Dynamic TCP Acknowledgement and Other Stories about e/(e-1)

Anna R. Karlin[*] Claire Kenyon[†] Dana Randall[‡]

ABSTRACT

We present the first optimal randomized online algorithms for the TCP acknowledgment problem [5] and the Bahncard problem [7]. 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.

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 don't 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 trivial deterministic online algorithm that achieves a competitive ratio of 2 [10], 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 [9].

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 acknowledgement packet to simultaneously acknowledge multiple outstanding packets, thereby reducing the overhead of the acknowledgements. On the other hand, delaying acknowledgements too much can interfere with 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 acknowledgement is sent to increase too much.

This motivated Dooly, Goldman and Scott to define the following problem [5]. 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 *acknowledgement 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.

Dooly *et al.* showed that the natural algorithm which waits until the latency since the previous acknowledgement equals the cost of the acknowledgement has a competitive ratio of 2. Subsequently, Seiden [13], and independently Noga [11], obtained a lower bound of $e/(e-1)$ on the competitive ratio of randomized online algorithms for this problem. A matching upper bound remained elusive, and in fact no randomized algorithm was known to beat the 2-competitive ratio achieved by the deterministic algorithm.

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.

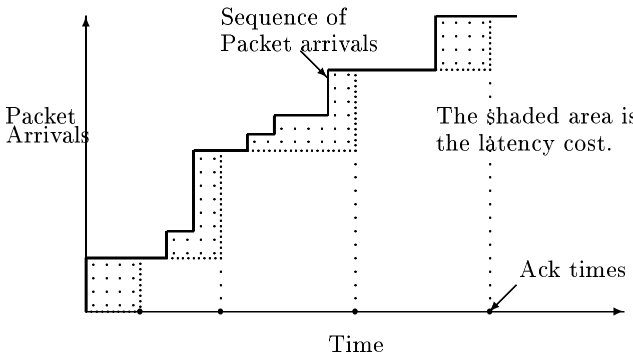
The Bahncard problem

The Bahncard problem models online ticket purchasing in the German Deutsche Bundesbahn, where one can opt to buy a Bahncard that entitles the traveler to a 50% discount

[*]Department of Computer Science and Engineering, University of Washington, Seattle, WA. Email: karlin@cs.washington.edu. Supported in part by NSF EIA-9870740 and a grant from the US-Israel Binational Science Foundation (BSF).

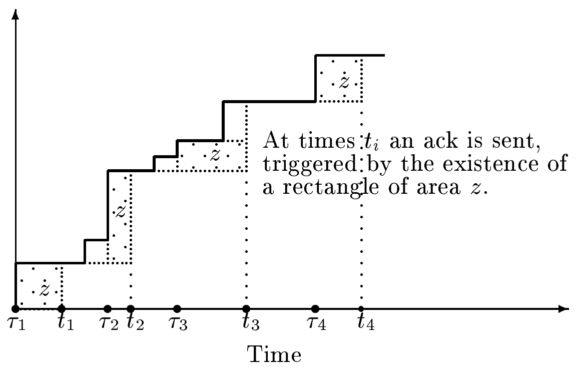
[†]LRI UMR 8623, Université Paris-Sud, Bât. 490, F91405 ORSAY Cedex, France. Email: kenyon@lri.fr. Part of this work was done while this author was visiting the University of Washington at Seattle and Microsoft.

[‡]College of Computing and School of Mathematics, Georgia Institute of Technology, Atlanta GA. Email: randall@cc.gatech.edu. Supported in part by NSF CCR-9703206. Part of this work was done while this author was visiting Microsoft.



Sequence of Packet arrivals
Packet Arrivals
The shaded area is the latency cost.
Ack times
Time

Figure 1: A pictorial representation of TCP acknowledgement.



At times t_i an ack is sent, triggered by the existence of a rectangle of area z.
z
z
z
z
z
τ_1
t_1
τ_2
t_2
τ_3
t_3
τ_4
t_4
Time

Figure 2: Algorithm A_z.

