The Seat Reservation Problem

Joan Boyar*†‡ Kim S. Larsen*§‡¶

Odense University

Abstract

We investigate the problem of giving seat reservations on-line. We assume that a train travels from a start station to an end station, stopping at k stations, including the first and last. Reservations can be made for any trip going from any station to any later station. The train has a fixed number of seats. The seat reservation system attempts to maximize income. We consider the case in which all tickets have the same price and the case in which the price of a ticket is proportional to the length of the trip. For both cases, we prove upper and lower bounds of $\Theta(1/k)$ on the competitive ratio of any "fair" deterministic algorithm. We also define the *accommodating ratio* which is similar to the competitive ratio except that the only sequences of requests allowed are sequences for which the optimal off-line algorithm could accommodate all requests. We prove upper and lower bounds of $\Theta(1)$ on the accommodating ratio of any "fair" deterministic algorithm, in the case in which all tickets have the same price, but $\Theta(1/k)$ in the case in which the ticket price is proportional to the length of the trip. The most surprising of these results is that all "fair" algorithms are at least 1/2-accommodating when all tickets have the same price. We prove similar results bounding the performance of any "fair" randomized algorithm against an adaptive on-line adversary. We also consider concrete algorithms; more specifically, First-Fit and Best-Fit.

Keywords: on-line algorithms, competitive ratio, accommodating ratio, graph coloring, interval graphs, seat reservation problem.

*Department of Mathematics and Computer Science, Odense University, Campusvej 55, DK-5230 Odense M, Denmark

†Email: joan@imada.ou.dk.

‡Supported in part by NSF (U.S.) grant CCR-9510244 and in part by the ESPRIT Long Term Research Programme of the EU under project number 20244 (ALCOM-IT).

§Email: kslarsen@imada.ou.dk.

¶Additional support from SNF (Denmark).

1 Introduction

With many train systems in Europe, passengers are required to buy seat reservations with their train tickets. Since the ticketing system must assign a passenger a single seat when that passenger purchases a ticket, without knowing what future requests there will be for seats, this is an on-line problem. Thus, a competitive analysis is appropriate.

In this paper, we investigate the problem of giving seat reservations on-line. We assume that a train with n seats travels from a start station to an end station, stopping at $k \geq 2$ stations, including the first and last. The seats are numbered from 1 to n. The start station is station 1 and the end station is station k. Reservations can be made for any trip from a station s to a station t where $1 \leq s < t \leq k$. The passenger is given a single seat number when the ticket is purchased, which can be any time before departure.

The algorithms (ticket agents) attempt to maximize income, i.e., the sum of the prices of the tickets sold. Thus, the performance of an on-line algorithm will depend on pricing policies for train tickets. We consider two different policies: one in which all tickets have the same price, the *unit price problem*, and one in which the price of a ticket is proportional to the distance traveled, the *proportional price problem*.

In addition, we define the *accommodating ratio* which is similar to the competitive ratio, except that the only sequences of requests allowed are sequences for which the optimal off-line algorithm could accommodate all requests. This assumption, that there are enough seats for the optimal fair on-line algorithm, is appropriate whenever the management has done a reasonable job of predicting ticket demand and has thus assigned an appropriate number of cars to the train. We show that there is a significant difference between the competitive and accommodating ratios for the unit price problem. This difference indicates that with that pricing policy, underestimating how many seats are needed can have a dramatic effect on earnings.

Since the accommodating ratio is essentially the competitive ratio, except that only specific restricted sets of request sequences are considered, the accommodating ratio is a special case of the performance ratio defined by Koutsoupias and Papadimitriou in [1], which restricts the power of the adversary by allowing only certain input distributions. Restricted sets of request sequences have also been considered for problems other than seat reservations. For example, Borodin, Irani, Raghavan and Shieber [2] restrict the inputs allowed to give more realistic analyses of the performance of various paging algorithms, taking into consideration locality of reference. In addition, on-line algorithms for bipartite matching [3] have restricted the inputs to graphs which actually have a perfect matching.

For political reasons, the ticket agent may not refuse a passenger if it is possible to accommodate him when he attempts to make his reservation. Thus, if there is any seat which is empty for the entire duration of that passenger's trip, the passenger must be assigned a seat. We will call an algorithm which has this restriction *fair* and will only consider fair algorithms.

The seat reservation problem is similar to the problem of optical routing with a limited number of wavelengths, which was considered in [4] for the off-line case for tree networks and in [5] for a number of on-line versions. In [6, 7], similar problems for the preemptive case are considered. Interval scheduling [8] and the one-wavelength version of the call control problem [9] are similar to trains with one seat. However, the fairness restriction makes the seat reservation problem different from the problems considered before. Without this restriction, the seat reservation problem is the same as the on-line call admission and wavelength selection problem [5] restricted to graphs which are simple paths. Their technique is not applicable to fair algorithms.

Note that since we are trying to maximize income rather than minimize cost, a lower bound is obtained by proving a bound on the worst case behavior of an algorithm, and an upper bound is obtained by giving an adversary argument. We prove upper and lower bounds of $\Theta(1)$ on the accommodating ratio of any fair deterministic on-line algorithm for the unit price problem, but $\Theta(1/k)$ for the proportional price problem. For both the unit price problem and the proportional price problem, we prove upper and lower bounds of $\Theta(1/k)$ on the competitive ratio of any fair deterministic algorithm.

