Critical examination of existing methods for communication/transportation networks, and identification of key-problems.
Start date of the project: January 2004
Duration: 48 months
Project Coordinator: Prof. Dr. math. Friedhelm Meyer auf der Heide
Heinz Nixdorf Institute, University of Paderborn, Germany
Due date of deliverable: December 2004
Actual submission date: January 2005
Dissemination level: PU – public

Work Package 3.2: Optimization methods for large-scale communication and transportation networks
Participants: Universität Paderborn (UPB), Germany
Research Academic Computer Technology Institute (CTI), Greece
Max-Planck-Gesellschaft zur Förderung der Wissenschaften E.V. (MPII), Germany
University of Cyprus (UCY), Cyprus
Charles University (CUNI), Czech Republic

Authors of deliverable: Robert Elssser (elsa@upb.de)
Sven Grothklags (sven@upb.de)
Georg Kliewer (geok1@upb.de)
Marios Mavronikolas (mavronic@cs.ucy.ac.cy)
Burkhard Monien (bm@upb.de)
Christos Zaroliagis (zaro@ceid.upatras.gr)
Overview

This report comprises the complete D3.2.1 deliverable as specified for workpackage WP3.2 in suproject SP3 of the DELIS (Dynamically Evolving Large-scale Information Systems) Integrated Project. The essential goal of the DELIS project is to understand, predict, engineer and control large evolving information systems.

Large-scale communication and transportation networks impose challenging optimization problems on the operators of such networks – among these are designing the basic network, planning the optimal usage and anticipating and/or reacting to environmental changes and disruptions. In section 1 the typical optimization problems are introduced that arise in large-scale networks. The existing solution methods for these optimization problems are described in section 2. Finally, open key-problems in the different areas of large-scale network optimization are presented in section 3.

1 Description of tasks and optimization problems

1.1 Communication networks

One of the most important tasks in communication networks is to efficiently spread the information to the nodes in the network and to distribute the work-load among the nodes almost equally. Several attempts have been made to develop efficient algorithms which are able to solve the problems mentioned above in general graphs. However, most communication networks have some special structural properties, which are not incorporated in these algorithms.

The behavior of deterministic broadcasting and routing algorithms depend on the connectivity, diameter, and expansion properties of the graph. Slight changes may modify their performance extremely. In contrast to this, simple randomized algorithms are robust against limited changes in the network. However, the behavior of such algorithms have been analyzed only for a few classes of graphs.

Load balancing algorithms have been extensively studied in the past. Tight bounds on the runtime of simple and complex nearest neighbor load balancing schemes in general graphs are known. However, these bounds do not necessarily reflect the behavior of these algorithms in the graph classes which model communication networks. Moreover, by using the special structural and spectral properties of these graphs, new algorithms should also be developed in order to improve the efficiency of known algorithms in communication networks.

1.2 Transportation networks

Network planning is necessary to define the infrastructure which is used for transport of persons, goods or information. First, the transportation demand must be evaluated. Forecasts are used to define the amount and structure of the future traffic in the network. Depending on these forecasts the network must be set up. The network design solves the task of defining the structure and capacity in the network. The transportation resources are targeted in the next step. Efficient usage of resources while fulfilling the transportation demand is the goal. When the plans and schedules are completed, they are going to be realized. Several disruptions can prevent the network operator from fulfilling his plans. The operator must control the resources and react on disruption in the network.

Traffic forecasting

In transportation networks the traffic is forecasted by market models in case of traveling persons or transportation requests for goods.

Passenger choice models are used to forecast the behavior of travelers. Possible travel alternatives have attributes which influence the decisions of passengers. For example, the duration of a flight
travel or the number of airplane changes are modeled as such attributes. Depending on the probabilistic component of the passenger decisions one obtains the logit or probit choice models. Another possibility is to model the passenger flow as a multicommodity flow in the network. Other methods for traffic forecasting are QSI-models (quality of service index), time series models, regression models, traffic simulation models, and user equilibrium models.

