Hildenbrandt, Regina

The k-Server Problem with Parallel Requests and the Compound Harmonic Algorithm

URN: urn:nbn:de:gbv:ilm1-2016200094

Original published in:

Baltic journal of modern computing, Vol.4 (2016), Nr. 3 p. 607-629.

Original published:	2016-06-17
ISSN (online):	2255-8950
ISSN (print):	2255-8942
URL:	http://www.bjmc.lu.lv/contents/vol-42016-no-3/

Baltic J. Modern Computing, Vol. 4 (2016), No. 3, pp. 607–629

The k-Server Problem with Parallel Requests and the Compound Harmonic Algorithm

R. HILDENBRANDT

Institute of Mathematics, Ilmenau Technical University,
PF 10 06 65, 98684 Ilmenau, Germany

r.hildenbrandt@tu-ilmenau.de

Abstract. In this paper the (randomized) compound Harmonic algorithm for solving the generalized k-server problem is proposed. This problem is an online k-server problem with parallel requests where several servers can also be located on one point. In 2000 Bartal and Grove have proved that the well-known Harmonic algorithm is competitive for the (usual) k-server problem. Unfortunately, certain techniques of this proof cannot be used to show that a natural generalization of the Harmonic algorithm is competitive for the problem with parallel requests. The probabilities, which are used by the compound Harmonic algorithm are, finally, derived from a surrogate problem, where at most one server must be moved in servicing the request in each step. We can show that the compound Harmonic algorithm is competitive with the bound of the ratio as which has been proved by Bartal and Grove in the case of the usual problem.

Keywords: Server problems, compound Harmonic algorithm, absorbing probabilities, competitive analysis

1 Introduction

This paper deals with a generalized online k-server problem. We want to present a new algorithm, the ”compound Harmonic algorithm”, and show its qualities for solving this problem.

In (Hildenbrandt, 2014) we have introduced a generalized k-server problem with parallel requests where several servers can also be located on one point. We are given initial locations of k servers in a metric space. Requests r^t for service at several points come in over time. It is sensible in the case of such requests to distinguish the surplus-situation where the request can be completely fulfilled by means of the k servers and and the scarcity-situation where the request cannot be completely met. Immediately after the t-th request is received, a sufficient number of servers must be moved from theirs current locations to the request points. The choice of which servers are moved, in the case of the surplus-situation or the choice of requests which should be fulfilled, in case of the scarcity-situation, respectively, must be made based only on the current servers configuration and on the requests seen so far; that is, the requests r^u for $u \leq t$. Moving servers costs the distances the servers are moved, and the goal is to minimize the total cost.

Thus in online computation, an algorithm must decide how to act on incoming requests without any knowledge of future inputs. In contrast, an offline procedure would be allowed to know the entire sequence of requests in advance, before it makes any decisions. We want online algorithms whose cost compares favorably to the cost of an optimal offline algorithm (in more detail, see Section 2).

In Section 3 we will develop the ”compound Harmonic algorithm” for the generalized k-server problem in the case of the surplus-situation. Certain multi-step transition probabilities and absorbing probabilities are used by the compound Harmonic algorithm. For their computation each step of the generalized k-server problem is replaced by a number of steps of other specific k-server problems. We will show that this algorithm is ”competitive”.

Until now, only the k-server problem with requests, where at most one server must be moved in servicing the request in each step, is considered in literature. This k-server problem was introduced by Manasse, McGeoch and Sleator (1988). Meanwhile it is the most studied problem in the area of competitive online problems. An important randomized algorithm for solving this problem is the Harmonic k-server algorithm. It has been introduced by Raghavan and Snir (1989). This algorithm is memoryless and time-efficient. The competitiveness of the randomized Harmonic k-server algorithm against an adaptive online adversary was proved in (Bartal and Grove, 2000). Unfortunately, certain methods used by Bartal and Grove cannot be applied to the k-server problems with parallel requests.

2 Principles of online optimization and the formulation of the model

2.1 The competitiveness of randomized online algorithm

It is important to understand the concept of the competitiveness of randomized online algorithms. One can find a good summary on this topic in (Bartal and Grove, 2000, Section 1). Nevertheless, we will state the key issues from this in the following:

Sleator and Tarjan (1985) founded the study of ”competitive” online algorithms by introducing the idea that the performance of such an algorithm should be measured by the maximum ratio of the cost it incurs on any sequence of inputs to the minimum offline cost for processing that sequence. Formally, a deterministic online algorithm is C-competitive if there exists a function I of the initial configuration so that for every finite input sequence the cost incurred by the algorithm is bounded by I plus C times the minimum cost of processing the input sequence. C is called the competitive ratio of the algorithm. The competitive ratio is a measure of how much better we could do if we knew the future.

In this paper, we study a randomized online algorithm for the k-server problem with parallel request and thus we need a generalization of the notion of the competitive ratio to the case of randomized online algorithms. Ben-David et al. (1994) introduced a framework for studying randomized algorithms for online problems called request-answer games. In this framework, we view the online algorithm as a player in a game against an adversary. The adversary produces a request sequence and gives requests to the algorithm one by one. The algorithm must respond with an answer specifying how it serves the given request and it incurs a cost accordingly. We view the adversary as if it generates the request sequence and must serve it as well. The competitive ratio of an algorithm is defined as the worst-case ratio between its cost and the cost of the adversary on the sequence generated by that adversary. For deterministic algorithms, this definition is equivalent to the former definition. For randomized algorithms, the definition depends on what power the adversary is given. In particular, we must specify in which way the request sequence may depend upon the random choices of the algorithm and how the adversary may serve the requests. We will restrict our attention to one type of adversary. An adaptive online adversary specifies the initial positions of k servers. The adversary and the algorithm each control k servers, respectively the offline servers and the online servers, whose positions are given by the initial configuration. The game proceeds in steps. The t-th step of the game corresponds to the t-th request in the request sequence. First, the adversary chooses offline servers, move they (possibly a distance 0) to some points, and then places corresponding requests at these points. Note that the online algorithm does not know which servers the adversary move to these points. The adversary may take into account the complete state of the online algorithm, including the results of random choices from previous phases. The adversary does not have the ability to predict the coin tosses in the current or future phases. Next, the online player move a servers to the request points. The adversary may choose to end the game after any number of steps. The algorithm is called $C(k)$-competitive if there exists a function I of the initial configuration specified by the adversary so that for every sequence generated by an adaptive online adversary

$$E[\text{cost}(\text{online algorithm})] \leq C(k) \cdot E[\text{cost}(\text{adversary})] + I, \tag{1}$$

where $\text{cost}(\text{online algorithm})$ and $cost(adversary)$ are the costs of the algorithm and the adversary on the sequence generated, respectively.

