Online Bounded Analysis

Joan Boyar[1], Leah Epstein[2], Lene M. Favrholdt[1],
Kim S. Larsen[1], and Asaf Levin[3]

[1] Dept. of Mathematics and Computer Science, University of Southern Denmark, Odense, Denmark. {joan,lenem,kslarsen}@imada.sdu.dk *
[2] Dept. of Mathematics, University of Haifa, Haifa, Israel. lea@math.haifa.ac.il.
[3] Faculty of IE&M, The Technion, Haifa, Israel. levinas@ie.technion.ac.il.

Abstract. Though competitive analysis is often a very good tool for the analysis of online algorithms, sometimes it does not give any insight and sometimes it gives counter-intuitive results. Much work has gone into exploring other performance measures, in particular targeted at what seems to be the core problem with competitive analysis: the comparison of the performance of an online algorithm is made to a too powerful adversary. We consider a new approach to restricting the power of the adversary, by requiring that when judging a given online algorithm, the optimal offline algorithm must perform as well as the online algorithm, not just on the entire final request sequence, but also on any prefix of that sequence. This is limiting the adversary's usual advantage of being able to exploit that it knows the sequence is continuing beyond the current request. Through a collection of online problems, including machine scheduling, bin packing, dual bin packing, and seat reservation, we investigate the significance of this particular offline advantage.

1 Introduction

An *online problem* is an optimization problem where requests from a request sequence I are given one at a time, and for each request an irrevocable decision must be made for that request before the next request is revealed. For a minimization problem, the goal is to minimize some cost function, and if ALG is an online algorithm, we let $\text{ALG}(I)$ denote this cost on the request sequence I. Similarly, for a maximization problem, the goal is to maximize some value function (a.k.a. profit), and if ALG is an online algorithm, we let $\text{ALG}(I)$ denote this value on the request sequence I.

1.1 Performance Measures

Competitive analysis [34, 27] is the most common tool for comparing online algorithms. For a minimization problem, an online algorithm is *c-competitive* if there exists a constant α such that for all input sequences I, $\text{ALG}(I) \leq c\,\text{OPT}(I) + \alpha$.

* Supported in part by the Danish Council for Independent Research, Natural Sciences, and the Villum Foundation.

Here, Opt denotes an optimal offline algorithm. The (asymptotic) *competitive ratio* of Alg is the infimum over all such c. Similarly, for a maximization problem, an online algorithm is *c-competitive* if there exists a constant α such that for all input sequences I, $\text{Alg}(I) \geq c\,\text{Opt}(I) - \alpha$. Again, Opt denotes an optimal offline algorithm. The (asymptotic) *competitive ratio* of Alg is the supremum over all such c. In both cases, if the inequality can be established using $\alpha = 0$, we refer to the result as being *strict* (some authors use the terms *absolute* or *strong*). Note that for maximization problems, we use the convention of competitive ratios smaller than 1.

For many online problems, competitive analysis gives useful and meaningful results. However, researchers also realized from the very beginning that this is not always the case: Sometimes competitive analysis does not give any insight and sometimes it even gives counter-intuitive results, in that it points to the worse of two algorithms as the better one. A recent list of examples with references can be found in [21]. Much work has gone into exploring other performance measures, in particular targeted at what seems to be the core problem with competitive analysis that the comparison of the performance of an online algorithm is made to a too powerful adversary, controlling an optimal offline algorithm.

Four main techniques for addressing this have been employed, sometimes in combination. We discuss these ideas below. No chronological order is implied by the order the techniques are presented in. First, one could completely eliminate the optimal offline algorithm by comparing algorithms to each other directly. Measures taking this approach include max/max analysis [8], relative worst order analysis [12], bijective and average analysis [3], and relative interval analysis [20]. Second, one could limit the resources of the optimal offline algorithm, or correspondingly increase the resources of the online algorithm, as is done in extra resource analysis [31, 34]. Thus, the offline algorithm's knowledge of the future is counter-acted by requiring that it solves a harder version of the problem than the online algorithm. Alternatively, the online algorithm could be given limited knowledge of the future in terms of some form of look-ahead, as has been done for paging. In those set-ups, one assumes that the online algorithm can see a fixed number ℓ of future requests, though it varies whether it is simply the next ℓ requests, or, for instance, the next ℓ expensive requests [36], the next ℓ new requests [16], or the next ℓ distinct requests [1]. Third, one could limit the adversary's control over exactly which sequence is being used to give the bound by grouping sequences and/or considering the expected value over some set as has been done with the statistical adversary [33], diffuse adversary [30], random order analysis [29], worst order analysis [12], Markov model [28], and distributional adversary [25].

Finally, one could limit the adversary's choice of sequences it is allowed to present to the online algorithm. An early approach to this, which at the same time addressed issues of locality of reference, was the access graph model [9], where a graph defines which requests are allowed to follow each other. Another locality of reference approach was taken in [2], limiting the maximum number of different requests allowed within some fixed-sized sliding window. Both of

these models were targeted at the paging problem, and the techniques are not meant to be generally applicable to online algorithm analysis. A resource-based approach is taken in [14], where only sequences that could be fully accommodated given some resource are considered, eliminating some pathological worst-case sequences. A generalization of this, where the competitive ratio is found in the limit, appears in [15, 13]. All of these approaches are aimed at removing pathological sequences from consideration such that the worst-case (or, in principle, expected case) behavior is taken over a smaller and more realistic set of sequences, thereby obtaining results corresponding better with observed behavior in practice. A similar concept for scheduling problems is the "known-OPT" model, where the cost of an optimal offline solution is known in advance [6]. Finally, loose competitive analysis [37] allows for a a set of sequences, asymptotically smaller than the whole infinite set of input sequences, to be disregarded, while the remaining sequences should either be c-competitive or have small cost. In this way, infrequent pathological as well as unimportant (due to low cost) sequences can be eliminated.