The lower bound on the accommodating ratio for any fair deterministic on-line algorithm is especially interesting because intuitively one first expects that it must be possible to design a "stupid" fair on-line algorithm which accommodates fewer than half the number of requests the best off-line algorithm could. Our result, however, shows that this is not possible when there are enough seats so that the optimal off-line algorithm can accommodate all requests.

We also consider randomized algorithms, and prove similar results bounding the performance of any "fair" randomized algorithm. In addition, we consider concrete algorithms; more specifically, First-Fit and Best-Fit, proving tight bounds on their performance for the unit price problem.

2 The unit price problem

If every ticket has the same price, regardless of how far the passenger is traveling, the seat reservation problem is similar to the problem of coloring an interval graph on-line. This is easy to see. The route the train travels from station 1 through station k is the section of the real line considered. The part of the route a passenger travels is an open interval, and the seat the passenger is assigned is the color the interval is given.

The problem of coloring an interval graph on-line has been well studied because of applications to dynamic storage allocation. Let $\chi(G)$ denote the chromatic number of a graph G and let $A(G)$ denote the number of colors that an on-line algorithm A uses in coloring G. Kierstead and Trotter [10] have shown that there exists an on-line algorithm A such that $A(G) \leq 3\chi(G)-2$ for all interval graphs G, and they prove a lower bound showing that this is best possible. When applying this result to trains, it says that processing requests for reservations on-line means that in order to accommodate all requests, the train need only have three times as many seats as would be necessary if the requests were processed off-line and an optimal algorithm were used. It also says that nearly this many seats can be necessary. One must recognize, however, that the construction used for the lower bound uses a large number of different endpoints for the intervals. In the train application, this would mean that there would have to be an incredibly large number of stations, so the lower bound may not be relevant for this application.

2.1 Bounds on the accommodating ratio

A more relevant measure of how well an algorithm does is the number of intervals which get colors 1 through n, rather than the total number of colors needed, i.e., the number of ticket requests that can be granted. We call the problem of maximizing this number the *unit price problem*. However, this number might not be too interesting if requests are such that no algorithm can grant very many, so, as is customary for on-line algorithms, we compare to an optimal fair off-line algorithm. This algorithm receives requests for tickets in the same arbitrary order. It can look at all of the requests before making any seat assignments, but it must process the requests in the given order and it must be fair.

An interesting case to consider is that in which there are enough seats so that the optimal fair on-line algorithm could accommodate all requests. (The decision as to how many cars the train should have is presumably based on expected ticket demand.)

Definition 1 Let $\mathit{earn}_A(I)$ denote how much a fair on-line algorithm A earns with the request sequence I, and let $\mathit{value}(I)$ denote how much could be earned if all requests in the sequence I were accommodated. A fair on-line algorithm A is *c-accommodating* if, for any sequence of requests which could all have been accommodated by the optimal fair off-line algorithm, $\mathit{earn}_A(I) \geq c \cdot \mathit{value}(I) - b$, where b is a constant which does not depend on the input sequence I. The *accommodating ratio* for A is the supremum over all such c. □

Thus if A is c-accommodating, A will always earn at least a fraction c of the total income possible, asymptotically. Note that in general the constant b is allowed to depend on k. This is because k is a parameter to the problem, and we quantify first over k. In most of the following proofs, this constant b is zero, so it is omitted.

Note that in the case where there are enough seats to accommodate all requests, the optimal fair off-line algorithm is polynomial time [11] since it is simply a matter of coloring an interval graph with the minimum number of colors. Recall that interval graphs are *perfect* [12], so the size of the largest clique is exactly the number of colors needed. Thus, when there is no pair of stations $(s, s+1)$ such that the number of people who want to be on the train between stations s and $s+1$ is greater than n, the optimal fair off-line algorithm will be able to accommodate all requests. The contrapositive is clearly also true; if there is a pair of stations such that the number of people who want to be on the train between those stations is greater than n, the optimal fair off-line algorithm will be unable to accommodate all requests. We will refer to the number of people who want to be on the train between two stations as the *density* between those stations.

First we show the somewhat surprising result that any fair on-line algorithm will be able to accommodate at least half of the requests, if the optimal fair on-line algorithm could have accommodated all the requests.

Since a request is also an interval, and since we use in the proofs that requests are intervals (which have subintervals, for instance), we use the terms *requests* and *intervals* interchangeably.

Theorem 2 Any fair on-line algorithm for the unit price problem is 1/2-accommodating.

Proof Consider any set of requests which the optimal fair off-line algorithm could accommodate with only n seats. Let S denote the seating assignment found by the fair on-line algorithm, and let U be the set of unseated intervals. It is only necessary to show that there are at least as many intervals in S as in U, and this can be done by assigning every interval in U to a distinct interval in S. Let S' be a seating assignment which is initialized to be the same as S, but which will be altered by the following process.

First order the intervals in U by increasing left endpoint (starting station), breaking ties arbitrarily. Now process these intervals, one by one, in increasing order. As an interval is processed, it will be removed from U, assigned to an interval in S, and used to alter an interval in S'. The following invariant will hold throughout: None of the intervals in U can be seated in S', and all of the intervals in U and S' could be seated by an optimal off-line algorithm.

The first part of the invariant holds initially since the algorithm is fair, and the second part holds initially by the definitions of U and S'.

