature. Bansal et al. [26] gave an $(3\log^2 n)$-approximation algorithm for OPTW. The main idea is the partition of vertices into different groups according to their time windows and in such a way that OPTW can be solved in each group ignoring time windows. The final solution is derived by stitching the solutions of these subproblems using a dynamic programming approach.

Chekuri and Kumar [51] gave a 5-approximation algorithm for OPTW with at most k distinct time windows that runs in time polynomial in $(n\Delta)^k$, where Δ is the maximum distance in the metric space and n is the number of vertices. They utilize an approximation algorithm for the maximum coverage problem with group budget constraints [3] and a 3-approximation algorithm of Bansal et al [26] for OP.

Later, Chekuri and Pal [52] gave an $O(\log OPT)$-approximation algorithm for rooted OPTW in directed graphs where the total weight of the vertices visited has the submodular property. Their approach, based on a variant of an algorithm for directed s-t connectivity due to Savitch [154], is recursive and greedy and runs in quasi-polynomial time. An application of this algorithm can be found in [57] where travel itineraries for a city are constructed from information collected in the social breadcrumb *Flickr* about the preferences of tourists visiting the city.

Also, Chekuri et al. [50] inspired by the technique of Bansal et al. [26] proved that an α-approximation algorithm for OP yields an $O(\alpha \max\{\log OPT, \log L\})$ approximation algorithm for OPTW in directed (and undirected) graphs, where OPT denotes the number of vertices in an optimal solution and L is the ratio of the longest to the shortest time window.

Finaly, Frederickson et al. [83] proposed approximation algorithms for the travelling repairman problem (TRP) in a metric graph or a tree. TRP requests the path passing through the maximum number of nodes with each node visited within its time window. First, the algorithm trims all time windows into subwindows with specific ends and then for the nodes of each time window, the optimum k-path from s to t [49] is solved. Last, the solutions found for each time window are combined into a solution to the TRP by applying a dynamic programming approach. For the case that all time windows have equal length, it is proved that the optimal solution for the trimmed time windows is within factor of 3 from the optimal solution before trimming. Using the above result, the algorithm has a 3-approximation ratio with running time $O(n^4)$ when the input graph is a tree and a $6+\epsilon$-approximation for a general graph with $n^4 \cdot n^{O(\frac{1}{\epsilon^2})}$ running time. Then, the authors generalize their method for time windows with different lengths and they derive an $O(\log L)$-approximation algorithm where L is the ratio of the maximum to minimum time length of all input windows.

4.5.3 Team Orienteering Problem (TOP)

The extension of the OP to multiple tours was defined as the Team Orienteering Problem by Chao et al. [48]. The TOP first appeared in the literature with the name Multiple Tour Maximum

[3] Given an integer k and a collection of subsets, of a set S, partitioned into groups, pick k subsets of that collection such that the cardinality of their union is maximized with the restriction that at most one set is picked from each group.

Collection Problem (MTMCP) by Butt and Cavalier [42]. TOP is an extension of OP where the goal is to find k paths (or tours) each with length bounded by B, that have the maximum total collected profit (each non-starting, non-terminal node is visited at most once along the k paths). TOP is NP-hard and APX-hard since OP is a special case of TOP.

TOP can be formulated as an integer problem as follows [172]:

Further to the notation for OP, given the integer k, let x_{ijm} be equal to 1 if node i is followed by node j in path m or equal to 0 otherwise, y_{im} be equal to 1 if node i is visited in path m or equal to 0 otherwise and u_{im} be the position of node i in path m. With this notation we have the following relations:

$$\max \sum_{m=1}^{k} \sum_{i=2}^{N-1} p_i y_{im}, \tag{8}$$

s.t.

$$\sum_{m=1}^{k} \sum_{j=2}^{N} x_{1jm} = \sum_{m=1}^{k} \sum_{i=1}^{N-1} x_{iNm} = k, \tag{9}$$

$$\sum_{m=1}^{k} y_{rm} \leq 1, \text{ for all } r = 2, \ldots, N-1, \tag{10}$$

$$\sum_{i=1}^{N-1} x_{irm} = \sum_{j=2}^{N} x_{rjm} = y_{rm}, \text{ for all } r = 2, \ldots, N-1, m = 1, \ldots, k \tag{11}$$

$$\sum_{i=1}^{N-1} \sum_{j=2}^{N} c_{ij} x_{ijm} \leq B \text{ for all } m = 1, \ldots, k, \tag{12}$$

$$2 \leq u_{im} \leq N, \text{ for all } i = 1, 2, \ldots, N, \ m = 1, \ldots, k \tag{13}$$

$$u_{im} - u_{jm} + 1 \leq (N-1)(1 - x_{ijm}), \text{ for all } i, j = 2, \ldots, N, \ m = 1, \ldots, k \tag{14}$$

$$x_{ijm}, y_{im} \in \{0, 1\}, \text{ for all } i, j = 1, \ldots, N, \ m = 1, \ldots, k \tag{15}$$

The objective function (8) is to maximize the total profit of visited vertices. Constraints (9) and (10) ensure that each of the k paths starts at node 1 and ends at node N and that each non-starting, non-terminal node is visited at most once. Constraint (11) ensures that each path starting at node 1 and ending at node N is connected. Constraint (12 ensures that the path meets the time budget. Finally, constraints (13) and (14) ensure that there are no closed subtours.

Exact algorithms for TOP are presented by Butt et al. [43] and Boussier et al. [39]. Butt et al. [43] give an algorithm that optimally solves TOP by solving the relaxation of the problem with the column generation technique together with a branch and bound technique for deriving increasingly better solutions. Specifically, the problem is formulated as a set-partitioning problem and then a column generation procedure is applied. When applying the branch and bound technique, the solution space is partitioned around a specific node pair $\{u, v\}$ with one subspace containing solutions where both u, v belong to the same tour and the other one containing solutions where these two nodes cannot be part of the same tour. The combination of column generation and branch-and-bound technique (also known as branch-and-price in the literature) has also been applied in [39] for optimally solving the TOP. The selection of the new columns to be included at each step of column generation is reduced to solving an instance of Elementary Shortest Path Problem with Resource Constraint by using a dynamic programming approach. Finally, in a branch and bound phase, different branches are created according to either whether a node should be visited or not or whether a particular edge should be included in a tour or not.

Blum et al. [37] present an approximation algorithm for variants of TOP in undirected graphs, where the paths have a common start point and not a fixed end point or they are mutually disjoint.

Their main idea is to iteratively apply algorithms for rooted OP setting already visited node profits to zero. For the former case, applying this procedure using an α-approximation algorithm for rooted OP, an $1/(1-e^{-\alpha)}$ approximation ratio is obtained. While, in the latter case where the paths are mutually disjoint, using an α-approximation algorithm for rooted OP, an $(\alpha+1)$ approximation ratio is obtained.

In the following, we outline heuristic approaches for TOP.

The first heuristic algorithm for TOP was presented by Butt and Cavalier [42]. They proposed a greedy algorithm that constructs the k tours succesively. Every pair of vertices obtains a weight that gives an estimate of how advantageous it is to include both vertices in the same tour. Every tour initialy contains the depot and the node pair with the greater weight. Then, at each step the node belonging to the heaviest pair of nodes with one of these nodes already in the tour is added to tour provided that this insertion is feasible.

The heuristic algorithm for the TOP presented by Chao et al. in [48] extends the one presented by the same authors for the OP in [47]. The main differences of the two algorithms are two. The first difference is that in TOP the current solution contains the k paths with the most profit instead of the one most profitable for OP. The other difference is that in TOP there are two perturbation moves instead of one that holds for OP. The one move is the same for both TOP and OP. In the second move of the TOP algorithm, a number of nodes with the lowest profit are removed from the paths of the solution.

A guided local search [177] metaheuristic algorithm for the TOP is presented by Vansteenwegen et al. [173]. A solution to the problem is initialized as in [48] and a local search procedure is applied to improve it. Finally, guided local search is employed to ameliorate the effectiveness of the local search.

Archetti et al.[18] presented three metaheuristics solving the TOP. After defining a number of local search moves that can be applied in the solution space of the problem at hand, they present two tabu search techniques and one variable neighborhood search[100] heuristic which iteratively apply some of the previously defined local search moves for gradually improving the solution derived at each step.

In [164] a tabu search heuristic for TOP is given consisting of three basic steps: initialization, solution improvement and evaluation. The tabu search heuristic is embedded in an adaptive memory procedure that alternates between small and large neighborhood stages during the solution improvement phase. Both random and greedy procedures for neighborhood solution generation are employed, and infeasible as well as feasible solutions are explored in the process.

An Ant Colony Optimization-based heuristic algorithm is proposed by Ke et al. [108] for TOP. Specifically, an iterative procedure is followed wherein the ants generate k feasible tours by succesively inserting promising edges from previous iterations associated with relatively low cost and high profit in their endnodes.

In [160] the authors employ the Greedy Randomised Adaptive Search Procedure (GRASP) to solve TOP. GRASP is a metaheuristic first introduced by Feo and Resende in [77]. GRASP performs a number of iterations that consist of a constructive procedure followed by a local search approach. The constructive procedure based on a ratio between greediness and randomness, inserts nodes one by one until all paths are full. Thus, a new initial solution is generated during every iteration. Then, the initial solution is improved by the local search procedure which alternates between reducing the total time of the solution and increasing its total profit, until the solution is locally optimal.

Bouly et al. [38] propose a genetic algorithm for TOP enhanced with local search techniques. A population of chromosomes is constructed where a chromosome is a sequence of nodes from which a solution to TOP is obtained by applying a PERT[4] like technique. A child chromosome is produced by a couple of chromosomes by applying a crossover technique followed by a local search procedure with a certain probability.

Muthuswamy et al. [141] tackle the TOP using discrete particle swarm optimization (PSO),

[4]program evaluation review technique

creating one tour at a time. At each step a population of particles is generated such that each particle represents a feasible tour. Then, using PSO particles are heading for more profitable solutions (tours). The whole procedure is enhanced with local search techniques.

4.5.4 Team Orienteering Problem with Time Windows (TOPTW)

The TOPTW extends TOP adding the constraint of limited time availability of serviced nodes (this corresponds to the opening and closing hours of a POI).

Team OP with TW (TOPTW) was introduced by Vansteenwegen P. [171]

Exact solutions for TOPTW are feasible for graphs with very restricted number of vertices (e.g. see the work by Z. Li and X. Hu [127] which is used on networks of up to 30 nodes).

Li and Hu formulated the Team Orienteering Problem with Capacity Constraint and Time Window (TOPCTW) [127] (an extension of TOPTW where each “customer” has a demand and the serving vehicle has a capacity limitation) and obtained exact solutions using an integer linear programming solver. However, this approach is inappropriate for real-time applications.

