Algorithmica (2003) 36: 209–224
DOI: 10.1007/s00453-003-1013-x

Algorithmica

Dynamic TCP Acknowledgment and Other Stories about $e/(e-1)$[1]

Anna R. Karlin,[2] Claire Kenyon,[3] and Dana Randall[4]

Abstract. We present the first optimal randomized online algorithms for the TCP acknowledgment problem [3] and the Bahncard problem [5]. These problems are well known to be generalizations of the classical online ski-rental problem, however, they appeared to be harder. In this paper we demonstrate that a number of online algorithms which have optimal competitive ratios of $e/(e-1)$, including these, are fundamentally no more complex than ski rental. Our results also suggest a clear paradigm for solving ski-rental-like problems.

Key Words. TCP acknowledgment, Online algorithms, Ski-rental, Basis inputs.

1. Introduction.

Consider the following online problems:

SKI RENTAL. Suppose you are about to go skiing for the first time in your life. Naturally, you ask yourself whether to rent skis or to buy them. Renting skis costs, say, \$30, whereas buying skis costs, say, \$300. Your goal is minimize your total cost on all future ski trips. Unfortunately, you do not know how many such trips there will be. You must make the decision online.

This is perhaps the simplest and most well-understood online problem. There is a natural deterministic online algorithm that achieves a competitive ratio of 2 [8], and a randomized online algorithm that achieves a competitive ratio of $e/(e-1)$ (which is about 1.58) in the limit as the ratio between the buy cost and the rent cost becomes large [7].

DYNAMIC TCP ACKNOWLEDGMENT. A stream of packets arrive at a destination. The TCP protocol requires that these packets be acknowledged. However, the possibility exists of using a single acknowledgment packet to simultaneously acknowledge multiple outstanding packets, thereby reducing the overhead of the acknowledgments. On the

[1] A preliminary version of this paper appeared in *Proceedings of the* 33*rd ACM Symposium on Theory of Computing*, Crete, July 2001, ACM Press, New York, pp. 502–509. The first author was supported in part by NSF EIA-9870740 and a grant from the US–Israel Binational Science Foundation (BSF). Part of this work was done while the second author was visiting the University of Washington at Seattle and Microsoft Research. The third author was supported in part by NSF CCR-0105639 and an Alfred P. Sloan research fellowship. Part of this work was done while this author was visiting Microsoft Research.

[2] Department of Computer Science and Engineering, University of Washington, Seattle, WA, USA. karlin@cs.washington.edu.

[3] LIX, Ecole Polytechnique, 91128 Palaiseau, France. kenyon@lix.polytechnique.fr.

[4] College of Computing and School of Mathematics, Georgia Institute of Technology, Atlanta, GA 30332, USA. randall@cc.gatech.edu.

Received April 15, 2002; revised November 18, 2002. Communicated by S. Albers.
Online publication March 14, 2003.

other hand, delaying acknowledgments too much can interfere with the TCP's congestion control mechanisms, and thus it is undesirable to allow the latency between a packet's arrival time and the time at which the acknowledgment is sent to increase too much.

This motivated Dooly et al. to define the following problem [3], [4]. The input is a sequence of n arrival times $a_1, a_2, \ldots, a_n$. The output is a set of times $t_1, \ldots, t_k$ at which acknowledgments occur such that

$$k + \sum_{1 \le j \le k} \text{latency}(j)$$

is minimized, where

$$\text{latency}(j) = \sum_{i \text{ s.t. } t_{j-1} < a_i \le t_j} (t_j - a_i).$$

(It is required that $t_k \ge a_n$ and $k \ge 1$.) The parameter k is called the *acknowledgment cost* and $\sum_j \text{latency}(j)$ is called the *latency cost* of the algorithm on that input. Of course in practice the acknowledgment times must be chosen online without knowledge of when future arrivals will occur. See Figure 1 for a pictorial representation.

Dooly et al. showed that the natural algorithm which waits until the latency since the previous acknowledgment equals the cost of the acknowledgment has a competitive ratio of 2. Subsequently, Seiden [11], and independently Noga [9], obtained a lower bound of $e/(e-1)$ on the competitive ratio of randomized online algorithms for this problem.

The variant of the problem where packet j has weight w_j and one wishes to minimize $k + \sum_j w_j \text{latency}(j)$ was also studied, but it is easy to see that for our purposes it is equivalent to the original problem.

The Bahncard Problem. The Bahncard problem models online ticket purchasing on the German Deutsche Bundesbahn, where one can opt to buy a Bahncard that entitles the traveler to a 50% discount on all trips within one year of the purchase date. In the more general setting, the (C, β, T) Bahncard problem offers a Bahncard for cost C which

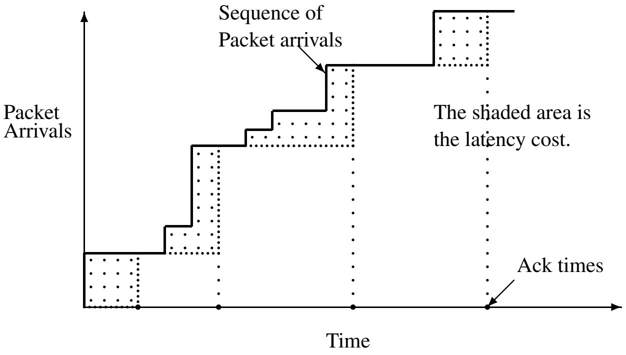


Fig. 1. A pictorial representation of TCP acknowledgment.

permits the price of tickets to be discounted by $\beta \in [0, 1]$ for time T from the date of purchase. This extends ski rental in three ways: first, the benefit of purchasing (instead of renting) comes with a time limit, second, the trip (rental) costs vary, and third, purchasing merely offers a discount rather than a free ride.

