Aproximační a online algoritmy

Tomáš Tichý

Matematický ústav Akademie věd České Republiky
Institut teoretické informatiky
Matematicko–fyzikální fakulta
Univerzita Karlova v Praze

Obor: I4 - Diskrétní modely a algoritmy

Autoreferát doktorské disertační práce

Duben 2008

Approximation and Online Algorithms

Tomáš Tichý

Institute of Mathematics of the Academy of Sciences
of the Czech Republic
Institute for Theoretical Computer Science
Faculty of Mathematics and Physics
Charles University in Prague

Branch: I4 - Discrete Models and Algorithms

Abstract of Doctoral Thesis

April 2008

Disertační práce byla vypracována v rámci doktorského studia, které uchazeč absolvoval na Matematicko–fyzikální fakultě Univerzity Karlovy v Praze v letech 2001–2008.

Doktorand: RNDr. Tomáš Tichý

Školitel: Doc. RNDr. Jiří Sgall, DrSc.

Školící pracoviště: Matematický ústav AV ČR
Žitná 25
115 67 Praha 1

Oponenti: Mgr. Petr Kolman, Ph.D.
kolman@kam.ms.mff.cuni.cz
Katedra aplikované matematiky
Matematicko-fyzikální fakulta Univerzity Karlovy
Malostranské náměstí 25
118 00 Praha 1

Rob van Stee, Ph.D.
vanstee@mpi-inf.mpg.de
Department 1: Algorithms and Complexity
Max-Planck-Institut für Informatik
Building 46.1, Room 305, Campus E 1 4
Germany, 661 23 Saarbrücken

Autoreferát byl rozeslán dne: ....

Obhajoba se koná dne .... v .... hodin před komisí pro obhajoby disertačních prací oboru I4 - Diskrétní modely a algoritmy, na Matematicko–fyzikální fakultě Univerzity Karlovy, místnost S5, 2. patro, na adrese Malostranské námestí 25, 118 00 Praha 1.

S disertací je možné se seznámit na Útvaru doktorského studia Matematicko–fyzikální fakulty Univerzity Karlovy v Praze, Ke Karlovu 3, 121 16 Praha 2.

Předseda oborové rady I4: Prof. RNDr. Jaroslav Nešetřil, DrSc.
Katedra aplikované matematiky, MFF UK

Chapter 1

Introduction

1.1 Overview

This thesis presents results of our research in the area of optimization problems with incomplete information—our research is focused on the online scheduling problems. Our research is based on the worst–case analysis of studied problems and algorithms; thus we use methods of the competitive analysis during our research.

Althrough there are many "real–world" industrial and theoretical applications of the online scheduling problems there are still so many open problems with so simple description. Therefore it is important, interesting and also challenging to study the online scheduling problems and their simplified variants as well.

In this thesis we have shown the following our results of our research on the online scheduling problems:

- A 1.58–competitive online algorithm for the problem of randomized scheduling of unit jobs on a single processor, where the jobs are arriving over time and the total weight of processed jobs is maximized.
- A lower bound 1.172 on the competitive ratio for the problem of randomized scheduling of 2–uniform unit jobs on a single processor, where the jobs are arriving over time and the total weight of processed jobs is maximized.
- A lower bound 1.25 on the competitive ratio for the problem of randomized scheduling of s–uniform unit jobs on a single processor where s is tending to infinity, the jobs are arriving over time and the total weight of processed jobs is maximized.
- A 1.5–competitive online algorithm for the problem of deterministic scheduling of equal–length jobs on a single processor, where restarts of jobs are allowed, the jobs are arriving over time and the total weight of processed jobs is maximized.
- There is no online 1–competitive algorithm with speed–up $s < 2$ for the problem of deterministic scheduling of tight jobs on a single processor,

where the preemptions of jobs are allowed, the jobs are arriving over time and the total weight of processed jobs is maximized.

- There is no 1–competitive k–relaxed online algorithm for any k for the problem of deterministic scheduling of jobs arriving over time with maximizing total weight of processed jobs.
- A lower bound 1.05099 on the competitive ratio for the problem of 1–relaxed deterministic scheduling of jobs arriving over time with maximizing total weight of processed jobs. We have shown a generalized lower bound for k–relaxed algorithms.

We present also some other results related to studied problems that are products of a joint work with other researchers.

1.2 Motivation

Our area of interest arises from a traditional combinatorial area—the area of optimization problems. The optimization problems are widely studied for a very long time, but there are many pretty hard and famous problems. Because of the huge scope of the optimization problems there arise some subareas like scheduling which try to solve some classes of optimization problems.

We focus on variants of the optimization problems with incomplete information. In such problems, the information about the problem arrives in steps and we are forced to make decisions while the information is still incomplete.

In the usual definition of the optimization problems it is assumed that the whole information is known and fixed. However it is not the case of the real world, where information arrives incrementally in time and we have to make decisions continuously. This is exactly the case when the *online algorithms* can help us. The reason is that they are defined to make decisions using partial knowledge of their input instance.

In the next sections we discuss basic definitions, methods and the current state of art. We also give an overview of the studied problems.

1.3 Optimization and approximation

Let us describe what an optimization problem is. Consider an input instance for the problem. This input instance together with the problem definition gives a set of discrete objects. These objects are feasible solutions of the problem for a given input instance. An objective (cost or profit) function of the problem measures the quality of any particular solution. The goal is to find an optimal solution among the feasible solutions for a fixed input instance, such that it minimizes its cost or maximizes its profit.

For many interesting optimization problems it is really hard to find an optimal solution in reasonable time or space, because of huge complexity in time or space. Interesting optimization problems are mostly NP–hard.

In practice there are many industrial applications of the optimization problems like making plans, working or logistic schedules etc. The advantage of practice is that we usually do not need an optimal solution but it is enough to find a sufficiently reasonable feasible solution.

One of the methods to find a reasonable solution is a method of approximation algorithms. The approximation algorithms are fast (with low complexity) and are provably close to the optimum.

