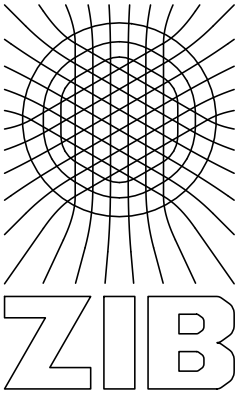
Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany



BENJAMIN HILLER TJARK VREDEVELD

Probabilistic alternatives for competitive analysis

ZIB-Report 11-55 (March 2011. This version will appear in Computer Science – Research and Development.)

Herausgegeben vom
Konrad-Zuse-Zentrum für Informationstechnik Berlin
Takustraße 7
D-14195 Berlin-Dahlem

Telefon: 030-84185-0
Telefax: 030-84185-125

e-mail: bibliothek@zib.de
URL: http://www.zib.de

ZIB-Report (Print) ISSN 1438-0064
ZIB-Report (Internet) ISSN 2192-7782

Probabilistic alternatives for competitive analysis

Benjamin Hiller[1] and Tjark Vredeveld[2]

[1]Zuse Institute Berlin, Takustraße 7, 14195 Berlin, Germany
[2]Maastricht University, Dept. of Quantitative Economics, P.O.Box 616, 6200 MD Maastricht, The Netherlands

April 19, 2012

Abstract

In the last 20 years competitive analysis has become the main tool for analyzing the quality of online algorithms. Despite of this, competitive analysis has also been criticized: It sometimes cannot discriminate between algorithms that exhibit significantly different empirical behavior, or it even favors an algorithm that is worse from an empirical point of view. Therefore, there have been several approaches to circumvent these drawbacks. In this survey, we discuss probabilistic alternatives for competitive analysis.

1 Introduction

Classical optimization techniques assume knowledge of all the data relevant to solve the optimization problem. However, this assumption is not always realistic, since quite often decisions need to be made without having access to the complete data. Such problems are called *online optimization problems*. The major feature of these problems is that the input is only revealed over time, while an *online algorithm* has to take decisions that cannot be revoked once more information becomes known. The following three problems are classical examples. We will use these problems to illustrate different kinds of analysis that have been developed for online algorithms.

Paging This problem models an optimization problem occuring in a two-level memory system. The *slow memory* stores a fixed set M of pages. To speed up access to the pages, up to k pages can be put in the *fast memory* or *cache*. The task is to serve a sequence $\sigma \in M^n$ of n page requests. In order to serve a page request $p \in M$, the page has to be in the cache, otherwise a *page fault* occurs. The requested page must then be loaded into the cache, and if the cache contains k pages, at least one page must be evicted from the cache. An online paging algorithm needs to decide which page(s) will be evicted from the cache on a page fault without knowing the remaining request sequence or its length. The goal is to minimize the number of page faults.

Bin Packing There is an (unlimited) number of unit capacity bins and a sequence of items, described by their item sizes $s_1, \ldots, s_n$. The task is to pack each item into a bin that still has sufficient capacity. Once an item has been packed in one bin, it is not possible to repack it into another bin. An online algorithm has to determine the bin for each item without knowing the number of items to come and their sizes. The goal is to minimize the total number of bins used.

Scheduling In machine scheduling, there are n jobs, each of which needs to be processed on one of m machines. A machine can process at most one job at a time, and each job can be processed by at most one machine at a time. Moreover, it cannot be processed before its release date. Processing a job j takes p_j amounts of time, and we consider both a model in which the processing may be interrupted (preemptive model) and one in which a job needs to be processed until completion once started (non-preemptive model). The goal is to schedule the jobs on the machines in such a way that a certain objective function is minimized. In this paper, we consider the total weighted completion time and the total flow-time. The completion time of job j, C_j, is the earliest point in time at which a job has received p_j units of processing time, whereas the flow-time, F_j, is the time a job is in the system, i. e., its completion time minus its release date. The total weighted completion time is then $\sum_j w_j C_j$ and the total flow time is $\sum_j F_j$.

In addition to the uncertainty about the future, there may be another type of uncertainty for online scheduling problems: Even if a job is known, its processing time may still be unknown until it has completed. We refer to this as non-clairvoyant scheduling.

When designing an online algorithm, one faces the question of how to measure the quality of the solution obtained by this algorithm. By far the most often used measure is *competitive analysis* [46], in which an online algorithm is compared to an optimal offline algorithm that knows the entire input in advance. In some cases, competitive analysis does not provide satisfying answers, e. g., it sometimes cannot discriminate between algorithms that exhibit significantly different empirical performance. The approach of competitive analysis and some of the criticism voiced against it are discussed in Section 2.

There are several different approaches to circumvent these drawbacks. In this survey, we focus on probabilistic approaches to analyze online algorithms to obtain more detailed results than possible with competitive analysis. Section 3 gives an overview on these approaches and shows how they help to get improved results for the three problems introduced above.