For the induction part, we proceed as follows. For a given interval I, find a seat which is empty in S' from the point where the passenger wants to get on until at least the next station. Some such seat must exist because the entire set of intervals could be accommodated by the optimal algorithm. By the invariant, the interval I cannot be placed on that seat, so there must be a first interval J assigned to that seat in S' which overlaps the interval I. Assign the interval I to the interval J. Now remove I from U and replace J on this seat in S' by an interval K, which is as much of $I \cup J$ as will currently fit on that seat. Clearly, all of the intervals which are now seated in S' and all of the unseated intervals currently in U could be seated by the optimal algorithm, since this operation cannot increase the density anywhere. Furthermore, S' has been expanded, so none of the remaining intervals in U will fit. This process can be repeated. The order of processing ensures that each interval $I \in U$ gets assigned to a distinct interval in S. Thus, the on-line algorithm is 1/2-accommodating. $\square$

Before continuing, we introduce the notation $[s, t]$ to mean the interval from station s to station t, where $1 \leq s < t \leq k$.

As an example of a specific on-line algorithm, one might consider First-Fit, which always processes a new request by placing it on the first seat which is unoccupied for the length of that journey. Since this is a fair algorithm, the lower bound of 1/2 on the accommodating ratio for any fair deterministic algorithm applies, so First-Fit is at least 1/2-accommodating. In fact, however, it is at most $\frac{k}{2k-6}$-accommodating, so it is essentially as bad as any fair deterministic algorithm can be, in the worst case. The same applies to Best-Fit, which always processes a new request by placing it on a seat so it leaves as little total free space as possible on that seat immediately before and after that passenger's trip.

Theorem 3 The accommodating ratios for First-Fit and Best-Fit are no better than $\frac{k}{2k-6}$, for the unit price problem, assuming that $k \geq 4$ and $k \equiv 4 \pmod 6$.

Proof Throughout this proof, we call the algorithm First-Fit, but Best-Fit would make exactly the same seat assignments on this sequence of requests as First-Fit would, so one obtains the same result for Best-Fit. Assume that n is divisible by 3. To produce a "difficult" sequence of requests, the adversary starts with $n/3$ requests for the interval $[1, 2]$. Then, for $s = 0, 1, ..., (k-10)/6$, it gives $n/3$ requests for intervals $[6s+4, 6s+8]$, and First-Fit places them on the first $n/3$ seats. Now, it gives $n/3$ requests for the interval $[1, 4]$, and for $s = 1, 2, ..., (k-4)/6$, it gives $n/3$ requests for intervals $[6s, 6s+4]$. First-Fit places them on the next $n/3$ seats. Next, for $s = 0, 1, ..., (k-10)/6$, it gives $n/3$ requests for intervals $[6s+2, 6s+6]$, and First-Fit places them on the last $n/3$ seats. The last set of requests are, for $s = 0, 1, ..., (k-8)/2$, $n/3$ requests for intervals $[2s+1, 2s+3]$. Since none of the seats has a gap of more than 2 and none of these gaps begins at an odd-numbered station, First-Fit will be unable to place them anywhere. Note that the density is no greater than n anywhere, so the optimal fair off-line algorithm would be able to accommodate all the requests. Thus First-Fit accommodates $\frac{n}{3}(\frac{3(k-4)}{6}+2)$ requests and the optimal algorithm accommodates $\frac{n}{3}(\frac{3(k-4)}{6}+2+\frac{(k-6)}{2})$ requests, so the accommodating ratio is no better than $\frac{k}{2k-6}$. □

For other k, not congruent to 4 modulo 6, similar results hold. Using the same sequence of requests (and thus not using the last stations) gives upper bounds of the form $\frac{k-c_1}{2k-c_2}$ for constants c_1 and c_2 which depend only on $k \pmod 6$.

It is unknown if any algorithm has a better accommodating ratio than First-Fit, but no fair on-line algorithm has an accommodating ratio much better than 4/5.

Theorem 4 No deterministic fair on-line algorithm for the unit price problem is more than $\frac{8k-9}{10k-15}$-accommodating, when k is divisible by 3.

Proof The following is an adversary argument. Assume that n is divisible by 2. The adversary begins with $n/2$ requests for the intervals $[3s+1, 3s+3]$, for $s = 0, 1, ..., (k-3)/3$. Suppose the fair on-line algorithm places these $kn/6$ intervals such that after these requests there are exactly q_i seats which contain both an interval $[3i+1, 3i+3]$ and an interval $[3i+4, 3i+6]$, $i \in \{0, 1, ...(k-6)/3\}$. Then there are exactly q_i seats which are empty from station $3i+2$ to station $3i+5$. For each i, the adversary determines which of two cases hold:

- Case 1: $q_i \leq 3n/10$, or
- Case 2: $q_i > 3n/10$.

For those values of i where case 1 occurs, there will now be $n/2$ requests for the interval $[3i+2, 3i+5]$. The algorithm will accommodate q_i of these requests, but the optimal off-line algorithm would be able to accommodate all of them.

For those values of i where case 2 occurs, there will now be $n/2$ requests for the interval $[3i+2, 3i+4]$ and $n/2$ requests for the interval $[3i+3, 3i+5]$. The algorithm will accommodate $n - q_i$ of these requests, but the optimal off-line algorithm would be able to accommodate all of them.

