Chapter 1

Introduction

The questions studied in this thesis are based on routing problems at a container terminal at the Hamburg harbor. Between the harbor's quay and a storage area, more than sixty vehicles transport containers with a planned throughput of nearly two million containers per year. These vehicles are automatically guided to drive along specified lanes on their own. In order to transport all containers to their destinations, these vehicles have to be supplied with appropriate routes along the lanes to ensure that they do not collide with each other. Herein, time plays an essential role. Ships have to be loaded and discharged in a short time in order for the containers to be quickly brought to their destinations.

Such routing problems are a common task in logistics. Typically the aim is to find paths and to send transportation vehicles along them, such that given requests are fulfilled as quickly as possible with a limited number of vehicles. Most routing problems are complex due to a large amount of transportation requests and a large variety of possible routes in the given area. Therefore, good routings can hardly be determined manually, but a good control is a substantial competitive edge. As such, more and more sophisticated tools are currently under development to support this type of planning.

The field of combinatorial optimization provides methods to model and solve routing problems. Such mathematical methods are characterized not only by their ability to suggest strategies for a routing but also by their capability to prove the quality of the proposals, which other approaches normally cannot do. A broadly–applied method to model and solve routing problems within a combinatorial optimization framework are so–called network flows. Commonly, lanes are taken as edges and crossings as nodes of the network. Restrictions such as driving times, lane capacities, and costs for using a lane are assigned as parameters to edges and nodes of this routing network.

Every routing problem stemming from an application involves a number of special requirements based on the conditions of the processes to be modeled. Some of these restrictions are covered by standard network flow models, such as lane capacities and directions in which lanes can be traversed. Standard methods do not include a time dimension, although it is essential for routing strategies. As a consequence, waiting policies cannot be handled. Mostly, standard approaches allow fractional flow portions, meaning that path flows cannot be seen as vehicles. Bounds on the number of paths used cannot be expressed because when formulating flow edgewise concrete paths are ignored.

In this thesis, we investigate two extensions of the standard model. The first extension is motivated by the fact that transportation tasks often have to be fulfilled with a limited number of vehicles. Due to this number, only a bounded number of paths can be used. Standard network flow algorithms are not designed to respect such a bound, meaning that flows may travel along a huge number of paths. We take this bound into account and study so–called *k–splittable flows* using no more than k paths. We investigate a problem related to the well–known maximum s, t–flow problem by looking for *maximum k–splittable s, t–flows* where k is a given integer or depends on graph parameters.

In the second part, we involve a time dimension that cannot be handled efficiently by standard methods. We study so–called *flows over time* that allow the modeling of movements of flow through a network. We investigate such flows over time in networks featuring a special structure, namely grids that often occur in storage areas. Regarding these grid flows, we consider further aspects such as waiting on edges and time windows that close edges for certain time intervals. Such aspects motivated by practical requirements lead to interesting research questions.

For both k–splittable flows and flows over time in grids, we concentrate on analyzing the complexity and approximability of such problems. We identify polynomially solvable and NP–hard cases for a variety of problem variants. Non–approximability results as well as approximation algorithms are shown.

Both flow problems are related to the *disjoint paths problem*: For a given network with sources and sinks, the question is, how many source–sink pairs can be connected by pairwise edge–disjoint paths. Finding a maximum integral multicommodity k–splittable flow with unit edge capacities and a single path for each commodity corresponds to the same question. Maximum flows over time in graphs with edge capacities 1 and integral flow portions reflect the disjoint paths problem in a dynamic environment, such that we call it the *disjoint paths over time* problem. As far as we know, disjoint paths over time have not been studied at all up until now.

Outline of this thesis

The main part of this thesis consists of three chapters.

In Chapter 2, we investigate k–splittable flows. This chapter is based on joint work with Ronald Koch and Martin Skutella of the University of Dortmund. Parts of it are published in [30], [31], and [32]. It deals with the *maximum k–splittable s,t–flow problem* (MkSF), expanding on the well-known maximum s,t–flow problem with the requirement that the solution can be split into no more than k path flows. In this chapter, the value k is either a constant integer or is dependant on graph parameters. For both instances we study the complexity and approximability of the problem MkSF. An overview of polynomially solvable cases and NP–hard variants is developed and bounds for approximation guarantees are proven. For solving MkSF, a new approach is introduced by decomposing the problem into two subproblems, a packing and a routing step, which can be solved consecutively. We succeed in the unusual way of first determining path flow values and then looking for corresponding routes. With this strategy, we are able to calculate solutions for a special graph class, namely graphs of bounded treewidth: In the case of a constant number k, the problem is solved to optimality, whereas if k is part of the input, a polynomial time approximation scheme (PTAS) is derived.