1.4 Online algorithms

The concept of an online algorithm formalizes the real–world scenario, where a real algorithm does not know the whole input instance while offline algorithms do. Instead of this the online algorithm gets pieces of the input instance in steps (in time) and the algorithm has to react to the new requests with only partial knowledge of the input.

Moreover, many heuristical or approximation algorithms applied on hard optimization problems are actually online algorithms. The main reason is that we need to design simple algorithms because of realizable analysis and implementation. Simple algorithms are usually unable to use the complete information of the input instance. They split input instance into smaller pieces, for example they sort objects of the input instance and process it one by one.

In the offline world we are usually interested in the time and/or space complexity of the studied algorithms. Instead, in the online world we are interested in the quality of the produced solution. When we have sufficient time and space resources in the offline world, we always find the optimal solution. But this does not hold in the online world because of the important role of the partial knowledge of the input instance. Moreover, online algorithms are usually simple and fast, hence there is much more interesting challenge in the quality of produced solutions than in their complexity.

The online algorithms are usually quite simple heuristical algorithms with very low time and space complexity. Nevertheless the analysis of the online algorithms is sometimes really complicated. Let us take a look on the common methods used for measuring the quality of the online algorithms.

1.4.1 Competitive analysis

The *competitive analysis* was introduced by Sleator and Tarjan [24] and it is a variation on the traditional *worst–case analysis* of optimization algorithms. The worst–case analysis studies the performance of an algorithm in the worst case. We have to define what does it mean exactly.

We measure the cost of an online algorithm on an input instance by a cost function. The cost function is unbounded and we are interested in the behaviour on all input instances. Therefore we compare the cost to the cost of another algorithm for each fixed input instance. We want to bound the ratio of these costs over all input instances.

We are interested in how much worse is the online algorithm against the optimal solution. Since we give no restrictions on time or space complexity we can assume that the offline algorithm always produces an optimal solution. Hence in the competitive analysis it is natural to compare the cost of the online algorithm to the cost of the offline algorithm on the same input instances.

According to the worst–case analysis we look for the comparison of an online algorithm and an offline algorithm on the worst instance. Formally we define this by the *competitive ratio* as follows.

We consider minimization or maximization problems. In the minimization problems an algorithm minimizes cost which it has to pay. In the maximization problems an algorithm maximizes profit which it gets. Let us denote an online algorithm as $\mathcal{A}$, an input instance as σ and the cost or profit of the algorithm on the instance as $\mathcal{A}(\sigma)$. Let OPT denote an (arbitrary fixed) offline (optimal) algorithm and $\mathrm{OPT}(\sigma)$ the optimal cost or profit on the instance.

For minimization problems we define the competitive ratio (over all possible input instances) as follows:

$$\mathcal{R}(\mathcal{A}) = \inf_{R \in \mathbb{R}} \{R : (\forall \sigma), \mathcal{A}(\sigma) \le R \cdot \mathrm{OPT}(\sigma)\}$$

and similarly for the maximization problem as follows:

$$\mathcal{R}(\mathcal{A}) = \inf_{R \in \mathbb{R}} \{R : (\forall \sigma), R \cdot \mathcal{A}(\sigma) \ge \mathrm{OPT}(\sigma)\}.$$

Because of these definitions, the competitive ratio is always greater than or equal to 1. The best online algorithm is the online algorithm with the lowest possible competitive ratio. An online algorithm is an optimal algorithm for the problem if and only if its competitive ratio is exactly 1.

When we consider a fixed scheduling problem then we also define the **competitive ratio of the problem** as

$$\mathcal{R} = \inf_{\mathcal{A}} \mathcal{R}(\mathcal{A}),$$

which is going over all online algorithms for the problem.

Obviously, the competitive ratio is a worst–case measure of online algorithms. It shows the strong influence of uncertain input instance (partial knowledge of instance) on the profit. Another userful feature of the measure is the following claim. Let us consider an algorithm $\mathcal{A}$ with the competitive ratio $c = \mathcal{R}(\mathcal{A})$ and any other better d–competitive algorithm $\mathcal{B}$ for some $d < c$. Then there exists an input instance such that the better algorithm is at least c/d–times better on the instance. Obviously the worst case input instance of the algorithm $\mathcal{A}$ proves this claim.

1.4.2 Randomization in competitive analysis

Randomization is a standard extension of the competitive analysis. It gives us ways of significant improvement of the competitive performance of our algorithms while we remain in the worst–case world. The basic idea of randomization is to allow the algorithm to use random bits in its decision process. Instead of the objective function on a fixed input instance we consider the expectation of the objective function on the fixed input instance.

The competitive ratio for a minimization problem is defined as

$$\mathcal{R}(\mathcal{A}) = \inf_{R \in \mathbb{R}} \{R : (\forall \sigma), \mathbb{E}[\mathcal{A}(\sigma)] \le R \cdot \mathrm{OPT}(\sigma)\}$$

and similarly for the maximization problem as follows:

$$\mathcal{R}(\mathcal{A}) = \inf_{R \in \mathbb{R}} \{R : (\forall \sigma), R \cdot \mathbb{E}[\mathcal{A}(\sigma)] \geq \mathrm{OPT}(\sigma)\}.$$

The important aspect is that we do not consider randomization over input instances but we consider randomization for each fixed input instance separately.

Therefore the nature of the analysis remains worst–case. The randomized computation model is stronger than deterministic model, hence the algorithm is more powerful. The model can significantly improve the competitive ratio of some problems.

1.4.3 Resource augmentation in competitive analysis

The resource augmentation is one of the common techniques used in the competitive analysis. Generally, results deloped under this technique allow us to better understand the studied problems—give use more complex view on the problems—and possibly allow us to design better algorithms for the problems.

For the first time, this technique was already used in 1966 by Graham in [16]. The technique was officially introduced and entitled in 1995 by Kalyanasundaram and Pruhs [17]. They demonstated this method on a certain scheduling problem.

The basic idea of the resource augmentation technique is to allow more resources to the online algorithm than to the corresponding offline algorithm on the same problem. For example, the online algorithm can be allowed to use more processors, faster processors, later deadlines, etc. Under this technique we study the problems from the point of view of the sufficiency of resources. We study changes of the competitive ratio when we break some of the resource constraints.