In order to prove (1) a potential function Φ could be used. Intuitively, Φ is an upper bound on the expected amount of work the algorithm can be forced to do if the offline servers do not move. Here we will use a potential function, which is a function of the current locations of the online and offline servers. Now, let Φ_t denote the value of Φ at the end of the t-th step (corresponding to the t-th request r^t in the request sequence) and let $\Phi_t^{\sim}$ denote the value of Φ after the first stage of the t-th step (i.e., after the adversary's move and before the algorithm's move). If such a potential function exists, which satisfies the following properties with regard to a randomized online algorithm ALG:

$$\Phi \geq 0 \tag{2}$$

$$\Phi_t^{\sim} - \Phi_{t-1} \leq C(k) D_t, \tag{3}$$

where D_t denotes the distance moved by the offline servers (controlled by the adversary) to serve the request in the t-th step,

$$E(\Phi_t^{\sim} - \Phi_t) \geq E(Z_t), \tag{4}$$

where Z_t represents the cost which incurred by the online algorithm to serve the request in the t-th step,

then algorithm ALG is C(k)-competitive, see (Bartal and Grove, 2000, Lemma 1).

2.2 The k-server problem with parallel requests

Now, we want to describe the generalized k-server problem. Let $k \geq 1$ be an integer, and $M = (M, d)$ be a finite metric space where M is a set of points with $|M| = N$. An algorithm controls k mobile servers, which are located on points of M. Several servers can be located on one point. Requests r^t for service at several points come in over time. Let $\sigma = r^1, r^2, \cdots, r^n$ be such a sequence of requests. A request r is defined as an N-ary vector of integers with $r_i \in \{0, 1, \cdots, k\}, i = 1, 2, \cdots, N$ ("parallel requests"). The request means that r_i servers are needed on point $i, i = 1, 2, \cdots, N$.

Principally, two cases of requests have to distinguished: $\sum\limits_{i=1}^{N} r_i \leq k$ describes the surplus-situation. The request can be completely fulfilled. We say a request r is served if at least r_i servers lie on $i, i = 1, 2, \cdots, N$. In contrast, $\sum\limits_{i=1}^{N} r_i \geq k$ means the scarcity-situation. The request cannot be completely met, however it should be met as much as possible. The request r is served if at most r_i servers lie on $i, i = 1, 2, \cdots, N$.

By moving servers, the algorithm must serve the requests $r^1, r^2, \cdots, r^n$ sequentially. For any request sequence σ and any k-server algorithm ALG, ALG(σ) is defined as the total distance (measured by the metric d) moved by the ALG's servers in servicing σ.

Analogous to (Borodin and El-Yaniv, p. 152) working with lazy algorithms ALG is sufficient. This means, servers are not moved in a step if they are not needed to fulfil requests in this step. For that reason we define the set of feasible servers' positions with respect to the previous servers' positions s and the request r in the following way

$$\hat{A}_{N;k}(s, r) \quad = \{s' \in S_N(k) \,|r_i \leq s'_i \leq \max\{s_i, r_i\},\ i = 1, \cdots, N\ \} \tag{5}$$

$$\text{where } \ S_N(k) := \left\{ s \in \mathbb{Z}_+^N \mid \sum_{i=1}^{N} s_i = k \right\} \tag{6}$$

in the case of the surplus-situation and

$$\hat{A}_{N;k}(s, r) \quad = \{s' \in S_N(k) \,|\min\{s_i, r_i\} \leq s'_i \leq r_i,\ i = 1, \cdots, N\} \tag{5a}$$

in the case of the scarcity-situation.

The metric d implies that $(S_N(k), \hat{d})$ is also a finite metric space where $\hat{d}$ are the optimal values of the classical transportation problems with availabilities s and requirements $s' \in S_N(k)$: $\sum_{i=1}^{N} \sum_{j=N}^{N} d(i,j)\, x_{ij} \;\rightarrow\; min$

subject to $\sum_{j=1}^{N} x_{ij} = s_i \; \forall i, \; \sum_{i=1}^{N} x_{ij} = s'_j \; \forall j, \; x \in \mathbb{Z}_{+}^{N} \times \mathbb{Z}_{+}^{N},$

see (Hildenbrandt, 1995, Lemma 3.6).

In the following we will develop the compound Harmonic k-server algorithm and show that this algorithm is competitive (see Theorem 1) against an adaptive online adversary in the case of the surplus-situation.

3 The compound Harmonic algorithm

Firstly, we examine a natural generalization of the well-know Harmonic algorithm in this section. Then we derive the compound Harmonic k-server algorithm for the generalized k-server problem.

3.1 Considerations concerning the Harmonic algorithm

The randomized Harmonic algorithm, applied to the usual k-server problem, sends each server with probability proportional to the inverse of its distance from the current request location. A natural generalization of this algorithm, adapted to the k-server problem with parallel requests, is the following: Harmonic serves a not completely covered request r with randomly chosen servers so that for the new servers' positions $s' \in \hat{A}_{N;k}(s,r)$ is valid with respect to the previous servers' positions s and the request r. More precisely, Harmonic leads to $s' \in \hat{A}_{N;k}(s,r)$ with probability

$$P_H(s'|s,r) = \frac{1/\hat{d}(s,s')}{\sum\limits_{s'':s'' \in \hat{A}_{N;k}(s,r)} 1/\hat{d}(s,s'')}. \tag{7}$$

(The quantity $\sum\limits_{s'':s'' \in \hat{A}_{N;k}(s,r)} 1/\hat{d}(s,s'')$ is referred to as the normalization factor.)

At first we give an example that the Harmonic algorithm is not competitive in general, where the case $\sum_{i=1}^{N} r_i \geq k$ is allowed.

Example 1 *We focus on a number of similar examples $E1, E2, \cdots, Ej, \cdots$ with $k = 1$, finite metric spaces $M_j = \{-j, -j+1, \cdots, 0, \cdots, j-1, j\}$ and request sequences $\{r^1, r^2, \cdots, r^n\}$, $n \leq j$. Let the following assumptions be valid for all examples. The usual metric of integers is used. If the server of the adversary is located on point $\bar{s} \in \mathbb{Z}$ and the server of the algorithm on point $s \in \mathbb{Z}$ then the adversary produces the request r with*

$$r_i = \begin{cases} \begin{cases} 1 & \textit{if } \; i = s-1 \textit{ or } \; i = s+1 \\ 0 & \textit{otherwise} \end{cases} & \textit{if } \; \bar{s} = s \\ \begin{cases} 1 & \textit{if } \; i = \bar{s} \textit{ or } \; i = s+1 \\ 0 & \textit{otherwise} \end{cases} & \textit{if } \; \bar{s} \neq s. \end{cases}$$

The adversary moves his server to another point, more precisely to $s-1$, if and only if the servers of the adversary and of the algorithm are located on the same point.