In [3] foundations of the discrete choice theory are given. In several application areas the traffic forecasting techniques are successfully applied [27, 31].

**Network design**

In network design, the structure and capacity of the operated network is defined. Depending on forecasted transportation demand, arcs and nodes are added to the network and the necessary capacity of transportation resources is calculated. Design of the topology is carried out by making an installation decision and paying a fixed price for a new road, flight, or wire. Several properties may be important in concrete applications, e.g. survivability of the network in case of damages or connectivity quality. The capacities of transportation edges or transshipment nodes must be determined. As a goal, the forecasted traffic should be routable in the network with low transportation costs.

In [1, 21] overviews of the field of network design can be found. Problem variations and application areas are discussed.

**Airline fleet assignment**

In airline scheduling, the resources which are to be scheduled are the planes and pilots with crews. The airline operates several fleet types which differ in their flight range, capacity and operation costs. Depending on forecasted number of passengers the airline has to decide which fleet type to put on a flight. This decision depends not only on characteristics of the given flight but also on the preceding and following flights (rotation). This problem is one of the most important optimization tasks in airline scheduling and is known as airline fleet assignment problem (FAP). Typically, real-world problem instances of an FAP are huge and have many hard and soft constraints. For these two reasons this class of problems attained a lot of attention from academical and industrial researchers.

In [16] the most used mathematical model for the FAP is presented. Since then several model variations and many optimization algorithms (exact and heuristic) were developed (e.g. [15]). For an overview see [18].

**Airline crew scheduling**

Since in most airlines the airplanes are scheduled first, scheduling for crews takes a concrete flight plan as an input. In a first step work plans for pilots and cabin crews are generated. In a second step an assignment of plans to concrete persons is calculated. A pairing (work plan) is a sequence of flights, which starts and ends on a crew base. The crew pairing problem is to find a least cost subset of pairings that partition the given set of flights from the flight plan. Crew schedules have to satisfy complex complex airline rules and regulations. Constraint programming is used to treat them efficiently. Typically, solution approaches to the airline crew rostering problem are based on column generation. For an overview of research in this area see [2].

**Railway optimization**

An important problem in public transportation systems is to model timetable information so that subsequent queries asking for optimal itineraries can be answered efficiently. The main target that underlies the modeling (and which applies not only to public transportation systems, but also to other systems as well like route planning for car traffic, database queries, web searching, etc) is
to process a vast number of on-line queries as fast as possible. A specific, query-intensive scenario arising in public railway transport concerns a central server that is directly accessible to any customer either through terminals in train stations or through a web interface, and has to answer a potentially infinite number of queries. The major question is how the modeling of timetable information and the preprocessing should be carried out such that the average response time for a query is reduced without sacrificing optimality (i.e., the quality of the answer returned). In addition, queries may not involve optimizing a single criterion (e.g., time) but several ones (e.g., number of transfers, price, etc).

Along the same setting (querying traffic information systems to find best routes or optimal itineraries as efficiently as possible), there are a few additional problem aspects. Answering a best route (or optimal itinerary) query translates in computing a minimum cost path on a suitably defined directed graph with nonnegative edge costs, rendering shortest path computation to be the underlying core algorithmic problem. Although the straightforward approach of precomputing and storing shortest paths for all pairs of nodes would allow to answer shortest-path queries optimally, the quadratic space requirements for graphs with more than $10^5$ nodes makes such an approach prohibitive for large-scale transportation networks. Consequently, two major questions in this setting are: (i) How the search space (number of nodes visited) of a shortest path algorithm can be reduced (and hence the time for answering a query) while simultaneously retaining data structures, created during a preprocessing phase, of size linear (i.e., optimal) to the size of the graph? (ii) How efficiently can shortest path computations be carried out in the case where the graph may dynamically change over time as certain “disruptions” to the transportation network occur (e.g., streets may be blocked, built, or destroyed, trains/buses may be added or canceled, etc)?