Chapter 2

Problems and results

The basic purpose of this chapter is to show a compact overview of all problems studied in this thesis. This overview also includes enumeration of our results related to the studied problems.

We do not intend this thesis to be a detailed description of a new entire consistent theory on scheduling and approximation algorithms. The area of the approximation and online algorithms is so wide and contains lots of interesting open problems that are waiting for their resolution. We focused our research effort on some of these problems and we tried to solve them or at least to solve its subproblems. We were successful in some cases and of course in some cases we were not successful. Instead of developing a consistent theory we are helping to assemble the mosaic of open problems of the online and approximation algorithms by providing our fragments of the mosaic—we provide a set of various results of our research.

When we look at the problems which we study we will see that we are solving similar scheduling problems for various special cases. Basically there are two reasons:

- the general case is usually too hard to be solved directly, we are getting closer by solving special cases,
- the general case is already solved, but the results are not satisfactory—they are too weak to be used in practice, then we are solving special cases to get stronger results for reasonable restrictions.

In the introduction we have generally discussed the area of scheduling problems. In the thesis we discuss the scheduling problems in more details including formal taxonomy. Let us continue the discussion. But now the discussion will be focused on our research, especially on the problems studied in this thesis.

We show a detailed overview on these problems, we mention related problems and known related results from other authors—the current state of art—and also results of our research.

2.1 Online scheduling of unit jobs

As the importance of the Internet is growing, people are searching for improvements of the general performance of the global network. In the network the packets are forwarded by network routers and switches. Unfortunatelly, the most of them implements the First–In–First–Out (FIFO) strategy for packet forwarding. However the communication protocols based on IP (Internet Protocol) are sufficiently robust—assumes unpredictable packet flows and heterogenous networks as well. Thus the research of the strategies based on QoS (Quality of Service) for network routers snad switches become more important.

We discuss the problem of online scheduling of unit jobs which arises from the area of buffer management problems in this section. In the buffer management problems we study how to manage buffers for storing network packets in the QoS networks. In such networks packets arrive and are buffered at network switches. Each packet has its QoS value which is the profit gained by forwarding the packet. The network switch works in steps—the switch can receive and transmit only one packet at each step. When the system is overloaded then some packets will not be delivered before their deadline (dropped packets). Such a buffer management problem can be formally described at the following problem of online scheduling of unit jobs.

2.1.1 Problem description

In the model the processing time of each job is equal to 1. Each job is specified by its release time, deadline and weight, where release times and deadlines are integral values and weight is a non–negative real value. These jobs are processed on a single processor—at most one job can be processed at each integral time. It is allowed to drop jobs which cannot be processed before their deadline. The profit is the total weight of all jobs completed before their deadline. The goal is to maximize the obtained profit.

2.1.2 Previous results

We consider the problem in the randomized computational model, thus the deterministic lower bounds apply for the problem. We also mention the deterministic upper bound—the best randomized algorithm must be better (or equal) than the deterministic algorithm.

- **Upper bound for deterministic algorithms**—The best known upper bound was recently presented in [12]—the 1.828–competitive deterministic algorithm. For a long time the best upper bound was a 2–competitive deterministic algorithm presented in [18]. This result was improved in our joint paper [5]—the 1.939–competitive algorithms, the first algorithm with the ratio strictly below 2. This was recently improved in two independent papers. The first was mentioned as the best known, the second one is the 1.854–competitive algorithm which is presented in [23].
- **Upper bound for memoryless deterministic algorithms**—The best known upper bound for memoryless deterministic algorithms is the 1.893–competitive algorithm presented in [12]. It is the first algorithm

with the competitive ratio strictly below 2 for the memoryless algorithms for the problem.

- **Lower bound for deterministic algorithms**—The best known deterministic lower bound on the competitive ratio for the problem is the $\phi \approx 1.618$—this lower bound is based on the 2–bounded input instances and is presented in [2] and [10].
- **Lower bound for randomized algorithms**—The best known lower bound on the competitive ratio for the problem in the randomized model is 1.25, it has been shown in [10] and also based on the 2–bounded instances.
- **Upper bound for randomized algorithms**—There are none, bounded by the deterministic upper bounds.

2.1.3 Our results

We have shown in Section 4.5.1 the following result for the studied problem, this result is published in [Pub–2]:

- **Upper bound for randomized algorithms**—We have improved the upper bound for the problem in the randomized model. We have shown the $\frac{e}{e-1} \approx 1.58$–competitive algorithm which is still the best known upper bound for the problem in the randomized model. This result is published in a joint paper [Pub–2] together with other results of other co–authors. This result is proven in Theorem 4.5.1.

2.2 Online scheduling of uniform jobs

The problem of the online scheduling of uniform jobs arises from the buffer management problems as the problem of the online scheduling of the unit jobs—the problems are closely related and have the same motivation.

This problem introduces a restriction on the input instances for the problem which can be considered as reasonable for practical point of view. Thus it is natural that we can develop stronger algorithms—with better competitive ratio than in the general case. In the considered problem the attribute "uniform jobs" means that the input instances are restricted—these consist of s–uniform jobs, where s is a parameter of considered problem, this means that the span of each of the jobs is equal to s.

2.2.1 Problem description

The problem of the online scheduling of uniform jobs is about the online scheduling of s–uniform input instances of unit jobs, where s is a fixed integer—the parameter of the considered problem. Each job has its span (difference of its deadline and release time) equal to s. Each job is specified by its release time and weight. Jobs are processed on a single processor, some jobs can be dropped. The objective is the total weight of scheduled jobs.

2.2.2 Previous results

Because of the parameter s for the problem we can consider the problem in two ways—we can consider the problem for some values of s and we can consider the problem in general—the worst case over all possible values of s as well. Naturally, we are interested in both cases. Thus we distinguish the lower and upper bounds according according to these two cases. Obviously, a lower bound for a special case applies in the general case.