1.2 Online Bounded Analysis

Much work can be done in all of these four categories. In this paper, we consider a new approach to restricting the power of the adversary that does not really fit into any of the known categories. Given an online algorithm, we require that the optimal offline algorithm perform as well as the online algorithm, not just on the entire final request sequence, but also on any prefix of that sequence. In essence, this is limiting the adversary's usual advantage of being able to exploit that it knows the sequence is continuing beyond the current request, without completely eliminating this advantage. Since the core of the problem of the adversary's strength is its knowledge of the future, is seems natural to try to limit that advantage directly.

This new measure is generally applicable to online problems, since it is only based on the objective function. Comparing with other measures, it is a new element that the behavioral restriction imposed on the optimal offline algorithm is determined by the online algorithm, which is the reason we name this technique *online bounded analysis*. It is adaptive in the sense that online algorithms attempting non-optimal behavior face increasingly harder conditions from the adversary the farther the online algorithm goes in the direction of non-optimality (on prefixes). The measure judges greediness more positively than does competitive analysis, since making greedy choices limits the adversary's options more, so the focus shifts towards the quality of a range of greedy or near-greedy decisions.

Behavioral restrictions on the optimal offline algorithm have been seen before, as in [17], where it is used as a tool to arrive at the final result. Here they first show a $O(1)$-competitive result against an offline algorithm restricted to, among other things, using shortest remaining processing time for job selection. Later they show that this gives rise to a schedule at most three times as bad as for an unrestricted offline algorithm. Thus, the end goal is the usual competitive ratio, and the restriction employed in the process is problem specific.

1.3 Our New Measure

If I is an input sequence for some optimization problem and A is a deterministic online algorithm for this problem, we let $A(I)$ denote the objective function value returned by A on the input sequence I.

We let $\textsc{Opt}_A$ denote the offline algorithm which is optimal under the restriction that it can never be worse than A on any prefix of an input sequence, i.e., for all sequences I, and all prefixes I' of I, for a minimization problem $\textsc{Opt}_A(I') \leq A(I')$ (for a maximization problem, $\textsc{Opt}_A(I') \geq A(I')$), and no algorithm with that property is strictly better than $\textsc{Opt}_A$ on any sequence. We say that $\textsc{Opt}_A$ is the *online bounded optimal solution* (for A).

For a minimization problem, if for some constant, c, it holds for all sequences I that $A(I) \leq c\,\textsc{Opt}_A(I)$, then we say that A has an *online bounded ratio* of at most c. The online bounded ratio of A is the infimum over all such c. Similarly, for a maximization problem, if for some constant, c, it holds for all sequences I that $A(I) \geq c\,\textsc{Opt}_A(I)$, then we say that A has an *online bounded ratio* of at least c. The online bounded ratio of A is the supremum over all such c. Note that we use the convention that an online bounded ratio for a minimization problem is at least 1, while this ratio for a maximization problem is at most 1.

1.4 Results

Through a collection of online problems, including machine scheduling, bin packing, dual bin packing, and seat reservation, we investigate the workings of online bounded analysis. As is apparent from the large collection of measures that have been defined, there is not any one measure which is best for everything. With our approach, we try to learn more about the nature of online problems, greediness, and robustness. As a first approach, we study this new idea in the simplest possible setting, but many measures combine ideas, so in future work, it would be natural to investigate this basic idea in combination with elements from other measures.

First, we observe that some results from competitive analysis carry over. Then we note that some problem characteristics imply that a greedy algorithm is optimal.

For machine scheduling, we obtain the following results. For minimizing makespan on $m \geq 2$ identical machines, we get an online bounded ratio of $2-\frac{1}{m-1}$ for Greedy. Though this is smaller than the competitive ratio of $2 - \frac{1}{m}$ [26], it is a comparable result, demonstrating that non-greedy behavior is not the key to the adversary performing better by a factor close to two for large m. For two uniformly related machines, we prove that Greedy has online bounded ratio 1. This is consistent with competitive ratio results, where Greedy has been proven optimal [24, 18]. For the case where the faster machine is at least ϕ (the golden ratio) times faster than the slower machine, competitive analysis finds that Greedy and Fast, the algorithm that only uses the faster machine, are equally good. Using relative worst order analysis, Greedy is deemed the better algorithm [23], which seems reasonable since Greedy is never worse on any

sequence than FAST, and sometimes better. We also obtain this positive distinction, establishing the online bounded ratios 1 and $\frac{s+1}{s}$ (if the faster machine is s times faster than the slower one) for GREEDY and FAST, respectively.