1.3 Operation control and disruption management

The problem faced during operation control and disruption management can be described as follows: At the beginning of a business period, an optimal or near-optimal operational plan is obtained by using certain optimization models and solution schemes. When such an operational plan is executed, disruptions may occur from time to time caused by internal and external uncertain factors. In the case of airline scheduling, such disruptions can be e.g. a snowstorm, that forces a complete airport to shut down for some time, a change in the predicted passenger demand on a certain flight, the break-down of an aircraft, illness of a pilot, etc. As a result, the original operational plan may not remain optimal, or even feasible. Consequently, we need to dynamically revise the original plan and obtain a new one that reflects the constraints and objectives of the evolved environment while minimizing the negative impact of the disruption.

Therefore, the key issue, why disruption management is necessary for most planning tasks, lies in the uncertainty of the future during planning time. Uncertainties can be dealt with via different approaches and at different times. On the one hand, we can try to generate an optimal initial operational plan based on the estimate of future uncertainties. The result of this is what we call a robust plan. Ideally, this already leads to a plan, that is immune against disruptions, but typically, one can only hope to get a plan that can be repaired cheaply. On the other hand, we can perform real-time re-planning: the task is to revise a given original plan in its execution period whenever needed. In many cases it is desirable to consider further possible disruptions during re-planning as well; this leads to a field that we call robust re-planning.
2 Existing solution methods and algorithms

2.1 Communication networks

Broadcasting. There are several deterministic and randomized models for the broadcasting problem in general graphs. Simple randomized algorithms, such as the push or pull model, have also been analyzed in random graphs. In the push model, after some rumor is received by a node, this starts to send the rumor, in every following round, to a randomly chosen neighbor. In the pull model, in every round each node calls some randomly chosen neighbor, and if the called node has some information, then this is sent to the calling node. In the so called agent based model, \( n \) agents are performing random walks in a network containing \( n \) nodes. At some time, a piece of information is placed on one of the nodes, which is then called informed. In the succeeding rounds, informed nodes inform the agents visiting them and informed agents carry the information to other nodes.

Frieze and Molloy [13] showed that in a random graph \( G_p \) with \( n \) vertices, an upper bound of \( O\left(\frac{\ln n}{n}\right) \) is required on the edge density in order to deterministically broadcast information in \([\log_2 n]\) steps (with high probability). This result has been improved by Chen in [9]. Randomized broadcasting has also been examined within geometric networks. In [17] it is shown that new information is spread to nodes at distance \( t \) with high probability in \( O(\ln^{1+\epsilon} t) \) steps. A similar broadcasting model has been analyzed under the name of rumor-spreading. There, one of \( n \) people knows some rumor and any 'knower', in our language an infected person, infects in each round another randomly chosen person of the population. The goal is to determine the number of rounds required for infecting all persons in the system. Pittel [25] proved a nice result, which shows that within \( \log_2(n) + \ln(n) + O(1) \) steps they are probably infected. Feige et. al. [14] extended the results to different graph classes. Karp et. al. [19] showed that the number of messages can be bounded by \( O(n \ln \ln n) \).

Load balancing. There are also several methods known to solve the load balancing problem. One of them is the so called general diffusion, which belongs to the class of nearest neighbor load balancing algorithms and has been introduced by Cybenko in [10]. This algorithm is called first order scheme (FOS). Improvements of FOS are the so called second order schemes (SOS), which converge faster than FOS by almost a quadratic factor [23]. In [11] the quality of the computed flow has also been analyzed and it was shown that known diffusion schemes compute an \( l_2 \)-optimal flow.

However, these schemes work well if it is assumed that the load entities are arbitrary divisible [23]. To handle the case of undivisible unit size load items, a new randomized strategy was introduced in [12]. Using this strategy, the error can be reduced to an asymptotic optimal value while only slightly increasing the run-time of the diffusion algorithm.