- **Upper bound for deterministic algorithms on general input instances—** The best known general upper bound in the deterministic model is the 1.75–competitive algorithm and it was shown in [3].
- **Upper bound for deterministic algorithms on 2–uniform input instances—**The best known upper bound in the deterministic model for input instances restricted on 2–uniform jobs is the $\sqrt{2} \approx 1.41$–competitive algorithm and it was shown in [2].
- **Lower bound for deterministic algorithms on 2–uniform input instances—**The best known lower bound in the deterministic model for the 2–uniform instances is approximately 1.366, this result was shown in [2].
- **Lower bound for deterministic algorithms on general input instances—** The currently best known upper bound is 1.36. This result follows from the lower bound for the 2–uniform instances shown in [2].

2.2.3 Our results

We have shown in Section 4.6.1 the following results for the studied problem:

- **Lower bound for randomized algorithms on 2–uniform input instances—**We have shown in [Pub–2] a lower bound 1.172 on the competitive ratio for the problem for 2–uniform instances in the randomized model. This is still the best known lower bound in randomized model. This result is proven in Theorem 4.6.2.
- **Lower bound for randomized algorithms on general input instances—** We have shown a lower bound 1.25 on the competitive ratio for the problem with the s–uniform instances, where the span s is tending to infinity. We have presented this result in [Pub–2]. This result is proven in Theorem 4.6.3.

2.2.4 Joint results

The following interesting results are the products of our joint research on the problem with other co–authors. These results were shown in paper [Pub–2] and [Pub–3]. There results are presented in Section 4.6.2.

- **Matching lower bound on the competitive ratio for deterministic algorithms on 2–uniform input instances—**We have shown that there is no deterministic online algorithm for the problem of online scheduling of

2–uniform input instances with competitive ratio smaller than approximately 1.376. This result is presented in Theorem 4.6.5 and matches the following upper bound.

- **Upper bound on the competitive ratio for deterministic algorithms on** 2**–uniform input instances**—We have shown 1.377–competitive algorithm for the problem of online scheduling of 2–uniform input instances in deterministic model. This is presented in Theorem 4.6.6.

2.3 Online scheduling of bounded jobs

The problem of online scheduling of bounded jobs is very similar to the previous problem of online scheduling of uniform jobs. This problem is also parametrized—again the parameter s restricts the span of jobs in the input instances. The difference is that in the problem with uniform jobs the span must be equal to the parameter s and in the problem with bounded jobs the span must be at most s.

2.3.1 Problem description

The problem of online scheduling of bounded jobs is about online scheduling of s–bounded input instances of unit jobs, where s is a fixed integer—the parameter of the considered problem. Each job has its span (difference of its deadline and release time) at most s. Each job is specified by its release time and weight. Jobs are processed on a single processor, some jobs can be dropped. The objective is the total weight of scheduled jobs.

2.3.2 Previous results

For the problem of the online scheduling of uniform jobs we consider the problem from two points of view—as the worst case over all possible values of the parameter s of the problem and the problem for a fixed parameter s. Observe that in the studied problem of the online scheduling of bounded jobs the set of input instances for a parameter s contains also all input instances for all parameters smaller than s. Thus the worst case over all possible values of the parameter s is given by the parameter s tending to infinity.

Moreover the problem for the parameter s tending to infinity colapses to the previously described problem of online scheduling of unit jobs because with growing parameter s we lose boundaries.

- **Upper bound for deterministic algorithms on general input instances**—the best known general deterministic upper bound for the problem—for unlimited value of parameter s—is the 2–competitive algorithm that has been shown in [18].

- **Upper bound and lower bound for deterministic algorithms on** 2**–bounded input instances**—The optimal algorithm is known for the problem for the 2–bounded instances—this algorithm is $\phi \approx 1.618$–competitive. This algorithm was introduced in [18] and corresponding lower bound was shown independently in [2] and [10].

- **Lower bound for randomized algorithms on 2–bounded input instances**—The best known lower bound on the competitive ratio in the randomized model is 1.25 and was shown in [10].

2.3.3 Joint results

The following interesting results are the products of our joint research on the problem with other co–authors. These results were shown in paper [Pub–2]. There results are presented in Section 4.7.

- **Upper bound for deterministic algorithms on s–bounded input instances**—We have shown an algorithm for the problem in the deterministic model, its competitive ratio is given by formula $2 - 2/s + o(1/s)$. The competitive ratio tends to general upper bound 2 for the growing parameter s. Presented in Theorem 4.7.5.
- **Upper bound for deterministic algorithms on 4–bounded input instances**—We have shown the approximately 1.732–competitive algorithm for the problem with 4–bounded input instances in the deterministic model. Presented in Theorem 4.7.4.
- **Upper bound for deterministic algorithms on 3–bounded input instances**—We have shown the $\phi = 1.618$–competitive algorithm for the problem with 3–bounded input instances in the deterministic model. Presented in Theorem 4.7.2.
- **Upper bound for randomized algorithms on 2–bounded input instances**—We have shown the 1.25–competitive algorithm for the problem with 2–bounded input instances in the randomized model. This upper bound matches the best known lower bound mentioned above thus the algorithm is optimal for the case of 2–bounded instances in the randomized model. Presented in Theorem 4.7.1.

2.4 Online scheduling of equal–length jobs

The problem of online scheduling of equal–length jobs is one of the fundamental problems in the area of real–time scheduling. The motivation for the problem is in the real–time scheduling of jobs in overloaded systems where the matching deadlines is very important. We can find practical application in packet switched networks (with and without preemptions)—various streaming and processing applications, when weights are allowed the problem is related to quality of service problems. Our motivation is that the studied problem is a simplified version of fundamental scheduling problem where only a little is known about its competitiveness.

The jobs in the considered problem are almost the same—the jobs have equal processing times and equal weights (weights are not specified). The goal is to maximize the total number of jobs completed before their deadlines.

2.4.1 Problem description

The studied problem is about online scheduling of jobs where the processing time of each job is equal to p where p is a parameter of the problem. Each job is specified by its release time, deadline, where release times and deadlines are integral values. Weights are not specified. The jobs are processed on a single processor. It is allowed to drop jobs that cannot be processed before their deadlines. The considered profit is the total number of jobs completed before their deadlines. The goal is to maximize the obtained profit.