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 permits the price of tickets to be discounted by $0 \leq \beta \leq 1$ 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 [7] introduced this model and provided a deterministic algorithm with competitive ratio 2. He also presented an $e/(e-1+\beta)$ bound in the case that the Bahncard never expires and conjectured that this is also the bound for finite expiration periods.

One-machine scheduling to minimize weighted completion time

In this scheduling problem, each job has a processing time p_j, a release time r_j and a weight w_j – all of these parameters become known only at the moment the job is released. The goal is to construct a nonpreemptive feasible schedule S that minimizes $\sum_j w_j C_j^S$, where C_j^S is the completion time of job j in schedule S.

Chekuri, Motwani, Natarajan and Stein [3] have shown that there is a deterministic 2-competitive algorithm for this problem and an $e/(e-1)$ competitive randomized algorithm. Both of these bounds are best possible.

1.1 Our Results

The main contribution of this paper is new randomized online algorithm for TCP acknowledgement 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.[1] Our secondary contribution is to show that, despite the appearance of greater complexity, all of the problems described above 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]We also generalize our solution to get an optimal algorithm for the case where the discount rate for different trips varies.

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. We measure 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

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

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.

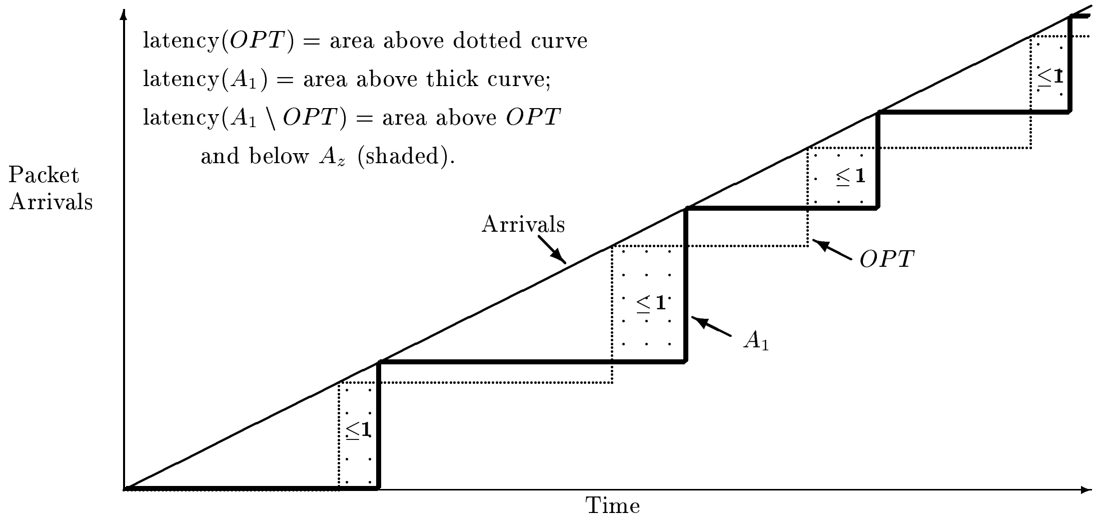
The randomized competitive ratio is then the infimum over c such that there is a c-competitive algorithm against an oblivious adversary.

2. A RANDOMIZED ALGORITHM FOR TCP ACKNOWLEDGEMENT

The most natural approach to the construction of a TCP acknowledgement problem is to consider algorithms which 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 [12].

The key to our solution is to define 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 acknowledgement at the first time $t_{i+1} > t_i$ for which there is a time τ_{i+1},



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

Figure 3: Proof of Lemma 3.

$t_i \leq \tau_{i+1} \leq t_{i+1}$, 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 acknowledgement 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}.