Routing. A significant class of packet routing protocols are bufferless protocols. In such protocols, there are no storage buffers at the nodes for the intermediate storage of the packets, and packets must depart from a node immediately ([6]). This creates a potential problem when there are conflicts among the departing packets (due to the limited capacity of the communication channels). In such cases, some kind of deflection is in order – some packet must deviate from its (pre-specified) path under the strong hope that it will very soon rejoin it. Such protocols are called hot-potato routing protocols and they have been recently under very intense investigation ([8]). An even more stringent requirement is to disallow deflections – in such case, packets must proceed to their destinations directly once scheduled. This is the Direct Routing problem; it has been studied recently from the point of view of Computational Complexity, and some natural connections with classical coloring problems have been identified ([7]). The identified limitations have raised hopes for approximate solutions to the problem, or to solutions that compromise the requirement of directness. Some such solutions are important for being universal – they can apply to any network and they achieve a
(common) upper bound on their performance which is independent of the particular network at hand.

2.2 Large scale transportation networks

2.2.1 Algorithms for airline planning problems

In transportation traffic forecasting, several passenger choice models are used. The methods there range from simulation to algorithms for linear multicommodity flows. In resource scheduling like airline fleet assignment large-scale mixed-integer linear problems must be solved. Sophisticated pre-processing techniques together with branch-and-cut algorithms enable efficient solving of real-life problem instances.

Three most common techniques for solving large-scale linear mixed integer models are: branch-and-cut, branch-and-price, and Lagrangian decomposition. They found several applications in large-scale optimization of communication and transportation problems in the last years.

Branch-and-cut is a branch-and-bound algorithm in which cutting planes are generated throughout the branch-and-bound-tree. Cutting planes are valid inequalities for a mixed-integer linear problem which separate the linear program solution from the optimal integer solution of the problem. The use of cutting planes to improve formulations and obtain tighter bounds is the area in which probably the most progress has been made in the last ten years [33].

Column generation is a useful solution technique in case of large-scale problem with a huge number of variables. The columns are described implicitly e.g. as possible working plans for airline crews. In such a case a restricted master problem with only a subset of possible columns is solved. In a subproblem columns with negative reduced costs are searched in order to be considered in the master problem. Often the subproblem consists of a constrained shortest path problem, but also generation of columns can be carried out by another technique, like constraint programming. Branch-and-price algorithm is a branch-and-bound algorithm, where the LP relaxations in every node are solved by column generation.

In Lagrangian decomposition the problem constraints are partitioned in easy and difficult constraints. If the difficult constraints could be removed, the problem would be easy solvable. In this approach, every difficult constraint gets a linear penalty and it is moved to the objective function. The resulting problem is called Lagrangian relaxation and it is a function of the penalties. For any fixed penalty values the problem is computationally easy. The goal is to find the best vector of penalty values, which is a non-linear optimization problem. Typically, subgradient optimization or bundle methods are used for this task. The overall algorithm is relatively easy to implement and can handle complex side constraints which often arise in real-life applications.

2.2.2 Algorithms for railway optimization problems

Modeling Timetable Information. Two main approaches have been proposed for modeling timetable information: the time-expanded and the time-dependent approach The common characteristic of both approaches is that a query is answered by applying some shortest path algorithm to a suitably constructed digraph. The time-expanded approach [30] constructs the time-expanded digraph in which every node corresponds to a specific time event (departure or arrival) at a station and edges between nodes represent either elementary connections between the two events (i.e., served by a train that does not stop in-between), or waiting within a station. Depending on the problem that we want to solve (see below), the construction assigns specific fixed costs to the edges. This naturally results in the construction of a very large (but usually sparse) graph. The time-dependent approach [5] constructs the time-dependent digraph in which every node represents a station and two nodes are connected by an edge if the corresponding stations are connected by an elementary connection. The
costs on the edges are assigned “on-the-fly”, i.e., the cost of an edge depends on the time in which the particular edge will be used by the shortest path algorithm to answer the query.

The two most frequently encountered timetable problems are the earliest arrival and the minimum number of transfers problems. In the earliest arrival problem, the goal is to find a train connection from a departure station $A$ to an arrival station $B$ that departs from $A$ later than a given departure time and arrives at $B$ as early as possible. There are two variants of the problem depending on whether train transfers within a station are assumed to take negligible time (simplified version) or not. In the minimum number of transfers problem, the goal is to find a connection that minimizes the number of train transfers when considering an itinerary from $A$ to $B$. Combinations of the above problems are considered as bicriteria optimization problems.