The resulting schedule—produced by the offline and online algorithms—has to be non–preemptive. Althrough we allow preemptions with restarts to the online algorithm in one of studied cases the requirement on the non–preemptive resulting schedule is still satisfied, because we obtain profit only for such a job that is completed before its deadline and its processing was not preempted because of the nature of restarts.

2.4.2 Previous results

Because of the importance of the problem for the area of real–time scheduling, the problem was extensively studied also in its offline version. The feasibility version of the problem was studied in [13]—to goal is to check whether it is possible to schedule all jobs of given input instance. They have shown a deterministic algorithm for the feasibility problem with time complexity $O(n \log n)$. The maximization version of the problem was studied in [11] and [4]—they have shown polynomial but very slow algorithm.

The following results are known for the topic of our interest—the online version of the problem:

- **Upper bound for deterministic algorithms**—In the paper [7] and [6] it was shown that the Greedy algorithm for the studied problem is 2–competitive. The shown result is more stronger—there was shown that any non–preemptive deterministic algorithm that never idles when jobs are available is also 2–competitive.

- **Lower bound for deterministic algorithms**—The fact that the Greedy algorithm is optimal was shown in [14], they have shown a lower bound 2 on the competitive ratio for the studied problem in the deterministic model.

- **Lower bound for randomized algorithms**—The lower bound $4/3 \approx 1.333$ on the competitive ratio for the studied problem in the randomized model was shown in [14].

- **Upper bound for deterministic algorithms on input instances with large slack**—The lower bound 2 on the competitive ratio for the problem in the deterministic model can be beaten when we require sufficiently large slack of jobs. In the paper [14] was shown a 1.5–competitive algorithm for input instances consisting of jobs with slack at least p, it means that each job j satisfies $d_j - r_j \geq 2 \cdot p$. An improvement of this result is presented in [15]—a $(1 + 1/\lambda)$–competitive algorithm for input instances consisting of jobs such that each job j satisfies $d_j - r_j \geq \lambda \cdot p$.

2.4.3 Our results

We have presented the following results for the studied problem in [Pub–1], these results are presented in Section 6.5:

- **Upper bound on deterministic algorithms allowed to restart jobs**—We have shown a 1.5–competitive algorithm for the studied problem such that it is an online scheduling algorithm for the problem which is allowed to restart jobs. Allowed restarts means that the algorithm is allowed to preempt running jobs and the preempted jobs can be completed later—the processing of the job cannot be continued but the job can be again processed from scratch. This result is presented in Theorem 6.5.2.
- **Lower bound on deterministic algorithms allowed to restart jobs**—We have shown a lower bound 1.5 on the competitive ratio of deterministic algorithms allowed to restart jobs. Thus our algorithm is optimal for the studied problem. This result is presented in Theorem 6.5.4.
- **Lower bound on randomized algorithms allowed to restart jobs**—We have shown a lower bound 1.2 on the competitive ratio of randomized algorithms allowed to restart jobs. This result is presented in Theorem 6.5.4.

2.4.4 Joint results

The following interesting results are the products of our joint research on the problem with other co–authors. These results were shown in paper [Pub–1].

- **Upper bound for randomized non–preemptive algorithms**—We present a barely random algorithm which uses only one random bit to choose between two deterministic non–preemptive algorithms. We show that this algorithm is $5/3 \approx 1.667$–competitive for the studied problem. This result is presented in Theorem 6.6.1.
- **Lower bound for barely random algorithms**—We show a lower bound on the competitive ratio of barely random algorithm such that randomly chooses between two deterministic algorithms. The competitive ratio of such an algorithm is at least 1.5. Moreover when the algorithm chooses between the deterministic algorithms with equal probability, then its competitive ratio is at least 1.6. These two results are presented in Theorem 6.6.5 and Theorem 6.6.6.

2.5 Overloaded systems

In this section we focus on preemptive online scheduling problems for overloaded systems. Recall that there are two types of task scheduling problems—first the minimization problems where we have to schedule all jobs and minimize objective like makespan and second the maximization problems where we do not need to schedule all jobs but our goal is to maximize our profit. This problem is about the second case—in the overloaded systems the number of jobs and their processing times exceeds the capacity of machines which we use to process these jobs and thus not all jobs can be completed.

2.5.1 Problem description

In the problem each job is specified by its release time, deadline, processing time and its weight which represents corresponding profit rate. We allow preemptions—the online algorithms are allowed to split each job into several pieces—arbitrary number of pieces with arbitrary granularity. There is only one machine, which can be used for processing of these jobs. The goal is to find schedule for 1 processor, that maximizes the total profit.

We studied the problem in two models with different measures—the standard model and the metered model. In the standard model we get the whole profit for completed tasks only. In the metered model we get the proportional part of the whole profit—given by the fraction of the execution time and processing time.

We study this problem in the deterministic model. For the analysis of the problem we used the methods of the competitive analysis including resource augmentation.

2.5.2 Previous results

Standard model

The problem in the standard profit model has been extensively studied. It was shown in [20] and [8] that there is no constant competitive online algorithm for the problem because the shown lower bound depends on so called importance factor—the ratio of jobs with maximum and minimum weights which is unbounded for general input instances. Since there is no constant competitive online algorithm it is natural to study the problem under the resource augmentation framework.

The following results related to the studied problem in the standard profit model were published:

- **Upper bound on the competitive ratio in the standard model**—The $(\sqrt{\xi}+1)^2$–competitive online algorithm for the studied problem, where $\xi = \max_j w_j / \min_j w_j$ is called the *importance factor*, was shown in [20].
- **Lower bound on the competitive ratio in the standard model—**It was shown in [20] and [8] that the algorithm showing the upper bound is optimal algorithm for the studied problem.
- **Upper bound on speed–up in the standard model**—The first online algorithm with constant competitive ratio with speed–up 32 for the problem in the standard model was shown in [17].
- **Upper bound on speed–up in the standard model for 1–competitiveness**—A 1–competitive online algorithm with speed–up $O(\xi)$ was shown in [22]. The parameter ξ is again the importance factor.
- **Upper bound on speed–up in the standard model for 1–competitiveness and tight jobs**—A 1–competitive online algorithm with speed–up $O(1)$ was shown in [19].