Fleischer [5] introduced this model and provided a deterministic algorithm with competitive ratio 2. He also presented an $e/(e-1+\beta)$ lower bound on the randomized competitive ratio. For the case that the Bahncard never expires, Fleischer presented a matching upper bound and conjectured that this is also the bound for finite expiration periods.

1.1. *Our Results*. The main contribution of this paper is a new randomized online algorithm for TCP acknowledgment that achieves the best possible competitive ratio of $e/(e-1)$. We extend these ideas to solve the Bahncard problem for finite expiration periods, thereby settling Fleischer's conjecture positively by presenting an $e/(e-1+\beta)$-competitive algorithm. We also generalize our solution to get an optimal algorithm for the case where the discount rate for different trips varies.

Our secondary contribution is to show that, despite the appearance of greater complexity, these problems are just glorified versions of ski rental (in a somewhat interesting and obscure way). We believe that there may be something fundamental, if simple, going on here in precisely this sense: online problems with competitive ratios of $e/(e-1)$, of which there are many examples, may need to abstract the ski-rental "phenomenon." Finally, our results suggest a clear paradigm for solving online problems of this nature.

The rest of the paper is organized as follows. In Sections 2 and 3 we present the $e/(e-1)$-competitive randomized algorithm for TCP acknowledgment and its analysis. In Section 4 we explain the connection with ski rental. In the following section we present the solution to the Bahncard problem. The general paradigm for solving problems of this nature is briefly discussed in Section 6.

1.2. *Definitions*. We consider randomized online algorithms against oblivious adversaries. (See, e.g., [2] for a more detailed discussion of randomized online algorithms.) An oblivious adversary must choose the entire request sequence without knowledge of the coin tosses made by the algorithm, but with full knowledge of the randomized algorithms. One measures the competitiveness of such algorithms as follows. A randomized, online algorithm A is *c-competitive against an oblivious adversary* if there exists a constant α such that for all oblivious adversaries

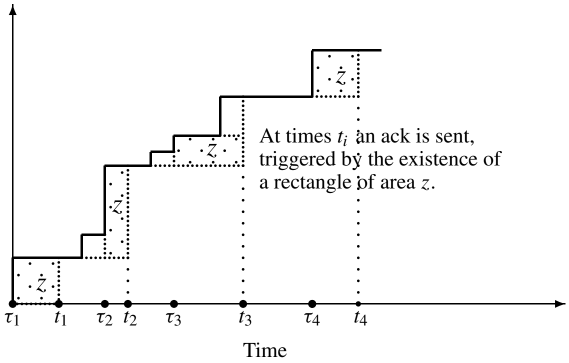
$$E(C_A(I)) \leq cC_{OPT}(I) + \alpha,$$

where I is the request sequence generated by the adversary, $E(C_A(I))$ is the expected cost of algorithm A on input I, and $C_{OPT}(I)$ is the optimal cost on input I.

The randomized competitive ratio is then the infimum over c such that there is a c-competitive algorithm against an oblivious adversary.

2. Randomized Algorithm for TCP Acknowledgment.

The most natural approach to the construction of a TCP acknowledgment problem is to consider algorithms which



At times t_i an ack is sent, triggered by the existence of a rectangle of area z.

Fig. 2. Algorithm A_z.

probabilistically vary the amount of latency they tolerate until an acknowledgment is performed. Unfortunately, Noga and Seiden have shown that the most natural variants of such algorithms do not give an $e/(e-1)$ competitive ratio [10].

Our solution defines a one parameter family of deterministic online algorithms A_z, where $0 \leq z \leq 1$, that measures cost that can be directly charged to the optimal offline algorithm.

Algorithm A_z is defined as follows: Let $P(t, t')$ be the set of packets that arrive between time t and time t', i.e., the set of i such that $t < a_i \leq t'$. Suppose that the ith acknowledgment occurred at time t_i (and assume that $t_0 = 0$). Algorithm A_z performs the next acknowledgment at the first time $t_{i+1} > t_i$ for which there is a time τ_{i+1}, $t_i \leq \tau_{i+1} \leq t_{i+1}$, such that $P(t_i, \tau_{i+1})(t_{i+1} - \tau_{i+1}) = z$. Intuitively, this time is chosen so that, given the fact that the previous acknowledgment occurred at time t_i, in hindsight, z units of latency cost would have been saved by performing an additional acknowledgment at time τ_{i+1}. See Figure 2.

We define a randomized algorithm A that chooses z between 0 and 1 according to the probability density function $p(z) = e^z/(e-1)$ and then runs A_z.

Theorem 1. *Let A be the randomized algorithm that picks z between* 0 *and* 1 *according to the probability density function $p(z) = e^z/(e-1)$ and runs the deterministic algorithm A_z. The competitive ratio of A is $e/(e-1)$.*

We will find the following pictorial representation of the input and algorithm very useful in explaining our algorithms and proofs. Figure 1 shows an example of the TCP acknowledgment problem. The x-axis represents time and on the y-axis we plot the number of packet arrivals by that time. The sequence of packet arrivals defines a step function of equation $y(t) = |P(0, t)|$. The dots on the x-axis indicate times at which acknowledgments are sent by the algorithm. The algorithm defines a staircase curve g such that if acknowledgments are sent at times $t_1, t_2, \ldots, t_k$, then for $t_i \leq t < t_{i+1}$, $g(t)$ is constant and equal to $|P(0, t_i)|$. It is easy to see that the latency cost of the algorithm is exactly the sum of the areas of the shaded regions on the figure, i.e., the area between the curve of the algorithm and the curve of the packet arrivals.

