On the Bahncard Problem

Rudolf Fleischer*

Max-Planck-Institut für Informatik, Im Stadtwald, 66123 Saarbrücken, Germany.
E-mail: `rudolf@mpi-sb.mpg.de`.

Abstract. In this paper, we generalize the *Ski-Rental Problem* to the *Bahncard Problem* which is an online problem of practical relevance for all travelers. The Bahncard is a railway pass of the Deutsche Bundesbahn (the German railway company) which entitles its holder to a 50% price reduction on nearly all train tickets. It costs 240 DM, and it is valid for 12 months.
For the common traveler, the decision at which time to buy a Bahncard is a typical online problem, because she usually does not know when and where to she will travel next. We show that the greedy algorithm applied by most travelers and clerks at ticket offices is not better in the worst case than the trivial algorithm which never buys a Bahncard. We present two optimal deterministic online algorithms, an optimistic one and and a pessimistic one. We further give a lower bound for randomized online algorithms and present an algorithm which we conjecture to be optimal; a proof of the conjecture is given for a special case of the problem. It turns out that the optimal competitive ratio only depends on the price reduction factor (50% for the German Bahncard Problem), but does not depend on the price or validity period of a Bahncard.

1 Introduction

In [7, p. 113], Raghavan introduced the *Ski-Rental Problem (SRP)* : A sportsman can either rent a pair of skis for 1 DM a day, or buy a pair for N DM. As long as he has not bought his skis, he must decide before each trip whether to buy the skis this time or to wait until the next trip (which might never come).

* The author was partially supported by the EU ESPRIT LTR Project No. 20244 (ALCOM-IT). He was further supported by a Habilitation Scholarship of the German Research Foundation (DFG).

The SRP is equivalent to a page replication problem on two nodes with distance 1 and replication cost N, or to a two-server problem on a triangle with side lengths $(1, N, N)$. For the page replication problem, there are optimal 2-competitive deterministic [3] and $\frac{(1+\frac{1}{N})^N}{(1+\frac{1}{N})^N-1}$-competitive randomized algorithms against an oblivious adversary [1, 6]. A similar bound was obtained by Karlin *et al.* [5] for the problem of two servers on a $(1, N, N)$-triangle.

In this paper, we consider the *Bahncard Problem* which contains the SRP as a special case. The Bahncard is a railway pass of the Deutsche Bundesbahn (the German railway company)[1]. It costs 240 DM, and it is valid for 12 months. Within this period, a traveler can buy train tickets[2] for half of the regular price. Looking back at her travel schedule of the last few years, a traveler can easily determine when several expensive trips had been sufficiently close together to justify the additional expense of a Bahncard. Unfortunately, at any given time the traveler cannot see far into the future, so her decision when to buy a Bahncard is made with a high degree of uncertainty.

Let $BP(C, \beta, T)$ denote the (C, β, T)*–Bahncard Problem*, where a Bahncard costs C, reduces any ticket price p to $\beta \cdot p$, and is valid for time T. For example, the *German Bahncard Problem* GBP is BP(240 DM, $\frac{1}{2}$, 1 year), and SRP is BP$(N, 0, \infty)$ with the additional constraint that each ticket costs $1\,DM$.

The SRP and the Bahncard Problem are *online problems*, i.e., all decisions must be made without any knowledge of the future. The quality of an *online algorithm* is measured by the ratio of its performance and the performance of an optimal offline algorithm with full knowledge of the future. The supremum of this ratio over all possible travel request sequences is called the *competitive ratio* of the online algorithm; the smaller the competitive ratio (it is always at least 1), the better the algorithm[3]

We show that no deterministic online algorithm for BP(C, β, T) can be better than $(2-\beta)$-competitive. This lower bound is achieved by a natural generalization, SUM, of the optimal deterministic 2-competitive Ski-Rental

[1] Of course, our theory can also be applied to other time bounded price reduction schemes which can be found all over the world.

[2] Well, most train tickets.

[3] Karlin *et al.* [2] coined the term "competitive analysis", but the model was used before by Sleator and Tarjan [9] in the analysis of list ranking and paging problems, and even earlier by Johnson *et al.* [4] in the analysis of bin packing heuristics.

algorithm. SUM is pessimistic about the future in the sense that it always buys at the latest possible time. Surprisingly, there is another optimal deterministic algorithm, *OSUM*, which usually buys much earlier than the pessimistic SUM (in fact, it buys at the earliest possible time). This gives the rare chance of combining competitive analysis with probabilistic analysis : A traveler with a low travel frequency should use the pessimistic algorithm, whereas a frequent traveler should use the optimistic algorithm. Then both travelers will be happy in the worst case (because both algorithms achieve an optimal competitive ratio), and on the average (because the pessimistic algorithm tries to avoid buying, in contrast to the optimistic algorithm).