- **Lower bound on speed–up in the standard model for 1–competitiveness and tight jobs**—A lower bound $\phi \approx 1.618$ on the competitive ratio was shown in [21].

Metered model

The problem in the metered profit model has been studied in [9], however, in a different terminology. They studied the problem in the context of thinwire visualization—user is viewing a low–resolution image and uses cursor to generate requests for higher resolution at specified positions. The problem is overloaded because of limited network bandwidth (thinwire). This is a good example of practical application where the partial processing of jobs is benefical for the user.

The following results related to the studied problem in the metered profit model were published:

- **Upper and lower bound on** FIRSTFIT **and** ENDFIT **algorithms in the metered model**—In the paper [9] was shown that the online scheduling algorithms known as FIRSTFIT and ENDFIT are both 2–competitive.
- **Lower bound on the competitive ratio in the metered model**—A lower bound on the competitive ratio for the problem was shown in [9]—there is no online algorithm with better competitive ratio than $2(2-\sqrt{2}) \approx 1.17$.

2.5.3 Our results

We have shown the following result for the studied problem, which we has been published in [Pub–4] and which we present in this thesis in Section 5.4.2:

- **Lower bound on speed–up for tight jobs in the standard model**—We have shown that in the standard profit model there is no online 1–competitive algorithm with speed–up $s < 2$ for the problem with the input instances consisting of tight jobs only. Presented in Theorem 5.4.1.

2.5.4 Joint results

The following interesting results are the products of our joint research on the problem with other co–authors. These results were shown in paper [Pub–4] and also these results are mentioned in Section 5.4.3.

- **Upper bound in the metered model**—We have shown a $e/(e-1) \approx 1.58$–competitive algorithm for the problem in metered profit model. Presented in Theorem 5.4.2.
- **Lower bound in the metered model**—We have shown a lower bound $\sqrt{5}-1 \approx 1.236$ on the competitive ratio for the problem in metered profit model. Presented in Theorem 5.4.3.
- **Lower bound on speed–up in the metered model for 1–competitiveness**—We have shown that there is no 1–competitive online algorithm with speed–up better than $\Omega(\log\log\xi)$. Presented in Theorem 5.4.4.

- **Lower bound in standard for 1–competitiveness**—Another resource augmentation result for the model where the online algorithm is allowed to use more machines than the offline algorithm. We have shown that the competitive ratio of online algorithms is at least $\Omega(\sqrt[m]{\xi}/m)$ where m is number of machines used by the online algorithm even if all jobs are tight. Presented in Theorem 5.4.5.

As we have shown that the online algorithms in the standard model cannot achieve the competitive ratio 1 using constant number of machines but these algorithms can achieve the competitive ratio 1 using constant speed–up. Thus the speed–up is more powerful than increasing of number of machines.

2.6 Resource augmentation in online scheduling problems

In this section we consider online scheduling problems with resource augmentation. The resources can be augmented in various ways—usually in time, speed, number of processors or simply by breaking or weaking some other constraints.

The resource augmentation techniques we use when we study interesting or important problems and we are not satisfied with the competitive ratio of the studied problem. It can happen that the competitive ratio is unbounded for some scheduling problems—then we are searching for the "strong" constraint which is forcing the unboundness of the competitive ratio. Or the competitive ratio is too high—altrough we have proven that the problem is constant competitive but the competitive ratio is too big that it cannot be applied in practice. Especially in the practical applications we need online algorithms because of their simplicity and we require similar quality of results like it is provided by the offline algorithms. Thus we are only satisfied with competitive ratio really close to 1. Therefore it is necessary to break or make weaker some of not so important constraints. Then the result is usually an online algorithm applicable in practice with very good performance, but breaking some minor constraints.

In previous section on equal–length jobs we have presented an online scheduling problem with resource augmentation—augmentation of speed of processors used by the online algorithm. In this section we present the augmentation of deadlines—this means that the online algorithm has more time for processing given jobs.

2.6.1 Problem description

In this problem we study the influence of the resource augmentation on the competitive ratio of one of common scheduling problems - the problem with equal processing times of jobs. The problem is a single processor online scheduling problem and the problem is considered in the deterministic computational model. The time axis is assumed to be integral and jobs arrive one by one in time. In the problem each job is specified by its integral release time and integral deadline and non–negative weight. The processing times of all jobs are equal to a constant.

In this problem we allow augmentation of time—we break the constraints given by deadlines. We consider some constant k and the deadlines for the online algorithm are shifted by k time units to the future while the offline algorithm must process the jobs before their deadlines.

As it is usual in the problems with allowed resource augmentation we are interested in the influence of the resource augmentation on the competitive ratio of the problem.

2.6.2 Previous results

Very similar problem was studied in [1], which is almost the same problem but in different terminology. They studied the problem of packet buffering and the problem matches our problem for unbounded buffers.

- **Upper bound for greedy algorithm with k–time faster transmissions—** They shown that the greedy algorithm is $(1 + 1/k)$–competitive when such a resource augmentation of speed is considered.

2.6.3 Our results

We present the following results for the studied problem in Section 5.3:

- **Lower bound for 1–competitiveness—**We have shown that there is no online 1–competitive k–relaxed algorithm for any k. Presented in Theorem 5.3.2.
- **Lower bound on the competitive ratio for 1–relaxed—**We have shown a lower bound ≈ 1.05099 on the competitive ratio for 1–relaxed online algorithms for the studied problem. Presented in Theorem 5.3.3.
- **Lower bound on the competitive ratio—**We have shown a general lower bound $1 + \frac{1}{2^{k+1}(k^2+3k+5)}$ for the online k–relaxed algorithms for the problem. Presented in Theorem 5.3.4.

Bibliography

[1] S. Albers, M. Schmidt: *On the Performance of Greedy Algorithms in Packet Buffering..* SIAM, Journal of Compututing 35(2): 278–304, 2005.