The time-expanded and the time-dependent approaches have been recently considered for the simplified earliest arrival problem, but little is known about their relative performance. So far, there are only theoretical arguments in favor of the time-dependent approach. Moreover, nothing is known about the relative behavior of the two approaches in more realistic settings.

In [26], the first experimental comparison of the time-expanded and the time-dependent approaches with respect to their performance in the specific, query-intensive scenario mentioned earlier. Besides the comparison of the two approaches regarding the simplified earliest arrival problem (where it is confirmed that the time-dependent approach is superior to the time-expanded one), new extensions of both approaches are presented in order to cope with more realistic settings. In particular, the proposed extensions can handle cases not tackled by most previous studies for the sake of simplification. These new cases are: (a) the waiving of the assumption that transfer of trains within a station takes negligible time; (b) the consideration of the minimum number of transfers problem; (c) the involvement of traffic days; and (d) the consideration of bicriteria optimization problems combining the earliest arrival and the minimum number of transfers problems.

The idea of extending the time-expanded and the time-dependent models to capture realistic scenarios is to enlarge the underlying graphs. In the time-expanded case, extensions that model more realistic requirements (like modeling train transfers) could be integrated in a more-or-less straightforward way and the central characteristic of the approach is that a solution to a given optimization problem could be provided by solving a shortest path problem in a static graph, even for finding all Pareto-optimal solutions in the considered bicriteria optimization problems. In the time-dependent case, the central characteristic of having one node per station had to be violated when more realistic requirements (like the integration of minimum transfer times at stations) were considered, and more sophisticated techniques in the bicriteria optimization problems had to be used for their effective solution. Nevertheless, all the problems under consideration could be efficiently modeled in an extension of the time-dependent model that, although enlarged considerably the initial (simplified) model, it still was more compact than its counterpart in the extended time-expanded model, and thus resulting in better performance in practice. However, although there is a big difference in favor of the time-dependent approach when comparison of the original version of the models is considered (speedup between 12 and 50), this large difference is reduced dramatically when considering the extensions of the models for the solution of realistic versions of optimization problems (speed-up is now reduced in the range of 1.5 to 3.3). The time-expanded approach benefits in this case from the straightforward modeling that allows more direct extensions and effective solutions.

**Efficient Query Answering in Timetable Information Systems.** One of the features of travel planning is the fact that the traffic networks do not change for a certain period of time while there are many queries for shortest paths. Consequently, a heavy preprocessing of the network (that does not blow up the space requirements) is justified in order to speed up the query time. The most commonly used approach for answering shortest path queries concerns variants of Dijkstra’s algorithm, targeting at reducing its search-space (number of nodes visited by the algorithm).
In [32], the possibility of reducing the search space of Dijkstra’s algorithm by using precomputed information that can be stored in $O(n + m)$ space is investigated, where $n$ (resp. $m$) is the number of nodes (resp. edges) of the graph. Note that $O(n + m)$ is an optimal space requirement. The main contribution in [32] is that the search space of Dijkstra’s algorithm can be significantly reduced – thus answering on-line shortest path queries fast – by extracting geometric information from a given layout of the graph. For traffic information systems such a layout is provided by the geographic locations of the nodes. In particular, the study shows that storing partial results based on geometric information reduces the number of nodes visited by Dijkstra’s algorithm to only 5% to 10%.

To achieve this space reduction, a very fundamental observation on shortest paths is used: an edge that is not the first edge on a shortest path to the target can be safely ignored in any shortest path computation to this target. In particular, the main idea in [32] is as follows. Assume that during preprocessing a set of nodes $S(e)$ is computed, for each edge $e$, containing all nodes that can be reached by a shortest path starting with $e$. Subsequently, when Dijkstra’s algorithm is executed, those edges $e$ for which the target is not in $S(e)$ are ignored. As storing all sets $S(e)$ would require $O(nm)$ space, this prohibitive space requirement is relaxed by storing instead a geometric object, called container, for each edge that contains at least the nodes in $S(e)$. The shortest path queries are then answered by Dijkstra’s algorithm restricted to those edges for which the target node is inside their associated geometric container. Note that this method still leads to a correct result (optimal path), although it may increase the number of visited nodes to more than the strict minimum (i.e., the number of nodes in the shortest path). The geometric containers are generated from the given layout of the graph.