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 $p(z) = \frac{e^z}{e-1}$ 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 acknowledgement 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 acknowledgements are sent by the algorithm. The algorithm defines a staircase curve g such that, if acknowledgements 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 acknowledgement between any pair of successive acknowledgements 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 acknowledgements, and assume that it does not send any acknowledgement in (t_i, t_{i+1}). Enrich this sequence by sending an additional acknowledgement at time τ_i. The acknowledgement 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 acknowledgement 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. The cost C_{A_1} of A_1 on input I satisfies

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

where n_{OPT} is the number of acknowledgements performed by OPT on input I, and latency($A_1 \setminus OPT$) is the latency incurred by A_1 that is not incurred by OPT. However, it is easy to see from Figure 3 that latency($A_1 \setminus OPT$) 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 acknowledgement, and its right side at the following time when A_1 sends an acknowledgement. By definition of algorithm A_1, all these rectangles have area at most 1. Hence, latency($A1 \setminus OPT$) $\leq n_{OPT}$ and we obtain that

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

□

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 acknowledgements of algorithm A_z on input I. Looking

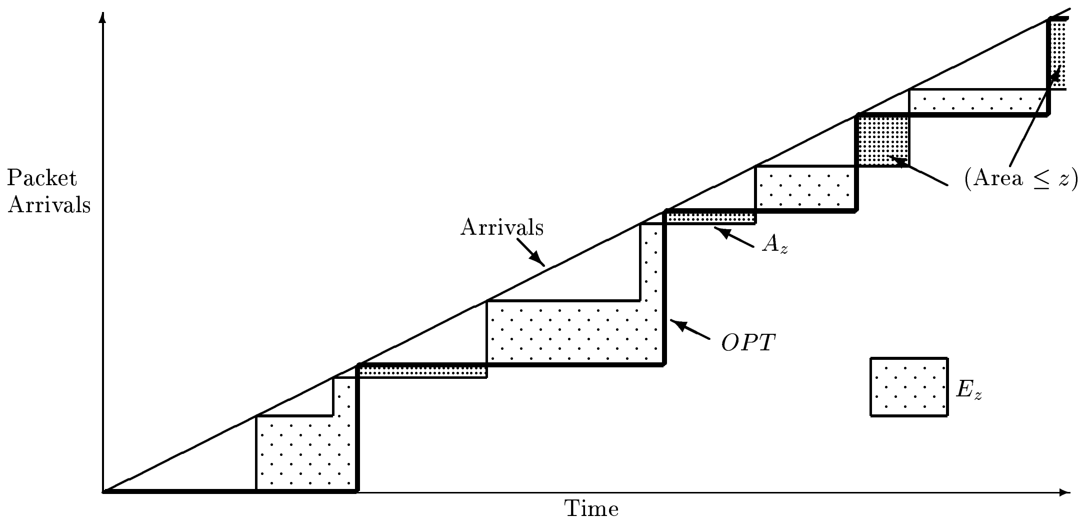


Figure 4: Analysis of A_z, proof of Equation 1 and definition of E_z.

at Figure 4, we see that the latency cost of A_z is bounded by (a) the area above the OPT curve, plus (b) the area under OPT and over A_z (dark shaded area in Figure 4), minus (c) the area, denoted $E_z(I)$, under A_z and over OPT (lightly shaded area in Figure 4).

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

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

LEMMA 4. *Let n_z denote the number of acknowledgements of algorithm $\mathcal{A}_z$ and n_{OPT} denote the number of acknowledgements of the optimal algorithm on some input I. Then the area E_z above the optimal curve and below the $\mathcal{A}_z$ 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 acknowledgements, of the area above the S curve that is *below* the A_z curve. $L(n, z)$ is a monotone non-decreasing function of n. Thus we have, since $n_{\mathrm{OPT}} \geq n_1$ by Lemma 2: $E_z \geq L(n_{\mathrm{OPT}}, z) \geq L(n_1, z)$. We will prove a lower bound on $L(n_u, z)$ for all $u \geq z$.