Given the complexity of the problem, the main body of TOPTW literature exclusively involves heuristic algorithms. Notably, existing methods are metaheuristics that involve, (a) an insertion step (adds a visit to one of the k tours) iteratively performed until a first feasible solution (or a set of feasible solutions) is obtained, and (b) a sort of local search step that aims at escaping from local optima. Those two steps are repeated until a termination criterion is met. Depending on the insertion step principle, existing methods are designated either as deterministic (those that always produce the same solution for given problem instances) or as stochastic or probabilistic (those that involve a degree of randomness in solutions generation). Probabilistic methods are generally shown to yield high quality solutions (as they perform more extensive search of the solution space) at the expense of increased execution time.

Labadi et al. [117] propose a local search heuristic algorithm for TOPTW based on a variable neighbourhood structure. In the local search routine the algorithm tries to replace a segment of a path with nodes not included in a path that offer more profit. For that, an assignment problem related to the TOPTW is solved and based on that solution the algorithm decides which arcs will be entered in the path.

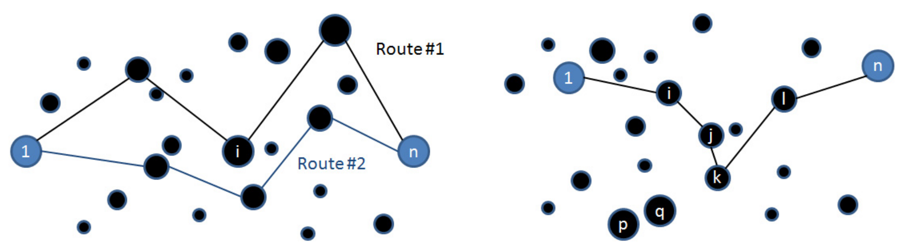
Lin et al. [131] propose a heuristic algorithm based on simulated annealing (SA) for TOPTW. In each iteration a neighbouring solution is obtained from the current solution by applying one of the moves swap, insertion or inversion, with equal probability. If the new solution is more profitable than the current and with a probability depending on the difference of profits of the two solutions in the opposite case, the new solution is adopted and becomes the current one. After applying the above procedure for a certain number of iterations the best solution found so far is further improved by applying local search.

The Iterated Local Search (ILS) heuristic proposed by Vansteenwegen et al. [174] is the fastest known algorithm proposed for TOPTW [172]. ILS defines an “insertion” and a “shake” step. The insertion step adds, one by one, new visits to a tour, ensuring that all subsequent visits (those scheduled after the insertion place) remain feasible, i.e. they still satisfy their time window constraint. For each visit i that can be inserted, the cheapest insertion time cost is determined. For each of these visits the heuristic calculates a ratio, which represents a measure of how profitable is to visit i versus the time delay this visit incurs. Among them, the heuristic selects the one with the highest ratio for insertion. The shake step is used to escape from local optima. During this step, one or more visits are removed in each tour in search of non-included visits that may either decrease the tour time length or increase the overall collected profit.

Overall, ILS represents a fair compromise in terms of speed versus deriving routes of reasonable quality. However, ILS presents a number of shortcomings:

- ILS first finalizes each tour before proceeding to the next one, instead of inserting visits interchangeably to generated tours. In several scenarios, this may lead to ILS incorporating nodes to the wrong tour, while those nodes could find a better match if connected to another itinerary (i.e. yield decreased route completion times). A side-effect is that ILS itineraries are unbalanced, as relatively "important" POIs are absorbed into the first routes (see Figure 17(a)).

- During the 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 that the route instance gradually grows and converges towards the high profit nodes, those may be no longer feasible to insert due to overall tour time constraints. For instance, in Figure 17(b), ILS inserts i, l, j and k. Although p and q have larger profit value, they are not selected on the first four insertion steps since they are associated with large $Shift$ values. On the next step, q is associated with the highest $Ratio$, however its insertion violates the tour feasibility constraint; hence, it is not performed.
- The ILS shake step examines a very narrow space of alternative solutions. For instance ILS neglects swaps among visits included on the same or different itineraries which could potentially decrease the involved tours' length, thereby creating room for accommodating new visits until a new local optima is reached.



(a) (b)

Figure 17: Weaknesses of ILS.

Montemanni and Gambardella proposed an ant colony system (ACS) algorithm [139] to derive solutions for a hierarchical generalization of TOPTW, wherein more than the k required routes are constructed. At the expense of the additional overhead, those additional fragments are used to perform exchanges/insertions so as to improve the quality of the k tours. The algorithm comprises two phases:

- Construction phase: Ants are sent out sequentially; when at node i, an ant chooses probabilistically the next node j to visit (i.e. to include into the tour) based on two factors:
  - The pheromone trail τ_{ij} (i.e. a measure on how good it has been in the past to include arc (i, j) in the solution).
  - The desirability n_{ij}, (a node j is more desirable when it is associated with high profit, it is not far from i, and its time window is used in a suitable way).
- Local search: performed upon the solutions derived from construction phase, aiming at taking them down to a local optimum.

ACS has been shown to obtain high quality results (that is, low average gap to the best known solution) at the expense of prolonged execution time, practically prohibitive for online applications.

Tricoire et al. [167] deal with the Multi-Period Orienteering Problem with Multiple Time Windows (MuPOPTW), a generalization of TOPTW, wherein each vertex may be assigned more than one time window on a given day, while time windows may differ on different days. Both mandatory and optional visits are considered. The motivation behind this modelling is to facilitate individual route planning of field workers and sales representatives. The authors developed two heuristic algorithms for the MuPOPTW: a deterministic constructive heuristic which provides a starting solution, and a stochastic local search algorithm, the Variable Neighbourhood Search (VNS), which considers random exchanges between chains of nodes.

Vansteenwegen et al. [172] argue that a detailed comparison of TOPTW solution approaches (i.e. ILS, ACS and the algorithm of Tricoire et al. [167]), is impossible since the respective authors have used (slightly) different benchmark instances. Nevertheless, it can be concluded that ILS has the advantage of being very fast, while ACS and the approach of Tricoire et al. [167] (2010) have the advantage of obtaining high quality solutions.

Labadi et al. [118], [119] recently proposed a method that combines the greedy randomized adaptive search procedure (GRASP) with the evolutionary local search (ELS). GRASP generates independent solutions (using some randomized heuristic) further improved by a local search procedure. ELS generates multiple copies of a starting solution (instead of a single copy generated in ILS) using a random mutation (perturbation) and then applies a local search on each copy to yield an improved solution. GRASP-ELS derives solutions of comparable quality and significantly less computational effort to ACS. However, its performance lags far behind ILS.

Garcia et al. modelled the TTDP as a Multi-Constrained Team Orienteering Problem with Time Windows (MCTOPTW) [85]; each visit in the MCTOPTW is associated with a number of attributes; the sum of those attributes values is bounded by a max value (e.g. the sum of attractions entrance fee should not exceed an overall budget or the total time spent in parks cannot exceed a given time threshold). The proposed algorithm is based on ILS [174], incorporating two different aspects: (a) The feasibility check of visit insertions caters for checking attribute constraints in addition to time feasibility; (b) the ratio function determining the candidate visit to be inserted is adapted so as to associate each attribute constraint with a special weight and include the available quantity of each constraint on the route. For instance, if the total entrance fee constraint is assigned a relatively high weight, the algorithm favours insertions of visits with relatively low entrance fee, even more so if currently selected visits sum to low overall fee (relatively to the fee threshold).

Souffriau et al. [161] studied the Multi-Constraint Team Orienteering Problem with Multiple Time Windows (MCTOPMTW), in effect an extension of MCTOPTW which allows defining different/ multiple time windows for different days. The proposed MCTOPMTW algorithm is based on a hybrid ILS-GRASP approach: GRASP yields an initial solution (GRASP involves a degree of randomness in the insertion phase) and the 'shake' routine of ILS is used thereafter to derive an improved solution. The authors report that the ILS-GRASP algorithm yields fairly quality solutions, while achieving computation time suitable for online applications.

4.5.5 Time Dependent Team Orienteering Problem with Time Windows (TDTOPTW)

Time-dependent route planning incorporates time dependency in calculating cost of edges, i.e. travelling times among vertices. Time dependency is useful for modeling transfers among nodes through multimodal public transportation, hence it is considered particularly relevant to eCOMPASS. Time-dependent graphs has been used in almost all variants of the orienteering problems, from the basic OP to the TOPTW.

Time Dependent OP (TDOP) was introduced by Formin and Lingas [82]. TDOP is MAX-SNP-hard since a special case of TDOP, time-depenent maximum scheduling problem is MAX-SNP-hard [162]. An exact algorithm for solving TDOP is given by Li et al [125] using a mixed integer programming model and a pre-node optimal labeling algorithm "based on the idea of dynamic programming". Moreover, Li [124] proposes an exact algorithm for TDTOP based on dynamic programming principles. However, both algorithms are of exponential complexity. Fomin and Lingas

[82] give a $(2+\epsilon)$ approximation algorithm for rooted and unrooted TDOP ("which runs in polynomial time if the ratio R between the maximum and minimum traveling time between any two sites is constant"). When considering unrooted TDOP its running time is $O\big(\big(2R^2(\frac{2+\epsilon}{\epsilon})\big)!\frac{2R^2}{\epsilon}n^{2R^2(\frac{2+\epsilon}{\epsilon})+1}\big)$, and for rooted TDOP its running time increases by the multiplicative factor $O(\frac{Rn}{\epsilon})$. (The key idea is derived from Spieksma's algorithm [162] for Job Interval Selection Problem). They use a divide-and-conquer approach. First the problem is split in smaller ones. Exact solutions are found to each smaller problem and later combined (stitch) to obtain an approximate solution.

Abbaspour et al. [3] investigated a variant of Time Dependent OP with Time Windows (TDOPTW) in urban areas, where the nodes are partitioned into the POIs (associated with profits and time windows) and multimodal transportation stops which do not have profit. A genetic algorithm is proposed for the problem that uses as a subroutine another genetic algorithm for solving the shortest path problem between POIs.

Time Dependent TOP with Time Windows (TDTOPTW) is the problem that best matches TTDP requirements among all problems and approaches surveyed in this Section. TDTOPTW is particulary complex as it adds time dependency of arcs to TOPTW.

Zenker et al. [186] described a tourism-inspired problem that refers to TDTOPTW and presented ROSE, a mobile application assisting pedestrians to locate events and locations, moving through public transport connections. ROSE incorporates three main services: recommendation, route generation and navigation. The authors identified the route planning problem to solve and they described it as a multiple-constrained destination recommendation with time windows using public transportation. However, no algorithmic solution to this problem has been proposed.