For the Santa Claus machine scheduling problem [7], we prove that GREEDY is optimal for identical machines with respect to the online bounded ratio. For two related machines with speed ratio s, we present an algorithm with an online bounded ratio better than $\frac{1}{s}$ and show that no online algorithm has a higher online bounded ratio. For this problem, it is known that the best possible competitive ratio for identical machines is $\frac{1}{m}$, and the best possible competitive ratio for two related machines is $\frac{1}{s+1}$ [35, 5, 22].

For classic bin packing, we show that any Any-Fit algorithm has an online bounded ratio of at least $\frac{3}{2}$. We observe that for bin covering, the best online bounded ratio is equal to the best competitive ratio [19]. For these problems, asymptotic measures are used. We show a connection between results concerning the competitive ratio on accommodating sequences and the online bounded ratio. For dual bin packing (namely, the multiple knapsack problem with equal capacity knapsacks and unit weights items), we show that the online bounded ratio is the same as the competitive ratio on accommodating sequences (that is, sequences where OPT packs all items) for a large class of algorithms including First-Fit, Best-Fit, and Worst-Fit. It then follows from results in [13] that any algorithm in this class has an online bounded ratio of at least $\frac{1}{2}$. Furthermore, the online bounded ratio of First-Fit and Best-Fit is $\frac{5}{8}$, and that of Worst-Fit is $\frac{1}{2}$. We also note that, for any dual bin packing algorithm, an upper bound on the competitive ratio on accommodating sequences is also an upper bound on the online bounded ratio. Using a result from [13], this implies that any (possibly randomized) algorithm has an online bounded ratio of at most $\frac{6}{7}$.

For seat reservation, we have preliminary results, and conjecture that results are similar to machine scheduling for identical machines, in that ratios similar to but slightly better than those obtained using competitive analysis can be established.

We found that the new measure sometimes leads to the same results as the standard competitive ratio, and in some cases it leads to a competitive ratio of 1. However, there are problem variants for which we obtain an intermediate value, which confirms the relevance of our approach.

The proofs which have been omitted can be found in the full paper [10].

2 Online Bounded Analysis

Before considering concrete problems, we discuss some generic properties.

2.1 Measure Properties

The online bounded ratio of an algorithm is never further away from 1 than the competitive ratio, since the online algorithm's performance is being compared to a (possibly) restricted "optimal" algorithm.

Since algorithms are compared with different optimal algorithms, one might be concerned that two algorithms, A and O, could have online bounded ratio 1, and yet one algorithm could do better on some sequences than the other. This is not possible. To see this, consider some sequence I and assume that A does better than O on I, yet both algorithms have online bounded ratio 1.

If their online bounded ratio is 1, there is no point where one algorithm makes a decision which changes the objective value more than the other does, since the adversary could end the sequence there and the one algorithm would not have online bounded ratio 1. Thus, both algorithms have the same objective function value at all points, so they always compete against the same adversary. If algorithm A performs better than algorithm O on I, then algorithm O does not have online bounded ratio 1.

For some problems, such as paging, OPT_A is the same as OPT under competitive analysis for all algorithms A, because OPT's behavior on any sequence is also optimal on any prefix of that sequence. Thus, the competitive analysis results for paging and similar problems also hold with this measure, giving the same online bounded ratio as competitive ratio.

2.2 Greedy is Sometimes Optimal

It is sometimes the case that there is one natural greedy algorithm that always has a unique greedy choice in each step. In such situations, the greedy algorithm is optimal with respect to this measure, having online bounded ratio 1. For example, consider the weighted matching in a graph where the edges arrive in an online fashion (the edge-arrival model) and the algorithm in each step decides if the current edge is added to the matching or discarded. Here, the greedy algorithm, denoted by GREEDY, adds the current edge if adding the edge will keep the solution feasible (that is, its two end-vertices are still exposed by the matching that the algorithm created so far) and the weight of the edge is strictly positive. Note that indeed the online bounded ratio of GREEDY is 1, as the solution constructed by $\text{OPT}_{\text{GREEDY}}$ must coincide with the solution created by GREEDY. The last claim follows by a trivial induction on the number of edges considered so far by both GREEDY and $\text{OPT}_{\text{GREEDY}}$. If GREEDY adds the current edge, then by the definition of $\text{OPT}_{\text{GREEDY}}$, we conclude that $\text{OPT}_{\text{GREEDY}}$ adds the current edge. If GREEDY discards the current edge because at least one of its end-vertices is matched, then $\text{OPT}_{\text{GREEDY}}$ cannot add the current edge either (using the induction assumption). Last, if GREEDY discards the current edge since its weight is non-positive, then we can remove the edge from the bounded optimal solution, $\text{OPT}_{\text{GREEDY}}$, if it was added (removing it from $\text{OPT}_{\text{GREEDY}}$ will not affect the future behavior of $\text{OPT}_{\text{GREEDY}}$ since $\text{OPT}_{\text{GREEDY}}$ must accept an edge whenever GREEDY does). Similar proofs hold in other cases when there is a unique greedy choice for OPT in each step. Note that for the weighted matching problem where vertices arrive in an online fashion and when a vertex arrives the edge set connecting this vertex to earlier vertices is revealed with their weights (the vertex-arrival model), the standard lower bound for weighted matching holds as can be seen in the following lower bound construction. The

first three vertices arrive in the order $1, 2, 3$ and when vertex 3 arrives, two edges $\{1, 3\}, \{2, 3\}$ are revealed each of which has weight of 1 (vertices 1 and 2 are not connected). At this point, an online algorithm with a finite online bounded ratio must add one of these edges to the matching. Then, either 1 or 2 are matched in the current solution, and in the last step, vertex 4 arrives with an edge of weight M connecting 4 to the vertex among 1 and 2 that was matched by the algorithm. Observe that when vertex 3 arrives, the algorithm adds an edge to the matching while the bounded optimal solution can add the other edge, and this will allow the bounded optimal solution to add the last edge as well.