Let S denote the set of indices for which case 1 is appropriate and $\bar{S}$ denote the set of indices for which case 2 is appropriate. The accommodating ratio is

$$\begin{aligned}
\frac{\frac{kn}{6}+\sum_{i\in S} q_i+\sum_{i\in\bar{S}}(n-q_i)}{\frac{kn}{6}+\sum_{i\in S}\frac{n}{2}+\sum_{i\in\bar{S}} n} &\leq \frac{\frac{k}{6}+\sum_{i\in S}\frac{3}{10}+\sum_{i\in\bar{S}}\frac{7}{10}}{\frac{k}{6}+\sum_{i\in S}\frac{1}{2}+\sum_{i\in\bar{S}} 1} = \frac{\frac{1}{2}+\sum_{i\in S}(\frac{1}{2}+\frac{3}{10})+\sum_{i\in\bar{S}}(\frac{1}{2}+\frac{7}{10})}{\frac{1}{2}+\sum_{i\in S} 1+\sum_{i\in\bar{S}}\frac{3}{2}} \\
&\leq \frac{\frac{1}{2}+\left(\frac{k-3}{3}\right)\left(\frac{4}{5}\right)}{\frac{1}{2}+\left(\frac{k-3}{3}\right)} = \frac{8k-9}{10k-15}.
\end{aligned}$$

The second inequality holds because in general $\frac{a}{c} = \frac{b}{d} < 1$ and $c < d$ implies that $\frac{e+xa+yb}{e+xc+yd} \leq \frac{e+(x+y)a}{e+(x+y)c}$. □

For other k, not divisible by 3, similar results hold.

2.2 Bounds on the competitive ratio

Note that the accommodating ratio which is calculated in the above theorems is not the same as the competitive ratio which is commonly used in evaluating on-line algorithms. Karlin, Manasse, Rudolph, and Sleator [13] defined an algorithm to be *c-competitive* if its cost on *any* sequence is within a factor of c of the cost of the optimal off-line algorithm, plus a constant, on that sequence. This value c is commonly referred to as the competitive ratio. In the case of seat reservations, where the goal is maximizing income instead of minimizing costs, the competitive ratio is still the ratio of how well the on-line algorithm does compared to how well the optimal off-line algorithm does on a worst case sequence of requests, but this ratio is no greater than one, instead of being at least one.

Definition 5 Let $earn_A(I)$ denote how much a fair on-line algorithm A earns with the request sequence I, and let $earn_{opt}(I)$ denote how much an optimal off-line algorithm could earn with the sequence I. A fair on-line algorithm A is *c-competitive* if, for any sequence of requests $earn_A(I) \geq c \cdot earn_{opt}(I) - b$, where b is a constant which does not depend on the input sequence I. The *competitive ratio* for A is the supremum over all such c. □

Proving an upper bound on the accommodating ratio, automatically gives the same bound for the competitive ratio, since the sets of sequences considered in the accommodating case is a subset of the ones considered in the competitive case. Thus, the competitive ratio can never be larger than the accommodating ratio. On the other hand, theorem 2 is false if, in the statement of the theorem, one substitutes "competitive" for "accommodating". Fair on-line algorithms can do very poorly compared to the optimal fair off-line algorithm.

Theorem 6 First-Fit and Best-Fit have competitive ratios which are no better than $\frac{2-\frac{1}{k-1}}{k-1}$ for the unit price problem.

Proof Again, we will state everything in terms of the First-Fit algorithm, but Best-Fit would behave exactly the same. We will assume that n is divisible by $k-1$. The request sequence will start with $n/(k-1)$ requests for each of the intervals $[1,2], [2,3], [3,4], ..., [k-1,k]$ which First-Fit will put in the first $n/(k-1)$ seats. Then there will be $n-n/(k-1)$ requests for $[1,k]$ intervals, which First-Fit will put in the remaining seats. At this point, the train will be full, but there will now be $n-n/(k-1)$ requests for each of the intervals $[1,2]$, $[2,3]$, $[3,4]$, ..., $[k-1,k]$, all of which First-Fit will be unable to accommodate. It will accommodate a total of $2n - n/(k-1)$ requests. The optimal off-line algorithm will put each of the original first intervals on a different seat, thus arranging that it can reject the longest intervals. Then, it will be able to accommodate all of the additional short intervals. Thus, it will accommodate n of the original short intervals, plus $(n - n/(k-1))(k-1)$ of the later short intervals. This gives a ratio of $\frac{2-\frac{1}{k-1}}{k-1}$. This argument assumes that $k \geq 3$, but the result trivially holds for $k = 2$ too. $\square$

As was the case with the accommodating ratio, we can show a fairly tight result for First-Fit and Best-Fit by proving that any fair algorithm does essentially as well as these algorithms do in the worst case.

Theorem 7 Any fair on-line algorithm for the unit price problem is $\frac{2}{k}$-competitive.

Proof Consider any fair on-line algorithm A and any request sequence I. Suppose A accepts m requests, while the optimal off-line algorithm OPT accepts m' requests, so the ratio $r = m/m'$. We will show that $r \geq 2/k$. Call the set of intervals which both A and the optimal off-line algorithm OPT accept, X; the set of intervals which A accepts, but OPT rejects, Y; and set of intervals which OPT accepts, but A rejects Z. Let $|S|$ denote the number of intervals in a set S of intervals, and let $ln(x)$ denote the length of an interval x. Let B denote the sum over all n seats of the number of unit intervals for which the on-line algorithm A does not seat any passengers.

Those intervals in Z which have length one can each be associated with a distinct unit length subinterval of an interval in Y. To see this, consider all intervals in Z of the form $[i, i+1]$, for some fixed station i. Call this set of intervals Z_i. Let $X_i \subset X$ be those intervals in X which include the subinterval $[i, i+1]$, and $Y_i \subset Y$ be those intervals in Y which include it. Since the on-line algorithm A rejected every interval in Z_i, that subinterval must be occupied on every seat, either by an interval in X_i or by an interval in Y_i. Since OPT is able to accommodate all intervals in $X_i \cup Z_i$, each interval in Z_i can be associated with a distinct subinterval $[i, i+1]$ of an interval in Y.