Figure 2 shows an example of what algorithm A_z might look like. (Subsequent figures will be simplified by drawing the packet arrival sequence as a straight line.)

3. Analysis of the Algorithm.

The main difficulty of our proof lies in the analysis of algorithm A_z for general values of z. As a warmup, we start with the (simpler) analysis of algorithm A_1.

3.1. *Analysis of Algorithm A_1.*

The following lemma, although not central to the analysis, will help clarify the picture.

LEMMA 2. *Without loss of generality, we can assume that the optimal algorithm sends an acknowledgment between any pair of successive acknowledgments of algorithm A_1.*

PROOF. Consider an arbitrary input sequence I, and suppose that A_1 acknowledges at times t_i. Consider any sequence S of acknowledgments, and assume that it does not send any acknowledgment in (t_i, t_{i+1}). Enrich this sequence by sending an additional acknowledgment at time τ_{i+1}. The acknowledgment cost increases by 1, and the latency cost decreases by at least 1, so this new sequence is at least as good as S. Hence there is an optimal sequence which sends at least one acknowledgment in each interval (t_i, t_{i+1}). □

With the help of this representation, we are ready to analyze algorithm A_1.

LEMMA 3. *Algorithm A_1 is 2-competitive.*

PROOF. Consider an arbitrary input sequence I. From Lemma 2, the cost C_{A_1} of A_1 on input I satisfies

$$C_{A_1} \leq n_{OPT} + \text{latency}(OPT) + \text{latency}(A_1 \backslash OPT),$$

where n_{OPT} is the number of acknowledgments performed by OPT on input I, and latency$(A_1 \backslash OPT)$ is the latency incurred by A_1 that is not incurred by OPT. However, it is easy to see from Figure 3 that latency$(A_1 \backslash OPT)$ is precisely the area of a set of rectangles (shaded in the figure), where each rectangle has its left side at the time when OPT sends an acknowledgment, and its right side at the following time when A_1 sends an acknowledgment. By definition of algorithm A_1, all these rectangles have area at most 1. Hence, latency$(A_1 \backslash OPT) \leq n_{OPT}$ and we obtain that

$$C_{A_1} \ \leq \ C_{OPT} + n_{OPT} \leq 2C_{OPT}. \qquad \square$$

3.2. *Analysis of Algorithm A_z.*

We now turn to the analysis of algorithm A_z. First, we need to understand how the cost of algorithm A_z relates to the cost of OPT on any input. Let $n_z(I)$ denote the number of acknowledgments of algorithm A_z on input I. Looking

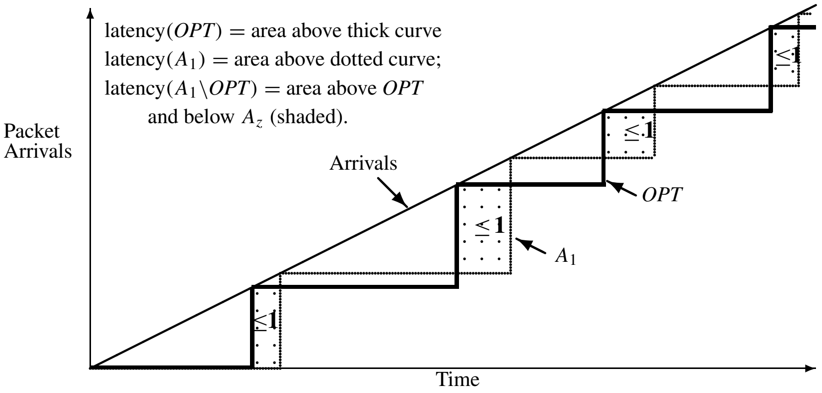

latency(*OPT*) = area above thick curve
latency(A_1) = area above dotted curve;
latency($A_1 \backslash OPT$) = area above *OPT* and below A_z (shaded).
Packet Arrivals
Arrivals
OPT
A_1
≤ 1
≤ 1
≤ 1
≤ 1
Time

Fig. 3. Proof of Lemma 3.

at Figure 4, we see that the latency cost of A_z is bounded by

- the area above the *OPT* curve, plus
- the area under *OPT* and over A_z (the dark shaded area in Figure 4), minus
- the area, denoted $E_z(I)$, under A_z and over *OPT* (the lightly shaded area in Figure 4).