The argument for the optimality of Greedy for the weighted matching problem in the edge-arrival model clearly holds if all weights are 1 also. This unweighted matching problem in the edge-arrival model is an example of a maximization problem in the online complexity class Asymmetric Online Covering (AOC) [11]:

Definition 1. *An online accept-reject problem, P, is in* Asymmetric Online Covering (AOC) *if, for the set Y of requests accepted:*

For minimization (maximization) problems, the objective value of Y is $|Y|$ if Y is feasible and ∞ ($-\infty$) otherwise, and any superset (subset) of a feasible solution is feasible.

For all maximization problems in the class AOC, there is an obvious greedy algorithm, Greedy, which accepts a request whenever acceptance maintains feasibility. The argument above showing that the online bounded ratio of Greedy is 1 for the weighted matching problem in the edge-arrival model generalizes to all maximization problems in AOC.

Theorem 1. *For any maximization problem in AOC, the online bounded ratio of* Greedy *is* 1. *Thus,* Greedy *is optimal according to online bounded analysis for Online Independent Set in the vertex-arrival model, Unweighted Matching in the edge-arrival model, and Online Disjoint Path Allocation where requests are paths.*

Note that this does not hold for all minimization problems in AOC. For example, Cycle Finding in the vertex-arrival model, the problem of accepting as few vertices as possible, but accepting enough so that there is a cycle in the induced subgraph accepted, is AOC-Complete. However, consider the first vertex requested in a graph with only one cycle. Greedy is forced to accept it, since the vertex could be part of the unique cycle, but $\text{Opt}_{\text{Greedy}}$ will reject the vertex if it is not in that cycle.

However, there are online bounded optimal greedy algorithms for minimization problems in AOC, such as Vertex Cover, which are *complements* of maximization problems in AOC (Independent Set in the case of Vertex Cover). By complement, we mean that set S is a maximal feasible set in the maximization problem if and only if the requests not in S are a feasible solution for the minimization problem. The greedy algorithm in the case of these minimization problems would be the algorithm that accepts exactly those requests that Greedy for the complementary maximization problem rejects.

3 Machine Scheduling: Makespan

We consider the load balancing problem of minimizing makespan for online job scheduling on m identical machines without preemption, and analyze the classic greedy algorithm (also known as list scheduling). At any point, Greedy schedules the next job on a least loaded machine. Since the machines are identical, ties can be resolved arbitrarily without loss of generality. It is known that the competitive ratio of Greedy is $2 - \frac{1}{m}$ [26]. With the more restricted optimal algorithm, we get a smaller value of $2 - \frac{1}{m-1}$ as the online bounded ratio of Greedy.

Lemma 1. *For the problem of minimizing makespan for online job scheduling on m identical machines,* Greedy *has online bounded ratio of at most $2 - \frac{1}{m-1}$.*

Proof. Consider a sequence I. Let j be the first job in I that is completed at the final makespan of Greedy, and assume that it has size w. Let t and s be the starting times of j in $\text{Opt}_{\text{Greedy}}$ and Greedy, respectively, and let ℓ and ℓ' be the makespans of $\text{Opt}_{\text{Greedy}}$ and Greedy, respectively, just before the arrival of j. Let V denote the total size of the jobs in I just before j arrives.

We have the following inequalities: $\text{Opt}_{\text{Greedy}} \geq t + w$ and $\text{Opt}_{\text{Greedy}} \geq \ell$. In addition, since, just before j arrived, the machine where $\text{Opt}_{\text{Greedy}}$ placed j had load t and the other machines had load at most ℓ, $V \leq t + (m-1)\ell$. Since $m - 1 \geq 1$, $V \leq (m-1)(t+\ell)$.

Because Greedy placed j on its least loaded machines, all machines had load at least s before j arrived. At least one machine had load ℓ', so $V \geq (m-1)s + \ell'$. By the definition of online bounded analysis, $\ell \leq \ell'$. Thus, $V \geq (m-1)s + \ell$. Combining the upper and lower bounds on V gives $(m-1)s \leq (m-1)t + (m-2)\ell$ and $s \leq t + \frac{m-2}{m-1}\ell$. We now bound Greedy's makespan:

$$\begin{aligned}\text{Greedy}(I) &= s + w = (s - t) + (t + w) \\ &\leq \left(\tfrac{m-2}{m-1}\right) \cdot \ell + \text{Opt}_{\text{Greedy}}(I) \leq \left(2 - \tfrac{1}{m-1}\right) \text{Opt}_{\text{Greedy}}(I)\end{aligned}$$

□

Lemma 2. *For the problem of minimizing makespan for online job scheduling on m identical machines,* Greedy *has online bounded ratio of at least $2 - \frac{1}{m-1}$.*

By Lemmas 1 and 2 we find the following.