2 Competitive analysis and its variants

The standard theoretical tool used to assess online algorithms is *competitive analysis*, which is widely used. Probably the first time competitive analysis has been performed was in 1966 by Graham [22]. However, he did not mention the name competitive analysis nor online optimization. Only with the seminal paper of Sleator and Tarjan [46] competitive analysis became the standard yardstick. The term competitive analysis was first proposed by Karlin et al. [30].

The basic idea is to evaluate the "loss" in solution quality due to lack of information in the worst case. Competitive analysis compares the performance of a given online algorithm ALG to that of an algorithm that knows the complete request sequence in advance and can serve the requests at minimum cost. This benchmark algorithm is called *optimal offline algorithm* OPT. For a request sequence σ, we denote the corresponding optimal offline cost and the cost of ALG by $\mathrm{OPT}(\sigma)$ and $\mathrm{ALG}(\sigma)$, respectively. The algorithm ALG is said to be c-competitive for $c \geq 1$ and some $b \geq 0$, if

$$\mathrm{ALG}(\sigma) \leq c \cdot \mathrm{OPT}(\sigma) + b \tag{1}$$

for all request sequences σ. The *competitive ratio* of ALG is the smallest value c such that ALG is c-competitive. ALG is called *competitive* if ALG is c-competitive for some constant $c \geq 1$ that does not depend on the length of the sequence. Frequently, only the case $b = 0$ is considered, which makes the results less asymptotic.

Ideally, competitive analysis helps to identify and to design algorithms that offer good performance both in theory and in practice. Clearly, an algorithm that is c-competitive for a small constant c is a good algorithm from a theoretical point of view; there are many cases where such results can be shown. However, there are also online optimization problems for which *all* algorithms have a competitive ratio depending on some problem-specific parameter that can be arbitrarily large. This is due to the worst-case nature of competitive analyis: The competitive ratio is large, if there is a single instance on which an online algorithm is much worse than the optimal offline algorithm. Even if competitiveness results with small constants can be shown, it happens that algorithms with the same competitive ratio have perform different empirically. We will highlight these issues for our three example problems.

Paging Well-known paging algorithms are flush-when-full (FWF), which evicts every page on a page fault, first-in-first-out (FIFO), which evicts the page that has been in the cache the longest, and least-recently-used (LRU), which evicts the page whose most recent request was earliest.

Sleator and Tarjan [46] showed that the competitive ratio of every online algorithm is at least k. They also proved that both LRU and FIFO attain this ratio and are thus optimal, whereas some other algorithms are not k-competitive. Later it was shown that any online algorithm that is a marking algorithm is k-competitive [50], which includes FWF and LRU.

These results did not match empirical observations, see e. g., [55, 1, 39]: LRU was observed to be much better than FWF and to outperform FIFO. Moreover, the "empirical" competitive ratio of LRU was much smaller than k.

Bin Packing Probably the simplest online algorithm for bin packing is NextFit, which looks only at the most recently opened bin. If the next item fits, it is put into this bin; if not, the item is put in a new bin and the other bin is never considered again. It was shown by Johnson [27] that the competitive ratio of NextFit is 2. The algorithm FirstFit scans through the bins in their opening order and puts the item in the first bin with sufficient capacity. If no bin is found, a new bin is opened to accomodate the item. BestFit works similar but puts the item in the open bin with least remaining capacity that suffices for

the item. Both FirstFit and BestFit have the small competitive ratio 1.7 [28], but BestFit gives better results in practice. Yao [54] proposed RefinedFirstFit, which achieves a competitive ratio of 5/3 based on a classification of items and bins according to sizes / remaining capacity. The Harmonic algorithm by Lee and Lee [35] classifies items by size and puts different items into the same bin if and only if they belong to the same class. Successively, this idea was pushed further and the best known algorithm so far achieves competitive ratio not more than 1.589 [41]. Van Vliet [51] proved a lower bound of 1.540 for the competitive ratio of any online algorithm.

Csirik and Johnson [19] present experimental results indicating that on average, BestFit performes significantly better than FirstFit. Moreover, BestFit also outperforms the Harmonic algorithm and achieves an empirical competitive ratio better than 1.01.

Scheduling On a single machine, the problem of minimizing the flow-time in the preemptive model can be solved optimally by an online algorithm known as Shortest Remaining Processing Time [43]. This algorithm processes at any time the unfinished job that has smallest remaining processing time. On the other hand, in non-clairvoyant scheduling, in which nothing is known about a job's processing time until the job is completed, it can be shown that any deterministic online algorithm has a competitive ratio of $\Omega(n^{1/3})$ or $\Omega(P)$ [38], where n is the number of jobs and P is the ratio between the maximum and minimum processing time. For the non-preemptive problem on a single machine to minimize the total weighted completion time there is a 2-competitive online algorithm [2], which is the best possible [26]. For multiple machines, the best known online algorithm achieves a competitive ratio of 2.62 [18], whereas no online algorithm can be better than 1.309-competitive [52]. For an overview on other results in online scheduling we refer to [40].