We assume that $\bar{s} \leq s$ at the beginning. Then

$$\bar{s} \leq s \tag{8}$$

is valid in every step. Furthermore, we use the following symbols

$$\delta = \begin{cases} s - \bar{s} \ \textit{if}\ s \neq \bar{s} \\ 1 \ \textit{if} \ \ s = \bar{s} \end{cases}$$

and

$h^l_\delta = E[\text{cost}(\text{Harmonic algorithm})]$
with regard to l steps and δ at the beginning,

$a^l_\delta = E[\text{cost}(\text{adversary})]$
with regard to l steps and δ at the beginning

for the the expected costs. Then

$$\lim_{l \to \infty} \frac{h^l_1}{a^l_1} = \infty \tag{9}$$

can be proved (see Appendix A). Thus the Harmonic algorithm is not competitive for such examples, with the severs' start position $s = \bar{s}$ and where the lengths l of the request sequences tends to infinity.

Bartal and Grove (2000) have used a potential function in order to prove that the Harmonic k-server algorithm against an adaptive online adversary is competitive for the (usual) k-server problem. As we will see in Example 2, such methods cannot be applied to show the competitiveness of the Harmonic algorithm for the k-server problem with parallel requests.

With regard to the Harmonic k-server algorithm Bartal and Grove have constructed the following potential function: Let OFF be the set of offline servers, and ON the set of online servers. Y. Bartal and E. Grove have defined a weighted bipartite graph G on the online and offline servers in the following way. Given an online server x and an offline server Y, then all paths from x to Y in $\{x\} \cup OFF$ are considered. The length of the j-th step of a path is weighted by a scaling function $\hat{f}_j$ that is very large for small j and decreases monotonically. The weight of the edge from x to Y in G is the minimum scaled length of a simple path from x to Y in $\{x\} \cup OFF$. Let p be an assignment of servers to points in the metric space then

$$w(x,Y) = \min_{\{Y_1,\cdots,Y_l = Y\} \subset OFF} \left\{ \hat{f}_1 \cdot d\left(p(x), p(Y_1)\right) + \sum_{2 \leq j \leq l} \hat{f}_j \cdot d\left(p(Y_{j-1}), p(Y_j)\right) \right\}, \tag{10}$$

where $\hat{f}_j$ are weights with $\hat{f}_1 > \hat{f}_2 > \cdots > \hat{f}_N$.

The potential function is:

$$\Phi = \min_{\bar{M}:\, ON \leftrightarrow OFF} \sum_{x \in ON} w(x, \bar{M}(x)). \tag{11}$$

This potential function is a function of the current locations of the online and offline servers. The weights $\hat{f}_j$ are computed in such a way that (4) is valid.

Additionally, let

$\bar{s}$ $(\in S_N(k))$ denote the (offline) servers' positions controlled by the adversary at the end of the $(t-1)$-th step (i.e., at the beginning of the t-th step),

s $(\in S_N(k))$ denote the (online) servers' positions controlled by the algorithm at the beginning of the t-th step,

s' $(\in \hat{A}_{N;k}(s, r^t))$ denote the online servers' positions at the end of the t-th step and

$\bar{s}'$ $(\in S_N(k))$ denote the (offline) servers' positions controlled by the adversary after the first stage of the t-th step.

Example 2 *Let $k = 4$ and let a metric space M consist of 3 points p_1, p_2, p_3 with the pairwise distance of 1 and certain points $p \in [2, \infty)$ on the line. The distance of two points p_i, p_j $(i, j \notin \{1, 2, 3\})$ on the line is $d(p_i, p_j) = |p_i - p_j|$ as usual. The distance $d(p_i, p)$ for $i \in \{1, 2, 3\}$ and $p \in [2, \infty)$ is defined as $d(p_i, p) := p$.*

At the beginning let the online and offline servers be located on $p_1, p_2, 2, 5$.

Additionally, we set:

$$d_0 = (d_0(l) =)\ 1 + l\ ,\ d_1 = (d_1(l) =) \begin{cases} 3 \ \text{ if } l = 1 \\ 3 + \frac{3}{2} \sum\limits_{q=3}^{l+1} \sqrt{q} \ \text{ if } \ l = 2, 3, \cdots \end{cases},$$

$d_2 = (d_2(l) =)\ \frac{3}{2}\sqrt{l+2}$ *and* $d_3 = (d_3(l) =)\ \frac{3}{2}\sqrt{l+1}$ *for* $l = 1, 2, \cdots$.

Possible server configurations $C_a(l), \cdots, C_i(l)$, corresponding requests r and answers by the adversary can be found in Appendix B. Here we focus on the configuration $C_c(l)$:

$C_c(l)$: *ON is located on* $p_1, p_2, p_3,\ d_0 + d_1$ *and*
OFF is located on $p_i, p_j,\ i, j \in \{1, 2, 3\},\ d_0, d_0 + d_1$ at the beginning of the t-step.

r : one server on $d_0 + 1$ and one server on $d_0 + d_1 + d_2$ are needed.

Answer by the adversary: the offline servers on $d_0, d_0 + d_1$ are moved.

We will show that property (4) of the potential function for sufficiently large l cannot be fulfilled.

According to (10) and (11), $\Phi_t^{\sim}(s, \bar{s}') = \hat{f}_1\ d_2 + \hat{f}_1 + \hat{f}_2 + \hat{f}_3\ (d_0 + 1)$ for sufficiently large l. $\hat{A}_{N;k}(s, r)$ includes the following 6 elements:

$s'^{(1)}$: *servers on $p_i, p_j,\ d_0 + 1, d_0 + d_1 + d_2$ (that means $s'^{(1)} = \bar{s}'$),*
$s'^{(2)}$: *servers on $p_i, p_q,\ q \in \{1, 2, 3\} \setminus \{i, j\},\ d_0 + 1, d_0 + d_1 + d_2$,*
$s'^{(3)}$: *servers on $p_j, p_q,\ q \in \{1, 2, 3\} \setminus \{i, j\},\ d_0 + 1, d_0 + d_1 + d_2$,*
$s'^{(4)}$: *servers on $p_i,\ d_0 + 1, d_0 + d_1, d_0 + d_1 + d_2$,*
$s'^{(5)}$: *servers on $p_j,\ d_0 + 1, d_0 + d_1, d_0 + d_1 + d_2$,*
$s'^{(6)}$: *servers on $p_q,\ q \in \{1, 2, 3\} \setminus \{i, j\},\ d_0 + 1, d_0 + d_1, d_0 + d_1 + d_2$.*