Theorem 2. *For the problem of minimizing makespan for online job scheduling on m identical machines,* Greedy *has online bounded ratio $2 - \frac{1}{m-1}$.*

Most interestingly, Theorem 2 establishes the existence of an online algorithm, Greedy, for makespan minimization on two identical machines with an online bounded ratio of 1. Next, we generalize this last result to the case of two uniformly related machines. Note that for two uniformly related machines we can assume that machine number 1 is strictly faster than machine number 2, and the two speeds are $s > 1$ and 1.

We define GREEDY as the algorithm that assigns the current job to the machine such that adding the job there results in a solution of a smaller makespan breaking ties in favor of assigning the job to the slower machine (that is, to machine number 2). If an algorithm breaks ties in favor of assigning the job to the faster machine (let this algorithm be called GREEDY$'$), then its online bounded ratio is strictly above 1, as the following example implies. The first job has size $s-1$ (and it is assigned to machine 1), and the second job has size 1 (and assigning it to any machine will result in the current makespan 1). The first job must be assigned to machine 1 by $\text{OPT}_{\text{GREEDY}'}$, and it assigns the second job to the second machine. A third job of size $s+1$ arrives. This job is assigned to the first machine by $\text{OPT}_{\text{GREEDY}'}$, obtaining a makespan of 2. GREEDY$'$ will have a makespan of at least $\min\{2+1/s, s+1\} > 2$ as $s > 1$.

Theorem 3. *For the problem of minimizing makespan for online job scheduling on two uniformly related machines,* GREEDY *has online bounded ratio* 1*.*

We now consider the algorithm FAST that simply schedules all jobs on the faster machine. In contrast to GREEDY, FAST does not have an online bounded ratio of 1. This also contrasts with competitive analysis, since FAST has an optimal competitive ratio for $s \geq \phi$, where $\phi = \frac{1+\sqrt{5}}{2} \approx 1.618$.

Theorem 4. *For two related machines with speed ratio* s*,* FAST *has an online bounded ratio of* $\frac{s+1}{s}$*.*

By Theorem 2, the result of Theorem 3 cannot be extended to three or more identical machines for GREEDY. We conclude this section by proving that such a generalization is impossible, not only for GREEDY, but for any deterministic online algorithm.

Theorem 5. *Let* $m \geq 3$*. For the problem of minimizing makespan for online job scheduling on* m *identical machines, any deterministic online algorithm* A *has online bounded ratio of at least* $\frac{4}{3}$*.*

An obvious next step would be to try to match the general lower bound of $\frac{4}{3}$ by designing an algorithm that places each job on the most loaded machine where the bound of $\frac{4}{3}$ would not be violated. However, even for $m = 3$, this would not work, as seen by the input sequence $I = \langle \frac{3}{4}, \frac{1}{4}, \frac{5}{12}, \frac{1}{6}, \frac{7}{12}, \frac{5}{6} \rangle$. The algorithm would combine the first two jobs on one machine and the following two on another machine. Since the optimal makespan at this point is $\frac{3}{4}$, the algorithm will schedule the fifth job on the third machine. When the last job arrives, all machines have a load of at least $\frac{7}{12}$, resulting in a makespan of $\frac{17}{12} > 1.4$. Note that I can be scheduled such that each machine has a load of exactly 1. Since the algorithm has a makespan of 1 already after the second job, the online bounded restriction is actually no restriction on OPT for this sequence.

4 Machine Scheduling: Santa Claus

In contrast to makespan, the objective in Santa Claus scheduling is to maximize the minimum load. Traditionally, the algorithm GREEDY for this problem assigns any new job to a machine achieving a minimum load in the schedule that was created up to the time just before the job is added to the solution (breaking ties arbitrarily). For identical machines, this algorithm is equivalent to the greedy algorithm for makespan minimization. Unlike the makespan minimization problem, where this algorithm has online bounded ratio of 1 only for two identical machines, here we show that GREEDY has an online bounded ratio of 1 for any number of identical machines.

Theorem 6. *For the Santa Claus problem on m identical machines,* GREEDY *has online bounded ratio* 1*.*

Proof. Let a configuration be a multi-set of the current loads on all of the machines, i.e., without any annotation of which machine is which. As long as $\text{OPT}_{\text{GREEDY}}$ also assigns each job to a machine with minimum load, the configurations of GREEDY and $\text{OPT}_{\text{GREEDY}}$ are identical.

Consider the first time $\text{OPT}_{\text{GREEDY}}$ does something different from GREEDY. If, when that job j arrives, there is a unique machine with minimum load, $\text{OPT}_{\text{GREEDY}}$ would have a worse objective value than GREEDY after placing j, so, by definition of online bounded analysis, this cannot happen. Now consider the situation where $k \geq 2$ machines have minimum load. Then, after processing j, GREEDY has $k-1$ machines with minimum load, whereas $\text{OPT}_{\text{GREEDY}}$ has k. In that case, no more than $k-2$ further jobs can be given. This is seen as follows: If $k-1$ jobs were given, GREEDY would place one on each of its $k-1$ machines with minimum load, and, thus, raise the minimum. $\text{OPT}_{\text{GREEDY}}$, on the other hand, would not be able to raise (at this step) the minimum of all of its k machines with minimum load, and would therefore not be optimal; a contradiction.