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

$$L(n_u, z) \geq (v-z)(n_v - n_u) + L(n_v, z). \qquad (2)$$

The proof is illustrated in Figures 5 and 6. Figure 5 shows three acknowledgement sequences for the given input: S, the acknowledgement sequence with n_u acks that minimizes $L(n_u, z)$, and the acknowledgement 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 acknowledgement. At most n_u such rectangles intersect the curve S, so there are at least $n_v - n_u$ of these area v rectangles which lie 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 acknowledgement 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 acknowledgements 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 Equation 2.

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

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

Rewriting and integrating from z to 1, we obtain

$$\int_z^1 dL(n_v, z) \geq \int_z^1 -(v-z)dn_v$$

which gives us

$$L(n_1, z) - L(n_z, z) \geq \int_z^1 n_v dv - (1-z)n_1.$$

Observing that $L(n_z, z) = 0$, and that $L(n_1, z)$ 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*

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

3.3 Analysis of the randomized algorithm

We can now prove the main theorem, of which Theorem 1 is a simple corollary obtained by plugging in the ski rental distribution $p(z) = e^z/(e-1)$.

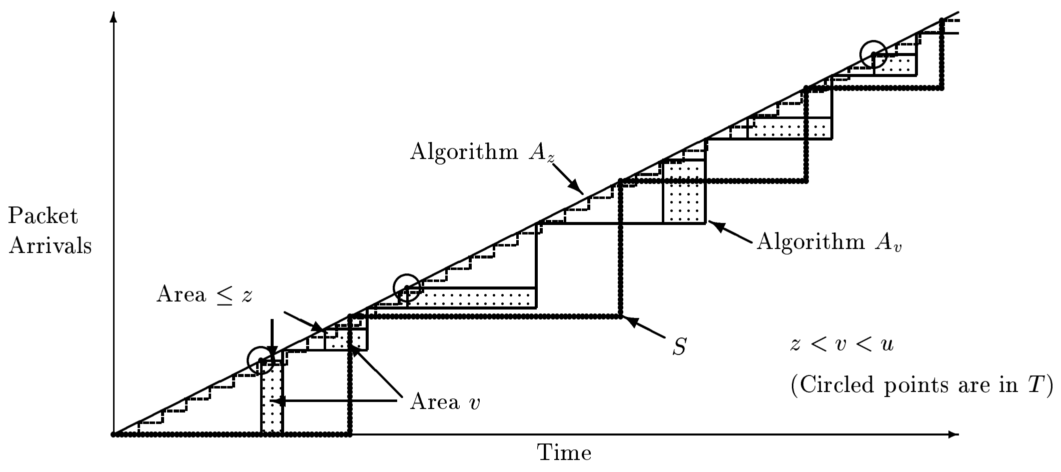


Figure 5: Proof of Lemma 4: setup.

THEOREM 6. *Let A be the randomized algorithm that picks z between 0 and 1 according to the 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*

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

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.

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

□

4. TCP ACKNOWLEDGEMENT 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 (to intentionally draw the analogy) as A_z.

The cost incurred by algorithm A_z on input I_u is

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

The optimal offline cost OPT on input I_u is

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

Therefore,

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

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 $0 \leq z \leq 1$. The optimal randomized online algorithm for ski rental is chosen so as to minimize c, such that for every u, $0 \leq u \leq 1$,

$$\int_0^u p(z)(1+z)dz + u \int_u^1 p(z)dz \leq c\, u. \tag{4}$$

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 Equation 4 gives us a competitive ratio of $e/e-1$.