The first term is just $C_{OPT}(I) - n_{OPT}(I)$, the latency cost of *OPT*. The second term can be analyzed as in the proof of Lemma 3: it is just a set of $n_{OPT}(I)$ rectangles, each of which has area at most z by definition of A_z, for a total of at most $zn_{OPT}(I)$. Hence,

$$C_{A_z}(I) \leq n_z(I) + C_{OPT} - n_{OPT}(I) + zn_{OPT}(I) - E_z(I). \tag{1}$$

Lemma 4. *Let n_z denote the number of acknowledgments of algorithm A_z and let n_{OPT} denote the number of acknowledgments of the optimal algorithm on some input I. Then*

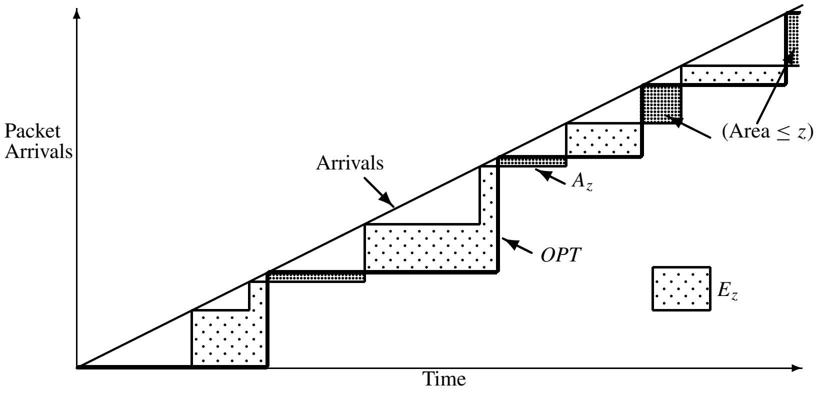


Fig. 4. Analysis of A_z, proof of (1) and definition of E_z.

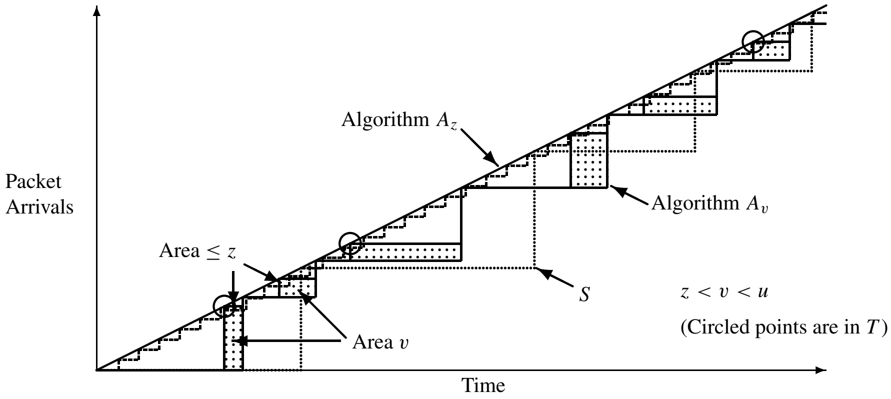


Fig. 5. Proof of Lemma 4: setup.

the area E_z above the optimal curve and below the A_z curve on input I is at least

$$E_z \geq \int_z^1 n_w \, dw - (1-z)n_{OPT}.$$

PROOF. Fix an input I. Let $L(n, z)$ be the minimum, over all acknowledgment sequences S with n acknowledgments, of the area above the S curve that is below the A_z curve. We will prove a lower bound on $L(n_u, z)$ for all $u \geq z$.

We claim that for any $u > v \geq z$,

$$L(n_u, z) \geq (v - z)(n_v - n_u) + L(n_v, z). \tag{2}$$

The proof is illustrated in Figures 5 and 6. Figure 5 shows three acknowledgment sequences for the given input: S, the acknowledgment sequence with n_u acknowledgments that minimizes $L(n_u, z)$, and the acknowledgment sequences of A_v and A_z. The shaded areas in Figure 5 represent the n_v area v rectangles which caused algorithm A_v to send an acknowledgment. At most n_u such rectangles intersect the curve S, since this is exactly the number of times the curve S meets the arrival curve. Therefore, there are at least $n_v - n_u$ of these area v rectangles which lie strictly above S; the upper left corners of these are circled in Figure 5. Let T be the set of times at which these $n_v - n_u$ rectangles begin (if there are more than $n_v - n_u$ of these, then we choose any $n_v - n_u$ of them to define the set T). We define a new acknowledgment sequence $S' = S \cup T$. The resulting curve is shown on Figure 6.

Since $|S| = n_u$ and $|T| = n_v - n_u$, the number of acknowledgments in S' is precisely n_v. The $n_v - n_u$ rectangles of T are all between S and S'. Each of them has area v, of which an area of at most z can lie above A_z (by definition of A_z). Thus the area above S is at least $(v - z)(n_v - n_u)$ plus the area above S'. These facts combine to give us (2).

Taking $u = v + dv$, we obtain from (2)

$$L(n_{v+dv}, z) \geq (v - z)(n_v - n_{v+dv}) + L(n_v, z).$$

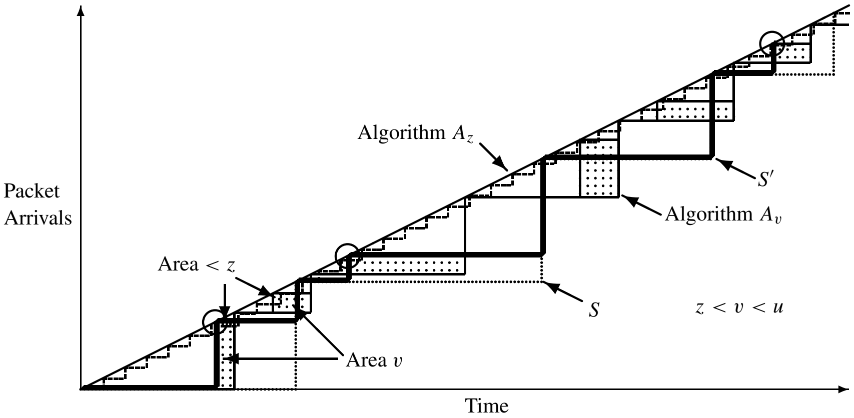


Fig. 6. Proof of Lemma 4, continued: defining S'.