Competitive analysis of online algorithms can be viewed as a two player game. The *online player*, i. e., the online algorithm, tries to achieve a small competitive ratio, whereas the other player, the *adversary*, tries to generate the request sequence such that he can serve it with small cost, but incurring high cost for the online player, which gives a large competitive ratio.

Basically there are three approaches to remedy the drawbacks of competitive analysis in order to get more realistic and useful results.

1. Keep the idea of comparing to the offline optimum, but reduce the power of the adversary online algorithms are compared to.

2. Compare online algorithms to a weaker optimum, and use this to compare the algorithms.

3. Compare online algorithms directly instead of indirectly via the offline optimum.

The first approach is the most prominent one. An obvious way to reduce the power of the adversary is to restrict the set of request sequences it may generate. This problem-specific technique is quite successful. Examples for paging are the restriction to sequences that exhibit locality of reference [1, 50] or that are derived from an access graph [11]. Another possibility is to allow the online algorithm to use more resources (e. g., faster machines) than the adversary [29];

this is known as resource augmentation. Many more deterministic models have been proposed; see e. g., see the article by Souza [47] in this issue for an overview on the paging problem.

The second and third approach are less common; some results of this type will be discussed in Section 3.5. From now on, we focus on probabilistic measures for online algorithms.

3 Alternative probabilistic measures

Randomness can be used in two ways in the analysis of online algorithms. First, the request sequence may be assumed to be drawn from some probability distribution. This is equivalent to replacing the adversary by some random process, which allows to study the "typical" or "average" behavior instead of a worst case. Second, the online algorithms themselves may be randomized, which makes it harder for the adversary to come up with bad sequences.

3.1 Randomized Online Algorithms

Using randomization to improve online algorithms has been suggested by Borodin et al. [12]. Usually, analyses are done with respect to an *oblivious adversary*, i. e., this adversary knows the probability distribution employed by the randomized online algorithm but not the random outcomes. The competitive ratio is then determined by replacing ALG by its expectation $\mathbb{E}[\mathrm{ALG}]$ in Equation (1).

Paging For the paging problem, Fiat et al. [20] showed a lower bound of H_k (H_k is the kth harmonic number) on the competitive ratio of every randomized paging algorithm. Moreover, they provided a randomized paging algorithm that is $2H_k$-competitive against the oblivious adversary. McGeoch and Sleator [36] presented an optimal randomized algorithm with competitive ratio H_k. Since $H_k \in \mathcal{O}(\log k)$, both results show that randomized paging algorithms can achieve a significantly better competitive ratio than deterministic ones. Similar improvements have been achieved for many online optimization problems.

Bin Packing Interestingly, randomized algorithms for bin packing can only be slightly better than deterministic ones. Chandra [13] proved a lower bound of 1.536 for the competitive ratio of any randomized online algorithm, whereas the best-known deterministic algorithms achieves 1.589.

Scheduling Becchetti and Leonardi [7] proposed a randomized online algorithm that achieves a competitive ratio of $\mathcal{O}(\log n)$ for the non-clairvoyant single machine scheduling problem of minimizing the total flow-time, which matches the lower bound of [38]. For the nonpreemptive problem of minimizing the total weighted completion time on multiple machines no randomized online algorithm can be better than 1.157 [44], whereas Correa and Wagner [18] provided a randomized online algorithm with competitive ratio strictly less than 2, which is the optimal competitive ratio for deterministic algorithms.

3.2 Average-case competitive analysis

Probably the first alternative for worst case analysis that comes to mind is *average case analysis*. In average case analysis, the request sequence is chosen according to some probability distribution. In most average case competitive analyses, the requests are chosen independently and identically distributed. In this context, the optimal value for a request sequence of length n, OPT_n, becomes a random variable, just as ALG_n, the value obtained by an (online) algorithm. The expected competitive ratio is defined as

$$R_{\mathrm{ALG}}(n) = \mathbb{E}\left[\frac{\mathrm{ALG}_n}{\mathrm{OPT}_n}\right].$$

An alternative measure for the average case performance would be the ratio of the expectations, i. e., the ratio $\mathbb{E}[\mathrm{ALG}_n]/\mathbb{E}[\mathrm{OPT}_n]$. This is the performance measure that is common to evaluate approximative stochastic scheduling policies, see, e. g., [37]. For an overview of results in stochastic (online) scheduling, we refer to the survey paper [53] which also appeared in this special issue. Scharbrodt et al. [42] and Souza and Steger [48] elaborate on the difference between the two measures. Although the ratio of the expectations is normally easier to compute, they prefer the expected competitive ratio for several reasons. First, they say that the ratio of expectations favours algorithms that perform well on instances for which the optimal solution value is large, due to the fact that instances with small solution value contribute little to the expected value of an algorithm. On the other hand, the expectation of the ratio compares the algorithm to the optimum on each instance separately and favours algorithms that perform well on many instances. Furthermore, by Markov's inequality, one can easily derive bounds on the probability that for a randomly generated instance the ratio $\mathrm{ALG}_n/\mathrm{OPT}_n$ is more than a certain factor away from the expected competitive ratio.