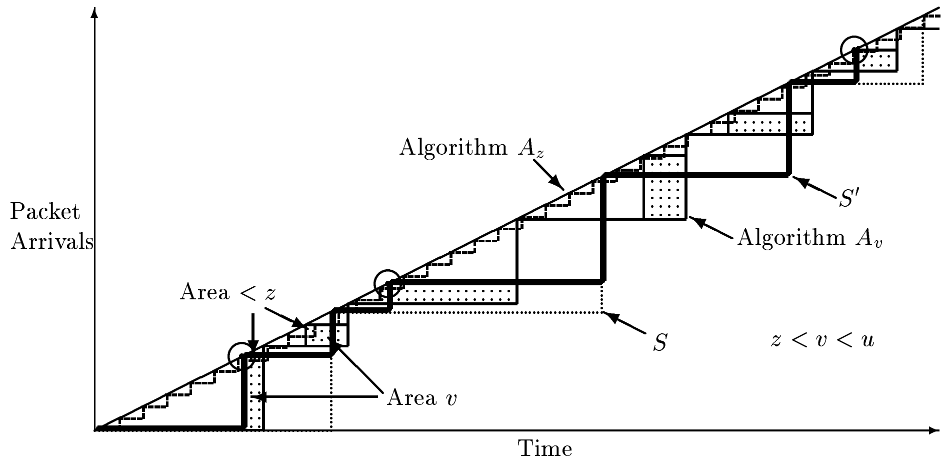


Figure 6: Proof of Lemma 4, continued: defining S'.

4.2 TCP Acknowledgment Basis Inputs

To explain the connection with ski rental, we describe a one parameter family of inputs I_u, $0 \leq u \leq 1$, 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 acknowledgement algorithms is precisely the behavior of ski rental algorithms.

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

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 acknowledgements up to that point, is 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 acks after seeing a rectangle of size z, we have that the cost incurred by A_z on input I_u is

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

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

$$\frac{C(A_z, I_u)}{OPT(I_u)} = \begin{cases} 1 & u \leq z \\ \frac{z+1}{u} & u > z \end{cases} \tag{5}$$

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 the 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

$$n_z(I_u) = \begin{cases} n & z \leq u \\ 0 & z > u \end{cases} \tag{6}$$

Therefore on the input I_u, Equation 3 becomes

$$\begin{aligned} \frac{C_A(I_u)}{C_{\text{OPT}}(I_u)} &\leq 1 + \frac{n\left(\int_0^u p(z)dz - \int_0^u P(z)dz\right)}{nu} \\ &= 1 + \frac{\int_0^u p(z)dz - uP(u) + \int_0^u zp(z)dz}{u} \\ &\qquad \text{(by integration by parts)} \end{aligned}$$

which is precisely the same equation we get for ski rental (obtained from Equation 4).

Finally, we recall that for any input I, $n_z(I)$ is a non-increasing function of z, defined over the range $0 \leq z \leq 1$. Thus, from Equation 6 we see that we can represent $n_z(I)$ as a linear combination of our basis functions

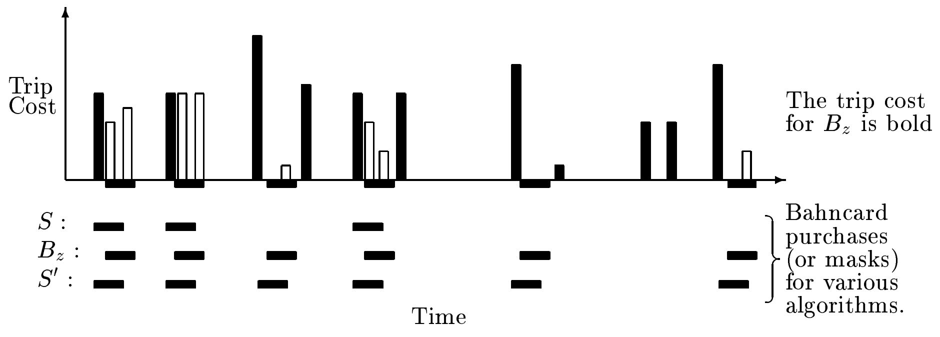
$$n_z(I) = \int_0^1 \alpha_u n_z(I_u).$$