After this association is done, assume that u unit subintervals of intervals in Y have a unit interval from Z associated with them. Since OPT is able to accommodate all intervals in $X \cup Z$, the remaining intervals in Z must fit in the space $B + \sum_{y \in Y} ln(y) - u$. Since these intervals have length at least two, $|Z| \leq u + \frac{B + \sum_{y \in Y} ln(y) - u}{2} \leq \frac{B}{2} + \sum_{y \in Y} ln(y)$.

Thus, the ratio

$$\begin{aligned} r &= \frac{|X|+|Y|}{|X|+|Z|} \\ &\geq \frac{|X|+|Y|}{|X|+\frac{B}{2}+\sum_{y\in Y} ln(y)} \\ &= \frac{|X|+|Y|}{|X|+\frac{1}{2}\left(\sum_{y\in Y} ln(y)+n(k-1)-\sum_{x\in X} ln(x)\right)}. \end{aligned}$$

Any interval has length at least one and at most $k-1$, so

$$r \geq \frac{|X|+|Y|}{\frac{1}{2}(|X|+|Y|(k-1)+n(k-1))}.$$

In addition, any fair algorithm must accept at least the first n requests from any request sequence, so $|X| \geq n$. Thus,

$$r \geq \frac{2(|X|+|Y|)}{|X|+(|X|+|Y|)(k-1)} = \frac{2}{\frac{|X|}{|X|+|Y|}+(k-1)} \geq \frac{2}{k}.$$

This shows that the competitive ratio for any fair on-line algorithm is at least $2/k$. □

One can combine the ideas from theorems 4 and 6 to prove that no fair on-line algorithm could have a competitive ratio better than $8/(k+5)$.

Theorem 8 No deterministic fair on-line algorithm for the unit price problem is more than $8/(k+5)$-competitive.

Proof The following is an adversary argument, so the request sequence depends on the algorithm's behavior. Assume that n is divisible by 2. The adversary begins with $n/2$ pairs of requests for $[1,2]$ and $[k-1,k]$ intervals. Suppose the fair on-line algorithm places them such that after these requests there are exactly q seats which contain two intervals. Then $n-2q$ of the seats have exactly one short interval scheduled. Next there will be q requests for $[1,k]$ intervals, followed by $(n-2q)/2$ requests for $[1,k-1]$ intervals, $(n-2q)/2$ requests for $[2,k]$ intervals, and q requests for $[2,k-1]$ intervals, all of which can be accommodated. Now the train is full. One of two cases will occur:

- Case 1: $q \geq n/4$, or
- Case 2: $q < n/4$.

If case 1 occurs, there will now be q requests for each of the intervals $[1,2]$, $[2,3]$, $[3,4]$,..., $[k-1,k]$, none of which can be accommodated by the fair on-line algorithm. On the other hand, the optimal fair off-line algorithm could put each of the short intervals on a separate seat, so that it would be unable to accommodate the q $[1, k]$ intervals, but all of the other intervals would fit. The on-line algorithm is able to accommodate $2n$ requests, while the off-line algorithm can accommodate $2n - q + q(k-1)$ requests. Since $q \geq n/4$, this ratio is less than $\frac{2n}{2n+(n/4)(k-2)} = 8/(k+6)$.

If case 2 occurs, there will now be $(n-2q)/2$ requests for each of the intervals $[2, 3]$, $[3, 4]$,..., $[k-1, k]$, none of which can be accommodated by the fair on-line algorithm. On the other hand, the optimal fair off-line algorithm could pair up the short intervals, putting two per seat, so that it would be unable to accommodate the $(n-2q)/2$ $[2, k]$ intervals, but all of the other intervals would fit. The on-line algorithm is able to accommodate $2n$ requests, while the off-line algorithm can accommodate $2n - (n-2q)/2 + ((n-2q)/2)(k-2)$ requests. Since $q < n/4$, this ratio is less than $\frac{2n}{2n+(n/4)(k-3)} = 8/(k+5)$. This argument assumes that $k \geq 4$, but the result clearly holds for $k = 2$ and $k = 3$, too. □

3 The proportional price problem

Another common situation is the one in which the price of the ticket is proportional to the length of the trip. In this section, we assume that the price of a ticket from station s to station t is proportional to $t - s$. Maximizing income in this case is solving the *proportional price problem*. Now if there are many stations, no fair on-line algorithm can ensure that revenues from ticket sales will be as much as a constant factor of the revenues which could theoretically be earned by the optimal fair off-line algorithm, even if one assumes that the optimal fair off-line algorithm could have accommodated all of the requests. We have, however, obtained upper and lower bounds on the accommodating and competitive ratios of any fair on-line algorithm, and these upper and lower bounds are within a constant factor of each other.

Theorem 9 Any fair on-line algorithm for the proportional price problem is $\frac{1}{k-1}$-competitive and $\frac{1}{k-1}$-accommodating.