The work of Garcia et al. [85], [86] is the first to address algorithmically the TDTOPTW extending previous work on TDOPTW [84]. The authors presented two different approaches to solve TDTOPTW, both applied on real urban test instances. The first approach involves a precalculation step, computing the average travel times between all pairs of POIs, allowing reducing the TDTOPTW to a regular TOPTW, solved using the insertion phase part of ILS. In case that the derived TOPTW solution is infeasible (due to violating the time windows of nodes included in the solution), a number of visits are removed. The second approach introduces four additional variables per vertex and is based on a fast evaluation of the possible insertion of an extra POI. The authors argue that their approach is suitable for real-time applications, requiring slightly longer computational time than fast TOPTW algorithms (i.e. ILS) to derive sufficiently quality solutions. The main weaknesses identified with the respect to their proposed algorithm, are the following:

- the modelling is based on the simplified assumption of periodic service schedules; multi-modal transports are not supported, i.e. moving from a POI to another may either be realized on foot or though a single mode of public transportation;
- the solutions yield using the second approach are not perturbed in order to reduce computational overhead; hence, those solutions are sensitive to the quality of the insertion phase;
- a time-dependent Dijkstra algorithm is used in the first approach for calculating average origin-destination travel times (computational time could be reduced through applying more efficient algorithms).

4.5.6 Variants of TTDP

The Travelling Salesman Problem with Profits (TSPP) is a bicriteria generalization of TSP with two conflicting objectives. In TSPP we are given a network in which nodes are associated with profits and links with travel costs, and the goal is to find a tour (which starts and finishes at a specified node - the depot) over a subset of nodes such that the collected profit is maximized while the travel cost is minimized. The problem was introduced under the name multiobjective vending problem in [109]. In [33] the authors gave the first exact Pareto fronts (sets of non-dominated solutions) for TSPP instances obtained from classical TSP instances available in the TSPLIB [151]. In [104] a hybrid meta-heuristic was presented that yields high-quality approximations of the efficient frontier for TSPP.

There are three single-criterion variants of TSPP based on how the two objectives of maximizing the collected profit and minimizing the travel cost are handled:

(i) The OP seeks for a tour that maximizes the total collected profit while maintaining the travel cost under a given value, i.e., the travel cost objective is stated as a constraint.

(ii) The Profitable Tour Problem (PTP) introduced in [63], searches for a tour that maximizes the collected profit minus the travel cost, i.e., the two objectives are combined in one objective function.

(iii) The Prize Collecting TSP (PCTSP) introduced in [25] aims at finding a tour that minimizes the travel cost and its total profit is not smaller than a given value, i.e., the profit objective is stated as a constraint.

TSP is a special case of both PTP and PCTSP and therefore, the two problems belong to the class of NP-hard problems. Bienstock et al. [35] developed the first approximation algorithm for PTP with a performance guarantee bound of 5/2. This bound was improved in [91] where a $2 - 1/(n-1)$-approximation algorithm was given, where n is the number of nodes. Awerbuch et al. [23] gave an approximation algorithm for the PCTSP based on an approximation algorithm for the k-minimum-spanning-tree problem ([22]). There also exists literature on exact, heuristic and metaheuristic algorithms for PTP and PCTSP as well as variants of these problems (see [76] for a survey).

A number of OP variants have been introduced in the literature to model different tourist trip design (as well as other practical) problems:

1. The Generalized Orienteering Problem (GOP) wherein each node of the network is assigned a set of benefit values. For example, in the case of a POI, the benefit values may be related to natural beauty, cultural interest, historical significance, educational interest. The overall objective function may comprise any combination of the different benefits. Nonlinear objective functions make the GOP more difficult to solve than OP. In [179] a heuristic was designed to solve GOP using artificial neural networks, while in [180] a straightforward genetic algorithm was given that yields comparable results. In [158] an iterative algorithm was presented for the problem.

2. The Multi-Objective Orienteering Problem (MOOP) is the multi-objective extension of the OP which was formulated in [155] as follows. Each node (POI) may be assigned to different categories (e.g., culture, history, leisure, shopping) and provide different benefits for each category. The aim of MOOP is to find all Pareto efficient solutions without violating the maximum travel cost restriction. In [155] two metaheuristic solution techniques for the bi-objective OP were presented. The first is an adaptation of the Pareto Ant Colony Optimization metaheuristic developed by Doerner et al. [73]. The second is a multi-objective extension of Variable Neighborhood Search (VNS)[100].

3. The following stochastic variants of the OP have been studied in the literature:

    - The Orienteering Problem with Stochastic Profits (OPSP), in which the nodes are associated with normally distributed profits. The problem was introduced in [103] and aims at finding a tour that starts and finishes at the depot, visits a subset of nodes within a time limit, and maximizes the probability of collecting more than a prespecified target profit level. In [103] the authors present an exact solution approach based on a parametric formulation of the problem for solving small problem instances and a Pareto-based bi-objective genetic algorithm for larger instances that is based on the conflict between high mean profit and low variance in a solution.

- The Stochastic Orienteering problem (SOP), in which each node is associated with a deterministic profit and a random service time. The visit time of a POI is not known until the visit is completed. The problem combines aspects of both the stochastic knapsack problem with uncertain item sizes and the OP. The stochastic orienteering problem was introduced in [97] where an $O(\log\log B)$-approximation algorithm was presented.
- The Orienteering Problem with Stochastic Travel and Service Times (OPSTS) which was introduced in [44]. In this problem both travel and service times are stochastic while if a node is visited, a reward is received, but if it is not, a penalty may be incurred. This problem reflects the challenges of a company who, on a given day, may have more customers than it can serve. In [44] heuristics for general problem instances and computational results for a variety of parameter settings were given.

4. The OP with Compulsory Vertices (OPCV) discussed in [89], models the variant of OP in which a subset of the nodes has to be visited. In tourist trip design problem modelling, these compulsory nodes may be significant POIs that should be included in any itinerary. Gendreau et al. ([89]) developed a branch-and-cut algorithm to solve to optimality problem instances with up to 100 nodes, of which some are compulsory.

The Vehicle Routing Problem (VRP) can be described as the problem of designing optimal delivery or collection routes from a depot to a number of nodes subject to certain constraints. The most common constraints are (i) capacity constraints i.e., a demand is attached to each node and the sum of weights loaded on any route may not exceed the vehicle capacity, (ii) time constraints over individual routes, (iii) time windows, and (iv) precedence relations between pairs of nodes. Although many variants of the classical VRP have been studied based on different constraints (e.g., the Capacity-constrained VRP (CVRP), the Time or Distance constrained VRP (DVRP), the Vehicle Routing Problem with Time Windows (VRPTW), etc.) only a few tourist trip design problems. We discuss in the sequel two problems that can formulate useful variants of TTDP: the DVRP and the Minimum Path Cover Problem (MPCP).

In DVRP, given a depot node r and a distance constraint D the goal is to find a minimum cardinality set of tours originating from r and corresponding to routes for vehicles, that covers all the nodes in the network ([121], [123], [143]). Each tour is required to have length at most D. DVRP may formulate the following problem: We are given a set of POIs and we are asked to determine the minimum number of days that will be needed to visit all POIs without violating the constraint of the available time per day. The unrooted version of DVRP, defined as the minimum path cover problem (MPCP) in [19], seeks for the minimum number of paths each of length at most D, that cover all the nodes of the network. Note that in MPCP, the paths may start and end at any two nodes. MPCP can be reduced to DVRP by adding a depot node that is located at some large distance L from all nodes, and setting the distance constraint to $D + 2L$. In [123] DVRP was studied under the objectives of total distance and number of tours. It was shown that the optimal solutions under both objectives are closely related, and any approximation guarantee for one objective implies a guarantee with an additional loss of factor 2, for the other objective. In [143] the authors presented an $(O(\log 1/\epsilon), 1 + \epsilon)$-approximation algorithm: i.e., for any $\epsilon > 0$, the algorithm provides a solution violating the length bound by a $1 + \epsilon$ factor while using at most $O(\log 1/\epsilon)$ times the optimal number of tours. The algorithm partitions the nodes of the network into subsets, according to their distance from the depot, and solves the unrooted DVRP with appropriate distance bounds in each subset. To solve the unrooted DVRP the $3-$approximation algorithm for the minimum path cover problem of Arkin et al.([19]) is employed that proceeds as follows. First, it guesses the solution k and then finds k paths with total length at most $2kD$ that cover the nodes of the network. Finally, it cuts the paths into smaller paths with length less than or equal to D.

The above variants of VRP assume that all nodes must be visited and there is no profit collected when visiting a node. Archetti et al. ([18]) name the extension of TSP with profits to multiple tours as Vehicle Routing Problem with Profits (VRPP). In VRPP not all nodes need to be necessarily visited, a profit is collected when visiting a node, while collecting the profits is distributed over several vehicles with limited capacity. Known variants of the VRPP is the Prize-Collecting VRP (PCVRP), the Capacitated Profitable Tour Problem (CPTP) [16], and the VRP with profits and time deadlines (VRPP-TD). In PCVRP the main objective is a linear combination of three objectives: minimization of total distance traveled, minimization of vehicles used, and maximization of prizes collected [165]. In CPTP the objective is to maximize the difference between the total collected profit and the total travel cost [16]. In VRPP-TD, in addition to the capacity constraints, there are node-specific temporal constraints referred to as time deadlines. The objective function is the same with the function of CPTP [9].

The extension of the OP to multiple tours, which is a special case of VRP with profits, was defined as the Team Orienteering Problem by Chao et al. [48]. Archetti et al. ([16]) introduced the Capacitated Team Orienteering Problem (CTOP) as a TOP with an additional constraint, i.e., a nonnegative demand is associated with each node and the total demand in each tour may not exceed the given capacity constraint. They present exact and heuristic algorithms that are extentions of schemes for solving the TOP. The exact algorithm is an adaptation of a branch-and-price scheme first presented in [39], while the heuristic algorithms are based on the heursitic solutions for TOP given in [18]. In [15] a new branch-and-price scheme is presented to solve the CTOP. A column-based heuristic is applied at each node of the branch-and-bound tree in order to obtain primal bound values.

In all the above cited problems the sites/customers are represented by the nodes of a network. Also, the network nodes are associated with profits and/or demands. There is a limited literature on arc routing problems with profits i.e., problems in which the sites/customers are represented by the arcs of a network and the profits/demands are associated with the arcs.

One such problem is the Prize-collecting Rural Postman Problem (PRPP) defined in [14]. In PRPP the arcs are associated with profits and costs, and the objective is to find a tour that maximizes the difference between the collected profit and the travel cost. Note that PRPP is the arc routing counterpart of the profitable tour problem (PTP). Problems related to post delivery and garbage collection can be modelled using PRPP, which has been studied from the algorithmic point of view in [75] and in [13].