Since an online algorithm must make its decisions in a state of uncertainty about future events, it seems plausible that randomization should help the algorithm (because this may help to average between good and bad unpredictable future developments). Ben-David *et al.* [2] defined several models for randomized competitive analysis and compared their relative strengths. In this paper, we assume an *oblivious adversary*. In this model, a request sequence is fixed in advance and the competitive ratio of a randomized algorithm is a random variable, only dependent on the random moves of the algorithm.

We show that randomized variants of SUM and R-SUM are $\frac{2}{1+\beta}$-competitive against an oblivious adversary. This beats the deterministic lower bound for $\beta \in (0,1)$, but it does not reach the lower bound of $\frac{e}{e-1+\beta}$ (except for the trivial case of $\beta = 1$) which we show to hold for any randomized algorithm. We give a randomized algorithm which achieves this bound in the case of $T = \infty$, i.e., a Bahncard never expires (in this case, the Bahncard Problem corresponds to a variant of the SRP where the price for renting the skis includes the daily fee for the lift, which has to be paid additionally each day after buying the skis). We conjecture that the algorithm is also optimal for $T < \infty$.

We note that introducing further "real life" restrictions like limiting the number of trips or upper bounds on ticket prices has no effect on the worst case behaviour of the problem as long as a single one-way ticket can already be more expensive than a Bahncard.

This paper is organized as follows. In section 2, we give all definitions. In section 3, we present an efficient offline algorithm. In section 4, we give a deterministic lower bound and two optimal deterministic algorithms. In section 5, we give some randomized online algorithms and prove a lower bound which is tight for a special case of the problem.

2 Definitions

Let $C > 0$, $T > 0$, and $\beta \in [0, 1]$ be fixed constants. The (C, β, T)*–Bahncard Problem* (or shortly $\mathrm{BP}(C, \beta, T)$) is a request-answer game between an algorithm A (the traveler) and an adversary (real life). The adversary presents a finite sequence of *travel requests* $\sigma = \sigma_1\sigma_2\cdots$. Each σ_i is a pair (t_i, p_i), where $t_i \geq 0$ is the travel time and $p_i \geq 0$ is the *regular price* of the ticket. The requests are presented in chronological order, i.e., $0 \leq t_1 < t_2 < \cdots$.

The task of A is to react to each travel request by buying a ticket (that cannot be avoided), but A can also decide to first buy a Bahncard. A Bahncard bought at time t is *valid* during the time interval $[t, t + T)$. A's *cost* on σ_i is

$$c_A(\sigma_i) = \begin{cases} \beta \cdot p_i & \text{if} \quad A \text{ has a valid Bahncard at time } t_i \\ p_i & \phantom{\text{if}} \quad \text{otherwise.} \end{cases}$$

We call $\beta \cdot p_i$ the *reduced price* of the ticket. Accordingly, σ_i is a *regular request* for A if A already had a valid Bahncard at t_i; otherwise it is a *reduced request*. Note that A might buy a Bahncard at a regular request and then pay the reduced price for the ticket.

If A buys Bahncards at times $0 \leq \tau_1 < \cdots < \tau_k$ then we call the sequence $\Gamma_A(\sigma) = (\tau_1, \ldots, \tau_k)$ the *B-schedule* of A on σ (since σ is finite, the B-schedule is also finite). We denote the *length* k of the B-schedule by $|\Gamma_A(\sigma)|$. The total cost of A on σ is then

$$c_A(\sigma) = |\Gamma_A(\sigma)| \cdot C + \sum_{i \geq 1} c_A(\sigma_i) .$$

In the following, we do not always distinguish clearly between an algorithm A and its B-schedule Γ_A, so $c_{\Gamma_A}(\sigma)$ means the same as $c_A(\sigma)$, for example.

Besides the total cost of A, we are interested in partial costs during some time interval I. Let

$$p^I(\sigma) = \sum_{i : t_i \in I} p_i$$

be the cost of all requests in I, and let

$$c_A^I(\sigma) = \sum_{i:t_i \in I} c_A(\sigma_i)$$

be the money spent by A on tickets during I. We call I *cheap* if

$$p^I(\sigma) < c_{crit} ,$$

where

$$c_{crit} = \frac{C}{1-\beta}$$

is the *critical cost*; otherwise, I is *expensive*. c_{crit} is the break-even point for any algorithm. Buying a Bahncard at the beginning of an expensive interval saves money in comparison to paying the regular price for all tickets in I. Observation 1 (b) below makes this more precise.