Rewriting and integrating from z to t, for any $z < t \le 1$, we obtain

$$\int_z^t dL(n_v, z) \ge \int_z^t -(v - z)\, dn_v$$

which implies

$$L(n_t, z) - L(n_z, z) \ge \int_z^t n_v\, dv - (t - z)n_t.$$

Observing that $L(n_z, z) = 0$, and that $n_v \le n_t$ for $v > t$, we have

$$L(n_t, z) \ge \int_z^1 n_v\, dv - (1 - z)n_t.$$

Taking $n_t = n_{OPT}$ and noting that $L(n_{OPT}, z)$ is a lower bound on E_z gives the lemma. □

Letting z tend towards zero, the A_z curve tends to the curve of packet arrivals, so that $\lim_{z\to 0} E_z$ is equal to the latency of *OPT*, and Lemma 4 then yields the following corollary.

Corollary 5. *The cost incurred by the optimal offline algorithm on input I, C_{OPT}, is at least*

$$C_{OPT} \ge \int_0^1 n_z\, dz.$$

3.3. *Analysis of the Randomized Algorithm.* We can now prove the main theorem.

Theorem 6. *Let A be the randomized algorithm that picks z between* 0 *and* 1 *according to a probability density function $p(z)$ and runs the resulting algorithm A_z. For any input I,* *the ratio between the expected cost incurred by A on I and the optimal cost on I satisfies*

$$\frac{C_A(I)}{C_{OPT}(I)} \le 1 + \frac{\int_0^1 (p(z) - P(z))n_z\,dz}{\int_0^1 n_z\,dz}, \tag{3}$$

where $P(z) = \int_0^z p(x)\,dx$.

Proof. Let C_A denote the expected cost incurred by the algorithm A. Combining the calculation below with Corollary 5 yields the theorem.

$$\begin{aligned}
C_A &\le C_{OPT} - n_{OPT} + \int_0^1 p(z)(n_z + zn_{OPT} - E_z)\,dz \\
&\qquad \text{(from (1))} \\
&\le C_{OPT} - n_{OPT} + \int_0^1 p(z)\left(n_z + zn_{OPT} - \int_z^1 n_w\,dw + (1-z)n_{OPT}\right)dz \\
&\qquad \text{(from Lemma 4)} \\
&= C_{OPT} + \int_0^1 p(z)n_z\,dz - \int_0^1 n_w \int_0^w p(z)\,dz\,dw \\
&\qquad \text{(by changing the order of integration)} \\
&= C_{OPT} + \int_0^1 p(z)n_z\,dz - \int_0^1 n_w P(w)\,dw \\
&= C_{OPT} + \int_0^1 (p(z) - P(z))n_z\,dz.
\end{aligned}$$

□

We obtain Theorem 1 as an immediate corollary:

Proof of Theorem 1. Applying Theorem 6 to the ski-rental distribution $p(z) = e^z/(e-1)$, we find

$$\begin{aligned}
\frac{C_A(I)}{C_{OPT}(I)} &\le 1 + \frac{\int_0^1 (p(z) - P(z))n_z\,dz}{\int_0^1 n_z\,dz} \\
&= 1 + \frac{\int_0^1 ((e^z/(e-1)) - ((e^z-1)/(e-1)))n_z\,dz}{\int_0^1 n_z\,dz} \\
&= \frac{e}{e-1}.
\end{aligned}$$

□

4. TCP Acknowledgment and Ski Rental.

To explain the sense in which the essence of the TCP acknowledgment problem is ski rental, we briefly review the basic ski-rental result.

4.1. *Ski Rental.* We focus here on the continuous version of the problem. The input, unknown to the online algorithm, is a nonnegative real number u, representing the length of time that the skier will actually end up skiing. We refer to this input as I_u. The skier, or online algorithm, must decide for what length of time she should rent skis before buying them, without knowing what u is. The cost of buying skis is 1.

Any deterministic algorithm for this problem is defined by a positive real number z, representing the time at which the user will buy skis. We refer to this algorithm (intentionally to draw the analogy) as A_z.

The cost incurred by algorithm A_z on input I_u is

$$C(A_z, I_u) = \begin{cases} u & \text{if } u \le z, \\ z+1 & \text{if } u > z. \end{cases}$$

The optimal offline cost OPT on input I_u is

$$OPT(I_u) = \min(u, 1).$$

Therefore,

$$\frac{C(A_z, I_u)}{OPT(I_u)} = \begin{cases} 1 & \text{if } u \le z, \\ (z+1)/u & \text{if } u > z. \end{cases} \tag{4}$$

We may assume without loss of generality that any online algorithm (deterministic or randomized) will buy by time 1, since thereafter the optimal offline does not increase, but the online cost does. Thus, in our discussion, we assume that both u and z are between 0 and 1.

Any randomized online algorithm A for ski rental is therefore a probability distribution $p(z)$ over algorithms A_z where $0 \le z \le 1$. The optimal randomized online algorithm for ski rental is chosen so as to minimize c, such that for every u, $0 \le u \le 1$,

$$\int_0^u p(z)(1+z)\,dz + u\int_u^1 p(z)\,dz \le cu. \tag{5}$$

A straightforward argument shows that we may assume equality for all u. We can derive a differential equation for $p(z)$ by differentiating twice with respect to u, the solution of which is $p(z) = e^z/(e-1)$. Plugging this distribution back into (5) gives a competitive ratio of $e/(e-1)$.