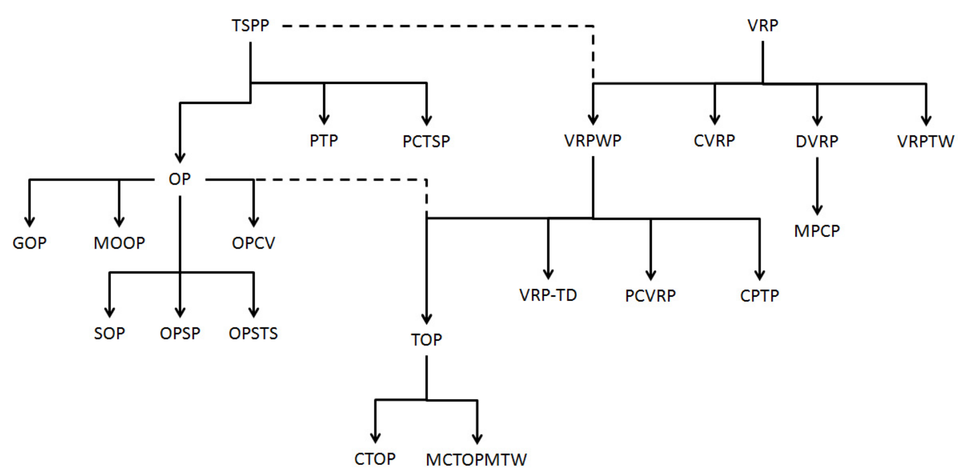


Figure 18: Variants of TTDP (solid arrows denote problem variants, while dashed arrows denote generalizations).

In [17] the undirected Capacitated Arc Routing Problem with Profits(CARPP) was considered which is the arc routing counetrpart of the capacitated TOP (CTOP): a profit and a nonnegative demand is associated with each arc and the objective is to determine a path for each available vehicle in order to maximize the total collected profit, without violating the capacity and time limit of each vehicle. The authors consider an application where carriers can select potential customers for transporting their goods. Another potential application is the creation of personal bicycle trips. Based on the biker's personal interests, starting and ending point and the available time, a personal trip can be composed using the selection of arcs that better match with the cyclist's profile.

The study of the combination of the orienteering problem and the arc routing problem with profits, under the name Mixed Orienteering Problem (MOP), where profits are associated to nodes as well as to arcs, is proposed in [172]. This problem is very interesting in the context of eCOMPASS as variants of MOP can be used to formulate TDDP variants where not only site attractions but also certain routes may be of tourist interest. To the best of our knowledge, no research has been done on MOP.

The aforementioned variants of TTDP and their relationship are shown schematically in Figure 18.

5 Discussion of new prospects in multi-modal route planning in urban areas

This Section highlights promising directions for dealing with multimodal route planning problems on the course of the WP3 of eCOMPASS. Subsection 5.1 discusses new prospects in multimodal route planning problems, Subsection 5.2 deals with promising lines of research with respect to route planning under uncertainty, while Subsection 5.3 suggests new prospects on tourist route planning.

5.1 New Prospects in Multi-Criteria Multi-Modal Route Planning

While the field of Route Planning in transportation networks has made rapid progress in the last ten years fully-realistic scenarios are far from solved, yet. As outlined in subsection 2.3.3, current approaches fall short on either the complexity of the considered scenario, the richness of the query response or on average query response time. For eCOMPASS we aim to overcome such weaknesses and solve realistic scenarios that involve the whole scale of urban human mobility in a city such as Berlin, Germany, and tend to address the needs of both residents, commuters as well as tourists.

Realistic multi-modal scenarios involve several tightly integrated layers of mobility such as walking, biking (private), usage of a private car navigation, different public transportation networks (underground, metro, bus, tram, ferry, night routes, regional trains, . . .), taxi or bike rentals. The goal of WP3 is to present a user several alternative journeys from door-to-door provably optimizing one of several criteria such as eco-friendliness, travel time, number of transfers between vehicles or between modes of transportation and overall price. It is also interesting to minimize or maximize the use of a specific mode of transportation in order to present journeys with short walks or journeys with long undisturbed rides in public transportation (good for working or reading). As traffic incidents, rush hours and delays on public transportation are more than common—and have a large impact on emissions—our algorithms will account for live updates of road and transit data.

A promising direction for the eCOMPASS consortium is to extend the previous research on multi-modal preprocessing techniques towards multi-criteria multi-modal search that is able to find several alternative optimal journeys through a multi-modal network while considering the modes used as well as time, transfers and price, and the environmental impact of each route. Integration with the newest accomplishments in multi-criteria transit route planning should enable fast multi-modal queries that can account for regular updates in the public transportation network. As for live delays on the road network, we will consider the integration of recent results in route planning for road networks (see Deliverable D2.1) that could enable this—even in a multi-modal setting.

When comparing road against public transportation travel, door-to-door earliest arrival queries tend to be biased: as the departure is an input to the query (and thus not accessible to optimization) waiting time at the first transit stop is included for the journey using public transportation whereas the car route departs immediately. It would be fairer and more realistic to first determine the departures of relevant nearby stops and calculate from that the needed departure at home. Multi-modal time-range (profile) queries would provide such a fairer evaluation. An extension of the recent research on multi-modal preprocessing techniques towards time-range multi-modal search is an important aspect of our work in WP3.

Route planning for park-and-ride scenarios has to account for even more details. Not only public transit stops that enable fast journeys with few transfers and low price are of interest. We will also search for stops that offer considerable savings on emissions. We will achieve this through tight cooperation with the partners of WP2. Last but not least, it is very important that such stops offer a high chance of service, i. e. that they have high service frequency and long operating hours. It might be necessary to relax on the optimality of journeys during computation in order to present more (optimal and sub-optimal) alternatives to the user to decide upon. We will look at the extension of known approaches for the computation of alternative sub-optimal journeys in road networks [5, 24] (also see Deliverable D2.1) to multi-modal scenarios.

Multi-modal route planning that can take into account time-dependency in the road network (historical data, modeling rush hours) has only been done in [112] so far but the resulting algorithm is rather slow and not very flexible with regards to mode constraints. It is an interesting challenge to find a different multi-modal approach that is both fast and time-dependent (the latter w. r. t. the road network). We have some very promising ideas for that based on recent results in route planning for road networks (see Deliverable D2.1).

As parallelization is ubiquitous in modern computer architectures we will also focus on parallel extensions of our algorithms already in the design phase. Fast algorithms are an important prerequisite for quality solutions to the multi-modal tourist trip design problem (TTDP) and parallelization is an important ingredient for performance.

5.2 New prospects in computing shortest paths under uncertainty

As regards robust multimodal route planning, we aim at developing new approaches to overcome the limitations outlined in Section 3.5 The main principle is to work with a set of noisy inputs from the past, not assuming anything else about the model. The goal is to generate a route that is very likely to be robust also with respect to the future data, and to give a confidence measure regarding quality of the solution with respect to the reality that generated the inputs.

Several challenges are specific to multimodal transportation. We already mentioned that disturbances can affect modes differently. An accurate model has to account for that. Also, we need to account for the fact that in many cities, buses and trams are prioritized against private vehicles at traffic lights and often have exclusively assigned separate lanes. In addition, the number of passengers varies over the day which not only affects the frequency of the timetable, but may have negative impact on the delay probability and propagation during peak hours.

Another difficulty when dealing with public transportation is that the objective function might not always be additive, which means that classical shortest path algorithms are not applicable. For example, minimizing the number of traversed tariff zones might be desirable to keep the ticket price low. The cost of an edge with respect to this criterion depends on whether the corresponding zone was traversed before (the price remains the same) or not (the price increases).

When planning a robust route for public transport, there are several criteria we may need to take into account to offer the user the freedom to choose among Pareto optimal solutions according to her personal preferences. The main criterion is travel time. Other criteria we may wish to optimize are the number of modal changes, the number of tariff zones, the passengers discomfort. Also, we may account for some passengers having a strong preference for or against certain means of transportation.

Both optimizing non-additive cost functions and multiple criteria are hard problems on their own. We cannot hope for efficient algorithms that also produce robust results. Novel approaches are necessary to cope with this inherent hardness of the problem.

5.3 New prospects in tourist route planning problems

5.3.1 Quality improvement upon existing approaches solutions

Evidently, extensions of the elementary OP problem (such as the TOPTW and the TDTOPTW), which strongly resemble TTDP modeling, are particularly complex; hence, even heuristic approaches that derive high quality approaches (e.g. the algorithms that deal with TOPTW [119], [139], [167]) cannot meet the real-time execution requirement of TTDP web/mobile applications. Notably, ILS [29] and the algorithm of Garcia et al. [86] (proposed for TOPTW and TDTOPTW, respectively) significantly reduce execution time. However, it appears that several promising new directions exist to further improve the quality of solutions derived by those algorithms.

For instance, the insertion phase of ILS overlooks attractive candidate nodes (i.e. POIs associated with relatively high profit) located far from currently selected nodes, as the insertion of such nodes would considerably increase route travel time. This is also the case when considering larger groups of nearby attractive candidate nodes located far from the current solution's nodes. In such scenarios, the increased travel cost of visiting the first node would be soon compensated by successive visits to other nearby interesting sites; yet, deterministic approaches like ILS will fail to incorporate such groups of nodes into derived solutions as they examine candidate nodes individually. In fact, realistic TTDP problem instances are likely to match such high-profit node distribution patterns. Namely, city tourist attraction maps typically include distinct areas (possibly far located from each other or from tourist hotels) with high density of must-see POIs.

A way around this problem would be the identification of node clusters located in close proximity with relatively high average profit, prior to executing the insertion phase. Several cluster analysis algorithms such as the k-means or the fuzzy c-means clustering [184] could serve for partitioning available nodes into separate groups (clusters). Certainly, the clustering criteria could be adjusted to incorporate several attributes (in addition to distance), such as the feasibility of successive visits to cluster nodes with respect to their time windows. Thereafter, several alternative insertion criteria could be examined to bias solutions including such node clusters. It is noted that the clustering procedure could be performed offline to save online queries execution time.

Another characteristic of TTDP overlooked by TOPTW algorithms is the fact that, typically, POIs time windows largely overlap. This fact could be utilized to effectively reduce TOPTW to TOP and thereafter apply perturbations typically used in TOP algorithms (such as 2-opt exchanges) to further improve derived solutions.