Bin Packing There is a vast literature on average case analysis of bin packing algorithms (see e. g., [15, 14] for surveys). Many results for bin packing focus on the asymptotic expected competitive ratio of ALG, i. e.,

$$R^{\infty}_{\mathrm{ALG}}(F) := \lim_{n\to\infty} R_{\mathrm{ALG}}(n),$$

assuming that the input sequence is generated by choosing each item size i. i. d. from distribution F. In most cases, F is a uniform distribution, e. g., $U[0,1]$, the uniform distribution on $[0,1]$. The first precise average case asymptotic analysis was given by Coffman et al. [17] for the Next Fit algorithm. They showed that the asymptotic expected competitive ratio for Next Fit is $R^{\infty}_{\mathrm{NF}}(U[0,1]) = 4/3$ (the deterministic competitive ratio is 2). Lee and Lee [35] proved that their Harmonic algorithm achieves $R^{\infty}_{\mathrm{H}}(U[0,1]) < 1.306$, which is better than the deterministic lower bound of 1.540. Bentley et al. [10] were able to show, somewhat suprisingly, $R^{\infty}_{\mathrm{FF}}(U[0,1]) = 1$, i. e., that FirstFit is asymptotically optimal. The result was surprising since the empirical value is significantly larger than 1 in simulations, even for large numbers of items. This motivated the study of the *expected waste after n items*, i. e., the expected difference between the number of bins used by ALG and the total size of n items. Analyzing the waste, Shor [45] was able to separate the performance of FirstFit and BestFit: While

the expected waste of FirstFit grows as $\Omega(n^{2/3})$, BestFit has expected waste $\Theta(\sqrt{n}\log^{3/4} n)$. Both FirstFit and BestFit are thus asymptotically optimal, but the performance of BestFit converges faster.

Scheduling In the area of scheduling, the studies that make an average case analysis assume that the processing times are independent random variables whereas all other data may be given by a (deterministic) adversary. The online aspect is that the algorithm does not know whether and/or when a new job arrives. The expected competitive ratio has been studied by Coffman and Gilbert in 1985 [16], which appears to be the first result on non-asymptotic expected competitive ratio. They consider list scheduling algorithms for minimizing the makespan, i. e., the completion time of the entire schedule, on identical machines when processing times are independent, identically distributed from the uniform or exponential distribution. Scharbrodt et al. [42] considered the scheduling problem on identical machines so as to minimize the total completion time. They performed an expected competitive ratio analysis for the SEPT list scheduling algorithm that schedules jobs according to nondecreasing expected processing time. Note that SEPT is only a semi-online algorithm as the jobs need to arrive in a specified order. They showed that SEPT has a constant expected competitive ratio whenever the processing times are exponentially distributed (not necessarily i. i. d.). Their analysis extends to a more general class of distributions. In the deterministic case, if the jobs are ordered in non-decreasing *realized* processing times, then Kawaguchi and Kyan [32] showed that the worst case performance ratio is bounded by $(1+\sqrt{2})/2$. This worst case bound actually holds also for the weighted version of the problem, whenever the jobs are ordered in nondecreasing realized processing time over weight. Souza and Steger [48] extended the result of Scharbrodt et al. to the weighted version and showed that the WSEPT rule admits a constant expected competitive ratio whenever the jobs are identically distributed, satisfying some additional probabilistic property.

3.3 Diffuse adversaries

As competitive analysis is often criticized for being too pessimistic due to its worst case character, average case analysis is often considered to be too optimistic. Moreover, in many cases the probability distributions analyzed are quite special and/or realistic probabilistic models cannot be analyzed or are not available.

In order to address these issues and to improve upon standard competitive analysis, Koutsoupias and Papadimitriou [33] proposed the *diffuse adversary model*. In this model, the offline adversary is replaced by the *diffuse adversary*, which chooses a probability distribution D out of a family of distributions Δ. The class Δ of distributions may be used to express some structural property of the inputs without sticking to a certain distribution. An algorithm ALG is called c-competitive against class Δ of request sequence distributions if there is a $b \geq 0$ such that

$$\mathbb{E}_D\left[\mathrm{ALG}(\sigma)\right] \leq c \cdot \mathbb{E}_D\left[\mathrm{OPT}(\sigma)\right] + b \tag{2}$$

for all $D \in \Delta$, where the request sequence σ is drawn according to D. Note that

this definition generalizes both competitive analysis and average case analysis for the ratio of expectations.