[2] N. Andelman, Y. Mansour, A. Zhu: *Competitive Queuing Polices in QoS Switches.* Proceedings of 14th Symposium on Discrete Algorithms (SODA), pages 761–770. ACM/SIAM, 2003.

[3] N. Bansal, L. Fleischer, T. Kimbrel, M. Mahdian, B. Schieber, M. Sviridenko: *Further Improvements in Competitive Guarantees for QoS Buffering.* Proceedings of 31st International Colloquium on Automata, Languages and Programming (ICALP), volume 3142 of Lecture Notes in Computer Science, pages 196–207. Springer 2004.

[4] P. Baptiste: *Polynomial–Time Algorithms for Minimizing the Weighted Number of Late Jobs on a Single Machine with Equal Processing Times.* Journal of Scheduling, 2:245–252, 1999.

[5] Y. Bartal, F. Y. L. Chin, M. Chrobak, S. P. Y. Fung, W. Jawor, R. Lavi, J. Sgall, T. Tichý: *Online Competitive Algorithms for Maximizing Weighted Throughput of Unit Jobs.* Proceedings of the 21st Annual Symposium on Theoretical Aspects of Computer Science (STACS). Berlin, Springer–Verlag, pages 187–198, 2004.

[6] S. K. Baruah, J. Haritsa, N. Sharma: *Online Scheduling to Maximize Task Completions.* Journal of Combinatorial Mathematics and Combinatorial Computing, 39:65–78, 2001.

[7] S. K. Baruah, J. Haritsa, N. Sharma: *Online Scheduling to Maximize Task Completions.* Proceedings of Real–Time Systems Symposium (RTSS), pages 228–236, 1994.

[8] S. Baruah, G. Koren, D. Mao, B. Mishra, A. Raghunathan, L. Rosier, D. Shasha, F. Wang: *On the competitiveness of on–line real–time task scheduling..* Real–Time Systems, 4:125—144, 1992.

[9] E. C. Chang, C. Yap: *Competitive online scheduling with level of service.* Proceedings of 7th Annual International Computing and Combinatorics Conference, volume 2108 of Lecture Notes in Compoter Science, pages 453—462, 2001.

[10] F. Y. L. Chin, S. P. Y. Fung: *Online Scheduling for Partial Job Values: Does Timesharing or Randomization Help?.* Algorithmica, 37:149–164, 2003.

[11] M. Chrobak, C. Dürr, W. Jawor, L. Kowalik, M. Kurowski: *A Note on Scheduling Equal–Length Jobs to Maximize Throughput.* Manuscript, 2004.

[12] M. Englert, M. Westermann: *Considering Suppressed Packets Improves Buffer Management in QoS Switches.* Proceedings of 18th Symposium on Discrete Algorithms (SODA), ACM/SIAM, 2007.

[13] M. Garey, D. Johnson, B. Simons, R. Tarjan: *Scheduling Unit–Time Tasks with Arbitrary Release Times and Deadlines.* SIAM, Journal of Computing, 10(2):256–269, 1981.

[14] S. A. Goldman, J. Parwatikar, S. Suri: *Online Scheduling with Hard Deadlines.* Journal of Algorithms, 34:370–389, 2000.

[15] M. H. Goldwasser: *Patience Is a Virtue: The Effect of Slack on the Competitiveness for for admission control.* Journal of Scheduling, 6:183–211, 2003.

[16] R. L. Graham: *Bounds for Certain Multiprocessing Anomalies.* Bell System Technical Journal 45,1563–1581, 1966.

[17] B. Kalyanasundaram, K. Pruhs: *Speed Is as Powerful as Clairvoyance.* Journal of the ACM, 47(4):214—221, 2000.

[18] A. Kesselman, Z. Lotker, Y. Mansour, B. Patt–Shamir, B. Schieber, M. Sviridenko: *Buffer Overflow Management in QoS Switches.* SIAM, Journal of Computing (SICOMP), pages 33:563–583. 2004.

[19] C. K. Koo, T. W. Lam, T. W. Ngan, K. K. To: *Online scheduling with tight deadlines.* Proceedings of 26th Mathematical Foundations of Computer Science, volume 2136 of Lecture Notes in Compoter Science, pages 464—473, 2001.

[20] G. Koren, D. Shasha: *d^{over}: an Optimal On–Line Scheduling Algorithm for Overloaded Uniprocessor Real–Time Systems.* SIAM, Journal on Computing, 24:318—339, 1995.

[21] T. W. Lam, T. W. Ngan, K. K To: *On the speed requirement for optimal deadline scheduling in overloaded systems..* Proceedings of 15th International Parallel and Distributed Processing Symposium, pages 202, 2001.

[22] T. W. Lam, K. K To: *Trade–offs between speed and processor in hard–deadline scheduling.* Proceedings of 10th Symposium on Discrete Algorithms (SODA), pages 755–764, 1999.

[23] F. Li, J. Sethuraman, C. Stein: *Better Online Buffer Management.* Proceedings of 18th Symposium on Discrete Algorithms (SODA), ACM/SIAM, 2007.

[24] D. D. Sleator, R. E. Tarjan: *Amortized Efficiency of List Update and Paging Rules.* Communications of the ACM, 28:202–208, 1985.

List of publications

[Pub–1] M. Chrobak, W. Jawor, J. Sgall, T. Tichý: *Online Scheduling of Equal–Length Jobs: Randomization and Restarts Help.* SIAM Journal of Computing, 36(6):1709–1728, 2007. A preliminary version appeared in Proceedings of the 31st International Colloquium on Automata, Languages, and Programming (ICALP). Berlin, Springer 358–370, 2004.

[Pub–2] F. Y. L. Chin, M. Chrobak, S. P. Y. Fung, W. Jawor, J. Sgall, T. Tichý: *Online Competitive Algorithms for Maximizing Weighted Throughput of Unit Jobs.* Journal of Discrete Algorithms, 4(2):255–276, 2006..

[Pub–3] Y. Bartal, F. Y. L. Chin, M. Chrobak, S. P. Y. Fung, W. Jawor, R. Lavi, J. Sgall, T. Tichý: *Online Competitive Algorithms for Maximizing Weighted Throughput of Unit Jobs.* Proceedings of the 21st Annual Symposium on Theoretical Aspects of Computer Science (STACS). Berlin, Springer–Verlag 187–198, 2004.