4.2. *TCP Acknowledgment Basis Inputs.* To explain the connection with ski rental, we describe a one parameter family of inputs I_u, $0 \le u \le 1$, to the TCP problem. For reasons that will become clear shortly, we call these inputs our *basis inputs*. We will show that when restricted to these basis inputs, the behavior of TCP acknowledgment algorithms is precisely the behavior of ski-rental algorithms.

The input I_u is defined as follows. Let K be a large constant. The input is a sequence of n groups of message arrivals of the following form: K^{i+1} messages arrive at time t_{i+1} where $t_{i+1} = t_i + uK^{-i}$.

The important property of this arrival sequence is that even though the latency between message arrivals is u, the total latency at time t_n, assuming no acknowledgments up to that point, is equivalent to nu. This is because in each interval the accumulation of latency due to messages other than those that arrived in the most recent burst is negligible.

Consider now the behavior of A_z on input I_u. Since A_z acknowledges after seeing a rectangle of size z, we have that the cost incurred by A_z on input I_u is

$$C(A_z, I_u) = \begin{cases} nu + 1 & \text{if} \quad u \leq z, \\ n(z+1) & \text{if} \quad u > z. \end{cases}$$

It is also easy to see that the optimal offline cost OPT on input I_u is

$$OPT(I_u) = nu + 1.$$

Therefore, in the limit, we have

$$\frac{C(A_z, I_u)}{OPT(I_u)} = \begin{cases} 1, & u \leq z, \\ (z+1)/u, & u > z, \end{cases} \tag{6}$$

which by no coincidence is precisely the same as the corresponding ski-rental bounds. From this we can conclude that, were our inputs restricted to the set I_u, we could easily use ski-rental results to construct a randomized TCP acknowledgment protocol that achieves the $e/(e-1)$ competitive ratio.

4.3. *The Final Piece of the Puzzle.* The question then becomes: why do the basis inputs capture the essence of the TCP acknowledgment problem? To understand this, we return to inequality (3) derived in Theorem 6 for the competitive ratio of the randomized algorithm A that uses probability distribution $p(z)$ over algorithms A_z. Returning to the basis inputs I_u, we observe that

$$n_z(I_u) = \begin{cases} n, & z \leq u, \\ 1, & z > u. \end{cases} \tag{7}$$

Therefore on the input I_u, (3) becomes

$$\frac{C_A(I_u)}{C_{OPT}(I_u)} \leq 1 + \frac{n(\int_0^u p(z)\,dz - \int_0^u P(z)\,dz)}{nu} + \frac{\int_u^1 p(z)\,dz - \int_u^1 P(z)\,dz}{nu}.$$

In the limit the third term vanishes and this yields

$$\frac{C_A(I_u)}{C_{OPT}(I_u)} \leq 1 + \frac{\int_0^u p(z)\,dz - uP(u) + \int_0^u zp(z)\,dz}{u},$$

by integration by parts, which is precisely the same equation we get for ski rental (obtained from (5)).

Finally, we recall that for any input I, $n_z(I)$ is a nonincreasing function of z, defined over the range $0 \leq z \leq 1$. Thus, from (7) we see that we can represent $n_z(I)$ as a linear combination of our basis functions

$$n_z(I) = \int_0^1 \alpha_u n_z(I_u).$$

(In fact, this is a finite sum, since $n_z(I)$ only changes a finite number of times in the interval 0 to 1, as there are only a finite number of message arrivals.)

Thus, we have that for any input I,

$$\begin{aligned}\frac{C_A(I)}{C_{OPT}(I)} &\leq 1+\frac{\int_0^1 \alpha_u(\int_0^u p(z)\,dz - uP(u) + \int_0^u zp(z)\,dz)\,du}{\int_0^1 \alpha_u u\,du}\\ &\leq 1+\max_u\left(\frac{\int_0^u p(z)\,dz - uP(z) + \int_0^u zp(z)\,dz}{u}\right)\\ &= \frac{e}{e-1},\end{aligned}$$

where the final equality is achieved, as in the ski-rental problem, with $p(z) = e^z/(e-1)$.

5. The Bahncard Problem. We now outline how the machinery set forth in Sections 3 and 4 easily translates into an optimal online algorithm for the Bahncard problem with a competitive ratio of $e/(e-1+\beta)$. In particular, we show that embedded in the Bahncard problem is another rendition of ski rental.

Following [5], we define the Bahncard problem input parameters (C, β, T), where C is the cost of a Bahncard, and $0 \leq \beta \leq 1$ is the discount awarded with a Bahncard on all trips within time T from the time of purchase (i.e., a discounted trip costs β times its full cost). Fleischer provides a randomized algorithm which is $e/(e-1+\beta)$-competitive when $T \to \infty$. When $\beta = 0$ and the trip costs are the same, the Bahncard problem is of course precisely ski rental. Here we present an algorithm which achieves the same competitive ratio for finite T and varying trip cost. The solution we present also generalizes to give an optimal algorithm in the case where there is a different discount β_i associated with each trip. For simplicity, we renormalize by setting $C = 1$ whereby each Bahncard expires after one time unit, say one year.

5.1. *A Randomized Algorithm for the Bahncard Problem.* The key to our analysis is defining another appropriate one parameter family of online algorithms B_z. The algorithm B_z buys a Bahncard at the first point when there would have been a cost of z saved had a Bahncard been purchased at some time earlier in the year. The break-even point for purchasing a Bahncard is $c_{\text{CRIT}} = 1/(1-\beta)$. We can assume without loss of generality that the optimal algorithm will have bought a card during any interval where the savings exceeds c_{CRIT}, and our randomized algorithm will be a linear combination of algorithms B_z with $0 \leq z \leq c_{\text{CRIT}}$.