We omit restrictions on the number of paths used and include a time dimension beginning with Chapter 3. Flows over time are considered in grid graphs. In Chapter 3, the single commodity case is investigated. We are interested in finding *quickest flows*, which means to determine the minimum possible time horizons needed to satisfy given demands. We consider a variety of specific problem configurations that differ by having uniform or edge–specific transit times. Secondly, waiting on edges can or cannot be allowed. Furthermore, edges can or cannot be closed for certain time intervals. In the case where edges are not temporarily closed, we provide polynomial solution methods. When in contrast to this, time windows close edges for certain periods of time, then some configurations are shown to be NP–hard. In some cases, the complexity remains open. Moreover, we prove certain non-approximability results.

In Chapter 4, the previous chapter is extended to more than one commodity. Again, the above–mentioned problem configurations are investigated. We show how to replace time windows with additional commodities. Thus, in contrast to the previous chapter, time windows are not the critical factor for the problem's complexity. All considered problem variants are proven to be NP–hard by a reduction from 3-COLORING. A second, much easier reduction from PARTITION, which applies only when non–uniform transit times are allowed, refines the results for some variants by already showing NP–hardness for very small constant numbers of commodities. We discuss the influence of waiting and of integrality requirements. Two approximation algorithms are introduced, one for all variants without time windows and the other for cases when additional transit times are uniform along all grid edges.

Preliminaries

We assume the reader is familiar with basic concepts of complexity theory. A good survey is provided by Garey and Johnson [19]. Some terms used in this thesis are briefly summarized in the following. Throughout this work we assume that $P \neq NP$.

A problem is said to be *strongly NP-hard* if it is still NP-hard even when the absolute values of all numbers in the input are bounded by some polynomial in the length of the input. Thus, NP-hard problems without numbers are always strongly NP-hard. An algorithm is called *pseudo–polynomial* if it is polynomial in its number of input values and in its maximum input value. If there is a pseudo–polynomial algorithm, then a problem is not strongly NP-hard. A problem that is NP-hard but not in the strong sense is called *weakly NP-hard*.

In the following chapters, we refer to some well–known NP–complete problems. To simplify the notation and to avoid redundancies, we outline them here and use the same notation throughout this work. More details can be found in Garey and Johnson's work [19].

- SUBSETSUM: Given q positive integers $u_1, ..., u_q$ and a number M, is there a subset $S \subseteq \{1, ..., q\}$, such that $\sum_{i \in S} u_i = M$?

- SAT: A set of variables $\{x_1, ..., x_r\}$ and a set of clauses $\{C_1, ..., C_q\}$ over the set of variables is given. Is there a truth assignment on the variable set satisfying all clauses?

- 3SAT: A set of variables $\{x_1, ..., x_r\}$ and a set of clauses $\{C_1, ..., C_q\}$ over the set of variables is given, such that each clause contains three variables. Is there a truth assignment for the variable set satisfying all clauses? This problem is NP–complete in the strong sense.

- Partition: Given positive integers $a_1, ..., a_r$, is there a partition of these integers into two groups with the same sum of elements?

- 3-Coloring: Is it possible to color the nodes of a given graph with three colors, such that adjacent nodes have different colors? The problem is NP–complete in the strong sense.

Some NP–hard problems can be handled by so–called *approximation algorithms*. These are polynomial algorithms that provide a feasible solution to a problem with a certain performance guarantee. Such guarantees are given as relative or absolute values referring to an optimal value OPT. An algorithm A is said to have a *relative guarantee* of α if, for all problem instances I, it gives a solution $A(I)$ with $A(I) \geq \alpha\, OPT(I)$ for a maximization problem and $A(I) \leq \alpha\, OPT(I)$ for a minimization problem. Obviously, $\alpha \leq 1$ in the case of a maximization problem and $\alpha \geq 1$ in a minimization problem. A has an *absolute guarantee* of β if its solutions fulfill $A(I) \geq OPT(I) + \beta$ in a maximization problem, and otherwise $A(I) \leq OPT(I) + \beta$. In a maximization problem $\beta \leq 0$ is required and in a minimization problem $\beta \geq 0$.

As a reference for the reader, a list of symbols is given at the end of this thesis.

Chapter 2

Static k-splittable flows

2.1 Introduction

Many planning tasks in transport, telecommunication, production, or traffic logistics can be modeled as network flow problems. In classic flow theory, flow is sent through a network from sources to sinks respecting edge capacities. It does not matter how many paths are used, such that small portions of flow may be sent along a large number of paths. However, for many practical purposes, it is not seen as favorable to use a huge number of paths. In logistic networks, for example, the number of paths that can be used simultaneously is naturally limited by the number of available vehicles. In communication networks, data are often split into several packages that are sent along different paths. Each of them has to carry a great deal of information on source, target, relations to other packages and so on, such that a high number of packages is not desired. As in these examples, various applications ask for a flow that does not involve a large number of paths. Such a restriction is not taken into account by classical flow algorithms and cannot be easily incorporated. An edge–wise flow formulation does not relate to a unique path decomposition and is not suitable for limiting path numbers. Flow considered in a path formulation cannot be dealt with efficiently due to an exponential number of paths. For such purposes, so–called *k–splittable flows* have been introduced with an upper bound on the number of paths as an explicit additional requirement. In this chapter, we look for such flows having a maximum value. We prove the NP–hardness of the *maximum k–splittable s,t–flow problem* for different specifications of k and solve some polynomial variants. A framework is also developped to solve problem variants to optimality or near–optimality on graphs of bounded treewidth whereas they are hard on general graphs.