Paging To apply this approach to the paging problem, Koutsoupias and Papadimitriou proposed a class of distributions Δ_ϵ. This class contains all distributions D such that the conditional probability $D[p|\sigma]$ of page p being requested after request sequence σ satisfies $D[p|\sigma] \leq \epsilon$ for all p and σ. Clearly, a small ϵ limits the power of the adversary since he has less control over the next request (note that Δ_1 is equivalent to all request distributions). Koutsoupias and Papadimitriou showed that LRU attains the optimal competitive ratio against Δ_ϵ for any $\epsilon \in [0, 1]$. However, they could not determine this ratio. In subsequent work, Young [56] gave lower and upper bounds for the optimal competitive ratio in terms of a function $\psi(\epsilon, k)$ that match up to a factor of two. The function $\psi(\epsilon, k)$ exhibits the following threshold behavior around $\epsilon = 1/k$:

$$\psi(\epsilon, k) \text{ is } \begin{cases} \leq 1 + \ln \frac{1}{\delta} & \epsilon = (1-\delta)/k, \\ \approx \ln k & \epsilon = 1/k, \\ \leq k \frac{\delta}{1+\delta} & \epsilon = (1+\delta)/k. \end{cases}$$

Thus the optimal competitive ratio (that of LRU) is constant for small ϵ and almost k for large ϵ. This bound holds for randomized algorithms, too, except for the case $\epsilon \geq 1/k$ where both the lower and upper bounds are $\mathcal{O}(\log k)$. Moreover, Young showed that FIFO and FWF have competitive ratio k for $\epsilon \geq 1/k$. Hence, these results generalize the standard competitive analysis results and those for randomized algorithms. They are able to discriminate between FWF or FIFO and LRU.

Becchetti [6] proposed a different diffuse adversary for the paging problem. The class Δ comprises distributions D whose conditional distributions $D[p|\sigma]$ favor pages p that are more recent w. r. t. σ in order to model the locality of reference often encountered in practice. He then showed that LRU achieves a constant competitive ratio against Δ, whereas that of FWF is $\Omega(k)$ if locality of reference is high.

Equation (2) defines the competitive ratio against a diffuse adversary essentially as a ratio of expectations. As for average-case analysis, it is also possible to consider the expecation of the ratio between online algorithm performance and the optimum offline cost. Panagiotou and Souza [39] consider a diffuse adversary which chooses a sequence, but the cache size is chosen at random without control of the adversary. They show that for this adversary, the ratio of expections approach as in Equation (2) gives misleading results, whereas the expectation of the ratios provides the correct picture.

3.4 Smoothed competitive analysis

An alternative compromise between worst case and average case analysis is to consider *smoothed* inputs. The notion of *smoothed complexity* was introduced by Spielman and Teng [49] in an attempt to explain the success of algorithms that are known to work well in practice while having a poor worst case performance. It can be seen as a hybrid between worst case and average case complexity. The basic idea is to randomly perturb the initial input instances and to analyze the expected performance of the algorithm on the perturbed instances. Becchetti et

al. [8] extended the idea of smoothed complexity to *smoothed competitive ratio*. Following the idea of Spielman and Teng, they smoothed the input instance according to some probability distribution f. Given an input instance $\bar{I}$, let $N(\bar{I})$ denote the set of instances that can be obtained by smoothing the input instance $\bar{I}$ according to f. The *smoothed competitive ratio* is defined as

$$\sup_{\bar{I}} \mathbb{E}_{I \xleftarrow{f} N(\bar{I})} \left[\frac{\mathrm{ALG}(I)}{\mathrm{OPT}(I)} \right],$$

where the supremum is taken over all input instances $\bar{I}$, and the expectation is taken according to f over all instances I in $N(\bar{I})$.

The adversarial input instance may be smoothed according to different smoothing models. The first class of models are symmetric models, which smooth a value by adding a random value. The distribution that is used to draw this random value from has mean 0 and is symmetric around this mean. The length of the interval from which the random value is taken may or may not depend on the original value.

The *partial bit randomization model*, introduced by [5, 9], is particularly useful if the input instance consists of K-bit integers. In this model the input values are not smoothed symmetrically. Assuming that each $\bar{x}_j$ is a K-bit integer, we perturb each $\bar{x}_j$ by replacing the k least significant bits, for $0 \leq k \leq K$, with some random number. Let f be a probablility distribution over the interval $[0, 2^k - 1]$. Then we define the smoothed value as

$$x_j = 2^k \left\lfloor \frac{\bar{x}_j}{2^k} \right\rfloor + \varepsilon, \quad \text{where } \varepsilon \to f.$$

For $k = 0$, the smoothed values are equal to the initial values, whereas for $k = K$ the smoothed values are independent identically distributed from $[0, 2^K - 1]$.

Scheduling Becchetti et al. [8] considered the so-called multilevel feedback (MLF) algorithm for preemptive scheduling on a single machine so as to minimize the total flow-time. When all processing times are between 1 and 2^K, any nonclairvoyant deterministic online algorithm has a competitive ratio of $\Omega(2^K)$ and $\Omega(n^{1/3})$. Becchetti et al. showed that when the processing times are smoothed according to the partial bit randomization model using a probability distribution with standard deviation σ, the smoothed competitive ratio of MLF is $\mathcal{O}\left((2^k/\sigma)^3 + (2^k/\sigma)^2 2^{K-k}\right)$. Whenever f is the uniform distribution over $[0, 2^k - 1]$, this simplifies to $\mathcal{O}\left(2^{K-k}\right)$. Becchetti et al. also showed that for the other smoothing models MLF has a lower bound of $\Omega(2^K)$ on the smoothed competitive ratio.