We are mainly interested in intervals of length T. The *T-recent-cost* (or *T-cost* for short) of σ at time t is

$$r^\sigma(t) = p^{(t-T,t]}(\sigma) .$$

The *regular T-cost*

$$rr_A^\sigma(t) = \sum_{i\,:\,\sigma_i \text{ is a regular request in } I} p_i$$

of A on σ at time t is the sum of all regular requests in $(t-T,t]$ with respect to A's B-schedule. Sometimes, we do not want the current request at time t to be included in the summation when computing $r^\sigma(t)$ or $rr_A^\sigma(t)$. Then we speak of the T$-$*cost* at t^- instead.

For any request sequence σ there is a B-schedule $OPT(\sigma)$ of minimal cost $c_{opt}(\sigma)$. In general, $OPT(\sigma)$ is not unique and can only be computed by an *offline algorithm* OPT which knows the entire sequence σ in advance.

Observation 1. *Let σ be a request sequence and $OPT(\sigma)$ be an optimal B-schedule for σ. Then we can assume w.l.o.g. that*

(a) OPT never buys a Bahncard at a reduced request.

(b) If I is an expensive time interval of length at most T then OPT has at least one reduced request in I.

Proof.

(a) Postponing the purchase of a new Bahncard until the next regular request cannot increase the cost of a B-schedule.

(b) Otherwise, buying a Bahncard at the first request in I would save money because

$$p^I(\sigma) \geq C + \beta \cdot p^I(\sigma)$$

for any expensive interval I. □

An *online algorithm* A must compute its B-schedule $\Gamma_A(\sigma)$ on the fly, i.e., whenever it receives a new request σ_i it must decide immediately if it wants to add t_i to its B-schedule, without knowing future requests $\sigma_{i+1}, \sigma_{i+2}, \ldots$. Once bought, a Bahncard cannot be reimbursed, so A cannot change its B-schedule later on. If A uses randomization then the cost of A on a fixed request sequence σ is a random variable whose expected value is also denoted by $c_A(\sigma)$. A is *d-competitive* if

$$c_A(\sigma) \leq d \cdot c_{opt}(\sigma)$$

for all request sequences σ. A is an *optimal online algorithm* if its *competitive ratio d* is the smallest possible among all online algorithms. If A is a randomized algorithm then this definition describes competitiveness against an *oblivious adversary* (see [2] for definitions of oblivious and adaptive adversaries and their respective strengths). Intuitively, an oblivious adversary must fix the request sequence σ before A starts serving the requests. In contrast, an *adaptive adversary* can construct the request sequence step by step, dependent on previous decisions of A. This makes it more difficult for a randomized online algorithm to be competitive. However, we do not expect real life to behave like an adaptive adversary (ignoring Murphy's Law), so we assume an oblivious adversary throughout this paper.

3 An Optimal Offline Algorithm

We note that the proof of Observation 1 (b) does not imply that an optimal algorithm will buy a Bahncard whenever it reaches the first regular request of an expensive time interval. In Fig. 1, both the intervals $[0, T)$ and $[T, 2T)$ are expensive, but if ϵ is small then the optimal algorithm would buy just one Bahncard at the second request.

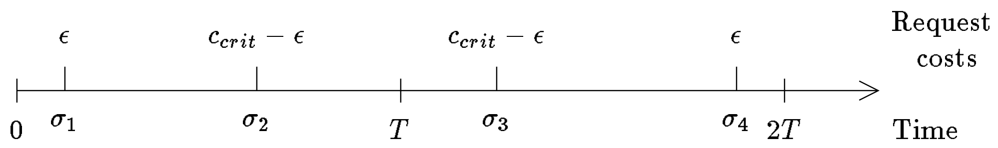

Fig. 1. Two expensive intervals, but an optimal algorithm buys at σ_2.

Theorem 2. *Given n travel requests, we can compute an optimal B-schedule and its minimal cost in time $O(n)$.*

Proof. Let $\sigma = \sigma_1 \cdots \sigma_n$ be a sequence of n travel requests. We construct a weighted acyclic directed graph G_σ with nodes $s = \sigma_0, \sigma_1, \ldots, \sigma_n, \sigma_{n+1} = t$, where $s = (0,0)$ and $t = (t_n + T, 0)$ are two new artificial requests. G_σ has the property that $(s \to^* t)$–paths in G_σ correspond to B-schedules, and any shortest $(s \to^* t)$–path corresponds to an optimal B-schedule.