(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

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

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



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

Figure 7: The Bahncard Problem

5. THE BAHNCARD PROBLEM

We now outline how the machinery set forth in the 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 [7], we define the Bahncard problem input parameters (C, β, T), where C is the cost of a Bahncard, $0 \leq \beta \leq 1$ 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 $T \rightarrow \infty$. When $\beta = 0$ and the rental 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 in this extended abstract, we assume that $\beta = 0$ (but allow varying trip costs and finite Bahncard expiration) and present an algorithm which achieves a competitive ratio of $e/(e-1)$. Fleischer's analysis which extends the the $\beta = 0$ analysis to $\beta > 0$ can be incorporated into our results to establish the conjectured competitive ratio of $e/(e-1+\beta)$ for the general Bahncard problem. In addition, we renormalize so that the Bahncard expires after one time unit (say, one year) and so the cost of a card is one (where units, in this case, turn out to be multiples of 240 DMs).

5.1 A randomized algorithm

The key to our analysis is defining another appropriate one parameter family of online algorithms B_z, where $0 \leq z \leq 1$. 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 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.

Following lemmas 4 and 5, we find:

LEMMA 7. *Let b_z denote the number of Bahncards purchased by algorithm $\mathcal{B}_z$ and b_{OPT} denote the number of Bahncards purchased by the the optimal algorithm on some input I. Then:*

1. *The cost incurred by the optimal offline algorithm on this input, $C_{OPT}(I)$, is at least*
$$C_{OPT}(I) \geq \int_0^1 b_z dz.$$
2. *The trip cost E_z incurred by the optimal algorithm that is not incurred by the $\mathcal{B}_z$ algorithm is at least*
$$E_z \geq \int_z^1 b_w dw - (1-z)b_{OPT}.$$

The proof of this lemma follows the same format as Lemma 4. The key idea is to bound $L(b_u, z)$, the minimum, over all Bahncard purchase sequences S with b_u Bahncards, of trips which cost full price according to S but which are free with B_z. As before, for $z \leq v < u$,

$$L(b_u, z) \geq (v-z)(b_v - b_u) + L(b_v, z). \qquad (7)$$

To prove this, we observe that there are at least $b_v - b_u$ periods in which B_v accumulates a trip cost of v (justifying one of its Bahncard purchases) in which S also pays full fare. During each of these periods B_z can only accumulate a trip cost of z. Let T be the times which are exactly one time unit before B_v buys these $b_v - b_u$ Bahncards, and as before, let $S' = S \cup T$ be a new purchasing schedule which purchases b_v Bahncards. (See Figure 7.) These facts give Equation 7. Following the remaining steps of the proof of Lemma 4 completes the proof.

This lemma can be used to prove the analogue of Theorem 6, giving

THEOREM 8. *Let B be the randomized algorithm that picks z between 0 and 1 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*

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

As before, letting $p(z) = \frac{e^z}{e-1}$ gives $e/(e-1)$ for the competitive ratio of B. This matches the lower bound proved by Fleischer.

5.2 Basis Inputs for Bahncard

The notion of basis inputs also generalizes from the TCP acknowledgement problem and provides a more explicit connection between the Bahncard problem and ski rental.

Again, we define a one parameter family of inputs I_u, $0 \leq u \leq 1$, to the Bahncard problem. Let m be a large constant. The input is a sequence of m trips, all costing u/m and occurring 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 $u < z$. Summarizing this and taking the limit, we have that the cost incurred by B_z on input I_u is

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

The optimal offline algorithm will never buy a Bahncard and will incur a cost of $OPT(I_u) = u$, giving the same ratios as ski rental and TCP acknowledgement. Since b_z is a non-increasing function of z, the appropriate linear combination of our basis functions shows that the behavior of Algorithm B on I_u completely captures its behavior on any input. This reestablishes the optimal competitive ratio for the Bahncard problem.