The dynamic version of the above scenario is also considered in [32]; namely, the case where the graph may dynamically change over time as certain “disruptions” to the traffic network occur (streets may be blocked, built, or destroyed, trains/buses may be added or canceled, etc). New algorithms are presented in [32] for this case that dynamically maintain geometric containers when the weight of an edge is increased or decreased (note that these cases cover also edge deletions and insertions). An accompanying experimental study shows that the new algorithms are 2 to 3 times faster than the naive approach of recomputing the geometric containers from scratch, although this is not evident from a worst-case theoretical comparison. As a final comment, it is worth mentioning that existing approaches for the dynamic all-pairs shortest paths problem are not applicable to maintain geometric containers, because of their inherent quadratic (and in some cases even cubic) space requirements.

2.3 Robust Scheduling

In this deliverable, ‘robust scheduling’ simply means, that we are generating plans with future uncertainties in mind. In literature, the term ‘robust scheduling’ is also used in a more specialized context, where the aim is to generate plans with some kind of worst-case guaranty (see below).

Rule-based robustness. Here, one tries to identify certain structural properties of a plan, that increase its robustness. During the optimization, only plans with these structural properties should be generated. This can be done implicitly by the optimization algorithm, explicitly by adding additional constraints to the problem, that must be fulfilled by a feasible plan, or by adding some kind of measure for these structural properties to the objective function.

E.g., in [28] a robust airline fleet assignment model is presented, that, besides minimizing planned operating cost and passenger spill, also favors plans with many short cycles and low hub connectivity. A cycle is a sequence of flights that begins and ends at the same airport; in case of a cancellation, typically a whole cycle must be canceled and therefore short cycles should increase the robustness of a fleet assignment. Hub connectivity is a measure how many aircrafts operate between hubs (central airports of an airline); the reason to look for fleet assignments with low hub connectivity is, that delays cannot spread out so easily over the whole network if there are only few connections between
the high-traffic airports. By simulations it is shown that solutions of this robust model perform better in operations than those of traditional fleet assignments.

**Classical robust scheduling.** When future uncertainties can be expressed by a set of possible scenarios, of which only one will come true, one can generate schedules that minimize the consequences of the worst case scenario. One can also minimize the (absolute or relative) deviation from scenario-dependent optimal schedules. No information is needed about the probability distribution of the scenarios, but, as a consequence, the average performance of such worst case optimal schedules can be very poor.

In [20] Kouvelis and Yu describe robust decision making for a number of discrete optimization problems, especially for one and two machine job scheduling. They describe a number of heuristics to generate robust schedules and present optimal branch and bound algorithms for worst deviation robust schedules. A comprehensive description on a large variety of robustness concepts can be found in [29].

**Stochastic optimization.** Similar to classical robust scheduling, future uncertainties are expressed by a set of possible scenarios, but here probability distributions governing the input data are known or can be estimated. The goal is to find a schedule that maximizes the expectation of some function of the decision and the random variables.

The most widely applied and studied stochastic optimization models are two-stage linear programs. Here the decision maker takes some action in the first stage, after which a random event occurs affecting the outcome of the first-stage decision. A recourse decision can then be made in the second stage that compensates for any bad effects that might have been experienced as a result of the random event. This model can be naturally extended to more than two stages, which leads to multi-stage linear programs.

