Chapter 1

Introduction

Online Optimization

In classical optimization, all input data of a problem instance is assumed to be available at once. Many real-life problems, however, require decisions before information about the data is complete. This insight has prompted the research in *online optimization*. In an online optimization problem, decisions have to be made while parts of the data are still missing.

Many real-life problems are naturally online. Picture, for instance, your next weekend trip to a European city. You want to see the most important sights, including monuments and museums, and walk around to immerse into the city's special atmosphere. When you arrive, one of the first decisions you will need to make is whether to buy a 4-day ticket, valid on all public transport, at a price of 18 Euro, or whether to buy tickets whenever you need them. Single ride tickets have a validity of 90 minutes, and their prices range from 1.50 Euro to 2.30 Euro, depending on the length of your ride and the means of transportation you choose. So, what should you do? A 4-day ticket would be convenient, of course: you don't have to bother about buying tickets any more. But is it worth it? On the one hand, if the weather is nice, then you'll probably walk most of the time, and you could save some money now and afford another (probably too expensive) coffee on one of the main boulevards later. On the other hand, if it rains a lot, then it would be nice to take a bus ride through the city or to catch the subway to the next museum. A lot of information such as the actual weather conditions, special events you might want to attend, etc., are not known to you in advance. This forces you to make decisions under uncertainty.

Incomplete information is a feature common to many real-life problems. Online problems arising in practice include distributed data management, foreign exchange and stock trade, the control of elevators, and the routing of calls in a telecommunication network.

In online optimization, the input data is usually modeled as a sequence of *requests* that is revealed piecewise. An online algorithm may base its decisions at any time only on the requests known to it so far. Request sequences can be classified into two types: the *sequence model* and the *time stamp model*.

In the sequence model, an online algorithm must process the requests one by one. It gets to know the next request only after it has made an irrevocable decision for the previous one. A well-known example is *paging* in a virtual memory system: a central processing unit (CPU) must decide which page to evict from its fast memory upon receiving a request for some page in its slow memory. The next request is only disclosed when the previous one has been processed, i.e., when the CPU has ensured that the requested page is in its fast memory. Time does not play any role in this model. Each decision of an online algorithm results in some gain or loss, and the objective function usually depends on the total gain or loss.

In contrast, time is decisive in the time stamp model. Here, each request has a non-negative *release time*, and an online algorithm gets to know further requests as time is progressing. The objective function usually depends on time. An online optimization problem in which new requests arrive over time is the *Online Traveling Repairman Problem*, an online variant of the famous *Traveling Salesman Problem*. In the Online Traveling Repairman Problem, a repairman has a set of jobs to do. Each job requires him to drive to a customer. While he is on his way, the repairman receives new requests and must decide how to rearrange his schedule such as to finish each job as early as possible.

Evaluation of Online Algorithms

For the comparison of online algorithms, it is necessary to have meaningful measures for assessing their quality. Several approaches have been taken to evaluate online algorithms.

In the traditional *distributional* or *stochastic* approach, a distribution over the problem instances is assumed, and the *expected* objective value of the online algorithm under consideration is computed. The major weakness of this average-case approach is that the assumed distribution is often either unrealistic, or too complex, making the computation of the expected value intractable. Another option is to compare the worst-case objective values of two algorithms. Alas, this approach is also problematic: what if all algorithms are equally bad in the worst-case? In the paging problem mentioned above, for instance, all algorithms show the same worst-case behavior: they incur a page fault in each step.

Competitive analysis tries to overcome this weakness by introducing a (hypothetical) benchmark algorithm, the *optimal offline algorithm*, that knows the given sequence in advance and can process it in an optimal way. The worst-case performance of an online algorithm is measured relative to the optimal offline algorithm:

Given a minimization problem, an online algorithm ALG is said to be *c-competitive*, if, for any problem instance σ, the objective function value of ALG on σ is at most c times the objective value of the optimal offline algorithm on σ. Since the ratio of the online and the offline objective value must be bounded by the constant c for all instances, competitive analysis is a worst-case approach. It is intended to answer the question what is lost in the worst case by the lack of complete information.

Competitive analysis has become the standard tool in online optimization. Nevertheless, it suffers from several conceptual deficiencies. For instance, it completely disregards complexity issues; an online algorithm is not required to be efficient or to make real-time decisions. Fast computation becomes important, however, when online algorithms are to be used in practice. Very frequently, decisions have to be made within a given time bound, often within seconds. Therefore, online algorithms intended for real world problems must be efficient, or at least workable.

An important instrument for the assessment of practical algorithms is simulation. Simulation experiments are indispensable to emulate and evaluate the behavior of online algorithms designed for practical use. In particular when worst-case performance bounds are not available or based on unrealistic scenarios, carefully chosen simulation runs provide valuable experimental performance guarantees. It is important to use real data for simulation whenever possible.

Other weaknesses of competitive analysis, as well as efforts to overcome them, will be addressed in Section 2.3.

Online Dial-a-Ride Problems

In the Dial-a-Ride Problem (Darp), one or several servers of given capacities and of unit speed have to transport objects between points in a metric space. Each transportation request specifies a *source* and a *destination*. The task is to design a sequence of moves for each server such that all transportation requests, also called *ride*s, are covered, the server's capacity bounds are not exceeded, and a given objective function is minimized. Moreover, unless specified otherwise, *preemption* is prohibited: once an object has been picked up, it can only be dropped at its destination. In some variants of the Darp, the servers are additionally required to eventually return to their initial position.