The further introductory part of the chapter contains some notation on static flows and k-splittability, preliminaries on bounded treewidth, an overview on the related literature, and an outline of the main part.

Static flows

Let $D = (V, A)$ be a directed graph with a node set V and an arc set A, $n := |V|$ and $m := |A|$. Arcs have capacities $u : A \to \mathbb{R}_{\geq 0}$. A source and a sink node $s, t \in V$ are given. An *s, t–flow* in D is a function on the arc set $f : A \to \mathbb{R}_{\geq 0}$ fulfilling *flow conservation* at each node except for s and t:

$$\sum_{a=(v,u)\in A} f(a) - \sum_{a=(u,v)\in A} f(a) = 0 \quad \forall u \in V \setminus \{s, t\}.$$

Such a flow is called *feasible* if capacity constraints are additionally met:

$$f(a) \leq u(a) \quad \forall a \in A.$$

The *value of a flow*, $|f|$, equates to the amount of flow leaving the source minus the amount of flow that reaches the source:

$$|f| := \sum_{a=(s,u)\in A} f(a) - \sum_{a=(u,s)\in A} f(a).$$

If we have more than one commodity, a set of d pairs of sources and sinks $(s_i, t_i) \in V \times V,\ i \in \{1, ..., d\}$, is given. Each node that belongs to the *set of sources* S or the *set of sinks* T is called a *terminal*. A function on the arc set $f : A \to \mathbb{R}_{\geq 0}$ respecting arc capacities, is a *feasible multicommodity flow* if f is the sum of feasible s_i, t_i–flows f_i, $i = 1, ..., d$.

We work with flows in undirected graphs in this thesis. Let $G = (V, E)$ be an undirected graph with a node set V and a set of edges E, and again $n := |V|$ and $m := |E|$. A function $f : E \to \mathbb{R}_{\geq 0}$ is a feasible s, t-flow in G if there is an edge orientation, such that f is a feasible s, t-flow in the resulting directed graph. Moreover, f is a feasible multicommodity flow if it can be divided into feasible s_i, t_i-flows. Edge orientations can be choosen for each commodity independently. Notice that for respecting capacity constraints, all flow along an edge is taken into account no matter its direction.

In the following, we will not distinguish between flows and feasible flows. Flows considered in this thesis have to fulfill flow conservation and capacity constraints in each case. Hence, feasibility is not mentioned explicitly.

Static s, t-flows have been addressed many times in the literature. Ford and Fulkerson [18] give an overview on the initial results. For more details see, e.g., Ahuja, Magnanti, and Orlin's work [1].

k-splittable flows

The concept of k–splittable flows has been introduced by Baier, Köhler, and Skutella [5]. Let $G = (V, E)$ be a connected undirected or directed graph. Additionally, a number $k \in \mathbb{N}$ is given. A flow is called k–splittable if it can be decomposed using at most k paths which are not required to be different or even disjoint. The *maximum k–splittable flow problem*(MkSF) asks for a k–splittable s, t–flow of maximum value similar to the well-known maximum s, t–flow problem. A standard result states that any s, t–flow can be decomposed into flow along, at most, m paths and cycles. This decomposition can be done in polynomial time. Thus, MkSF is an interesting problem for values of k that are smaller than m.

Of course, k–splittability can also be considered more generally in a multicommodity context. In this case, a bound on the number of all paths or an individual bound for each commodity is given. An *unsplittable* flow is a multicommodity flow with a single path per commodity. If all paths are required to carry the same amount of flow, then the flow is called *uniform*. If all edge capacities and demands are 1, then asking for a maximum integral k–splittable flow is equivalent to the problem of finding a maximum number of disjoint paths.

Kleinberg [28] investigates unsplittable flows. He studies their complexity and introduces approximation algorithms for different unsplittable flow problems, e.g., for minimizing the congestion along edges or equivalently maximizing the throughput, for the problem of minimizing the number of graph copies to satisfy all demands and for the problem of maximizing the total demand which can be routed simultaneously.

Baier, Köhler, and Skutella [5] investigate k–splittable flows with one or more commodities. They prove the NP–hardness of the MkSF problem in directed graphs for all constant values of $k \geq 2$. Considering MkSF, they provide a maximum flow – minimum cut result and an optimal solution algorithm working in $O(k\ m \log n)$. For the general MkSF problem with different path flow values, a 1/2–approximation is derived based on results for the uniform case.