Proof Consider any sequence of requests. If the on-line algorithm assigns seats in such a way that it collects less in revenues than the optimal fair off-line algorithm, then it must reject at least one request, so it must assign at least one interval to each of the n seats. The optimal fair off-line algorithm can do no better than to fill every seat at all times, so its income is no more than $n(k-1)$. This gives a ratio of at least $\frac{1}{k-1}$. □

Although this lower bound is trivial, it is tight if $k-1 = n$. In that case, consider a fair on-line algorithm which puts the first n requests on different seats, no matter what they are. Then, a request sequence consisting of $[1,2],[2,3]...[k-1,k]$, followed by $n-1$ requests for the interval $[1,k]$ gives the ratio $\frac{k-1}{(k-1)+(n-1)(k-1)} = \frac{1}{k-1}$.

Theorem 10 First-Fit and Best-Fit have accommodating ratios which are no better than $\frac{4}{k+2}$ for the proportional price problem.

Proof Again, we state everything in terms of the First-Fit algorithm, but Best-Fit would behave exactly the same. We assume that n is divisible by 2. The request sequence will start with $n/2$ requests for $[1,2]$ intervals and $n/2$ requests for $[3,4]$ intervals, all of which First-Fit will put in the first $n/2$ seats. Then there will be $n/2$ requests for $[1,3]$ intervals, which First-Fit will put in the last $n/2$ seats. Finally, there will be $n/2$ requests for $[2,k]$ intervals, which First-Fit will be unable to accommodate. The total length it accommodates is $2n$ units. The optimal off-line algorithm will put the $[1,2]$ intervals in the first $n/2$ seats, but the $[3,4]$ and $[1,3]$ intervals in the last $n/2$ seats. Then it will also be able to accommodate the $[2,k]$ intervals. The total length the optimal algorithm accommodates is thus $2n+(k-2)(n/2)$. This gives a ratio of $\frac{2n}{2n+(k-2)(n/2)} = \frac{4}{k+2}$. □

One can prove that no fair on-line algorithm has an accommodating ratio better than $\frac{4+2\sqrt{13}}{3+2\sqrt{13}+k} \approx 11.211/(10.211+k)$.

Theorem 11 No deterministic fair on-line algorithm for the proportional price problem is more than $\frac{4+2\sqrt{13}}{3+2\sqrt{13}+k}$-accommodating, when $k \geq 4$.

Proof The following is an adversary argument and is fairly similar to the proof of theorem 4. We will refer to $[1,k]$ intervals as *long* intervals. The request sequence I, begins with $m = \lfloor (4-\sqrt{13})n \rfloor$ pairs of $[1,2]$ and $[3,4]$ intervals. Suppose the fair on-line algorithm A places them such that after these requests there are exactly q seats which contain two intervals. One of two cases will occur:

- Case 1: $q \geq n-2m = 2\lceil (\sqrt{13}-7/2)n \rceil$, or
- Case 2: $q < n-2m$.

If case 1 occurs, the remainder of the request sequence I will consist of $(n-q)/2$ pairs of $[1,3]$ and $[2,4]$ intervals, followed by q requests for long intervals. The on-line algorithm will be unable to accommodate any of the requests for long intervals, so the total length it accommodates is $2m+2(n-q)$. On the other hand, the optimal fair off-line algorithm could match each interval of length two with an interval of length one (since $n-q \leq n-(n-2m) = 2m$) and place any extra intervals of length one in pairs, two per seat. This fills up

$$(n-q) + (2m-(n-q))/2 = m + (n-q)/2$$

seats, so there are $(n+q)/2 - m$ empty seats when the requests for the long intervals arrive. Thus, $(n+q)/2 - m \geq n - 2m = 2\lceil(\sqrt{13}-7/2)n\rceil$ of the long intervals will fit, and the accommodating ratio is

$$\begin{aligned}
\frac{2(m+n-q)}{2(m+n-q)+(n-2m)(k-1)} &\leq \frac{6m}{6m+(n-2m)(k-1)} \\
&\leq \frac{6(4-\sqrt{13})n}{6(4-\sqrt{13})n+(2\sqrt{13}-7)n(k-1)} \\
&= \frac{4+2\sqrt{13}}{3+2\sqrt{13}+k}.
\end{aligned}$$

If case 2 occurs, then the remainder of the request sequence I will consist of $n-2m+q$ requests for $[1,4]$ intervals, followed by $m-q$ requests for long intervals. The on-line algorithm will be unable to accommodate any of the requests for long intervals, so the total length it accommodates is $2m+3(n-2m+q) = 3n+3q-4m$. The optimal fair off-line algorithm would seat each $[1,2]$ interval request on the same seat as some $[3,4]$ interval request, so the optimal algorithm would satisfy all of the requests. Thus,

$$\begin{aligned}
earn_A(I) &= \frac{earn_A(I)}{value(I)} \cdot value(I) \\
&= \frac{3n+3q-4m}{3n+3q-4m+(m-q)(k-1)} value(I) \\
&= \frac{3n+3q-4m-10-3(k-1)}{3n+3q-4m+(m-q)(k-1)} value(I) + (10+3(k-1)) \\
&\leq \frac{6n-10m-10-3(k-1)}{6n-10m+(3m-n)(k-1)} value(I) + (10+3(k-1)) \\
&\leq \frac{6n-10(4-\sqrt{13})n+10-10-3(k-1)}{6n-10(4-\sqrt{13})n+(3(4-\sqrt{13})n-n)(k-1)-3(k-1)} value(I) + (10+3(k-1)) \\
&\leq \frac{6n-10(4-\sqrt{13})n}{6n-10(4-\sqrt{13})n+(3(4-\sqrt{13})n-n)(k-1)} value(I) + (10+3(k-1)) \\
&= \frac{4+2\sqrt{13}}{3+2\sqrt{13}+k} value(I) + (10+3(k-1)).
\end{aligned}$$

Thus no fair on-line algorithm has an accommodating ratio which is better than $\frac{4+2\sqrt{13}}{3+2\sqrt{13}+k}$.