For $i = 0, \ldots, n$, there is an edge $\sigma_i \to \sigma_{i+1}$ of weight p_i, and an edge $\sigma_i \to \sigma_i^{+T}$ of weight q_i, where σ_i^{+T} is the first request after (or at) time $t_i + T$, and q_i is the accumulated cost of buying a Bahncard at request σ_i and paying reduced ticket prices until this Bahncard expires, i.e.,

$$q_i = C + \sum_{j: t_i \le t_j < t_i + T} \beta \cdot p_j \ .$$

Fig. 2 shows the graph G_σ corresponding to the requests of Fig. 1.

Since we can in time $O(n)$ compute all the edge weights as well as a shortest $(s \to^* t)$–path by scanning the nodes $\sigma_0, \ldots, \sigma_{n+1}$ in increasing order [10], the theorem follows. □

4 Deterministic Online Algorithms

The *Buy-Never-Algorithm (NEVER)* which never buys a Bahncard is obviously $\frac{1}{\beta}$-competitive. Unfortunately, this is only optimal if $\beta = 1$, i.e., buying a Bahncard does not save any money. Before we analyze other algorithms, we show a lower bound on the deterministic competitive ratio.

Theorem 3. *No deterministic online algorithm for BP(C, β, T) can be better than $(2 - \beta)$-competitive.*

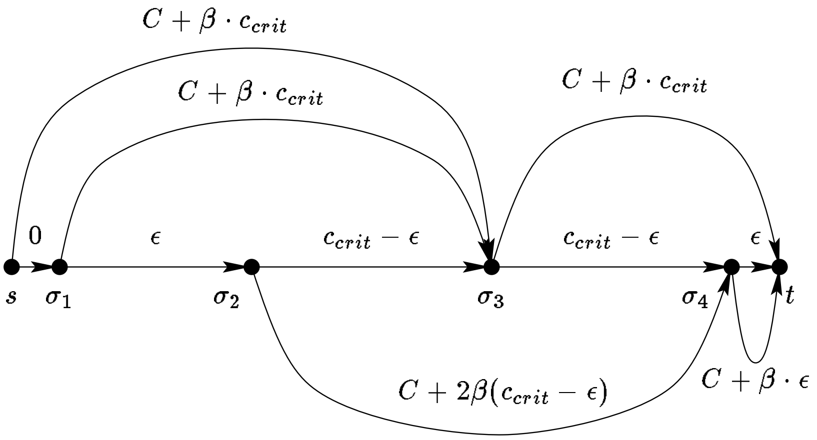

Fig. 2. The graph G_σ corresponding to the requests of Fig. 1.

Proof. Let A be an online algorithm for $\mathrm{BP}(C, \beta, T)$. Let $\epsilon > 0$ be an arbitrarily small constant. As long as A has not got a Bahncard, the adversary continues showing requests of cost ϵ (arbitrarily dense, so that all requests are in the interval $[0, T)$). If A wants to be better than $\frac{1}{\beta}$-competitive, it must eventually buy a Bahncard. Then the adversary stops showing requests. Let s be the accumulated cost of the requests so far, not including the current request. Then

$$c_A = C + s + \beta\epsilon$$

and

$$c_{opt} = \begin{cases} s + \epsilon & \text{if} \quad s + \epsilon \leq c_{crit} \\ C + \beta(s + \epsilon) & \phantom{\text{if}} \quad s + \epsilon \geq c_{crit} \end{cases} .$$

Hence

$$\begin{aligned} \frac{c_A}{c_{opt}} &= \begin{cases} \frac{C+s+\beta\epsilon}{s+\epsilon} & \text{if} \quad s + \epsilon \leq c_{crit} \\ \frac{C+s+\beta\epsilon}{C+\beta(s+\epsilon)} & \phantom{\text{if}} \quad s + \epsilon \geq c_{crit} \end{cases} \\ &\geq 2 - \beta - \frac{\epsilon(1-\beta)^2}{C} , \end{aligned}$$

because both quotients achieve their minimum at $s = c_{crit} - \epsilon$. Since ϵ can be arbitrarily small, the theorem follows. □

Clerks at railway ticket offices usually advise their customers to buy a Bahncard iff they are planning to buy one or more tickets of total cost at least c_{crit}. This *Ticket-Office-Algorithm (TOA)* has the advantage of being memoryless (cf. [8]), however its competitive ratio is the same as that of NEVER : If the request sequence consists of many travel requests of cost slightly less than c_{crit} within a short time interval, then TOA never buys a Bahncard, whereas the optimal algorithm would buy one at the first request.