The distances between the online servers' positions at the beginning and at the end of the t-th step, in other words, the cost which incurred by the online algorithm to serve the request, are $\hat{d}(s, s'^{(1)}) = Z_t(s, s'^{(1)}) = d_0 + 1 + d_2,\ \hat{d}(s, s'^{(2)}) = Z_t(s, s'^{(2)}) = d_0 + 1 + d_2,\ \hat{d}(s, s'^{(3)}) = Z_t(s, s'^{(3)}) = d_0 + 1 + d_2,\ \hat{d}(s, s'^{(4)}) = Z_t(s, s'^{(4)}) = 2\ d_0 + 1 + d_1 + d_2,\ \hat{d}(s, s'^{(5)}) = Z_t(s, s'^{(5)}) = 2\ d_0 + 1 + d_1 + d_2$ *and* $\hat{d}(s, s'^{(6)}) = Z_t(s, s'^{(6)}) = 2\ d_0 + 1 + d_1 + d_2$.

The Harmonic algorithm realizes $s'^{(i)}$ *with probability:*

$P_H(s'^{(i)}|s, r) = \frac{1/\hat{d}(s,s'^{(i)})}{N_f}$ *for* $i = 1, 2, \cdots 6$, *where* $N_f = \frac{3}{d_0+1+d_2} + \frac{3}{2\ d_0+1+d_1+d_2}$ *is referred to as the normalization factor.*

Using (10) and (11) computations yield

$$\Phi_t(\bar{s}', s'^{(1)})\ P_H(s'^{(1)}|s, r)\ N_f = 0,$$

$$\Phi_t(\bar{s}', s'^{(i)})\ P_H(s'^{(i)}|s, r)\ N_f = \frac{\hat{f}_1}{d_0+1+d_2} \ \ (i = 2, 3),$$

$$\Phi_t(\bar{s}', s'^{(i)})\ P_H(s'^{(i)}|s, r)\ N_f = \frac{\hat{f}_1\ d_2+\hat{f}_2\ (d_1+d_2-1)+\hat{f}_3\ (d_0+1)}{2\ d_0+1+d_1+d_2} \ \ (i = 4, 5),$$

$$\Phi_t(\bar{s}', s'^{(6)})\ P_H(s'^{(6)}|s, r)\ N_f = \frac{\hat{f}_1+\hat{f}_1\ d_2+\hat{f}_2\ (d_1+d_2-1)+\hat{f}_3\ (d_0+1)}{2\ d_0+1+d_1+d_2}$$

for sufficiently large l.

Condition (4) is equivalent to $N_f\ \ \Phi_t^{\sim} - N_f\ \ E(\Phi_t) \geq 6$. *This inequality can be written in the following representation:*

$$\frac{\hat{f}_1\ d_2+\hat{f}_1+\hat{f}_2+\hat{f}_3\ (d_0+1)}{d_0+1+d_2} + 2\ \frac{\hat{f}_1\ d_2+\hat{f}_2+\hat{f}_3\ (d_0+1)}{d_0+1+d_2} + 2\ \frac{\hat{f}_1-\hat{f}_2\ (d_1+d_2-2)}{2\ d_0+1+d_1+d_2} + \frac{-\hat{f}_2\ (d_1+d_2-2)}{2\ d_0+1+d_1+d_2}$$
$$= \frac{3\ \hat{f}_1\ d_2+\hat{f}_1+3\ \hat{f}_2+3\hat{f}_3\ (d_0+1)}{d_0+1+d_2} + \frac{2\ \hat{f}_1-3\ \hat{f}_2\ (d_1+d_2-2)}{2\ d_0+1+d_1+d_2} \geq 6.$$

If l *tends to infinity then* $\hat{f}_3 - \hat{f}_2 \geq 2$ *follows. This inequality is false since* $\hat{f}_2 > \hat{f}_3$ *is assumed.*

Until now, we do not even have found a potential function in order to proof the competitiveness of the Harmonic algorithm. So it remains an open question whether the Harmonic algorithm is competitive or not in the case of the surplus-situation. Thats why, we will introduce the new "compound Harmonic algorithm" in the following sections and prove the same bound of the competitive ratio as Bartal and Grove.

Let P_C denote the probabilities which are used by the compound Harmonic algorithm. These probabilities are, finally, derived from a surrogate problem, where at most one server must be moved in servicing the request in each step. In this way, the methods of Bartal and Grove can be applied. In order to construct surrogate problems, we will distinguish whether r is a multiple request on one point, this means $r_i > 0$ for at most one i, or not. In the second case we call r proper parallel request. (See for example Figure 1). Furthermore, we can use "blocking of servers" to compute P_C. This concept will be introduced in Section 3.2. There, we will also adapt the potential function (see (10) and (11)) for the generalized k-server problem. The probabilities P_C will be computed as multi-step transition probabilities for each multiple request on one point and as absorbing probabilities for proper parallel requests, respectively (Sections 3.3, 3.4). In Section 3.5 the competitiveness of the compound Harmonic algorithm will be proved.

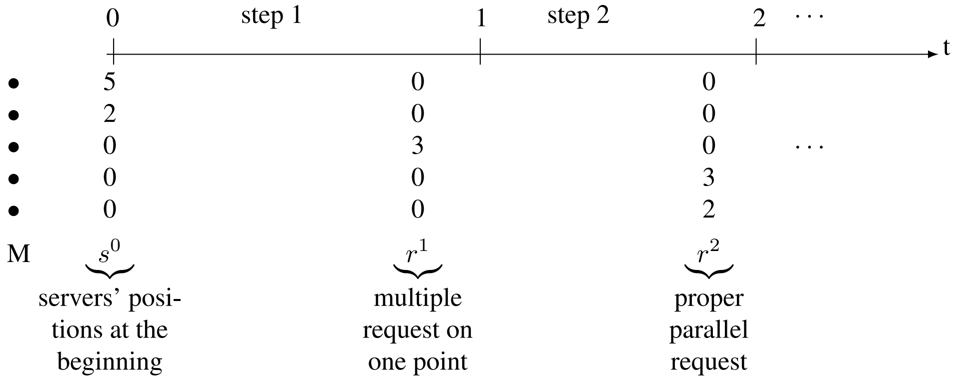

Fig. 1. The generalized k-server problem: two cases of parallel requests

3.2 Blocking of servers and a corresponding k-server problem

In this section more than one server can be located on a point. For certain considerations we will also use the same potential function as Bartal and Grove (see (10) and (11)).