□

Corollary 12 No deterministic fair on-line algorithm has a competitive ratio better than $\frac{4+2\sqrt{13}}{3+2\sqrt{13}+k}$.

4 Upper bounds for randomized algorithms

After proving upper bounds on how well deterministic algorithms can do, one is tempted to look for a good randomized algorithm. When analyzing a randomized on-line algorithm, one must compare it to some adversary, an *oblivious adversary*, an *adaptive on-line adversary*, or an *adaptive off-line adversary*. Ben-David, Borodin, Karp, Tardos, and Wigderson [14] have shown that it is not very interesting to consider adaptive off-line adversaries since if there is a randomized algorithm which is c-competitive against an adaptive off-line adversary, there is also a deterministic algorithm which is c-competitive. It can easily be seen that the techniques in their proof give the same result for the accommodating ratio as for the competitive ratio. The only significant difference is that the adversary is restricted in which request sequences it is allowed to produce (of course the direction of the inequality must also be switched).

For the same reason, the following two results in [14] also hold if one substitutes "accommodating" for "competitive". Their theorem 2.2 states

> Suppose G is α-competitive against any on-line adaptive adversary and there is a β-competitive randomized algorithm against any oblivious adversary. Then G is $\alpha \circ \beta$-competitive against any off-line adaptive adversary.

and corollary 2.2 states

> If an α-competitive randomized strategy against any adaptive on-line adversary exists, then it is $\alpha \circ \alpha$-competitive against any adaptive off-line adversary and thus there is a deterministic $\alpha \circ \alpha$-competitive strategy.

In [14], a notation slightly different from ours is used. We would say that an algorithm has competitive ratio c and is c-competitive if $earn_A(I) \geq c \cdot earn_{opt}(I) - b$, for all request sequences I. In [14], they are concerned with minimizing costs, rather than optimizing earnings. They would say that an algorithm has competitive ratio c if $cost_A(I) \leq c \cdot cost_{opt}(I) + b$, for all request sequences I, and such an algorithm is called α-competitive, where α is the linear function $\alpha(x) = c \cdot x + b$. Since we are dealing with linear functions, the function composition in the statements above corresponds to multiplication of the ratios.

Thus, our theorems 4, 8, 11 and corollary 12, together with their corollary 2.2, automatically give some bounds on how well a randomized algorithm can do against an adaptive on-line adversary—simply take the square roots of the values from those results. However, one can do better.

The bounds from theorems 4 and 8 also hold for randomized algorithms against adaptive on-line adversaries. In the proofs of both theorems, the adversary gives the algorithm some sets of pairs of unit length intervals, checks how many seats have pairs of these intervals at the end of this, and performs differently depending on how these numbers q_i (in theorem 8 just q) compare to some specified value v. The adaptive on-line adversary A behaves differently depending on the expected number of seats on which the randomized algorithm R will place pairs of these intervals; call these numbers $E[q_i]$. Before presenting R with any requests, A decides on case 1 or case 2 for each set of pairs by comparing this value $E[q_i]$ to v. Now A will present the initial pairs of requests, placing each interval from a pair on a different seat if case 1 was chosen, or placing both requests from a pair on the same seat if case 2 was chosen. Meanwhile, the randomized algorithm R will place some number q'_i as pairs. The adversary A will continue acting as the adversary from the proof, in the case it has chosen, using q'_i in place of q_i. Since the amounts earned in each case by the on-line algorithm and by the adversary are linear functions of the q_i, by the linearity of expectations, the expected amounts earned are these same amounts with $E[q_i]$ substituted for q_i. Thus the ratios are unchanged.

This argument almost works for theorem 11. The only problem with this is in case 1. If $q' < n-2m$, then the off-line algorithm will only be able to accommodate q' of the long intervals. If, however, we let m have the value $n/3$ (assume n is divisible by three), then we will always have $q' \leq m = n - 2m$, so the off-line algorithm always accommodates exactly q' of the long intervals. If one divides into the two cases depending on whether $E[q]$ is at least or no more than $\frac{(\sqrt{257}-15)n}{6}$, then in case 1, the ratio will be $\frac{2(m+n-E[q])}{2(m+n-E[q])+E[q](k-1)} \leq \frac{11+\sqrt{257}}{9+\sqrt{257}+2k}$, and in case 2, the ratio will be $\frac{3n+3E[q]-4m}{3n+3E[q]-4m+(m-E[q])(k-1)} \leq \frac{11+\sqrt{257}}{9+\sqrt{257}+2k}$. Thus no fair randomized algorithm for the proportional price problem has an accommodating ratio which is better than $\frac{11+\sqrt{257}}{9+\sqrt{257}+2k} \approx \frac{13.516}{12.516+k}$ against an adaptive on-line adversary.

The result from theorem 4 also holds for randomized algorithms against oblivious adversaries. Since the requests the adversary makes only depend on the case chosen and not on q in any other way, the adaptive on-line adversary described above for proving a bound on the accommodating ratio for the unit price problem is actually an oblivious adversary. Unfortunately, it does not seem to be quite so simple to prove results corresponding to theorems 8 and 11 for randomized algorithms against oblivious adversaries. This leaves open the possibility that there are randomized algorithms for these problems which can be proven to do well against an oblivious adversary. We leave this as an open problem: Do there exist good randomized algorithms for these two problems?