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 4 whose solution yields a competitive ratio of $e/e-1$. We will expand upon these ideas in the full paper.

One of the most interesting unsolved mysteries about $\frac{e}{e-1}$ concerns the problem of scheduling on m uniform machines to minimize the makespan. In this problem, a sequence of jobs is presented one by one, each characterized by its running time. The online algorithm is required to schedule each job on one of the m machines before seeing the next job, with performance measured by the makespan. The classic result of Graham shows that List Scheduling achieves a competitive ratio of $2 - 1/m$ [6]. The best bounds currently known on the competitive ratio for deterministic algorithms are an upper bound of approximately 1.92 [1, 8] and a lower bound of approximately 1.85 [1]. For randomized algorithms, there is a lower bound of $\frac{e}{e-1}$ [4, 13]. However, it is currently not known whether there are randomized algorithms that beat the deterministic ones for m large. This is one of the most notorious open problems in online scheduling and we do not resolve it here. However, we are intrigued by the possibility that there may be a ski rental problem buried within this one that we simply do not see at this time.

Acknowledgements

We gratefully acknowledge John Noga and Steve Seiden for sharing their early results on the TCP acknowledgement problem and for pointing us to the Bahncard problem.

7. REFERENCES

[1] S. Albers. "Better bounds for online scheduling", in Proc. 29th Annual ACM Symposium on Theory of Computing (STOC97) , pages 130-139, 1997. Accepted for publication in SIAM Journal on Computing.

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

[3] C. Chekuri, R. Motwani, B. Natarajan and C. Stein. "Approximation Techniques for Average Completion Time Scheduling". SIAM J. of Computing, to appear. Preliminary version in Proceedings of ACM-SIAM Symposium on Discrete Algorithms (SODA), 1997.

[4] B. Chen, A. van Vliet and G. J. Woeginger. "A lower bound for randomized on-line scheduling algorithms". *Inf. Process. Lett.*, 51:219-222, 1994.

[5] Daniel R. Dooly, Sally A. Goldman and Stephen D. Scott. "TCP Dynamic Acknowledgement Delay: Theory and Practice", Proceedings of the thirtieth annual ACM symposium on theory of computing (STOC), 1998, Dallas, TX, USA, pp. 389-398.

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

[7] Rudolf Fleischer. "On The Bahncard Problem". In Proceedings of the 4th Annual International Computing and Combinatorics Conference (COCOON-98), Taipei (ROC), August 12-14, 1998. Springer LNCS 1449, pp. 65-74. To appear in Theoretical Computer Science, special issue on online algorithms.

[8] D. Karger, S. Phillips and E. Torng. "A better algorithm for an ancient scheduling problem." *Journal of Algorithms*, 20:400-430, 1996.

[9] Anna R. Karlin, Mark S. Manasse, Lyle A. McGeoch, and Susan Owicki. "Competitive Randomized Algorithms for Non-Uniform Problems". In Proceedings of the First Annual ACM-SIAM Symposium on Discrete Algorithms, pages 301-309, San Francisco, California, 22-24 January 1990. Also in Algorithmica 11(6): 542-571 (1994).

[10] Anna R. Karlin, M. Manasse, L. Rudolph and D. Sleator "Competitive Snoopy Caching," *Algorithmica*, Special Issue on Parallel and Distributed Computing, Number 3, pp. 79-119, 1988.

[11] John Noga. Private Communication.

[12] John Noga and Steven S. Seiden. Private Communication.

[13] Steven S. Seiden. "A Guessing Game and Randomized Online Algorithms". Proceedings of the thirty-second annual ACM symposium on theory of computing (STOC), 2000, Portland, OR, USA, pp. 592-601.

[14] J. Sgall. "A lower bound for randomized on-line multiprocessor scheduling." *Inf. Process. Lett.*, 63(1):51-55, 1997.