The histogram depicted in Figure 7 will be useful. We indicate a trip of cost y at time x by a vertical bar of height y at point x. The sum of the heights of these bars is the total cost if no Bahncard is ever purchased. Below the histogram are masks, representing three possible algorithms, where the horizontal bars indicate the periods during which a purchased Bahncard was valid. The total trip cost incurred by each algorithm is the sum of the bars in the histogram that do not coincide with a bar in the mask. To demonstrate this, the trip cost for algorithm B_z is highlighted in bold.

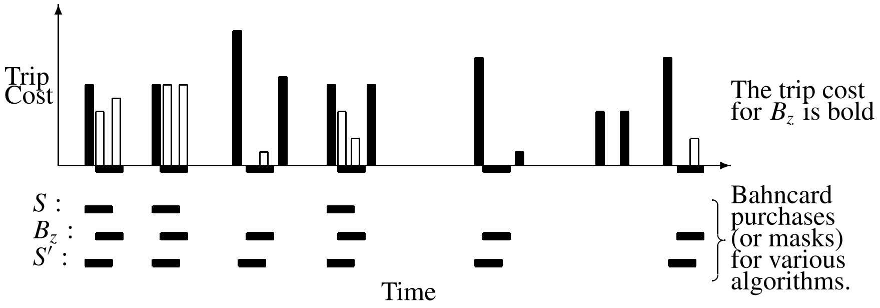

Trip Cost
The trip cost for B_z is bold
S :
B_z :
S' :
Time
Bahncard purchases (or masks) for various algorithms.

Fig. 7. The Bahncard problem.

Following Lemma 4 and Corollary 5, we find:

LEMMA 7. *Let b_z denote the number of Bahncards purchased by algorithm B_z and let b_{OPT} denote the number of Bahncards purchased by the optimal algorithm on some input I. Then*:

1. *The trip cost E_z incurred by the optimal algorithm that is not incurred by the B_z algorithm is at least*

$$E_z \geq \int_z^{c_{\text{CRIT}}} b_w \, dw - (c_{\text{CRIT}} - z) b_{OPT}.$$

2. *The cost incurred by the optimal offline algorithm on this input*, $C_{OPT}(I)$, *is at least*

$$C_{OPT}(I) \geq \int_0^{c_{\text{CRIT}}} b_z \, dz.$$

PROOF. The proof of this lemma follows the same format as Lemma 4. For any fixed input I, let $L(b_u, z)$ be the minimum, over all Bahncard purchase sequences S with b_u Bahncards, of trips which cost full price according to S but which are discounted with B_z. We find that for $z \leq v < u$,

$$L(b_u, z) \geq (v - z)(b_v - b_u) + L(b_v, z). \tag{8}$$

To see this, first observe that each of the b_v Bahncard purchases, algorithm A_v was caused by a recent accumulated trip cost of v. At least $b_v - b_u$ of these are periods in which both A_v and S are paying full fare. Let T be the times which are exactly one time unit before B_v buys these $b_v - b_u$ Bahncards, and let $S' = S \cup T$ be a new purchasing schedule which purchases b_v Bahncards. (See Figure 7.) This definition of S' might require a new Bahncard to be purchased before an old one has expired, but this will not affect subsequent arguments. These facts give (8).

Following Lemma 4 we find that that for any $z < t \leq 1$,

$$L(b_t, z) \geq \int_z^{c_{\text{CRIT}}} b_v \, dv - (c_{\text{CRIT}} - z) b_t.$$

Taking $b_t = b_{OPT}$ establishes the first part of the theorem.

Now, let T be the total cost of trips without purchasing any Bahncards. Letting z tend towards zero, we find

$$C_{OPT}(I) \geq n_{OPT} + \beta T + (1-\beta) E_z.$$

Noting that $E_z < T$ as z tends towards zero establishes the second part of the theorem. □

This lemma can be used to prove the following theorem, which is an analogue of Theorem 6.

THEOREM 8. *Let B be the randomized algorithm that picks z between 0 and c_{CRIT} according to the probability density function $p(z)$ and runs the resulting algorithm B_z. For any input I, the ratio between the expected cost incurred by B on I and the OPT cost on I satisfies*

$$\frac{C_B(I)}{C_{OPT}(I)} \leq 1 + \frac{(1-\beta)(\int_0^{c_{\text{CRIT}}} (p(z) - P(z)) b_z \, dz)}{\int_0^{c_{\text{CRIT}}} b_z \, dz}.$$

PROOF. Let C_B denote the expected cost incurred by the algorithm B. We find

$$\begin{aligned} C_B &\leq C_{OPT} - b_{OPT} + \int_0^{c_{\text{CRIT}}} p(z) b_z + (1-\beta) \int_0^{c_{\text{CRIT}}} p(z)(z b_{OPT} - E_z) \, dz \\ &\leq C_{OPT} - b_{OPT} + \int_0^{c_{\text{CRIT}}} p(z) \left(b_z + (1-\beta) z b_{OPT} - (1-\beta) \int_z^{c_{\text{CRIT}}} b_w \, dw \right. \\ &\qquad \left. + (1-z) b_{OPT} \right) dz \\ &= C_{OPT} + \int_0^{c_{\text{CRIT}}} p(z) b_z \, dz - (1-\beta) \int_0^{c_{\text{CRIT}}} b_w \int_0^w p(z) \, dz \, dw \\ &= C_{OPT} + \int_0^{c_{\text{CRIT}}} (p(z) - (1-\beta) P(z)) b_z \, dz. \end{aligned}$$

□

As an immediate corollary, we obtain the following theorem, matching the lower bound proved by Fleischer.