Thus, $\text{OPT}_{\text{GREEDY}}$ can only have a different configuration than GREEDY after GREEDY (and $\text{OPT}_{\text{GREEDY}}$) have obtained their final (and identical) objective value, and so, the online bounded ratio of GREEDY is 1. □

Next, we show that unlike the makespan minimization problem, for which there is an online algorithm with online bounded ratio of 1 even for the case of two uniformly related machines (Theorem 3), such a result is impossible for the Santa Claus problem on two uniformly related machines.

Theorem 7. *For the Santa Claus problem on two uniformly related machines with speed ratio s, no deterministic online algorithm has an online bounded ratio larger than $\frac{1}{s}$.*

Proof. For any online algorithm A, we consider a setting of two uniformly related machines with speeds 1 and s. The input consists of exactly two jobs. After the first job is assigned by A, the objective function value remains zero, and only if

the algorithm assigns the two jobs to distinct machines, will it have a positive objective function value. Thus, when there are only two jobs, $\textsc{Opt}_A$ is simply the optimal solution for the instance. The first job is of size 1. If A assigns the job to the machine of speed s, then the next job is of size s. At this point $\textsc{Opt}_A$ has value 1 (by assigning the first job to the slower machine and the second to the faster machine), but A has either zero value (if both jobs are assigned to the faster machine) or a value of $\frac{1}{s}$. In the second case where A assigns the first job (of size 1) to the slower machine of speed 1, the second job has size $\frac{1}{s}$. At this point $\textsc{Opt}_A$ has value $\frac{1}{s}$ (by assigning the first job to the faster machine and the second to the slower machine), but A has either zero value (if both jobs are assigned to the slower machine) or a value of $\frac{1/s}{s}$. □

Interestingly, the online bounded ratio of the following simple algorithm matches this bound. The algorithm G assigns each job to the least loaded machine. While for identical machines, this algorithm and Greedy are equivalent, for related machines this is not the case. The same algorithm is the one that achieves the best possible competitive ratio $\frac{1}{s+1}$ [22].

Theorem 8. *For the Santa Claus problem on two uniformly related machines with speed ratio s, the online bounded ratio of G is $\frac{1}{s}$.*

5 Classic Bin Packing and Bin Covering

In classic bin packing, the input is a sequence of items of sizes s, $0 < s \leq 1$, that should be packed in as few bins of size 1 as possible. We say that a bin is open if at least one item has been placed in the bin. An Any-Fit algorithm is an algorithm that never opens a new bin if the current item fits in a bin that is already open. In this section, we use the *asymptotic online bounded ratio*. Thus, we allow for an additive constant, exactly as with the asymptotic (non-strict) competitive ratio.

Theorem 9. *Any Any-Fit algorithm has an online bounded ratio of at least $\frac{3}{2}$.*

In classic bin covering, the input is as in bin packing, and the goal is to assign items to bins so as to maximize the number of bins whose total assigned size is at least 1. For this problem, it is known that a simple greedy algorithm (which assigns all items to the active bin until the total size assigned to it becomes 1 or larger, and then it moves to the next bin and defines it as active) has the best possible competitive ratio $\frac{1}{2}$. The negative result [19] is proven using inputs where the first batch of items consists of a large number of very small items, and it is followed by a set of large identical items of sizes close to 1 (where the exact size is selected based on the actions of the algorithm). The total size of the very small items is strictly below 1, so as long as large items were not presented yet, the value of any algorithm is zero. An optimal offline solution packs the very small items such that packing every large item results in a bin whose contents have a total size of exactly 1. Thus, no algorithm can perform better on any prefix, and this construction shows that the online bounded ratio is at most $\frac{1}{2}$.

6 Dual Bin Packing

As in the previous section, we use the asymptotic online bounded ratio here. Dual bin packing is like the classic bin packing problem, except that there is only a limited number, n, of bins and the goal is to pack as many items in these n bins as possible. Known results concerning the competitive ratio on accommodating sequences can be used to obtain results for the online bounded ratio.

In general, accommodating sequences [14, 15] are defined to be those sequences for which OPT does not get a better result by having more resources. For the dual bin packing problem, accommodating sequences are sequences of items that can be fully accommodated in the n bins, i.e., OPT packs all items.

We show that, for a large class of algorithms for dual bin packing containing First-Fit and Best-Fit, the online bounded ratio is the same as the competitive ratio on accommodating sequences. To show that this does not hold for all algorithms, we also give an example of a $\frac{2}{3}$-competitive algorithm on accommodating sequences that has an online bounded ratio of 0.

Dual bin packing is an example of a problem in a larger class of problems which includes the seat reservation problem discussed below. A problem is an *accept/reject accommodating problem* if algorithms can only accept or reject requests, the goal is accept as many requests as possible, and the accommodating sequences are those where OPT accepts all requests.

Theorem 10. *For any online algorithm* ALG *for any accept/reject accommodating problem, the competitive ratio of* ALG *on accommodating sequences is equal to the online bounded ratio of* ALG *on accommodating sequences.*

Note that this result applies to all algorithms for dual bin packing. Since any accommodating sequence is also a valid adversarial sequence for the case with no restrictions on the sequences, we obtain the following corollary of Theorem 10.