(Example: *If we have three points* p_1, p_2 *and* p_3 *with the distances* $d(p_1, p_2) = 1, d(p_1, p_3) = d(p_2, p_3) = 5$ *and the online servers' positions are given by* $s = (3, 0, 0)$*, the offline servers' positions by* $\bar{s} = (0, 2, 1)$ *then* $\Phi(s, \bar{s}) = \hat{f}_1 + \hat{f}_1 + (\hat{f}_1 + 0 * \hat{f}_2 + 5 * \hat{f}_3)$ *if* $\hat{f}_3 < \frac{4}{5}\hat{f}_1$.)

It is simple to prove the following property of this potential function:

Lemma 1. *If* s_i (> 0) *online severs and* $\bar{s}_i$ (> 0) *offline severs are located on point* i *then the number of* $min\{s_i, \bar{s}_i\}$ *online servers on point* i *are assigned to this number of offline severs on point* i *for the computation of the potential function* $\Phi_t(s, \bar{s})$ *by means of the minimum weight matching (see (10) and (11)).*

Blocking of servers: Besides creating the request the adaptive online adversary can additionally block the same number of online and offline servers on points in a step, which are then not to be used in order to serve the request in this step.

Corresponding potential functions must be independent of blocking. Otherwise $\Phi_t \neq \Phi_{(t+1)-1}$ in general, where Φ_t is the potential function at the end of step t and $\Phi_{(t+1)-1}$ is the potential function at the beginning of step $t + 1$, where some servers could be blocked.

As basics for further considerations we introduce the following more specific k-server problem. The term "k-server problem with blocking" describes a k-server problem with the following three properties:

(i) More than one server can be located on a point.
(ii) The problem possesses the possibility of blocking of servers by the adversary.
(iii) The problem only contains request where at most one server must be moved in servicing the request in each step. In more detail, a request r with one $r_i > 0$ and $r_i = b_i + 1$, where b_i is the number of blocked online and offline servers, is allowed.

This also means that the probabilities which are implied by the Harmonic algorithm are different to those for models without blocking:

$$P_{H_{b(i)}}(s'|s,r) = \frac{1/d(l_0,i)}{\sum\limits_{l:s_l>b_l} 1/d(l,i)}, \text{ where } s' : s'_l = \begin{cases} s_l - 1 \text{ if } l = l_0 \\ s_l \text{ otherwise} \end{cases}.$$

A statement which is analogous with Lemma 1 in (Bartal and Grove, 2000) is also valid:

Lemma 2. *If there exists a potential function* Φ*, satisfying the properties (2), (3) and (4) with respect to some randomized online algorithm for the corresponding k-server problem with blocking then this algorithm is C(k)-competitive.*

The proof is analogous to the proof in (Bartal and Grove, 2000). Merely the random choices by the algorithm must also satisfy the conditions of blocking. [1]

Now, we consider k-server problems with blocking only on the point where the current request is placed by the adversary. Briefly speaking, k-server problems with blocking on the request point.

Lemma 3. *The Harmonic k-server algorithm applied to the k-server problem with blocking on the request point is* $((k+1)(2^k-1)-k)$*-competitive against an adaptive online adversary.*

The proof is similar to the proof in (Bartal and Grove, 2000). We use a potential function as above. If b_i online and offline servers are blocked on a point, then these online and offline severs are assigned to each other for the computation of the potential function by means of the minimum weight matching, which corresponds to Lemma 1. The values of the scaling function $\hat{f}$ and $C(k)$ can be computed analogous to the proof by Bartal and Grove.

Remark 1. The Harmonic k-server algorithm applied to the k-server problem with blocking, but not necessary on the request point, is also competitive against an adaptive online adversary (for the proof see Appendix C). However, the bound of the competitive ratio, proved in Lemma 3, cannot be shown for such a k-server problem.

3.3 The generalized k-server problem where a multiple request on one point is allowed in each step

In the case of such problems the probabilities P_C, which are used by the compound Harmonic algorithm, and the proof of its competitiveness are derived from a surrogate problem. That is a k-server problem which can be described by means of blocking on the request point. In order to construct this problem we replace each step of the original problem by a number of steps in the surrogate problem.

In more detail, let s be the online servers' positions at the beginning of a step t and let r with $k > r_i > min\{1,\ s_i\}$ and $r_j = 0$ for $j \neq i$ be a multiple request on point i in the t-th step. Then we replace the t-th step of the generalized k-server problem by

[1] Also in case of parallel requests such a lemma would be valid. However in order to use the lemma, a corresponding potential function must be found.

$r_i - s_i =: \bar{j}$ steps $t_1, t_2, \cdots, t_{\bar{j}}$ of a corresponding k-server problem with blocking on the request point i. More detailed, that means the request

$$r'^j \text{ of the } t_j \text{-th step is } s_i + j \text{ on point } i,\ j \in \{1, 2, \cdots, \bar{j}\} \text{ and } s_i + j - 1 \text{ online and offline servers on point } i \text{ are blocked in this step.} \tag{12}$$

See for example Figure 2.

If s' denotes the online servers' positions at the end of step t in case of such a generalized k-server problem then several sequences $(s'^0, s'^1, \cdots, s'^{\bar{j}})$ with $s'^j \in \hat{A}_{N;k}(s'^{j-1}, r'^j)$ for $j = 1, 2, \cdots, \bar{j}$ exist in general, where $s'^0 = s, s'^{\bar{j}} = s'$ and s'^j denotes the online servers' positions at the end of step t_j.