5 Summary of results

We list a brief schematic summary of the results obtained in this paper, leaving out some details which can be found in the theorems in the previous sections. The table shows upper and lower bounds on the competitive and accommodating ratios obtained for the unit price and the proportional price seat reservation problems. It shows results for deterministic algorithms and for randomized algorithms against adaptive on-line adversaries and against oblivious adversaries.

Ratio \ Algorithm	Deterministic	Randomized – vs. adaptive	Randomized – vs. oblivious
Accommodating – unit price	$\frac{1}{2} \leq r \leq \frac{8k-9}{10k-15}$	$r \leq \frac{8k-9}{10k-15}$	$r \leq \frac{8k-9}{10k-15}$
Competitive – unit price	$\frac{2}{k} \leq r \leq \frac{8}{k+5}$	$r \leq \frac{8}{k+5}$	$r \leq 1$
Accommodating/ Competitive – proportional price	$\frac{1}{k-1} \leq r \leq \frac{4+2\sqrt{13}}{3+2\sqrt{13}+k}$	$r \leq \frac{11+\sqrt{257}}{9+\sqrt{257}+2k}$	$r \leq 1$

Of course, the lower bounds on the ratio r for the deterministic algorithms also hold for the randomized algorithms. The upper bounds of 1 indicate a lack of nontrivial results.

In addition, we have shown some upper bounds for the specific algorithms, First-Fit and Best-Fit, which are better than the general ones listed above.

First-Fit / Best-Fit	Unit price	Proportional price
Accommodating ratio	$\frac{1}{2} \leq r \leq \frac{k}{2k-6}$	$\frac{1}{k-1} \leq r \leq \frac{4}{k+2}$
Competitive ratio	$\frac{2}{k} \leq r \leq \frac{2-\frac{1}{k-1}}{k-1}$	$\frac{1}{k-1} \leq r \leq \frac{4}{k+2}$

It is unknown if any algorithms perform better than First-Fit or Best-Fit. We leave this as an interesting open problem.

Acknowledgments

We would like to thank Susanne Albers for pointing out that our results for the First-Fit algorithm also apply to Best-Fit. We would also like to thank an anonymous referee for motivating us to prove some stronger results.

References

[1] Elias Koutsoupias and Christos H. Papadimitriou. Beyond Competitive Analysis. In *Proc. 35th IEEE Symp. on Foundations of Computer Science*, pages 394–400, 1994.

[2] Allan Borodin, Sandy Irani, Prabhakar Raghavan, and Baruch Schieber. Competitive Paging with Locality of Reference. *J. Comp. Sys. Sci.*, 50:244–258, 1995.

[3] Richard M. Karp, Umesh V. Vazirani, and Vijay V. Vazirani. An Optimal Algorithm for On-Line Bipartite Matching. In *Proc. 22nd Annual ACM Symp. on Theory of Computing*, pages 352–358, 1990.

[4] Prabhakar Raghavan and Eli Upfal. Efficient Routing in All-Optical Networks. In *Proc. 26th Annual ACM Symp. on Theory of Computing*, pages 134–143, 1994.

[5] Baruch Awerbuch, Yossi Azar, Amos Fiat, Stefano Leonardi, and Adi Rosén. On-Line Competitive Algorithms for Call Admission in Optical Networks. In *Proc. Fourth Annual European Symp. on Algorithms*, volume 1136 of *Lecture Notes in Computer Science*, pages 431–444. Springer-Verlag, 1996.

[6] Juan A. Garay, Inder S. Gopal, Shay Kutten, Yishay Mansour, and Moti Yung. Efficient On-Line Call Control Algorithms. *J. Algorithms*, 23:180–194, 1997.

[7] Amotz Bar-Noy, Ran Canetti, Shay Kutten, Yishay Mansour, and Baruch Schieber. Bandwidth Allocation with Preemption. In *Proc. 27th Annual ACM Symp. on Theory of Computing*, pages 616–625, 1995.

[8] Richard J. Lipton and Andrew Tomkins. Online Interval Scheduling. In *Proc. Fifth Annual ACM-SIAM Symp. on Discrete Algorithms*, pages 302–311, 1994.

[9] Baruch Awerbuch, Yair Bartal, Amos Fiat, and Adi Rosén. Competitive Non-Preemptive Call Control. In *Proc. Fifth Annual ACM-SIAM Symp. on Discrete Algorithms*, pages 312–320, 1994.

[10] Henry A. Kierstead and Jr. William T. Trotter. An Extremal Problem in Recursive Combinatorics. *Congressus Numerantium*, 33:143–153, 1981.

[11] Fănică Gavril. Algorithms for Minimum Coloring, Maximum Clique, Minimum Covering by Cliques, and Maximum Independent Set of a Chordal Graph. *SIAM J. Comput.*, 1(2):180–187, 1972.

[12] Tommy R. Jensen and Bjarne Toft. *Graph Coloring Problems.* John Wiley & Sons, 1995.

[13] Anna R. Karlin, Mark S. Manasse, Larry Rudolph, and Daniel D. Sleator. Competitive Snoopy Caching. *Algorithmica*, 3:79–119, 1988.

[14] S. Ben-David, A. Borodin, R. Karp, G. Tardos, and A. Wigderson. On the Power of Randomization in On-Line Algorithms. *Algorithmica*, 11(1):2–14, 1994.