TOA seems to fail because it tries to handle expensive requests optimally but it cannot safeguard against a sequence of several cheap requests. To achieve a good performance for both types of request sequences we must allow for non-optimal behaviour in both cases. The proof of the lower bound indicates that the following *Sum-Algorithm (SUM)* might behave better than TOA. SUM buys a Bahncard at a regular request (t, p) iff the regular T-cost at t is at least c_{crit}, i.e.

$$rr^{\sigma}_{sum}(t) \geq c_{crit} .$$

In the example of Fig. 1, SUM would buy a Bahncard at the second request (and thus incidentally behave optimally).

Theorem 4. *SUM is $(2-\beta)$-competitive for BP(C, β, T).*

Proof. Let $\sigma = \sigma_1\sigma_2\cdots$ be a request sequence and let $OPT(\sigma) = (\tau_1, \ldots, \tau_k)$ be an optimal B-schedule for σ. This divides time into *epochs* $[\tau_j, \tau_{j+1})$, $0 \leq j \leq k$, where we assume $\tau_0 = 0$ and $\tau_{k+1} = \infty$. Each epoch (except for, possibly, the first and last one) starts with an *expensive phase* $[\tau_j, \tau_j + T)$, followed by a *cheap phase* $[\tau_j + T, \tau_{j+1})$.

SUM will buy at most one Bahncard during any epoch. This follows from Observation 1 (b) and the fact that $(t - T, t]$ must be an expensive interval if SUM buys a Bahncard at time t. Therefore, we can upper bound SUM's total cost of buying Bahncards by assuming that SUM spends C in every expensive phase, in addition to ticket costs.

Clearly , $c^I_{sum}(\sigma) \leq c^I_{opt}(\sigma)$ for a cheap phase I. So let I be any expensive phase. Let c_{sum} and c_{opt} denote SUM's and OPT's cost during I, respectively (including the cost of buying Bahncards). We divide I into three subphases I_1, I_2, I_3 (some of which can be empty); in I_1 and I_3, SUM has a valid Bahncard, whereas it must pay regular prices in I_2. For $i \in \{1, 2, 3\}$, let

$$s_i = p^{I_i}(\sigma)$$

be the total cost of requests in I_i. Then

$$c_{sum} \leq C + s_2 + \beta \cdot (s_1 + s_3)$$

and

$$c_{opt} = C + \beta(s_1 + s_2 + s_3)\,.$$

Hence

$$\begin{aligned}\frac{c_{sum}}{c_{opt}} &\leq \frac{C+s_2+\beta\cdot(s_1+s_3)}{C+\beta(s_1+s_2+s_3)}\\ &\leq \frac{C+c_{crit}}{C+\beta\cdot c_{crit}}\\ &= 2-\beta\,,\end{aligned}$$

because the second quotient is maximal if $s_1 = s_3 = 0$ and if s_2 is maximal, and the definition of SUM implies $s_2 \leq c_{crit}$. □

So SUM is optimal for the Bahncard Problem. In particular, it is $\frac{3}{2}$-competitive for the GBP. For the SRP, it behaves like the well-known optimal 2-competitive algorithm which buys at the N-th request [3].

However, SUM tends to be pessimistic about the future : It always buys at the latest possible time, namely after it has seen enough regular requests to know for sure that an optimal algorithm would already have bought a Bahncard. In contrast to that, we consider the *Optimistic-Sum-Algorithm (OSUM)* which buys a Bahncard at a regular request (t, p) iff

$$p \geq \frac{C - s(1-\beta)}{2(1-\beta)}\,,$$

where s is the regular T-cost at t^-, i.e.

$$s = rr^{\sigma}_{osum}(t^-)\,.$$

Observe that OSUM will never buy its i-th Bahncard later than SUM (because OSUM buys when s reaches c_{crit}), but often will buy earlier. Consider for example the GBP. Then OSUM buys a Bahncard whenever $p \geq C - \frac{1}{2}s$. On the request sequence (Jun 22, 250 DM), (Jun 26, 100 DM), (Jul 17, 50 DM), (Jul 31, 200 DM), for example, OSUM would buy a Bahncard at the first request on Jun 22 and spend 540 DM for all four tickets (which is optimal), whereas SUM would pay the regular price for the first three tickets and buy a Bahncard only at the fourth request on Jul 31, thus spending 740 DM.