Then the probabilities $P_C(s'|s,r)$ which will be defined for a multiple request on one point are $\bar{j}$-step transition probabilities. In more detail,

$$P_C(s'|s,r) :=$$

$$\sum_{\{(s'^1, s'^2, \cdots, s'^{\bar{j}-1}, s')\}} P_{H_{b(i)}}(s'^1|s, r'^1) \cdot P_{H_{b(i)}}(s'^2|s'^1, r'^2) \cdot \cdots \cdot P_{H_{b(i)}}(s'|s'^{\bar{j}-1}, r'^{\bar{j}}), \tag{13}$$

where $P_{H_{b(i)}}(s'^j|s'^{j-1}, r'^j)$ $(j = 1, \cdots, \bar{j}, s'^0 = s, s'^j = s')$ are be computed according to the Harmonic algorithm with the blocked servers in mind:
If $s'^j_{l_0} = s'^{j-1}_{l_0} - 1$ $(l_0 \neq i)$ then

$$P_{H_{b(i)}}(s'^j|s'^{j-1}, r'^j) = \frac{1/d(l_0,i)}{\sum_{l:s'^{j-1}_l > 0} 1/d(l,i)}. \tag{14}$$

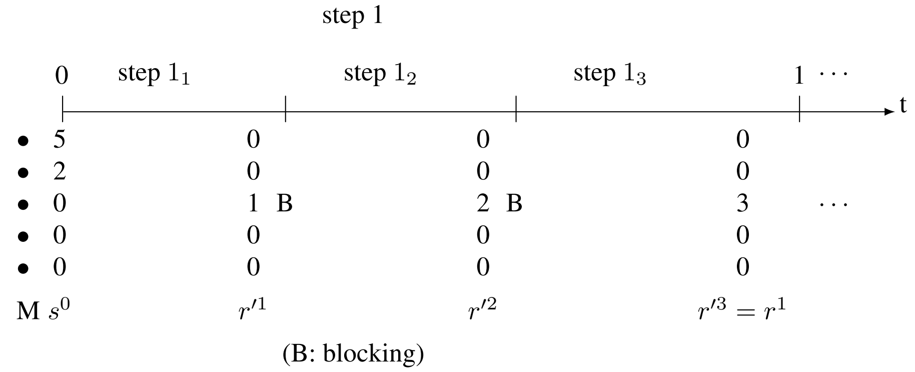

Fig. 2. Steps of the surrogate problem, which replace step 1 in Figure 1.

Lemma 4. *The compound Harmonic algorithm applied to the generalized k-server problem, where a multiple request on one point is allowed in each step, is $((k+1)(2^k - 1) - k)$-competitive against an adaptive online adversary.*

PROOF. In order to prove the statement, we consider on the one hand the given generalized k-server problem, where a multiple request on one point is allowed in each step and on the other hand the surrogate problem with blocking on the request point (as above introduced).

Without loss of generality, we can assume that the adversary moves the same servers in the surrogate problem (sp) (in $\bar{j}$ steps) as in the original problem (op) (in one step). Then the expected values $E_{\text{op}}[\text{cost}(\text{adversary})(\sigma)]$ and $E_{\text{sp}}[\text{cost}(\text{adversary})(\sigma')]$ are equal, where the request sequence σ' of the surrogate problem is constructed according to (12).

$E_{\text{op}}[\text{cost}(\text{compound Harmonic algorithm})(\sigma)]$ and
$E_{\text{sp}}[\text{cost}(\text{compound Harmonic algorithm})(\sigma')]$ are also equal because of (13).
Then the statement follows by means of Lemma 3. ■

3.4 The generalized k-server problems with proper parallel requests

The probabilities P_C, which are used by the compound Harmonic algorithm, and the proof of Theorem 1 are again derived from a surrogate problem, which is in this case a k-server problem where a multiple request on one point is allowed in each step. Similar as above, each step of the original problem will be replaced by a number of steps in the surrogate problem.

In more detail, let s denote the online servers' positions at the beginning of a step t and let r be a request in the t-th step with (w.l.o.g.)

$$\begin{cases} r_i > 0 \ \text{ for } \ i = 1, \cdots, \bar{N} \\ r_i = 0 \ \text{ for } \ i = \bar{N}+1, \bar{N}+2, \cdots, N \end{cases}, \ 2 \leq \bar{N} \leq N \text{ and } \sum_{i=1}^{N} r_i < k.$$

Then we replace the t-th step of the generalized k-server problem with proper parallel requests by a number of steps $t_1, t_2, \cdots$ of a corresponding k-server problem where a multiple request on one point is allowed. In more detail, the following request sequence $(\bar{r}^j)_{j=1,2,\cdots}$ for the surrogate steps should be created:

$$\bar{r}_i^j = \begin{cases} r_i \text{ if } j \equiv i \ mod(\bar{N}) \\ 0 \text{ otherwise} \end{cases}. \tag{15}$$

See for example Figure 3.

If s' denotes the online servers' positions at the end of step t in case of the generalized k-server problem then several sequences $(s'^0, s'^1, s'^2, \cdots, s'^{\bar{j}})$ with several length $\bar{j}$, $s'^j \in \hat{A}_{N;k}(s'^{j-1}, \bar{r}^j)$ for $j = 1, 2, \cdots, \bar{j}$ exist, where $s'^0 = s, s'^{\bar{j}} = s'$ and s'^j denotes the online servers' positions at the end of step t_j. If $s_i'^j \geq \bar{r}_i^{j+1} > 0$ then the corresponding surrogate step could be also omitted.

Such sequences represent realizations of a time-homogeneous Markov chain with transient states $(s'^{j-1}, \bar{r}^j)$, absorbing states $s' \in \hat{A}_{N;k}(s, r)$ and transition probabilities $P_C(s'^j, \bar{r}^{j+1} | s'^{j-1}, \bar{r}^j) := P_C(s'^j | s'^{j-1}, \bar{r}^j)$.

The probabilities $P_C(s'|s, r)$, $s' \in \hat{A}_{N;k}(s, r)$ for proper parallel requests, which are used by the compound Harmonic algorithm, are defined as absorbing probabilities. Absorbing probabilities can be computed by means of linear systems, see e.g. (Langrock and Jahn, Theorem 6.6 and the following Example 3). For this purpose all states of the above mentioned Markov chains must be known and the corresponding transition probabilities are the coefficients of these linear systems. The number of these states is finite. Furthermore

$$\sum_{s' \in \hat{A}_{N;k}(s,r)} P_C(s'|s,r) = 1 \tag{16}$$

is valid and the solutions of the linear systems are unique.

Clearly, if $\bar{N} = 1$ then the special case of a multiple request on one point is given in the current step.

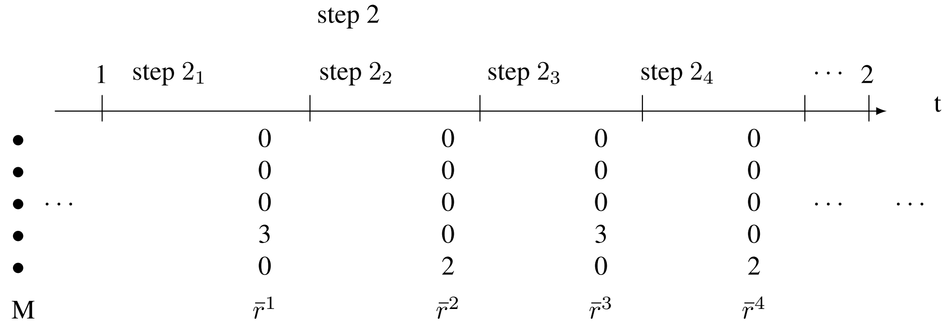

Fig. 3. Steps of the surrogate problem, which replace step 2 in Figure 1.