5.3.2 Modeling and solving TOPTW generalizations

The state-of-the-art relevant to the OP and VRP families of problems presented in Section 4.5 reveals that little has been done in regards with problems that closely match TTDP requirements, e.g. allowing modeling multiple user constraints and transfers through public transportation. This highlights a promising field of research which calls for modeling and solving extensions of TOPTW and TDTOPTW that take into account realistic TTDP issues or constraints like the following:

- Weather conditions: museums may be more appropriate to visit than open-air sites in rainy or relatively cold days, while the contrary may be true in sunny days; hence, route planning could take into account weather forecast information in recommending daily itineraries.
- Accessibility features of sites should be taken into account when recommending visits to individuals with motor disabilities.
- Tourists are commonly under inflexible budget restrictions when considering accommodation, meals, means of transport or visits to POIs with entrance fees. Hence, next to the time budget, money budget further constrains the selection of POI visits.
- Recommended tourist routes that exclusively comprise POI visits and last longer than a few hours are unlikely to be followed closely. Tourists typically enjoy relaxing and breaks as much as they enjoy visits to POIs. A realistic route should therefore provide for breaks either for resting (e.g. at a nearby park) or for a coffee and meal. Coffee and meal breaks are typically specific in number, while respective recommendations may be subject to strict time window (e.g. meal should be scheduled around noon) and budget constraints.
- The assumption of POIs having periodic time windows is invalid. POIs typically operate at specific days weekly, possibly with varying opening and closing hours. Hence, TTDP modelling should take into account multiple time windows.
- Max-n Type [159] constrains the selection of POIs by allowing stating a maximum number of certain types of POIs, per day or for the whole trip. e.g. maximum two museum visits on the first day. Likewise, mandatory visits (i.e. tours including at least one visit to a POI of certain type, such as a visit to a church) could also be asked for.

- Tourists commonly prefer strolling downtown rather than visiting museums. In such cases, tourists may prefer to walk along routes featuring buildings and squares with historical value or routes with scenic beauty. Such routes are likely to be preferred also when moving among POIs, e.g. a detour through a car-free street along a medieval castle walls would be more appreciated than following a shortest path though streets with car traffic.

5.3.3 Modeling and solving relevant problems

Modeling and solving of problems relevant to TTDP represents another promising research direction for eCOMPASS consortium. For instance, hotel selection is often a cumbersome task for tourists unfamiliar with hotels and POIs locations within the urban space, or with the structure of the public transportation network. Several criteria could apply in hotel recommendation, including cost, amenities or cost-for-profit (i.e. select an affordable hotel suitably located so as to maximize the overall profit collected from POI visits throughout the whole trip). Another example is the problem of determining the minimum number of days that one needs to visit all selected POIs without violating the constraint of the available time per day. This problem may be formulated using the distance constrained vehicle problem (DVRP) described in section 4.5.6. Other interesting variants of TTDP may be formulated using the mixed orienteering problem discussed also in 4.5.6.

5.3.4 Fast tourist routes updates

Existing TTDP solutions deal with tourist queries for multiple days' route planning, considering routes with the same starting/ending location. However, there is no provision for user deviations from the originally planned routes, although such deviations are highly probable to occur.

Dynamic rescheduling functionality should detect route invalidation (infeasibility) and present a new route schedule in real time. This should exclude POIs already visited and recommend a tour for the remainder of the current day (starting from the user's current position) as well as the next days of stay at the destination.

5.3.5 Parallel computation

One of the most important objectives in the design of algorithmic methods for the TTDP is the real time response to user queries. Parallel computing is a promising approach for attaining this important objective. Considering all the solution methods for TTDP, heuristics and metaheuristics are most amenable to parallel computation since the huge solution space arising in this kind of problems enables a lot of variation in parallelizing solution searching. Specifically, according to [62] one could parallelize the local search for good neighboring solutions or partition the solution space in number of subspaces and run a heuristic in each of these subspaces in parallel. Alternatively, a number of search threads could be created working on the same solution space, starting from different or the same initial solution and applying same or different heuristics. These threads could work independently or could cooperate periodically exchanging information about their progress and the good solutions they have found so far. An interesting aspect of these approaches is that they may as well provide new heuristic solutions with improved solution quality since they can search the solution space and combine solutions in such a way that it is very costly to simulate with a sequential implementation. Although, parallel heuristics has been proposed in the literature for the VRP and TSP [54], [60], [61], [145], [163] parallel solutions for TTDP are missing and the design of new parallel heuristics for TTDP may solve the problem of the fast derivation of the tourist itineraries.

References

[1] City trip planner, http://www.citytripplanner.com/, Last accessed: March 2012.

[2] mtrip travel guides, http://www.mtrip.com/, Last accessed: March 2012.

[3] R. A. Abbaspour and F. Samadzadegan. Time-dependent personal tour planning and scheduling in metropolises. *Expert Syst. Appl.*, 38:12439–12452, September 2011.

[4] R.A. Abbaspour and F. Samadzadegan. An evolutionary solution for multimodal shortest path problem in metropolises. *Computer Science and Information Systems*, 7(4):789–811, 2010.

[5] I. Abraham, D. Delling, A. V. Goldberg, and R. F. Werneck. Alternative Routes in Road Networks. In Festa [78], pages 23–34.

[6] I. Abraham, D. Delling, A. V. Goldberg, and R. F. Werneck. A Hub-Based Labeling Algorithm for Shortest Paths on Road Networks. In Panos M. Pardalos and Steffen Rebennack, editors, *Proceedings of the 10th International Symposium on Experimental Algorithms (SEA'11)*, volume 6630 of *Lecture Notes in Computer Science*, pages 230–241. Springer, 2011.

[7] I. Abraham, A. Fiat, A. V. Goldberg, and R. F. Werneck. Highway Dimension, Shortest Paths, and Provably Efficient Algorithms. In Moses Charikar, editor, *Proceedings of the 21st Annual ACM–SIAM Symposium on Discrete Algorithms (SODA'10)*, pages 782–793. SIAM, 2010.

[8] S. Ahmed and S.S. Kanhere. Cluster-based forwarding in delay tolerant public transport networks. In *Proceedings of the 32nd IEEE Conference on Local Computer Networks.*, pages 625–634. IEEE, 2007.

[9] D. Aksen and N. Aras. Customer selection and profit maximization in vehicle routing problems. In *Operations Research Proceedings 2005*, Operations Research Proceedings. Springer Berlin Heidelberg.

[10] *Proceedings of the 14th Meeting on Algorithm Engineering and Experiments (ALENEX'12)*. SIAM, 2012.

[11] D. Ambrosino and A. Sciomachen. A shortest path algorithm in multimodal networks: a case study with time varying costs. In *Proceedings of the International Network Optimization Conference*, 2009.

[12] K. N. Androutsopoulos and K. G. Zografos. Solving the multi-criteria time-dependent routing and scheduling problem in a multi-modal fixed schedule network. *European Journal of Operational Research*, 192:18–28, 2009.

[13] J. Aráoz, E. Fernández, and O. Meza. Solving the prize-collecting rural postman problem. *European Journal of Operational Research*, 196(3):886 – 896, 2009.

[14] J. Aráoz, E. Fernández, and C. Zoltan. Privatized rural postman problems. *Computers & Operations Research*, 33(12):3432–3449, 2006.

[15] C. Archetti, D. Feillet, A. Hertz, and M. G. Speranza. Optimal solutions for routing problems with profits.

[16] C. Archetti, D. Feillet, A. Hertz, and M. G. Speranza. The capacitated team orienteering and profitable tour problems. *J. Oper. Res. Soc.*, 60:831–842, 2009.

[17] C. Archetti, D. Feillet, A. Hertz, and M. G. Speranza. The undirected capacitated arc routing problem with profits. *Computers & Operations Research*, 37(11):1860 – 1869, 2010.

[18] C. Archetti, A. Hertz, and M. Speranza. Metaheuristics for the team orienteering problem. *Journal of Heuristics*, 13:49–76, 2007.

[19] E. M. Arkin, R. Hassin, and A. Levin. Approximations for minimum and min-max vehicle routing problems. *Journal of Algorithms*, 59(1):1 – 18, 2006.

[20] E. M. Arkin, J. S. B. Mitchell, and G. Narasimhan. Resource-constrained geometric network optimization. In *Proceedings of the fourteenth annual symposium on Computational geometry*, SCG '98, pages 307–316, New York, NY, USA, 1998. ACM.

[21] S. Arora. Polynomial time approximation schemes for euclidean traveling salesman and other geometric problems. *J. ACM*, 45(5):753–782, sep 1998.

[22] S. Arora and G. Karakostas. A $2 + \epsilon$ approximation algorithm for the k-mst problem. *Mathematical Programming*, 107:491–504.

[23] B. Awerbuch, Y. Azar, A. Blum, and S. Vempala. New approximation guarantees for minimum-weight k-trees and prize-collecting salesmen. *SIAM J. Comput.*, 28(1):254–262, 1998.

[24] R. Bader, J. Dees, R. Geisberger, and P. Sanders. Alternative Route Graphs in Road Networks. In Alberto Marchetti-Spaccamela and Michael Segal, editors, *Proceedings of the 1st International ICST Conference on Theory and Practice of Algorithms in (Computer) Systems (TAPAS'11)*, volume 6595 of *Lecture Notes in Computer Science*, pages 21–32. Springer, 2011.

[25] E. Balas. The prize collecting traveling salesman problem. *Networks*, 19(6):621–636, 1989.

[26] N. Bansal, A. Blum, S. Chawla, and A. Meyerson. Approximation algorithms for deadline-tsp and vehicle routing with time-windows. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, STOC '04, pages 166–174, New York, NY, USA, 2004. ACM.

[27] C. Barrett, R. Jacob, and M. V. Marathe. Formal-Language-Constrained Path Problems. *SIAM Journal on Computing*, 30(3):809–837, 2000.

[28] H. Bast. Car or Public Transport – Two Worlds. In Susanne Albers, Helmut Alt, and Stefan Näher, editors, *Efficient Algorithms*, volume 5760 of *Lecture Notes in Computer Science*, pages 355–367. Springer, 2009.

[29] H. Bast, E. Carlsson, A. Eigenwillig, R. Geisberger, C. Harrelson, V. Raychev, and F. Viger. Fast Routing in Very Large Public Transportation Networks using Transfer Patterns. In *Proceedings of the 18th Annual European Symposium on Algorithms (ESA'10)*, volume 6346 of *Lecture Notes in Computer Science*, pages 290–301. Springer, 2010.

[30] R. Bauer, D. Delling, and D. Wagner. Experimental Study on Speed-Up Techniques for Timetable Information Systems. *Networks*, 57(1):38–52, January 2011.

[31] A. Berger, A. Gebhardt, M. Müller-Hannemann, and M. Ostrowski. Stochastic delay prediction in large train networks. In *Proceedings of the 11th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems*, page 100, 2011.