Normally, it is assumed that the number of scenarios is finite (and small). In this case, it is possible to compute a solution to the stochastic programming problem by solving a deterministic equivalent linear program. These problems are typically very large scale problems and specialized decomposition techniques, that exploit the structure of the deterministic equivalent linear programs, were developed to actually solve these LPs. Additional problems arise when dealing with stochastic integer programs (NP-hardness of subproblems, loss of convexity, ...). An introduction to stochastic programming and details of solution approaches can be found in [4]. One of the rare stochastic programming approaches for large scale transportation problems, namely airline crew scheduling, can be found in [34].

3 Key problems

3.1 Communication networks

As previously described, real world communication networks have some specific structural and spectral properties. One key problem is to study the relationship between structural and spectral properties of such graphs. This relationship will be helpful to improve the efficiency of diffusion based deterministic and randomized algorithms on these graphs.

The behavior of broadcasting algorithms depends on the diameter, expansion and density of graphs. In order to analyze the run-time and message complexity of simple randomized broadcasting methods, it is necessary to determine the structural properties of the communication networks studied here. Then, the run-time of existing algorithms can be determined and more efficient new algorithms can be developed, which exploit the structural properties already specified.
3.2 Large scale transportation networks

3.2.1 Integration of planning tasks

The process of airline planning is organized in several planning steps. It starts about one year before operation with strategic network and schedule design. After that an airline creates plans for the fleet/aircraft. In the next planning step operational constraints must be considered like aircraft maintenances etc. Another important question is the planning and scheduling of pilots and cabin crews. This is the main issue in the field of crew scheduling and rostering.

These problems are solved sequentially, rather than simultaneously, primarily because of limitations in computer technology and solution algorithms.

As a result of economic considerations in generating the plans and schedules, all the resources are tightly coupled. The drawback to the sequential approach is that it might not yield to feasible solutions, and even if feasible solutions are achieved, these solutions might be far from optimal. In contrast to the sequential approach, cooperative solution approaches will produce more economical plans by considering the interdependencies between the different optimization problems.

3.2.2 Railway planning problems

A key problem in timetable information modeling and query answering is to consider multiple (in particular more than 2) criteria when itinerary queries are issued. Some preliminary efforts towards this goal have been made in [22, 24], where the discussion in [22] is focused on a distributed approach for timetable information problems, while the case of 3 criteria has been considered in [24], under the (unrealistic) assumption, however, that transfer times are negligible.

A second key problem in the same setting is to improve the time for updating the data structures supporting efficient answering of itinerary queries, when certain dynamic changes to the railway network (and hence to the underlying graph) occur. A particular challenge is to develop dynamic algorithms that are theoretically better than their static counterparts. Further exploitation of explicit or implicit geometric information seems to be a promising avenue of research.

A third key problem concerns line planning in a railway (and in general in any public transportation) network. A major problem is that in such a network customer demands change with the provided quality of service (QoS): customers tend to switch to other transport services when the offered QoS drops. Moreover, this loss of customers is non-uniform, since it depends on their particular type. The main question is how to maximize the total flow of customers weighted by the price they pay for the service provided.

3.3 Robust scheduling

In practice, optimization problems for transportation networks hardly ever deal with deterministic input data, but with stochastic or even uncertain input data. Nevertheless, most of the time these problems are solved as if they were deterministic for simplicity and computational efficiency. This may result in a significant loss of actual performance when it comes to operations.

A basic key problem is the evaluation of robustness for solutions of specific transportation problems, like e.g. the fleet assignment problem. Ideally, this can be done by defining suitable (problem specific) performance measures for robustness. But for many problems such measures are not known and hard to find. One solution to this problem is the use of a simulation environment to be able to analyze the behavior of different solutions during operations under realistic circumstances. On the one hand, this gives us an instrument to directly compare different solutions and solution methods. On the other hand, such a simulation environment can help to find performance measures for robustness and to verify that such measures are suitable.
The main challenge is to integrate the concept of robustness into additional real-world planning tasks for transportation problems. On the one hand, existing solution methods and models can be extended by identifying robustness properties and adding these to the constraints and/or objective function. On the other hand, the more general concepts of stochastic optimization can be applied to transportation problems. This will typically lead to multi-stage discrete stochastic optimization problems, for which we want to develop new optimization methods.

References