Of course, OSUM's advantage over SUM shrinks if there are many cheap requests, and if all requests are infinitesimally small then OSUM converges to SUM. Nevertheless, OSUM should be used by frequent travelers who expect to buy more tickets in the near future, whereas SUM should be preferred by sporadic travelers with a low probability of traveling. The next theorem shows that OSUM is as optimal as SUM.

Theorem 5. *OSUM is $(2-\beta)$-competitive for BP(C,β,T).*

Proof. We would like to argue similar to the proof of Theorem 4. However, OSUM might buy more Bahncards than OPT, so we can no longer charge the cost of OSUM's Bahncards to the expensive phases. Therefore, we augment the proof of Theorem 4 by introducing *critical phases.* If OSUM buys a Bahncard at some time t, then let t' be the maximum of $t-T$ and the expiration time of OSUM's previous Bahncard. If the interval $I=(t',t]$ has a non-empty intersection with an expensive phase then we can, as before, charge OSUM's cost for this Bahncard to this expensive phase. Otherwise, we call I a critical phase and charge OSUM's cost for this Bahncard to this critical phase.

Now, an epoch consists of an expensive phase, followed by an alternating sequence of cheap phases and critical phases. The cheap and expensive phases can be analyzed as in the proof of Theorem 4. A critical phase is induced by a request (t,p) with

$$p \geq \tfrac{C-s(1-\beta)}{2(1-\beta)} =: a \ ,$$

where $s = rr^{\sigma}_{osum}(t^-)$. Hence

$$c_{osum} = C + s + \beta p$$

and

$$c_{opt} = s + p \ .$$

Therefore

$$\begin{aligned} \tfrac{c_{osum}}{c_{opt}} &= \tfrac{C+s+\beta p}{s+p} \\ &\leq \tfrac{C+s+\beta a}{s+a} \\ &= 2-\beta \ , \end{aligned}$$

because the second quotient is maximal if p is minimal. □

5 Randomized Online Algorithms

We define *R-SUM (R-OSUM)* as a randomized variant of SUM (OSUM) which, with probability $q = \frac{1}{1+\beta}$, buys a Bahncard at time t iff SUM (OSUM) would buy one at time t. It is easy to see from the proof of the next theorem that $\frac{1}{1+\beta}$ is the optimal choice for this probability.

Theorem 6. *R-SUM and R-OSUM are $\frac{2}{1+\beta}$-competitive for BP(C, β, T).*

Proof. We use the same notation as in the proofs of Theorems 4 and 5. Cheap phases are analyzed as in the deterministic case. If $I = I_1 \cup I_2 \cup I_3$ is an expensive phase then

$$c_{rsum} \leq qC + s_2 + (q\beta + 1 - q) \cdot (s_1 + s_3)$$

and

$$c_{opt} = C + \beta(s_1 + s_2 + s_3)\ .$$

Hence

$$\begin{aligned} \frac{c_{rsum}}{c_{opt}} &= \frac{qC+s_2+(q\beta+1-q)\cdot(s_1+s_3)}{C+\beta(s_1+s_2+s_3)} \\ &\leq \frac{C\cdot\frac{2}{(1+\beta)(1-\beta)}+\frac{2\beta}{1+\beta}\cdot(s_1+s_3)}{C\cdot\frac{1}{1-\beta}+\beta(s_1+s_3)} \\ &= \frac{2}{1+\beta}\ , \end{aligned}$$

because the second quotient is maximal if s_2 is maximal, and $s_2 \leq c_{crit}$ by definition of SUM (or OSUM).

If I is a critical phase then

$$c_{rosum} = s + q(C + \beta p) + (1 - q)p$$

and

$$c_{opt} = s + p\ .$$

Hence

$$\begin{aligned}
\frac{c_{rosum}}{c_{opt}} &= \frac{s+q(C+\beta p)+(1-q)p}{s+p} \\
&\leq \frac{s+q(C+\beta a)+(1-q)a}{s+a} \\
&= \frac{s+\frac{1}{1+\beta}(C+\beta\cdot\frac{C-s(1-\beta)}{2(1-\beta)})+\frac{\beta}{1+\beta}\cdot\frac{C-s(1-\beta)}{2(1-\beta)}}{s+\frac{C-s(1-\beta)}{2(1-\beta)}} \\
&= \frac{s\cdot\frac{1}{1+\beta}+C\cdot\frac{1}{(1+\beta)(1-\beta)}}{s\cdot\frac{1}{2}+C\cdot\frac{1}{2(1-\beta)}} \\
&= \frac{2}{1+\beta}\ ,
\end{aligned}$$

because the second quotient is maximal if p is minimal. □

Note that $\frac{2}{1+\beta} < 2-\beta$ if $\beta \in (0,1)$, so R-SUM usually beats SUM. It is $\frac{4}{3}$-competitive for the GBP, but for the SRP it is identical to the deterministic SUM algorithm.