3.5 Other measures

As mentioned before, one weakness of competitive analysis is the comparison to the offline optimum, which is due to the fact that in a deterministic setting there is no reasonable concept of "optimal online algorithm" that can serve as a yardstick. In contrast, such an optimal online algorithm can be defined if the input is generated by some random process.

This approach has been applied in the analysis of paging algorithms even before the advent of competitive analysis. Franaszek and Wagner [21] studied

the paging problem with request sequences generated according to the independent reference model. In this model, each request is generated independently and identically from the same fixed page distribution. They considered the *page fault rate*, i. e., the (asymptotic) expected number of page faults per time unit, instead of the number of page faults as their performance measure for paging algorithms. It was shown that no paging algorithm from a certain class of algorithms including LRU and FIFO achieves a page fault rate that is at most a constant factor larger than the optimal one. However, it turns out that if LRU is allowed to use (slightly) larger cache than the optimal online algorithm, the ratio of the page fault rates becomes bounded, which is not true for FIFO.

Karlin et al. [31] generalized this approach by considering a request sequence generated by a fixed Markov chain, which is a probabilistic version of the access graph model. Using the theory of Markov decision processes, they were able to characterize the optimal online algorithm for the given Markov chain, which is a deterministic algorithm whose decisions depend only on the current request and the state of the cache. They showed that there are Markov chains such that all marking algorithms exhibit a page fault rate that is $\Omega(k)$ times the optimal one. Surprisingly, this includes LRU, which performs well under deterministic locality of reference models. The authors describe a polynomial-time algorithm whose page fault rate is not more than a constant times the optimal one. In contrast to most paging algorithms, this algorithm is not independent of the input but depends crucially on the Markov chain generating the request sequence.

All the measures discussed so far use a single number to evaluate the performance of an algorithm, e. g., the maximum or the average. It may be more descriptive to look at the *distribution* of the performance of an algorithm for various inputs to assess the algorithm.

This approach was successfully applied by Hiller and Vredeveld [25] to the analysis of bin coloring algorithms. For the bin coloring problem [34], the input consists of a sequence of unit-sized items, each of which has one of C colors. These items need to be packed sequentially into one of m initially empty bins of capacity B. As soon as a bin is *full*, i. e., has exactly B items, it is replaced by an empty one. As for bin packing, repacking is not allowed. An online algorithm must decide upon the bin for each item without knowing future item colors. The goal is to minimize the maximum number of different colors in one bin, which is called *colorfulness*. A natural algorithm for this problem is the algorithm GreedyFit: it packs an item with an already present color in the bin with that color. Otherwise, it chooses a bin which currently has the least number of different colors. Another simple algorithm, OneBin, packs all items in the same bin. Krumke et al. [34] analyzed these algorithms, showing that the competitive ratio of the trivial algorithm OneBin is strictly better than that of GreedyFit. However, GreedyFit outperforms OneBin significantly in simulations.

Hiller and Vredeveld studied the bin coloring problem with random input, where a color sequence is generated by choosing each color i. i. d. according to a fixed color distribution. To compare the performance of algorithms, they used the concept of stochastic dominance. A random variable X is *stochastically dominated* by a random variable Y, written $X \leq_{\text{st}} Y$, if

$$\Pr[X \geq x] \leq \Pr[Y \geq x] \quad \text{for all } x \in \mathbb{R}.$$

Note that a stochastic dominance relation is a statement about the distributions

of the random variables. Hiller and Vredeveld showed that for every color distribution the maximum colorfulness of GreedyFit after n items is stochastically dominated by that of OneBin.

This result is particularly interesting for the uniform color distribution, since in this case the color sequences are uniformly distributed, too. It is then possible to interpret the probabilistic result deterministically as a counting result: Stochastic dominance of the maximum colorfulness distribution implies that GreedyFit achieves a low colorfulness on more instances than OneBin. In fact, stochastic dominance in this special case is equivalent to a recent deterministic way to compare online algorithms known as *Bijective Analysis* [3]. Let S_n denote the sequences of length n and consider two online algorithms ALG_1 and ALG_2. ALG_1 is said to dominate ALG_2 w. r. t. Bijective Analysis if there is a bijective mapping $\phi\colon S_n \to S_n$ such that $\mathrm{ALG}_1(\sigma) \leq \mathrm{ALG}_2(\phi(\sigma))$ for any $\sigma \in S_n$.

Very recently, Angelopoulos and Schweitzer [4] applied this idea to paging and showed that LRU is the unique optimal algorithm w. r. t. to Bijective Analysis for a restricted class of sequences exhibiting locality of reference. For the more general analysis of stochastic dominance, Hiller and Vredeveld [24, 23] showed that LRU is optimal with respect to probability distributions that model locality of reference.

Observe that both stochastic dominance analysis and Bijective Analysis compare two online algorithms directly, without using any other (hypothetical) reference algorithm.