THEOREM 9. *Let B be the randomized algorithm that picks z between 0 and c_{CRIT} according to the probability density function $p(z) = (1-\beta) e^{z(1-\beta)} / (e - 1 + \beta)$ and runs the resulting algorithm B_z. The competitive ratio of B is $e/(e-1+\beta)$.*

PROOF. Plugging $p(z) = (1-\beta) e^{z(1-\beta)} / (e - 1 + \beta)$ into Theorem 8 yields the result. □

5.2. *Basis Inputs for Bahncard.* The notion of basis inputs also generalizes from the TCP acknowledgment problem and provides a more explicit connection between the Bahncard problem and ski rental. For simplicity we consider the case when $\beta = 0$.

The key, again, is to define a one parameter family of inputs I_u, $0 \leq u \leq 1$, to the Bahncard problem. In the simplified Bahncard problem where we take $\beta = 0$, the only difference from ski rental is the finite expiration of the card. For our basis inputs all trips will occur within a single unit of time. Hence they precisely model ski-rental inputs.

Let m be a large constant. We define the input I_u to be a sequence of m trips, all costing u/m, all of which occur within one unit of time. Algorithm B_z will purchase a Bahncard after spending amount z if $u \geq z$ and will never purchase if $u < z$. Taking the limit as m tends to infinity we have that the cost incurred by B_z on input I_u is equivalent to

$$C(B_z, I_u) = \begin{cases} u & \text{if} \quad u \leq z, \\ z + 1 & \text{if} \quad u > z. \end{cases}$$

As in ski rental, the optimal offline algorithm will never buy a Bahncard and will incur a cost of $OPT(I_u) = u$, giving the same competitive ratio as ski rental given in (4).

For these very simple inputs I_u we find that the number of Bahncard purchases is

$$b_z(I_u) = \begin{cases} n, & z \leq u, \\ 0, & z > u. \end{cases} \tag{9}$$

Thus, Theorem 8 with $\beta = 0$ gives

$$\frac{C_B(I_u)}{C_{OPT}(I_u)} \leq 1 + \frac{(\int_0^u p(z)\,dz - \int_0^u P(z)\,dz)}{u}.$$

Again, recalling that $b_z(I)$ is always a nonincreasing function of z and writing the input I as a linear combination of basis inputs, we can conclude that the competitive ratio is always at most $e/(e-1)$.

6. Final Remarks.

The solutions to ski rental, TCP acknowledgment, the Bahncard problem and scheduling to minimize weighted completion time all fall within a common framework. There is a one-parameter family of algorithms, each defined (though not always explicitly) in terms of savings the optimal offline algorithm would have incurred had it "acted" earlier. This seems to be a principled approach to solving such problems, and inherently leads to the ski-rental equation (5) whose solution yields a competitive ratio of $e/(e-1)$.

Acknowledgments. We are very grateful to Gagan Aggarwal for correcting an error in the proof of Theorem 4 in the STOC version of this paper. We also gratefully acknowledge John Noga and Steve Seiden for sharing their early results on the TCP acknowledgment problem and for pointing us to the Bahncard problem.

References

[1] S. Albers. Better bounds for online scheduling. *SIAM Journal on Computing*, 29: 459–473, 1999. Preliminary version in *Proceedings of the* 29*th Annual ACM Symposium on Theory of Computing* (*STOC* 97), pp. 130–139, 1997.

[2] A. Borodin and R. El Yaniv. *Online Computation and Competitive Analysis*, Cambridge University Press, Cambridge, 1998. ISBN: 0521563925.

[3] D. R. Dooly, S. A. Goldman and S. D. Scott. TCP dynamic acknowledgment delay: theory and practice. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing* (*STOC*), Dallas, TX, pp. 389–398, 1998.

[4] D. R. Dooly, S. A. Goldman and S. D. Scott. On-line analysis of the TCP acknowledgment delay problem. *Journal of the ACM*, 48: 243–273, 2001.

[5] R. Fleischer. On the Bahncard problem. *Theoretical Computer Science*, 268: 161–174, 2001. Preliminary version appeared in *Proceedings of the* 4*th Annual International Computing and Combinatorics Conference* (*COCOON*-98), Taipei, August 12–14, 1998, pp. 65–74. LNCS 1449. Springer-Verlag, Berlin, 1998.

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

[7] A. R. Karlin, M. S. Manasse, L. A. McGeoch, and S. Owicki. Competitive randomized algorithms for non-uniform problems. In *Proceedings of the First Annual ACM–SIAM Symposium on Discrete Algorithms*, San Francisco, CA, 22–24 January 1990, pp. 301–309. Also in *Algorithmica*, 11(6): 542–571, 1994.

[8] A. R. Karlin, M. Manasse, L. Rudolph and D. Sleator. Competitive snoopy caching. *Algorithmica*, 3(1): 79–119, 1988.

[9] J. Noga. Private communication, 2000.

[10] J. Noga and S. S. Seiden. Private communication, 2000.

[11] S. S. Seiden. A guessing game and randomized online algorithms. In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing* (*STOC*), Portland, OR, pp. 592–601, 2000.