We now consider the case that $T=\infty$, i.e., a Bahncard never expires. This makes the problem more similar to the well-understood SRP. In this case, time is no longer important, and we can w.l.o.g. assume that the behaviour of an algorithm at any moment is completely determined by the sum of all previous requests. A deterministic algorithm A can thus be described by a single positive number s_A, meaning that A buys a Bahncard if the cost has reached s_A. A randomized algorithm Q can be described by a monotone increasing function $p_Q : [0,\infty] \to [0,1]$, where $p_Q(s)$ is the probability that Q has a Bahncard after the cost has reached s. Since small requests work in favour of the adversary, we can further assume w.lo.g. that the total ticket cost is a continous function of time (and monotone increasing, of course). Then, a request sequence is also just a positive number s, namely the sum of all requests, and the expected cost of Q on s is

Lemma 7. *Let Q be a randomized algorithm. Then Q has expected cost*

$$c_Q(s) = p_Q(s)\cdot C + s - (1-\beta)\cdot\int_0^s p_Q(x)dx$$

for any request sequence s.

Proof. The expected cost of all tickets is

$$\beta\cdot\int_0^s p_Q(x)dx + \int_0^s (s-p_Q(x))dx\ .$$ □

We now define the randomized algorithm $RAND$ by

$$p_{rand}(s) = \begin{cases} \frac{e^{\frac{s}{c_{crit}}}-1}{e-1+\beta} & \text{if} \quad s \le c_{crit} \\ \frac{e-1}{e-1+\beta} & \phantom{\text{if}} \quad s \ge c_{crit} \end{cases} .$$

Since p is monotone increasing and $p(c_{crit}) < 1$, RAND is well defined.

Theorem 8. *RAND is $\frac{e}{e-1+\beta}$-competitive for BP(C, β, ∞).*

Proof. Let s be a request sequence. If $s \le c_{crit}$ then

$$c_{opt}(s) = s$$

and, by Lemma 7,

$$\begin{aligned} c_{rand}(s) &= \frac{e^{\frac{s}{c_{crit}}}-1}{e-1+\beta} \cdot C + s + (1-\beta) \cdot \int_0^s \frac{e^{\frac{x}{c_{crit}}}-1}{e-1+\beta} dx \\ &= \frac{e^{\frac{s}{c_{crit}}}-1}{e-1+\beta} \cdot C + s - \frac{1-\beta}{e-1+\beta} \cdot \left(\frac{C}{1-\beta} e^{\frac{s}{c_{crit}}} - s - \frac{C}{1-\beta}\right) \\ &= s \cdot \frac{e}{e-1+\beta} . \end{aligned}$$

If $s \ge c_{crit}$ then

$$c_{opt}(s) = C + \beta s$$

and

$$\begin{aligned} c_{rand}(s) &= c_{rand}(c_{crit}) + (1 - p_{rand}(c_{crit})(1-\beta)) \cdot (s - c_{crit}) \\ &= \frac{e}{e-1+\beta} \cdot \frac{C}{1-\beta} + s - \frac{C}{1-\beta} - \frac{e-1}{e-1+\beta}(1-\beta)\left(s - \frac{C}{1-\beta}\right) \\ &= C \cdot \left(\frac{e}{(e-1+\beta)(1-\beta)} - \frac{1}{1-\beta} + \frac{e-1}{(e-1+\beta)(1-\beta)}\right) + s \cdot \left(1 - \frac{(e-1)(1-\beta)}{e-1+\beta}\right) \\ &= C \cdot \frac{e}{e-1+\beta} + \beta s \cdot \frac{e}{e-1+\beta} . \end{aligned}$$

Hence the competitive ratio is $\frac{e}{e-1+\beta}$ in both cases. □

Note that RAND always beats R-SUM or R-OSUM. If $\beta = 0$ then the Bahncard Problem becomes the SRP, and RAND behaves like the optimal $\frac{e}{e-1}$-competitive randomized Ski-Rental algorithm with $N = \infty$ [1, 6].