3.5 The compound Harmonic algorithm applied to the generalized k-server problem

In general, the compound Harmonic algorithm uses more-step transition and absorbing probabilities P_C from Sections 3.3 and 3.4 instead of the probabilities P_H (see (7)).

For the computation of the absorbing probabilities transient and absorbing states of the corresponding Markov chain must be found (see Langrock and Jahn, Section 6). This can be done according to following Algorithm 1.

In Example 3 we compute probabilities P_C, used by the compound Harmonic algorithm, and compare they with probabilities P_H of the Harmonic algorithm.

Example 3 *Let $k = 4$ and let the metric space M consist of 6 points $p_1, p_2, \cdots, p_6$ of the two-dimensional Euclidean space with the distances*
$d(p_3, p_1) = 5, d(p_4, p_1) = 3,85, d(p_5, p_1) = 1,6, d(p_6, p_1) = 4,5, d(p_2, p_1) = 2,4$ *and* $d(p_3, p_2) = 4, d(p_4, p_2) = 5, d(p_5, p_2) = 2,1, d(p_6, p_2) = 4,55.$

The current online servers' positions are given by $s = (0,0,1,1,1,1)^T$ and the current (proper parallel) requests by $r = (1,1,0,0,0,0)^T$.

Then we have 6 feasible online servers' positions with respect to s and r:

$s'^{(1)} = (1,1,0,0,1,1)^T, s'^{(2)} = (1,1,0,1,0,1)^T, s'^{(3)} = (1,1,0,1,1,0)^T,$
$s'^{(4)} = (1,1,1,0,0,1)^T, s'^{(5)} = (1,1,1,0,1,0)^T, s'^{(6)} = (1,1,1,1,0,0)^T.$

Corresponding distances $\hat{d}(s, s'^{(i)})$, probabilities $P_H(s'^{(i)}|s,r)$ (according to the Harmonic algorithm) and $P_C(s'^{(i)}|s,r)$ (according to the compound Harmonic algorithm) can be found in Table 1.

Table 1.

i	1	2	3	4	5	6
$\hat{d}(s, s'^{(i)})$	7,85	5,60	8,50	2,95	8,40	6,15
$P_H(s'^{(i)}\|s, r)$	0,1459	0,2045	0,1347	0,1924	0,1363	0,1862
$P_C(s'^{(i)}\|s, r)$	0,0836	0,2504	0,0781	0,2582	0,0829	0,2466

$P_H(s'^{(i)}|s, r)$ can be calculated according to (7) and $\hat{d}(s, s'^{(i)})$ by means of the classical transportation problem, see Section 2. For the computation of $P_C(s'^{(i)}|s, r)$ see Appendix D. We can observe that $P_C(s'^{(i)}|s, r) < P_H(s'^{(i)}|s, r)$ for greater distances $\hat{d}(s, s'^{(i)})$ and $P_C(s'^{(i)}|s, r) > P_H(s'^{(i)}|s, r)$ for smaller $\hat{d}(s, s'^{(i)})$.

Theorem 1. *The compound Harmonic algorithm applied to the generalized k-server problems with parallel requests is $((k + 1)(2^k - 1) - k)$-competitive against an adaptive online adversary in the case of the surplus-situation.*

PROOF. In order to prove the statement, we consider on the one hand the given generalized k-server problem with parallel requests and on the other hand the surrogate problem where a multiple request on one point is allowed in each step (as above introduced).

Without loss of generality, we can assume that the adversary moves the same servers in the surrogate problem (sp), in the first $\bar{N}$ steps, as in the original problem (op) in one step. Then the expected values $E_{\text{op}}[\text{cost}(\text{adversary})(\sigma)]$ and $E_{\text{sp}}[\text{cost}(\text{adversary})(\sigma')]$ are equal, where the request sequence σ' of the surrogate problem is constructed according to (15). We can use such surrogate sequences since (16) is valid.

By means of the triangle-inequality
$E_{\text{op}}[\text{cost}(\text{compound Harmonic algorithm})(\sigma)] \leq$
$E_{\text{sp}}[\text{cost}(\text{compound Harmonic algorithm})(\sigma')]$ follows in general.

Then the application of Lemma 4 leads to the bound of the competitive ratio. ■

Corollary 1. *The compound Harmonic algorithm applied to the generalized k-server problems with parallel requests $r^1, r^2, \cdots, r^n$, where $r_i^j \leq 1$ for any i, j and $\sum_{i=1}^{N} r_i^j \leq k$ for any j, have the same competitive ratio as the Harmonic algorithm applied to the (usual) k-server problems against an adaptive online adversary.*

PROOF. Because of $r_i^j \leq 1$ for any i and j, sequences σ' present request sequences for usual k-server problems and

$$E_{\text{sp}}[\text{cost}(\text{compound Harmonic algorithm})(\sigma')]$$
$$= E_{\text{sp}}[\text{cost}(\text{Harmonic algorithm})(\sigma')].$$ ■

Remark 2. (i) In the case of unit distances, that means $d(i, j) = 1 \ \forall \ i \neq j$, all probabilities $P_C(s'|s, r)$ are the same for $s' \in \hat{A}_{N;k}(s, r)$. Hence, $P_C(s'|s, r) = P_H(s'|s, r)$ and the compound Harmonic algorithm and the Harmonic algorithm are identical.