[32] A. Berger, M. Grimmer, and M. Müller–Hannemann. Fully Dynamic Speed-Up Techniques for Multi-criteria Shortest Path Searches in Time-Dependent Networks. In Festa [78], pages 35–46.

[33] J.-F. Bérubé, M. Gendreau, and J.-Y. Potvin. An exact epsilon-constraint method for bi-objective combinatorial optimization problems: Application to the traveling salesman problem with profits. *European Journal of Operational Research*, 194(1):39–50, 2009.

[34] M. Bielli, A. Boulmakoul, and H. Mouncif. Object modeling and path computation for multimodal travel systems. *European Journal of Operational Research*, 175(3):1705 – 1730, 2006.

[35] D. Bienstock, M. X. Goemans, D. Simchi-Levi, and D. Williamson. A note on the prize collecting traveling salesman problem. *Mathematical Programming*, 59:413–420.

[36] A. Blum, S. Chawla, D. R. Karger, T. Lane, A. Meyerson, and M. Minkoff. Approximation algorithms for orienteering and discounted-reward tsp. In *Foundations of Computer Science, 2003. Proceedings. 44th Annual IEEE Symposium on*, pages 46 –55, oct. 2003.

[37] A. Blum, S. Chawla, D. R. Karger, Lane T., A. Meyerson, and M. Minkoff. Approximation algorithms for orienteering and discounted-reward tsp. *SIAM J. Comput.*, 37(2):653–670, 2007.

[38] H. Bouly, D.-C. Dang, and A. Moukrim. A memetic algorithm for the team orienteering problem. *4OR: A Quarterly Journal of Operations Research*, 8:49–70.

[39] S. Boussier, D. Feillet, and M. Gendreau. An exact algorithm for team orienteering problems. *4OR: A Quarterly Journal of Operations Research*, 5:211–230.

[40] G. Brodal and R. Jacob. Time-dependent Networks as Models to Achieve Fast Exact Time-table Queries. In *Proceedings of the 3rd Workshop on Algorithmic Methods and Models for Optimization of Railways (ATMOS'03)*, volume 92 of *Electronic Notes in Theoretical Computer Science*, pages 3–15, 2004.

[41] C. Büsing. The exact subgraph recoverable robust shortest path problem. *Robust and Online Large-Scale Optimization*, pages 231–248, 2009.

[42] S. E. Butt and T. M. Cavalier. A heuristic for the multiple tour maximum collection problem. *Computers & Operations Research*, 21(1):101 – 111, 1994.

[43] S. E. Butt and D. M. Ryan. An optimal solution procedure for the multiple tour maximum collection problem using column generation. *Computers and Operations Research*, 26(4):427 – 441, 1999.

[44] A. Campbell, M. Gendreau, and B. Thomas. The orienteering problem with stochastic travel and service times. *Annals of Operations Research*, 186:61–81.

[45] M. Carey and A. Kwieciński. Stochastic approximation to the effects of headways on knock-on delays of trains. *Transportation Research Part B: Methodological*, 28(4):251–267, 1994.

[46] M. Carey and A. Kwieciński. Properties of expected costs and performance measures in stochastic models of scheduled transport. *European Journal of Operational Research*, 83(1):182–199, 1995.

[47] I-M. Chao, B. L. Golden, and E. A. Wasil. A fast and effective heuristic for the orienteering problem. *European Journal of Operational Research*, 88(3):475 – 489, 1996.

[48] I-M. Chao, B. L. Golden, and E. A. Wasil. The team orienteering problem. *European Journal of Operational Research*, 88(3):464 – 474, 1996.

[49] K. Chaudhuri, B. Godfrey, S. Rao, and K. Talwar. Paths, trees, and minimum latency tours. In *Foundations of Computer Science, 2003. Proceedings. 44th Annual IEEE Symposium on*, pages 36 – 45, oct. 2003.

[50] C. Chekuri, N. Korula, and M. Pál. Improved algorithms for orienteering and related problems. In *Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, SODA '08, pages 661–670, Philadelphia, PA, USA, 2008. Society for Industrial and Applied Mathematics.

[51] C. Chekuri and A. Kumar. Maximum coverage problem with group budget constraints and applications. In *PROC. OF APPROX, SPRINGER LNCS, 722̆01383*, pages 72–83, 2004.

[52] C. Chekuri and M. Pal. A recursive greedy algorithm for walks in directed graphs. In *Foundations of Computer Science, 2005. FOCS 2005. 46th Annual IEEE Symposium on*, pages 245 – 253, oct. 2005.

[53] K. Chen and S. Har-Peled. The orienteering problem in the plane revisited. In *Proceedings of the twenty-second annual symposium on Computational geometry*, SCG '06, pages 247–254, New York, NY, USA, 2006. ACM.

[54] L. Chen, H.-Y. Sun, and S. Wang. A parallel ant colony algorithm on massively parallel processors and its convergence analysis for the travelling salesman problem. *Information Sciences*, 2012. in press.

[55] K. Cheverst, N. Davies, K. Mitchell, A. Friday, and C. Efstratiou. Developing a context-aware electronic tourist guide: some issues and experiences. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '00, pages 17–24, New York, NY, USA, 2000. ACM.

[56] K. Cheverst, K. Mitchell, and N. Davies. The role of adaptive hypermedia in a context-aware tourist guide. *Commun. ACM*, 45(5):47–51, May 2002.

[57] M. De Choudhury, M. Feldman, S. Amer-Yahia, N. Golbandi, R. Lempel, and C. Yu. Automatic construction of travel itineraries using social breadcrumbs. In *Proceedings of the 21st ACM conference on Hypertext and hypermedia*, HT '10, pages 35–44, New York, NY, USA, 2010. ACM.

[58] N. Christofides, A. Mingozzi, and P. Toth. State-space relaxation procedures for the computation of bounds to routing problems. *Networks*, 11(2):145–164, 1981.

[59] S. Cicerone, G. Di Stefano, M. Schachtebeck, and A. Schöbel. Dynamic algorithms for recoverable robustness problems. In *Proceedings of the 8th Workshop in Algorithmic Approaches for Transportation Modeling, Optimization, and Systems.*, 2008.

[60] J.-F. Cordeau and M. Maischberger. A parallel iterated tabu search heuristic for vehicle routing problems. *Computers & Operations Research*, 39:2033–2050, 2012.

[61] T. G. Crainic. Parallel solution methods for vehicle routing problems. In B. Golden, editor, *The Vehicle Routing Problem*. Springer Science+Business Media.

[62] T. G. Crainic and M. Toulouse. Parallel meta-heuristics. In M. Gendreau and J.-Y. Potvin, editors, *Handbook of Metaheuristics*, International Series in Operations Research & Management Science. Springer Science+Business Media.

[63] M. Dell'Amico, F. Maffioli, and P. Värbrand. On prize-collecting tours and the asymmetric travelling salesman problem. *International Transactions in Operational Research*, 2(3):297–308, 1995.

[64] D. Delling, B. Katz, and T. Pajor. Parallel Computation of Best Connections in Public Transportation Networks. In *24th International Parallel and Distributed Processing Symposium (IPDPS'10)*, pages 1–12. IEEE Computer Society, 2010.

[65] D. Delling, T. Pajor, and D. Wagner. Engineering Time-Expanded Graphs for Faster Timetable Information. In *Proceedings of the 8th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems (ATMOS'08)*, OpenAccess Series in Informatics (OASIcs), September 2008.

[66] D. Delling, T. Pajor, and D. Wagner. Accelerating Multi-Modal Route Planning by Access-Nodes. In A. Fiat and P. Sanders, editors, *Proceedings of the 17th Annual European Symposium on Algorithms (ESA'09)*, volume 5757 of *Lecture Notes in Computer Science*, pages 587–598. Springer, September 2009.

[67] D. Delling, T. Pajor, D. Wagner, and C. Zaroliagis. Efficient Route Planning in Flight Networks. In *Proceedings of the 9th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems (ATMOS'09)*, OpenAccess Series in Informatics (OASIcs), 2009.

[68] D. Delling, T. Pajor, and R. F. Werneck. Round-Based Public Transit Routing. In ALENEX'12 [10].

[69] J. Dibbelt, T. Pajor, and D. Wagner. User-Constrained Multi-Modal Route Planning. In ALENEX'12 [10].

[70] E. W. Dijkstra. A Note on Two Problems in Connexion with Graphs. *Numerische Mathematik*, 1:269–271, 1959.

[71] Y. Disser, M. Müller–Hannemann, and M. Schnee. Multi-Criteria Shortest Paths in Time-Dependent Train Networks. In McGeoch [135], pages 347–361.

[72] M. Doering, T. Pögel, and L. Wolf. DTN routing in urban public transport systems. In *Proceedings of the 5th ACM Workshop on Challenged Networks*, pages 55–62. ACM, 2010.

[73] K. Doerner, W. Gutjahr, R. Hartl, C. Strauss, and C. Stummer. Pareto ant colony optimization: A metaheuristic approach to multiobjective portfolio selection. *Annals of Operations Research*, 131:79–99.

[74] M. Dunlop, P. Ptasinski, A. Morrison, S. McCallum, C. Risbey, and F. Stewart. Design and development of taeneb city guide - from paper maps and guidebooks to electronic guides. In *Proceedings of the International Conference Information and Communication Technologies in Tourism (ENTER 2004)*, pages 58–64, 2004.

[75] D. Feillet, P. Dejax, and M. Gendreau. The profitable arc tour problem: Solution with a branch-and-price algorithm. *Transportation Science*, 39(4):539–552, 2005.

[76] D. Feillet, P. Dejax, and M. Gendreau. Traveling salesman problems with profits. *Transportation Science*, 39(2):188–205, 2005.

[77] T. A. Feo and M. G. C. Resende. A probabilistic heuristic for a computationally difficult set covering problem. 8:67–71, 1989.

[78] P. Festa, editor. *Proceedings of the 9th International Symposium on Experimental Algorithms (SEA'10)*, volume 6049 of *Lecture Notes in Computer Science*. Springer, May 2010.

[79] M. Fischetti, J. J. S. González, and P. Toth. Solving the orienteering problem through branch-and-cut. *INFORMS Journal on Computing*, 10(2):133–148, 1998.

[80] M. Fischetti and M. Monaci. Light robustness. *Robust and Online Large-Scale Optimization*, pages 61–84, 2009.

[81] M. Fischetti, D. Salvagnin, and A. Zanette. Fast Approaches to Improve the Robustness of a Railway Timetable. *Transportation Science*, 43(3):321–335, 2009.

[82] F. V. Fomin and A. Lingas. Approximation algorithms for time-dependent orienteering. *Information Processing Letters*, 83(2):57 – 62, 2002.