[Pub–4] M. Chrobak, L. Epstein, J. Noga, J. Sgall, R. van Stee, T. Tichý, N. Vakhania: *Preemptive Scheduling in Overloaded Systems.* Journal of Computer and System Sciences, 67 (1) 183–197, 2003; Proceedings of the 29th International Colloquium on Automata, Languages, and Programming. Berlin, Springer–Verlag 800–811, 2002.

[Pub–5] M. Chrobak, W. Jawor, J. Sgall, T. Tichý: *Improved Online Algorithms for Buffer Management in QoS Switches.* ACM Transactions on Algorithms, 4(3) article 50, 2007.

Other publications

[Pub–6] D. Král', J. Sgall, T. Tichý: *Randomized Strategies for the Plurality Problem.* To appear in Discrete Applied Mathematics. KAM–DIMATIA Series 2005–722 and ITI Series 2005–238, Charles University, Prague, 2005.

[Pub–7] T. Tichý: *Randomized Online Scheduling on 3 Processors.* Operation Research Letters, 32 (2) 152–158, 2004.

[Pub–8] D. Král', V. Majerech, J. Sgall, T. Tichý, G. Woeginger: *It Is Tough to Be a Plumber.* Theoretical Computer Science, 313 (3) 473–484, 2004.

[Pub–9] T. Tichý: *Multiprocessor Randomized Online Scheduling.* ITI Series, 2002–069, Charles University, Prague, 2002.

[Pub–10] T. Tichý: *A Lower Bound for Restricted Randomized Online Algorithms for Scheduling.* Proceedings of the Week for Doctoral Students 2002. Praha, MATFYZPRESS 21–26, 2002.

List of citations

[1] I. Yahav, A. Gal, N. Larson: *Bid–based approach for pricing Web service.* volume 4275 of Lecture Notes in Computer Science, pages 360–376. Springer, 2006; refers to [Pub–5].

[2] F.F. Zheng, Y.F. Xu, E. Zhang: *On–line production order scheduling with preemption penalties.* volume 13 of Journal of combinatorial optimization, pages 189–204, 2007; refers to [Pub–1].

[3] T. Ebenlendr, W. Jawor, J. Sgall: *Preemptive online scheduling: Optimal algorithms for all speeds.* Proceeding of ESA 2006, volume 4168 of Lecture Notes in Computer Science, pages 327–339. Springer, 2006; refers to [Pub–2].

[4] J.H. Ding, G.C. Zhang: *Online scheduling with hard deadlines on parallel machines.* volume 4041 of Lecture Notes in Computer Science, pages 32–42. Springer, 2006; refers to [Pub–1].

[5] Y. Azar, N. Levy: *Multiplexing packets with arbitrary deadlines in bounded buffers.* Proceedings of SWAT 2006, volume 4059 of Lecture Notes in Computer Science, pages 5–16. Springer, 2006; refers to [Pub–2].

[6] M.H. Goldwasser, M. Pedigo: *Online, non–preemptive scheduling of equal–length jobs on two identical machines.* Proceedings of SWAT 2006, volume 4059 of Lecture Notes in Computer Science, pages 113–123. Springer, 2006; refers to [Pub–1].

[7] P. Damaschke: *Scheduling search procedures: The wheel of fortune.* volume 9 of Journal of Scheduling, pages 545–557, 2006; refers to [Pub–4].

[8] S.P.Y. Fung, R.Y.L. Chin, C.K. Poon: *Laxity helps in broadcast scheduling.* Proceedings of TCS'2005, volume 3701 of Lecture Notes in Computer Science, pages 251–264, Springer, 2005; refers to [Pub–1] and [Pub–2].

[9] S.P.Y. Fung, F.Y.L. Chin, H. Shen: *Online scheduling of unit jobs with bounded importance ratio.* International Journal of Foundations of Computer Science, volume 16, pages 581–598, 2005; refers to [Pub–3].

[10] H.L. Chan, T.W. Lam, K.K. To: *Nonmigratory online deadline scheduling on multiprocessors.* Journal on Computing, SIAM'2005, volume 34, pages 669–682, 2005; refers to [Pub–4].

[11] Y. Takenaga, T. Walsh: *Tetravex is NP–complete.* Information Processing Letters, volume 99 (5), pages 171–174, 2006; refers to [Pub–8].

[12] F.Y.L. Chin, S.P.Y. Fung: *Improved competitive algorithms for online scheduling with partial job values.* TCS'2004, volume 325 (3), pages 467–478, 2004; refers to [Pub–8].

[13] C. Koo, T. Lam, T. Ngan, K. To: *Extra Processors versus Future Information in Optimal Deadline Scheduling.* In Proc. SPAA, pages 133–142, 2002; refers to [Pub–4].

[14] T. Lam, T. Ngan, K. To: *Performance Guarantee of EDF under Overload.* Journal of Algorithhm, 52(2), pp. 193–206, 2004; refers to [Pub–4].

[15] T. Lam, T. Ngan: *Aggressive Online Deadline Scheduling.* Electronic Notes in Theoretical Computer Science, 91, pages 148–157, 2004; refers to [Pub–4].

[16] R. Lavi, N. Nissan: *Online ascending auctions for gradually expiring items.* In Proc. 16th ACM–SIAM symposium on Discrete algorithms, pages 1146–1155, 2005; refers to [Pub–3].

[17] K. Pruhs, E. Torng, J. Sgall: *Online scheduling.* In Handbook of Scheduling: Algorithms, Models, and Performance Analysis, ed. Joseph Y.–T. Leung, chapter 15, pages 15–1—15–41, CRC Press, 2004; refers to [Pub–7].

[18] J. Ding, T. Ebenlendr, J. Sgall, G. Zhang: *Online scheduling of equal–length jobs on parallel machines.* In Proc. of the 15th European Symp. on Algorithms (ESA), Lecture Notes in Comput. Sci. 4698, pages 427–438. Springer, 2007; refers to [Pub–1].