References

[1] S. Albers, L. M. Favrholdt, and O. Giel. On paging with locality of reference. *J. Comput. System Sci.*, 70(2):145–175, 2005.

[2] E. J. Anderson and C. N. Potts. On-line scheduling of a single machine to minimize total weighted completion time. *Mathematics of Operations Research*, 29:686–697, 2004.

[3] S. Angelopoulos, R. Dorrigiv, and A. López-Ortiz. On the separation and equivalence of paging strategies. In *Proceedings of the 18th ACM-SIAM Symposium on Discrete Algorithms*, pages 229–237, 2007.

[4] S. Angelopoulos and P. Schweitzer. Paging and list update under bijective analysis. In *Proceedings of the 20th ACM-SIAM symposium on Discrete algorithms*, pages 1136–1145, 2009.

[5] C. Banderier, R. Beier, and K. Mehlhorn. Smoothed analysis of three combinatorial problems. In *Proceedings of the 28th International Symposium on Mathematical Foundations of Computer Science*, volume 2747 of *Lecture Notes in Computer Science*, pages 198–207. Springer, 2003.

[6] L. Becchetti. Modeling locality: A probabilistic analysis of LRU and FWF. In *Proceedings of the 12th European Symp. on Algorithms (ESA)*, pages 98–109, 2004.

[7] L. Becchetti and S. Leonardi. Nonclairvoyant scheduling to minimize the total flow time on single and parallel machines. *J. ACM*, 51:517–539, 2004.

[8] L. Becchetti, S. Leonardi, A. Marchetti-Spaccamela, G. Schäfer, and T. Vredeveld. Average case and smoothed competitive analysis for the multi-level feedback algorithm. *Math. Oper. Res.*, 31(1):85–108, 2006.

[9] R. Beier, A. Czumaj, P. Krysta, and B. Vöcking. Computing equilibria for congestion games with (im)perfect information. In *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 739–748, 2004.

[10] J. L. Bentley, D. S. Johnson, F. T. Leighton, C. C. McGeoch, and L. A. McGeoch. Some unexpected expected behavior results for bin packing. In *Proceedings of the 16th Annual ACM Symposium on Theory of Computing*, pages 279–288, 1984.

[11] A. Borodin, S. Irani, P. Raghavan, and B. Schieber. Competitive paging with locality of reference. *J. Comput. System Sci.*, 50(2):244–258, 1995.

[12] A. Borodin, N. Linial, and M. E. Saks. An optimal on-line algorithm for metrical task systems. *J. ACM*, 39(4):745–763, 1992.

[13] B. Chandra. Does randomization help in on-line bin packing? *Inform. Process. Lett.*, 43(1):15–19, 1992.

[14] E. G. Coffman, Jr., C. Courcoubetis, M. R. Garey, D. S. Johnson, P. W. Shor, R. R. Weber, and M. Yannakakis. Perfect packing theorems and the average-case behavior of optimal and online bin packing. *SIAM Rev.*, 44(1):95–108, 2002.

[15] E. G. Coffman, Jr., M. R. Garey, and D. S. Johnson. Approximation algorithms for bin packing: A survey. In D. S. Hochbaum, editor, *Approximation Algorithms for NP-hard problems*. PWS Publishing, 1997.

[16] E. G. Coffman, Jr. and E. N. Gilbert. On the expected relative performance of list scheduling. *Operations Research*, 33(3):548–561, 1985.

[17] E. G. Coffman, Jr., K. So, M. Hofri, and A. C. Yao. A stochastic model of bin-packing. *Information and Control*, 44:105–115, 1980.

[18] J. Correa and M. Wagner. LP-based online scheduling: From single machine to parallel machines. *Mathematical Programming*, 119(1):109–136, 2009.

[19] J. Csirik and D. S. Johnson. Bounded space on-line bin packing: Best is better than first. *Algorithmica*, 11:115–138, 2001.

[20] A. Fiat, R. M. Karp, M. Luby, L. A. McGeoch, D. D. Sleator, and N. E. Young. Competitive paging algorithms. *Journal of Algorithms*, 12:685–699, 1991.

[21] P. A. Franaszek and T. J. Wagner. Some distribution-free aspects of paging algorithm performance. *J. ACM*, 21(1):31–39, 1974.

[22] R. L. Graham. Bounds for certain multiprocessor anomalies. *Bell System Technical Journal*, 45:1563–1581, 1966.

[23] B. Hiller. *Online Optimization: Probabilistic Analysis and Algorithm Engineering*. PhD thesis, TU Berlin, 2009.

[24] B. Hiller and T. Vredeveld. On the optimality of least recently used. ZIB-Report 08-39, Zuse Institute Berlin, 2008.

[25] B. Hiller and T. Vredeveld. Probabilistic analysis of online bin coloring algorithms via stochastic comparison. In *Proceedings of the 16th Annual European Symposium on Algorithms*, volume 5193 of *Lecture Notes in Computer Science*, pages 528–539. Springer, 2008.