[83] G. Frederickson and B. Wittman. Approximation algorithms for the traveling repairman and speeding deliveryman problems. *Algorithmica*, 62:1198–1221.

[84] A. Garcia, O. Arbelaitz, P. Vansteenwegen, W. Souffriau, and M. Linaza. Hybrid approach for the public transportation time dependent orienteering problem with time windows. In *Hybrid Artificial Intelligence Systems*, volume 6077 of *Lecture Notes in Computer Science*, pages 151–158. Springer Berlin / Heidelberg, 2010.

[85] A. Garcia, M. Linaza, O. Arbelaitz, and P. Vansteenwegen. Intelligent routing system for a personalised electronic tourist guide. In Wolfram Hpken, Ulrike Gretzel, and Rob Law, editors, *Information and Communication Technologies in Tourism 2009*, pages 185–197. Springer Vienna.

[86] A. Garcia, P. Vansteenwegen, O. Arbelaitz, W. Souffriau, and M. Linaza. Integrating public transportation in personalised electronic tourist guides. *Computers & Operations Research*, (0):–, 2011.

[87] D. Gavalas, M. Kenteris, C. Konstantopoulos, and G. Pantziou. A web application for recommending personalized mobile tourist routes. *IET Software*, 2012. in press.

[88] R. Geisberger, P. Sanders, D. Schultes, and D. Delling. Contraction Hierarchies: Faster and Simpler Hierarchical Routing in Road Networks. In McGeoch [135], pages 319–333.

[89] M. Gendreau, G. Laporte, and F. Semet. A branch-and-cut algorithm for the undirected selective traveling salesman problem. *Networks*, 32(4):263–273, 1998.

[90] M. Gendreau, G. Laporte, and F. Semet. A tabu search heuristic for the undirected selective travelling salesman problem. *European Journal of Operational Research*, 106(2-3):539 – 545, 1998.

[91] M. X. Goemans and D. P. Williamson. A general approximation technique for constrained forest problems. *SIAM J. Comput.*, 24(2):296–317, 1995.

[92] M. Goerigk, M. Knoth, M. Müller-Hannemann, M. Schmidt, and A. Schöbel. The price of robustness in timetable information. In *Proceedings of the 11th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems*, page 76, 2011.

[93] A. V. Goldberg and C. Harrelson. Computing the Shortest Path: A* Search Meets Graph Theory. In *Proceedings of the 16th Annual ACM–SIAM Symposium on Discrete Algorithms (SODA'05)*, pages 156–165. SIAM, 2005.

[94] B. Golden, Q. Wang, and L. Liu. A multifaceted heuristic for the orienteering problem. *Naval Research Logistics*, 35.

[95] B. L. Golden, L. Levy, and R. Vohra. The orienteering problem. *Naval Research Logistics (NRL)*, 34(3):307–318, 1987.

[96] R.M.P. Goverde. A delay propagation algorithm for large-scale railway traffic networks. *Transportation Research Part C: Emerging Technologies*, 18(3):269–287, 2010.

[97] A. Gupta, R. Krishnaswamy, V. Nagarajan, and R. Ravi. Approximation algorithms for stochastic orienteering. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '12, pages 1522–1538. SIAM, 2012.

[98] Z. Habbas P. Bouvry H. Ayed, D. Khadraoui and J.F. Merche. Transfer graph approach for multimodal transport problems. In Hoai An Le Thi, Pascal Bouvry, and Tao Pham Dinh, editors, *Modelling, Computation and Optimization in Information Systems and Management Sciences*, volume 14 of *Communications in Computer and Information Science*, pages 538–547. Springer Berlin Heidelberg, 2008.

[99] R.W. Hall. The Fastest Path through a Network with Random Time-Dependent Travel Times. *Transportation Science*, 20(3):182–188, 1986.

[100] P. Hansen and N. Mladenovic. An introduction to variable neighborhood search. In *Meta-heuristics, Advances and Trends in Local Search Paradigms for Optimization.*

[101] A. Higgins and E. Kozan. Modeling train delays in urban networks. *Transportation Science*, 32(4):346–357, 1998.

[102] T. Huisman and R.J. Boucherie. Running times on railway sections with heterogeneous train traffic. *Transportation Research Part B: Methodological*, 35(3):271–292, 2001.

[103] T. Ilhan, S. M. R. Iravani, and M. S. Daskin. The orienteering problem with stochastic profits. *IIE Transactions*, 40(4):406–421, 2008.

[104] N. Jozefowiez, F. Glover, and M. Laguna. Multi-objective meta-heuristics for the traveling salesman problem with profits. *Journal of Mathematical Modelling and Algorithms*, 7:177–195.

[105] M. G. Kantor and M. B. Rosenwein. The orienteering problem with time windows. *The Journal of the Operational Research Society*, 43(6):pp. 629–635, 1992.

[106] S. Kataoka and S. Morito. An algorithm for single constraint maximum collection problem. *Journal of the Operations Research Society of Japan*, 31(4):515–530, 1988.

[107] D.E. Kaufman and R.L. Smith. Fastest Paths in Time-Dependent Networks for Intelligent Vehicle-Highway Systems Application. *Journal of Intelligent Transportation Systems*, 1(1):1–11, 1993.

[108] L. Ke, C. Archetti, and Z. Feng. Ants can solve the team orienteering problem. *Comput. Ind. Eng.*, 54(3):648–665, April 2008.

[109] C. P. Keller and M. F. Goodchild. The multiobjective vending problem: a generalization of the travelling salesman problem. *Environment and Planning B: Planning and Design*, 15:447–460, 1988.

[110] M. Kenteris, D. Gavalas, and D. Economou. Electronic mobile guides: a survey. *Personal and Ubiquitous Computing*, 15:97–111.

[111] M. Kenteris, D. Gavalas, and D. Economou. An innovative mobile electronic tourist guide application. *Personal and Ubiquitous Computing*, 13:103–118.

[112] D. Kirchler, L. Liberti, T. Pajor, and R. W. Calvo. UniALT for Regular Language Constraint Shortest Paths on a Multi-Modal Transportation Network. In *Proceedings of the 11th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems (ATMOS'11)*, volume 20 of *OpenAccess Series in Informatics (OASIcs)*, pages 64–75, 2011.

[113] N. J. Korula. *Approximation algorithms for network design and orienteering*. PhD thesis, University of Illinois at Urbana-Champaign, 2010.

[114] R. Kramer, M. Modsching, and K. ten Hagen. A city guide agent creating and adapting individual sightseeing tours based on field trial results. *International Journal of Computational Intelligence Research (IJCIR)*, 2(2):191–206, 2006.

[115] L. Kroon, R. Dekker, and M. Vromans. Cyclic railway timetabling: a stochastic optimization approach. *Algorithmic Methods for Railway Optimization*, pages 41–66, 2007.

[116] A. Krüger and R. Malaka. Artificial intelligence goes mobile. *Applied Artificial Intelligence*, 18(6):469–476, 2004.

[117] N. Labadi, R. Mansini, J. Melechovský, and R. Wolfler Calvo. The team orienteering problem with time windows: An lp-based granular variable neighborhood search. *European Journal of Operational Research*, 220(1):15 – 27, 2012.

[118] N. Labadi, J. Melechovský, and R. Calvo. An effective hybrid evolutionary local search for orienteering and team orienteering problems with time windows. In Robert Schaefer, Carlos Cotta, Joanna Kolodziej, and Gnter Rudolph, editors, *Parallel Problem Solving from Nature PPSN XI*, volume 6239 of *Lecture Notes in Computer Science*, pages 219–228. Springer Berlin / Heidelberg.

[119] N. Labadi, J. Melechovský, and R. Wolfler Calvo. Hybridized evolutionary local search algorithm for the team orienteering problem with time windows. *Journal of Heuristics*, 17:729–753.

[120] S. Lan, J.P. Clarke, and C. Barnhart. Planning for robust airline operations: Optimizing aircraft routings and flight departure times to minimize passenger disruptions. *Transportation Science*, pages 15–28, 2006.

[121] G. Laporte, M. Desrochers, and Y. Nobert. Two exact algorithms for the distance-constrained vehicle routing problem. *Networks*, 14(1):161–172, 1984.

[122] G. Laporte and S. Martello. The selective travelling salesman problem. *Discrete Applied Mathematics*, 26(2-3):193 – 207, 1990.

[123] C.-L. Li, D. Simchi-Levi, and M. Desrochers. On the distance constrained vehicle routing problem. *Operations Research*, 40(4):pp. 790–799, 1992.

[124] J. Li. Model and algorithm for time-dependent team orienteering problem. In Song Lin and Xiong Huang, editors, *Advanced Research on Computer Education, Simulation and Modeling*, volume 175 of *Communications in Computer and Information Science*, pages 1–7. Springer Berlin Heidelberg, 2011.

[125] J. Li, Q. Wu, X. Li, and D. Zhu. Study on the time-dependent orienteering problem. In *E-Product E-Service and E-Entertainment (ICEEE), 2010 International Conference on*, pages 1–4, nov. 2010.

[126] J.Q. Li, K. Zhou, L. Zhang, and W.B. Zhang. A Multimodal Trip Planning System Incorporating the Park-and-Ride Mode and Real-time Traffic/Transit Information. In *Proceedings of the ITS World Congress*, 2010.

[127] Z. Li and X. Hu. The team orienteering problem with capacity constraint and time window. *The Tenth International Symposium on Operations Research and Its Applications (ISORA 2011)*, pages 157–163, August 2011.

[128] C. Liebchen, M. Lübbecke, R. Möhring, and S. Stiller. The concept of recoverable robustness, linear programming recovery, and railway applications. *Robust and Online Large-Scale Optimization*, pages 1–27, 2009.

[129] C. Liebchen, M. Lübbecke, and R.H. Möhring. Recoverable robustness. Technical report, 2007.

[130] C. Liebchen, M. Schachtebeck, A. Schöbel, S. Stiller, and A. Prigge. Computing delay resistant railway timetables. *Computers & Operations Research*, 37(5):857–868, 2010.

[131] S.-W. Lin and V. F. Yu. A simulated annealing heuristic for the team orienteering problem with time windows. *European Journal of Operational Research*, 217(1):94 – 107, 2012.

[132] A. Lozano and G. Storchi. Shortest viable path algorithm in multimodal networks. *Transportation Research Part A: Policy and Practice*, 35(3):225 – 241, 2001.

[133] R. Malaka and A. Zipf. Deep map - challenging it research in the framework of a tourist information system. In *Proceedings of the International Conference on Information and Communication Technologies in Tourism (ENTER 2000)*, pages 15–27, 2000.

[134] Á. Marín, J. Mesa, and F. Perea. Integrating robust railway network design and line planning under failures. *Robust and Online Large-Scale Optimization*, pages 273–292, 2009.