The class of Dial-a-Ride Problems comprises many well-studied problems in combinatorial optimization. For instance, the *Traveling Salesman Problem* is the special case of the single-server Darp with the makespan as objective function in which source and destination of each ride coincide. Also many *Vehicle Routing Problem*s can be formulated within the Darp framework. Moreover, Dial-a-Ride Problems can be used to model *scheduling problems* in which the jobs have order dependent setup costs.

We are interested in the online version of the Dial-a-Ride Problem with a single server, further referred to as the *Online Dial-a-Ride Problem* (OlDarp). In the OlDarp, transportation requests are not known beforehand but become known over time. That is, in addition to source and destination, each ride specifies a *release time*. In the online setting, the server has at no point in time any information about requests whose release time is greater than that point in time. In particular, it neither knows the total number of requests, nor the release time of the last request. Objective functions that have been considered for the OlDarp are the *makespan* (completion time of the schedule), the *latency* (weighted sum of completion times of all requests), the *average flow time* (average time in which a request remains in the system), and the *maximum flow time*. In this thesis, we present new results for the OlDarp with the latency and with the maximum flow time as objective functions.

Online Dial-a-Ride Problems occur frequently in practice, in particular in logistics. Applications are machine scheduling, field service, delivery and courier services, elevator and stacker crane control, transportation of disabled persons, and the dispatching of automobile service units, among others.

Online Call Admission in All-Optical Networks

All-optical telecommunication networks are the optical networks of the next generation. While in today's networks, signals are already transmitted as light pulses via glass fibers, but still switched electronically in intermediate nodes, new devices will shortly allow to process signals completely within the optical domain.

The *Wavelength Division Multiplexing* (WDM) technique, deployed for the first time in the early 1990ies, brought a substantial increase in transmission capacities of telecommunication networks. By installing so-called *multiplexer*s and *demultiplexer*s at the beginning and the end of a fiber, respectively, the available bandwidth of a fiber is separated into different wavelengths that can be used in parallel by different signals. Recently, new devices for the switching and the insertion/extraction of signals in optical form have been developed, the so-called *Optical Cross-Connects* and *Optical Add-Drop-Multiplexers*. They are expected to be commercially available very soon. Moreover, *wavelength converters* are being devised that enable the (optical) switching of signals from one wavelength to another. Altogether, these new devices supersede current time-consuming conversions between optics and electronics. *All-optical networks* refer to optical networks that deploy these new switches in addition to the WDM technique.

All-optical networks require new mathematical models and give rise to new problems. Their crucial difference to the networks currently in use is that a signal sent through an all-optical network remains in optical form on its whole path from start to end node. Therefore, a connection in an all-optical network is realized via a *light-*

path, which is a path in the network together with a wavelength. Resources are limited: each wavelength may only be used once per fiber; consequently, two lightpaths that use the same fiber must have different wavelengths. This crucial restriction is called *wavelength conflict constraint*. A natural online problem is the *dynamic configuration of optical networks*. In its simplest variant it can be stated as follows: new connection requests arrive over time, and an (online) algorithm has to decide for each request whether to accept or reject it (*call admission*). If the request is accepted, the algorithm must provide a lightpath to realize the required connection without violating the wavelength conflict constraint (*routing and wavelength assignment*).

Overview

This thesis is divided into two major parts: in Part I, we investigate various Online Dial-a-Ride Problems; Part II is concerned with the dynamic configuration of all-optical networks.

Preceding these two major parts is Chapter 2, which is intended as a short reference to the concepts and the basic notation used in this thesis. We give a formal introduction to online optimization and competitive analysis, including deterministic as well as randomized online algorithms. We also introduce a useful technique for obtaining lower bounds on the competitive ratio of randomized algorithms. Moreover, we discuss the weaknesses of competitive analysis and cover known modifications and extensions of it.

In Part I, we present new results for several Online Dial-a-Ride Problems. After a formal introduction to Online Dial-a-Ride Problems in Chapter 3, Chapter 4 deals with Online-Dial-a-Ride Problems with the *latency* as objective function ($\sum w_j C_j$-OlDarp). The latency is defined as the weighted sum of completion times, where the *completion time* of a request is the time when the corresponding object is dropped at its destination. We present new lower bounds on the competitive ratio of any online algorithm, both for the general $\sum w_j C_j$-OlDarp and for the special case in which each ride's source and destination coincide, the $\sum w_j C_j$-OlTsp, also known as the *Online Traveling Repairman Problem* (OlTrp). The main result of this chapter is a $(1+\sqrt{2})^2$-competitive deterministic online algorithm for the $\sum w_j C_j$-OlDarp in general metric spaces. This algorithm significantly improves previous upper bounds for both the $\sum w_j C_j$-OlDarp and the $\sum w_j C_j$-OlTsp. Moreover, a modification of the algorithm yields a new randomized upper bound.

In Chapter 5, we investigate the OlDarp with the maximum flow time $F_{\max}$ as objective function, shortly $F_{\max}$-OlDarp. Again, we also consider the special case in which each ride's source and destination coincide, the $F_{\max}$-OlTsp. Easy worst-case sequences reveal that there is no competitive online algorithm for neither problem, showing that competitive analysis fails to evaluate and distinguish online