[26] H. Hoogeveen and A. P. A. Vestjens. Optimal on-line algorithms for single-machine scheduling. In W. H. Cunningham, S. T. McCormick, and M. Queyranne, editors, *Proceedings of the Fifth Conference on Integer Programming and Combinatorial Optimization IPCO*, volume 1084 of *Lecture Notes in Computer Science*, pages 404–414, Berlin, 1996. Springer.

[27] D. S. Johnson. Fast algorithms for bin packing. *J. Comput. System Sci.*, 8(8):272–314, 1974.

[28] D. S. Johnson, A. Demers, J. D. Ullman, M. R. Garey, and R. L. Graham. Worst-case performance bounds for simple one-dimensional packing algorithms. *SIAM J. Comput.*, 3(4):299–325, 1974.

[29] B. Kalyanasundaram and K. Pruhs. Speed is as powerful as clairvoyance. *J. ACM*, 47(4):617–643, 2000.

[30] A. R. Karlin, M. S. Manasse, L. Rudolph, and D. D. Sleator. Competitive snoopy caching. *Algorithmica*, 3:70–119, 1988.

[31] A. R. Karlin, S. J. Phillips, and P. Raghavan. Markov paging. *SIAM J. Comput.*, 30(2):906–922, 2000.

[32] T. Kawaguchi and S. Kyan. Worst case bound of an lrf schedule for the mean weighted flow-time problem. *SIAM J. Comput.*, 15:1119–1129, 1986.

[33] E. Koutsoupias and C. Papadimitriou. Beyond competitive analysis. In *Proceedings of the 35th Annual IEEE Symposium on Foundations of Computer Science*, pages 394–400, 1994.

[34] S. O. Krumke, W. E. de Paepe, L. Stougie, and J. Rambau. Online bin coloring. In F. M. auf der Heide, editor, *Proceedings of the 9th Annual European Symposium on Algorithms*, volume 2161 of *Lecture Notes in Computer Science*, pages 74–84, 2001.

[35] C. C. Lee and D. T. Lee. A simple online bin-packing algorithm. *J. ACM*, 32(3):562–572, 1985.

[36] L. A. McGeoch and D. D. Sleator. A strongly competitive randomized paging algorithm. *Algorithmica*, 6:816–825, 1991.

[37] R. H. Möhring, A. S. Schulz, and M. Uetz. Approximation in stochastic scheduling: the power of LP-based priority policies. *Journal of the ACM*, 46:924–942, 1999.

[38] R. Motwani, S. Phillips, and E. Torng. Non-clairvoyant scheduling. *Theoretical Computer Science*, 130:17–47, 1994.

[39] K. Panagiotou and A. Souza. On adequate performance measures for paging. In *STOC '06: Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 487–496, 2006.

[40] K. Pruhs, J. Sgall, and E. Torng. Online scheduling. In J. Leung, editor, *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*. CRC Press, 2004.

[41] M. B. Richey. Improved bounds for harmonic-based bin packing algorithms. *Discrete Appl. Math.*, 34(1-3):203–227, 1991.

[42] M. Scharbrodt, T. Schickinger, and A. Steger. A new average case analysis for completion time scheduling. *J. ACM*, pages 121–146, 2006.

[43] L. Schrage. A proof of the optimality of the shortest remaining processing time discipline. *Operations Research*, 16:687–690, 1968.

[44] S. Seiden. A guessing game and randomized online algorithms. In *Proceedings of the 32nd ACM Symposium on Theory of Computing*, pages 592–601, 2000.

[45] P. W. Shor. The average-case analysis of some on-line algorithms for bin packing. *Combinatorica*, 6(2):179–200, 1986.

[46] D. D. Sleator and R. E. Tarjan. Amortized efficiency of list update and paging rules. *Comm. ACM*, 28(2):202–208, 1985.

[47] A. Souza. Adversarial models in paging – bridging the gap between theory and practice. *Computer Science – Research and Development*, page this issue, 2010.

[48] A. Souza and A. Steger. The expected competitive ratio for weighted completion time scheduling. *Theory of Computing Systems*, 39:121–136, 2006.

[49] D. A. Spielman and S. H. Teng. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *J. ACM*, 51:385–463, 2004.

[50] E. Torng. A unified analysis of paging and caching. *Algorithmica*, 20(1):175–200, 1998.

[51] A. van Vliet. On the asymptotic worst case behavior of harmonic fit. *J. Algorithms*, 20(1):113–136, 1996.

[52] A. P. A. Vestjens. *On-line Machine Scheduling*. PhD thesis, Eindhoven University of Technology, Netherlands, 1997.

[53] T. Vredeveld. Stochastic online scheduling. *Computer Science – Research and Development*, page this issue, 2010.

[54] A. C. Yao. New algorithms for bin packing. *J. ACM*, 27(2):207–227, 1980.

[55] N. E. Young. The k-server dual and loose competitiveness for paging. *Algorithmica*, 11(6):525–541, 1994.

[56] N. E. Young. On-line paging against adversarially biased random inputs. *J. Algorithms*, 37(1):218–235, 2000.