[135] C. C. McGeoch, editor. *Proceedings of the 7th Workshop on Experimental Algorithms (WEA'08)*, volume 5038 of *Lecture Notes in Computer Science*. Springer, June 2008.

[136] L.E. Meester and S. Muns. Stochastic delay propagation in railway networks and phase-type distributions. *Transportation Research Part B: Methodological*, 41(2):218–230, 2007.

[137] E.D. Miller, H.S. Mahmassani, and A. Ziliaskopoulos. Path search techniques for transportation networks with time-dependent, stochastic arc costs. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics: Humans, Information and Technology*, pages 1716–1721. IEEE, 1994.

[138] J. S. B. Mitchell. Guillotine subdivisions approximate polygonal subdivisions: A simple polynomial-time approximation scheme for geometric tsp, k-mst, and related problems. *SIAM J. Comput.*, 28(4):1298–1309, 1999.

[139] R. Montemanni and L. M. Gambardella. An ant colony system for team orienteering problems with time windows. *Foundations of Computing and Decision Sciences*, 34(4):287–306, 2009.

[140] M. Müller-Hannemann and M. Schnee. Efficient timetable information in the presence of delays. *Robust and Online Large-Scale Optimization*, pages 249–272, 2009.

[141] S. Muthuswamy and S. Lam. Discrete particle swarm optimization for the team orienteering problem. *Memetic Computing*, 3:287–303.

[142] V. Nagarajan and R. Ravi. The directed orienteering problem. *Algorithmica*, 60:1017–1030, August 2011.

[143] V. Nagarajan and R. Ravi. Approximation algorithms for distance constrained vehicle routing problems. *Networks*, 59(2):209–214, 2012.

[144] T. Pajor. Multi-Modal Route Planning. Master's thesis, Universität Karlsruhe (TH), Fakultät für Informatik, 2009. Online available at `http://i11www.ira.uka.de/extra/publications/p-mmrp-09.pdf`.

[145] K.-W. Pang. An adaptive parallel route construction heuristic for the vehicle routing problem with time windows constraints,. *Expert Systems with Applications*, 38:11939–11946, 2011.

[146] I. Pohl. Bi-directional Search. In B. Meltzer and D. Michie, editors, *Proceedings of the Sixth Annual Machine Intelligence Workshop*, volume 6, pages 124–140. Edinburgh University Press, 1971.

[147] E. Pyrga, F. Schulz, D. Wagner, and C. Zaroliagis. Efficient Models for Timetable Information in Public Transportation Systems. *ACM Journal of Experimental Algorithmics*, 12(2.4):1–39, 2008.

[148] L. Qu and Y. Chen. A hybrid MCDM method for route selection of multimodal transportation network. *Advances in Neural Networks*, pages 374–383, 2008.

[149] R. Ramesh and K. M. Brown. An efficient four-phase heuristic for the generalized orienteering problem. *Computers & Operations Research*, 18(2):151 – 165, 1991.

[150] R. Ramesh, Y.-S. Yoon, and M. H. Karwan. An optimal algorithm for the orienteering tour problem. *ORSA Journal on Computing*, 4(2):155–165, Spring 1992.

[151] G. Reinelt. Tsplib - a traveling salesman problem library. *ORSA Journal on Computing*, 3(4):376–384, 1991.

[152] G. Righini and M. Salani. New dynamic programming algorithms for the resource constrained elementary shortest path problem. *Networks*, 51(3):155–170, 2008.

[153] G. Righini and M. Salani. Decremental state space relaxation strategies and initialization heuristics for solving the orienteering problem with time windows with dynamic programming. *Computers & Operations Research*, 36(4):1191 – 1203, 2009.

[154] W. J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *J. Comput. Syst. Sci.*, 4(2):177–192, 1970.

[155] M. Schilde, K. Doerner, R. Hartl, and G. Kiechle. Metaheuristics for the bi-objective orienteering problem. *Swarm Intelligence*, 3:179–201.

[156] A. Schöbel and A. Kratz. A bicriteria approach for robust timetabling. *Robust and Online Large-Scale Optimization*, pages 119–144, 2009.

[157] M. Sede, X. Li, D. Li, M.-Y. Wu, M. Li, and W. Shu. Routing in Large-Scale Buses Ad Hoc Networks. In *Proceedings of the IEEE Wireless Communications and Networking Conference.*, pages 2711–2716, 2008.

[158] J. Silberholz and B. Golden. The effective application of a new approach to the generalized orienteering problem. *Journal of Heuristics*, 16:393–415.

[159] W. Souffriau and P. Vansteenwegen. Tourist trip planning functionalities: Stateoftheart and future. In F. Daniel and F. Facca, editors, *Current Trends in Web Engineering*, volume 6385 of *Lecture Notes in Computer Science*, pages 474–485. Springer Berlin / Heidelberg.

[160] W. Souffriau, P. Vansteenwegen, G. Vanden Berghe, and D. Van Oudheusden. A greedy randomised adaptive search procedure for the team orienteering problem. In *EU/MEeting 2008 on metaheuristics for logistics and vehicle routing*, October 2008.

[161] W. Souffriau, P. Vansteenwegen, G. Vanden Berghe, and D. Van Oudheusden. The multiconstraint team orienteering problem with multiple time windows. *Transportation Science*, 2011. in press.

[162] F. C. R. Spieksma. On the approximability of an interval scheduling problem. *Journal of Scheduling*, 2:215–227, 1999.

[163] A. Subramanian, L. M. A. Drummonda, C. Bentes, L. S. Ochi, and R. Farias. A parallel heuristic for the vehicle routing problem with simultaneous pickup and delivery. *Computers & Operations Research*, 37:1899–1911, 2010.

[164] H. Tang and E. Miller-Hooks. A tabu search heuristic for the team orienteering problem. *Computers & Operations Research*, 32(6):1379 – 1407, 2005.

[165] L. Tang and X. Wang. Iterated local search algorithm based on very large-scale neighborhood for prize-collecting vehicle routing problem. *The International Journal of Advanced Manufacturing Technology*, 29:1246–1258.

[166] Google City Tours. Google labs dropouts and failures: http://google.about.com/od/blogs/ss/google-labs-dropouts-and-failures_ 2.htm, Last accessed: March 2012.

[167] F. Tricoire, M. Romauch, K. F. Doerner, and R. F. Hartl. Heuristics for the multi-period orienteering problem with multiple time windows. *Computers & Operations Research*, 37(2):351 – 367, 2010.

[168] T. Tsiligirides. Heuristic methods applied to orienteering. *The Journal of the Operational Research Society*, 35(9):pp. 797–809, 1984.

[169] J. N. Tsitsiklis. Special cases of traveling salesman and repairman problems with time windows. *Networks*, 22:263–282, 1992.

[170] D. van der Meer, R.M.P. Goverde, and I.A. Hansen. Prediction of train running times and conflicts using track occupation data. In *Proceedings of the 12th World Conference on Transport Research*, 2010.

[171] P. Vansteenwegen. *Planning in Tourism and Public Transportation - Attraction Selection by Means of a Personalised Electronic Tourist Guide and Train Transfer Scheduling*. PhD thesis, Katholieke Universiteit Leuven, 2008.

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

[173] P. Vansteenwegen, W. Souffriau, G. Vanden Berghe, and D. Van Oudheusden. A guided local search metaheuristic for the team orienteering problem. *European Journal of Operational Research*, 196(1):118 – 127, 2009.

[174] P. Vansteenwegen, W. Souffriau, G. Vanden Berghe, and D. Van Oudheusden. Iterated local search for the team orienteering problem with time windows. *Comput. Oper. Res.*, 36:3281–3290, December 2009.

[175] P. Vansteenwegen, W. Souffriau, G. Vanden Berghe, and D. Van Oudheusden. The city trip planner: An expert system for tourists. *Expert Systems with Applications*, 38(6):6540 – 6546, 2011.

[176] P. Vansteenwegen and D. Van Oudheusden. The mobile tourist guide: An or opportunity. *Operational Research Insight*, 20(3):21–27, 2007.

[177] C. Voudouris and E. Tsang. Guided local search and its application to the traveling salesman problem. *European Journal of Operational Research*, 113(2):469 – 499, 1999.

[178] P.T.R. Wang, L.A. Schaefer, and L.A. Wojcik. Flight connections and their impacts on delay propagation. In *Proceedings of the 22nd Digital Avionics Systems Conference*, page 5.B.4(9). IEEE, 2003.

[179] Q. Wang, X. Sun, B. L. Golden, and J. Jia. Using artificial neural networks to solve the orienteering problem. *Annals of Operations Research*, 61:111–120.

[180] X. Wang, B. L. Golden, and E. A. Wasil. Using a genetic algorithm to solve the generalized orienteering problem. In *The Vehicle Routing Problem: Latest Advances and New Challenges*, volume 43 of *Operations Research/Computer Science Interfaces Series*, pages 263–274. Springer US, 2008.

[181] M.P. Wellman, M. Ford, and K. Larson. Path planning under time-dependent uncertainty. In *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence*, pages 532–539. Morgan Kaufmann Publishers Inc., 1995.

[182] Q. Wu, J. Hartley, and D. Al-Dabass. Time-dependent stochastic shortest path(s) algorithms for a scheduled transportation network. *International Journal of Simulation*, 6(7-8):53–60, 2005.

[183] N. Xu, G. Donohue, K.B. Laskey, and C.H. Chen. Estimation of delay propagation in the national aviation system using Bayesian networks. In *Proceedings of the 6th USA/Europe Air Traffic Management Research and Development Seminar*. Citeseer, 2005.

[184] R. Xu and D. Wunsch. Survey of clustering algorithms. *Neural Networks, IEEE Transactions on*, 16(3):645 –678, may 2005.

[185] J. Yuan. Dealing with stochastic dependence in the modeling of train delays and delay propagation. In *Proceedings of the First International Conference on Transportation Engineering*, pages 3908–3914, 2009.

[186] B. Zenker and B. Ludwig. Rose: assisting pedestrians to find preferred events and comfortable public transport connections. In *Proceedings of the 6th International Conference on Mobile Technology, Application & Systems*, Mobility '09, pages 16:1–16:5, New York, NY, USA, 2009. ACM.

[187] K. G. Zografos and K. N. Androutsopoulos. Algorithms for Itinerary Planning in Multimodal Transportation Networks. *IEEE Transactions on Intelligent Transportation Systems*, 9(1), March 2008.

[188] K. G. Zografos, K. N. Androutsopoulos, and V. Spitadakis. Design and Assessment of an Online Passenger Information System for Integrated Multimodal Trip Planning. *IEEE Transactions on Intelligent Transportation Systems*, 10(2):311–323, June 2009.