Corollary 1. *For any online algorithm* ALG *for any accept/reject accommodating problem, any upper bound on the competitive ratio of* ALG *on accommodating sequences is also an upper bound on the online bounded ratio of* ALG.

A *fair* algorithm for dual bin packing is an algorithm that never rejects an item that it could fit in a bin. A *rejection-invariant* algorithm is an algorithm that does not change its behavior based on rejected items.

Theorem 11. *For any fair, rejection-invariant dual bin packing algorithm* ALG*, the online bounded ratio of* ALG *equals the competitive ratio of* ALG *on accommodating sequences.*

One algorithm which is fair and rejection-invariant is First-Fit, which packs each item in the first bin it fits in (and rejects it if no such bin exists). Another example of a fair, rejection-invariant algorithm is Best-Fit, which packs each item in a most full bin that can accommodate it. Worst-Fit is the algorithm that packs each item in a most empty bin.

Corollary 2. *Best-Fit and First-Fit have online bounded ratios of* $\frac{5}{8}$. *Worst-Fit has an online bounded ratio of* $\frac{1}{2}$.

Corollary 3. *Any fair, rejection-invariant dual bin packing algorithm has an online bounded ratio of at least* $\frac{1}{2}$. *Any (possibly randomized) dual bin packing algorithm has an online bounded ratio of at most* $\frac{6}{7}$.

The algorithm Unfair-First-Fit (UFF) defined in [4] is designed to work well on accommodating sequences. Whenever an item larger than $\frac{1}{2}$ arrives, UFF rejects the item unless it will bring the number of accepted items below $\frac{2}{3}$ of the total number of items that are accepted by an optimal solution of the prefix of items given so far (for an accommodating sequence this is the number of items in the prefix). The competitive ratio of UFF on accommodating sequences is $\frac{2}{3}$ [4]. We show that, in contrast to Theorem 11, UFF has an online bounded ratio of 0.

Theorem 12. *Unfair-First-Fit has an online bounded ratio of* 0.

7 Unit Price Seat Reservation

Since even the unit price seat reservation problem has a terrible competitive ratio, depending on the number of stations, this problem has often been studied using the competitive ratio on accommodating sequences, which for the seat reservation problem restricts the input sequences considered to those where OPT could have accepted all of the requests. By Theorem 10, for accommodating sequences, the competitive ratio and the online bounded ratio are identical.

The unit price seat reservation problem has competitive ratio $\Theta(1/k)$, where k is the number of stations. This does not change for the online bounded ratio, even though, both the original proof, showing that no deterministic fair online algorithm (that does not reject an interval if it is possible to accept it) for the unit price problem is more than $\frac{8}{k+5}$-competitive [14], and the proof improving this to $\frac{4}{k-2\sqrt{k-1}+4}$ [32], used an optimal offline algorithm which rejected some requests before the online algorithm did. The main ideas in these proofs was that the adversary could give small request intervals which OPT could place differently from the algorithm, allowing it to reject some long intervals and still be fair. Rejecting long intervals allowed it to accept many short intervals which the algorithm was forced to reject. By using small intervals involving only the last few stations, one can arrange that the online algorithm has to reject intervals early. Then, giving nearly the same sequence as for the $\frac{8}{k+5}$ bound, using two fewer stations, OPT can still reject the same long intervals and do just as badly asymptotically. Note that we reuse the $[k-3, k-2)$ intervals.

Theorem 13. *No deterministic fair online algorithm for the unit price seat reservation problem has an online bounded ratio of more than* $\frac{11}{k+7}$.

Using a similar proof, one can show that the online bounded ratios of First-Fit and Best-Fit are at least $\frac{5}{k+1}$. The major difference is that in the first part,

First-Fit and Best-Fit each reject $n/2$ intervals, so in the second part, O can also reject $n/2$ intervals. Since any fair online algorithm for the unit price problem is $2/k$-competitive, any fair online algorithm for the unit price problem has an online bound ratio of at least $2/k$.