Theorem 9. *No randomized online algorithm for BP(C, β, T) can be better than $\frac{e}{e-1+\beta}$-competitive.*

Proof. We first prove the theorem for BP(C, β, ∞). We have seen in the proof of Theorem 8 that RAND achieves the same competitive ratio of $\frac{e}{e-1+\beta}$ on *all* request sequences.

Let Q be a randomized algorithm, defined by p_Q. We will show that if Q differs from RAND (if the two probability functions only differ on singular points then we do not consider them different because they have the same expected cost) then there exists a request sequence s with $c_Q(s) > c_{rand}(s)$. Hence, Q has a worse competitive ratio than RAND.

Let $z = p_Q(\infty)$ (note that this limit always exists for bounded monotone increasing functions). Then

$$\begin{aligned}\frac{c_Q(\infty)}{c_{opt}(\infty)} &= \frac{z\beta+(1-z)}{\beta}\\ &= \frac{1-z(1-\beta)}{\beta}\,.\end{aligned}$$

Therefore, Q can only be as good as RAND if $p_Q(\infty) \geq p_{rand}(\infty)$.

If $p_Q(s)$ is smaller than $p_{rand}(s)$ for some values of s then let $s_1 > 0$ be minimal such that there exists $\epsilon > 0$ with $p_Q(s) < p_{rand}(s)$ for $s_1 \leq s \leq s_1+\epsilon$. Let $s_2 > s_1$ be minimal such that $p_Q(s_2) \geq p_{rand}(s_2)$. Then, by Lemma 7, $c_Q(s_2) > c_{rand}(s_2)$.

Otherwise, let $s_1 > 0$ be minimal such that there exists $\epsilon > 0$ with $p_Q(s) > p_{rand}(s)$ for $s_1 \leq s \leq s_1+\epsilon$. Then, by Lemma 7, $c_Q(s_1+\epsilon) > c_{rand}(s_1+\epsilon)$. □

To generalize RAND to arbitrary T, we define another randomized algorithm *RAND2*: For $0 \leq \gamma \leq c_{crit}$, let γ-*SUM* be the deterministic algorithm which buys a Bahncard whenever its current T-cost rises above γ (so SUM is c_{crit}-SUM). RAND2 chooses $\gamma \in [0, c_{crit}]$ randomly such that the probability of $\gamma \in [0, s]$ is $p_{rand}(s)$, for $s \in [0, c_{crit}]$. If $T = \infty$ then RAND2 is identical to RAND and hence optimal.

Conjecture 10. *RAND2 is $\frac{e}{e-1+\beta}$-competitive for BP(C, β, T).* □

Acknowledgements

We want to thank Kurt Mehlhorn for reading a preliminary manuscript.

References

1. S. Albers and H. Koga. New on-line algorithms for the page replication problem. In *Proceedings of the 4th Scandinavian Workshop on Algorithm Theory (SWAT'94), Springer Lecture Notes in Computer Science 824*, pages 25–36, 1994.
2. 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.
3. D.L. Black and D.D. Sleator. Competitive algorithms for replication and migration problems. Technical Report CMU-CS-89-201, Carnegie Mellon University, 1989.
4. D.S. Johnson, A. Demers, J.D. Ullman, M.R. Garey, and R.L. Graham. Worst-case performance bounds for simple one-dimensional packing algorithms. *SIAM Journal on Computing*, 3(4):299–325, 1974.
5. A.R. Karlin, M.S. Manasse, L.A. McGeoch, and S. Owicki. Competitive randomized algorithms for nonuniform problems. *Algorithmica*, 11(6):542–571, 1994.
6. C. Lund, N. Reingold, J. Westbrook, and D. Yan. On-line distributed data management. In *Proceedings of the 2nd European Symposium on Algorithms (ESA'94), Springer Lecture Notes in Computer Science 855*, pages 202–214, 1994.
7. P. Raghavan. Lecture notes on randomized algorithms. Technical Report RC 15340 1/9/90, IBM Research Division, T.J. Watson Research Center, Yorktown Heights, New York, 1990.
8. P. Raghavan and M. Snir. Memory versus randomization in on-line algorithms. In *Proceedings of the 16th Colloqium on Automata, Languages and Programming (ICALP'89), Springer Lecture Notes in Computer Science 372*, pages 687–703, 1989.
9. D.D. Sleator and R.E. Tarjan. Amortized efficiency of list update and paging rules. *Communications of the ACM*, 28(2):202–208, 1985.
10. R.E. Tarjan. *Data Structures and Network Algorithms.* CBMS-NSF Regional Conference Series in Applied Mathematics. SIAM, Philadelphia, Pennsylvania, 1983.