References

1. S. Albers. On the influence of lookahead in competitive paging algorithms. *Algorithmica*, 18:283–305, 1997.
2. S. Albers, L. M. Favrholdt, and O. Giel. On paging with locality of reference. In *34th Annual ACM Symposium on the Theory of Computing (STOC)*, pages 258–267, 2002.
3. S. Angelopoulos, R. Dorrigiv, and A. López-Ortiz. On the separation and equivalence of paging strategies. In *18th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 229–237, 2007.
4. Y. Azar, J. Boyar, L. Epstein, L. M. Favrholdt, K. S. Larsen, and M. N. Nielsen. Fair versus unrestricted bin packing. *Algorithmica*, 34(2):181–196, 2002.
5. Y. Azar and L. Epstein. On-line machine covering. *Journal of Scheduling*, 1(2):67–77, 1998.
6. Y. Azar and O. Regev. On-line bin-stretching. *Theoretical Computer Science*, 268(1):17–41, 2001.
7. N. Bansal and M. Sviridenko. The Santa Claus problem. In *38th Annual ACM Symposium on the Theory of Computing (STOC)*, pages 31–40, 2006.
8. S. Ben-David and A. Borodin. A new measure for the study of on-line algorithms. *Algorithmica*, 11(1):73–91, 1994.
9. A. Borodin, S. Irani, P. Raghavan, and B. Schieber. Competitive paging with locality of reference. *Journal of Computer and System Sciences*, 50(2):244–258, 1995.
10. J. Boyar, L. Epstein, L. M. Favrholdt, K. S. Larsen, and A. Levin. Online bounded analysis. *CoRR*, abs/1602.06708, 2016.
11. J. Boyar, L. Favrholdt, J. Mikkelsen, and C. Kudahl. Advice complexity for a class of online problems. In *32nd International Symposium on Theoretical Aspects of Computer Science, (STACS)*, volume 30 of *Leibniz International Proceedings in Informatics*, pages 116–129, 2015.
12. J. Boyar and L. M. Favrholdt. The relative worst order ratio for on-line algorithms. *ACM Transactions on Algorithms*, 3(2):article 22, 24 pages, 2007.
13. J. Boyar, L. M. Favrholdt, K. S. Larsen, and M. N. Nielsen. Extending the Accommodating Function. *Acta Informatica*, 40(1):3–35, 2003.
14. J. Boyar and K. Larsen. The seat reservation problem. *Algorithmica*, 25:403–417, 1999.
15. J. Boyar, K. S. Larsen, and M. N. Nielsen. The accommodating function—a generalization of the competitive ratio. *SIAM Journal on Computing*, 31(1):233–258, 2001.
16. D. Breslauer. On competitive on-line paging with lookahead. *Theoretical Computer Science*, 209(1–2):365–375, 1998.
17. S.-H. Chan, T.-W. Lam, L.-K. Lee, C.-M. Liu, and H.-F. Ting. Sleep management on multiple machines for energy and flow time. In L. Aceto, M. Henzinger, and J. Sgall, editors, *Automata, Languages and Programming (ICALP)*, volume 6755 of *LNCS*, pages 219–231. Springer-Verlag Berlin Heidelberg, 2011.

18. Y. Cho and S. Sahni. Bounds for list schedules on uniform processors. *SIAM Journal on Computing*, 9(1):91–103, 1980.
19. J. Csirik and V. Totik. On-line algorithms for a dual version of bin packing. *Discrete Applied Mathematics*, 21:163–167, 1988.
20. R. Dorrigiv, A. López-Ortiz, and J. I. Munro. On the relative dominance of paging algorithms. *Theoretical Computer Science*, 410:3694–3701, 2009.
21. M. R. Ehmsen, J. S. Kohrt, and K. S. Larsen. List Factoring and Relative Worst Order Analysis. *Algorithmica*, 66(2):287–309, 2013.
22. L. Epstein. Tight bounds for bandwidth allocation on two links. *Discrete Applied Mathematics*, 148(2):181–188, 2005.
23. L. Epstein, L. M. Favrholdt, and J. S. Kohrt. Separating online scheduling algorithms with the relative worst order ratio. *Journal of Combinatorial Optimization*, 12(4):363–386, 2006.
24. L. Epstein, J. Noga, S. S. Seiden, J. Sgall, and G. J. Woeginger. Randomized online scheduling on two uniform machines. In *Tenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 317–326, 1999.
25. Y. Giannakopoulos and E. Koutsoupias. Competitive analysis of maintaining frequent items of a stream. *Theoretical Computer Science*, 562:23–32, 2015.
26. R. L. Graham. Bounds for certain multiprocessing anomalies. *Bell Systems Technical Journal*, 45:1563–1581, 1966.
27. A. R. Karlin, M. S. Manasse, L. Rudolph, and D. D. Sleator. Competitive snoopy caching. *Algorithmica*, 3:79–119, 1988.
28. A. R. Karlin, S. J. Phillips, and P. Raghavan. Markov paging. *SIAM Journal on Computing*, 30(3):906–922, 2000.
29. C. Kenyon. Best-fit bin-packing with random order. In *7th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 359–364, 1996.
30. E. Koutsoupias and C. H. Papadimitriou. Beyond competitive analysis. In *35th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 394–400, 1994.
31. E. Koutsoupias and C. H. Papadimitriou. Beyond competitive analysis. *SIAM Journal on Computing*, 30(1):300–317, 2000.
32. S. Miyazaki and K. Okamoto. Improving the competitive ratios of the seat reservation problem. In *6th IFIP TC 1/WG 2.2 International Conference on Theoretical Computer Science (IFIP TCS)*, volume 323 of *IFIP Advances in Information and Communication Technology*, pages 328–339. Springer, 2010.
33. P. Raghavan. A statistical adversary for on-line algorithms. In *On-Line Algorithms*, volume 7 of *Series in Discrete Mathematics and Theoretical Computer Science*, pages 79–83. American Mathematical Society, 1992.
34. D. D. Sleator and R. E. Tarjan. Amortized efficiency of list update and paging rules. *Communications of the ACM*, 28(2):202–208, 1985.
35. G. J. Woeginger. A polynomial-time approximation scheme for maximizing the minimum machine completion time. *Operations Research Letters*, 20(4):149–154, 1997.
36. N. Young. Competitive paging and dual-guided algorithms for weighted caching and matching (thesis). Technical Report CS-TR-348-91, Computer Science Department, Princeton University, 1991.
37. N. E. Young. The k-server dual and loose competitiveness for paging. *Algorithmica*, 11:525–541, 1994.