Algorithm 1: Computation of absorbing probabilities, step t

```
Data:
s: the (online) servers' positions at the beginning of the t-th step
$r$: the request in the t-th step with (w.l.o.g.)
    $\begin{cases} r_i > 0 \ \text{ for } \ i = 1, \cdots, \bar{N} \\ r_i = 0 \ \text{ for } \ i = \bar{N}+1, \bar{N}+2, \cdots, N \end{cases}$, $2 \leq \bar{N} \leq N$ and $\sum\limits_{i=1}^{N} r_i < k$.
;       // Computation of the set of transient states $\hat{S}^T$, the set of
  absorbing states $\hat{S}$ and of corresponding transition probabilities: 1
  - 16
1  $\bar{r}_i^j := \begin{cases} r_i \text{ if } j \equiv i \; mod(\bar{N}) \\ 0 \text{ otherwise} \end{cases}$, $j = 1, 2, \cdots, \ i = 1, 2, \cdots N$;
2  $\hat{S}^{T1} := \emptyset, \hat{S}^{T2} := \emptyset, \cdots$;
3  if $s_1 \geq r_1, \cdots, s_{j-1} \geq r_{j-1}$ and $s_j < r_j$ then $\hat{S}^T := \hat{S}^{Tj} := \{s\}$;
4  for $j = 1, 2, \cdots$ do
5      if $\hat{S}^{Tj} \neq \emptyset$ then
6          find all $\tilde{s} \in \{\hat{A}_{N;k}(s'', \bar{r}^j) | s'' \in \hat{S}^{Tj}\}$;
7      for $\tilde{s} \in \{\hat{A}_{N;k}(s'', \bar{r}^j) | s'' \in \hat{S}^{Tj}\}$ do
8          if $\tilde{s}_{j+1} \geq r_{j+1}, \cdots, \tilde{s}_{j+l-1} \geq r_{j+l-1}$ and $\tilde{s}_{j+l} < r_{j+l}$ then
9              $\hat{S}^{Tj+l} := \hat{S}^{Tj+l} \cup \{\tilde{s}\}$;
10             $\hat{S}^T := \hat{S}^T \cup \{(\tilde{s}, \bar{r}^{j+l})\}$;
11             compute $P_C(\tilde{s}, \bar{r}^{j+l} | s'', \bar{r}^j) := P_C(\tilde{s} | s'', \bar{r}^j)$ according to (13);
12         if $\tilde{s} \in \hat{A}_{N;k}(s, r)$ then
13             $\hat{S} := \hat{S} \cup \{\tilde{s}\}$;
14             compute $P_C(\tilde{s} | s'', \bar{r}^j)$ according to (13);
15     if no new transient or absorbing state can be found then
16         Stop;
17 Compute $P_C(s' | s, r)$ for $s' \in \hat{A}_{N;k}(s, r)$ as absorbing probabilities by means of a linear
   systems, see e.g. (Langrock and Jahn, Theorem 6.6);
```

(ii) For k-server problems with proper parallel requests we could also introduce another compound Harmonic algorithm, where $\sum_{i=1}^{\bar{N}}(r_i - s_i)$-step transition probabilities would be used instead of the absorbing probabilities, and where several servers on several points must be blocked. Then the computation of such probabilities would be more simple. Remark 1 would also imply the competitiveness of such an algorithm, however with a weaker bound of the competitive ratio.

(iii) Another method to dealing with a multiple request on one point can be found in (N. Bansal *et al.*), for example. There, such a point is replaced by k points at epsilon distance. However, the complexity for the computation of absorbing probabilities would then be very large.

4 Conclusion

In this paper we have considered a generalized k-server problem with parallel requests. In the case of the surplus-situation it is difficult to answer the question whether a corresponding Harmonic algorithm is competitive or not against an adaptive online adversary. At least we have given an example that the potential function which was introduced by Bartal and Grove is not helpful to prove competitiveness. We have constructed the compound Harmonic algorithm which is also a memoryless algorithm. We were able to prove the same bound of the competitive ratio as for the Harmonic algorithm applied to the (usual) k-server problem, see (Bartal and Grove, 2000). However, multi-step transition probabilities and absorbing probabilities must be computed by the compound Harmonic algorithm. In the case of unit distances the Harmonic algorithm and the compound Harmonic algorithm are identical.

References

Bansal, N., Buchbinder N., Naor J. (2010). Towards the randomized k-server conjecture: A primal-dual approach. In: *Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 40-55.

Bartal, Y., Grove, E. (2000). The Harmonic k-Server Algorithm Is Competitive. *Journal of the ACM*, 47 (1), 1-15.

Borodin A., El-Yaniv, R. (1998) *Online computation and competitive analysis*. University Press, Cambrigde.

Hildenbrandt, R. (1995). *Methoden aus ganzzahliger Optimierung und Verbandstheorie zur Behandlung eines stochastischen dynamischen Transportproblems*. Habilitationsschrift, TU Ilmenau, (Libri BoD 2000).

Hildenbrandt, R. (2014). A k-server problem with parallel requests and unit distances. *Information Processing Letters* 114(5), 239-246.

Koutsoupias, E., Papadimitriou, C. (1995). On the k-server conjecture. *Journal of the ACM*, 42(5), 971-983.

Langrock, P., Jahn, W. (1979) *Einführung in die Theorie der Markovschen Ketten und ihre Anwendungen*. Teubner, Leipzig.

Manasse, M.S., McGeoch L.A., Sleator, D.D. (1988). Competitive algorithms for on-line problems. In: *Proceeding of the 20th Annual ACM Symposium on Theory of Computing*, pp. 322-333 (Journal version).

Raghavan P., Snir, M. (1989). Memory versus randomization in on-line algorithms. In: *Proceeding of the 16th ICALB*, Vol. 372 of Lecture Notes in Computer Sciences, pp. 687 - 703.

Appendices

A Proof of (9) concerning Example 1

According to the Harmonic algorithm

$$h_\delta^1 = \frac{1}{\delta+1} \cdot \delta + \frac{\delta}{\delta+1} \cdot 1 = 2 \cdot \frac{\delta}{\delta+1} \quad \text{for } \delta = 1, 2, ... \tag{A.1}$$

$$\text{and} \qquad a_1^1 = 1, \; a_\delta^1 = 0 \;\; \text{for } \delta = 2, 3, ... \tag{A.2}$$

follow. Using (A.1) and (A.2) recursion computations lead to

$$h_\delta^{l+1} = \tfrac{1}{\delta+1}(\delta + h_1^l) + \tfrac{\delta}{\delta+1}(1 + h_{\delta+1}^l) = \tfrac{1}{\delta+1}(h_1^l + \delta h_{\delta+1}^l) + h_\delta^1 \tag{A.3}$$

$$a_\delta^{l+1} = \qquad \tfrac{1}{\delta+1}(a_1^l + \delta a_{\delta+1}^l) + a_\delta^1 \tag{A.4}$$

for $\delta = 1, 2, ...$

At first we consider more general sequences $(g_\delta^l)_{l=1,2,...}$ with
$g_\delta^{l+1} = \frac{1}{\delta+1}(g_1^l + \delta g_{\delta+1}^l) + g_\delta^1$ and any given g_δ^1 for $\delta = 1, 2, ...$
We will show by mathematical induction that

$$\begin{aligned}
g_\delta^l = {} & g_1^1 \left[\tfrac{1}{\delta+1} a_1^{l-1} + \tfrac{\delta}{(\delta+1)(\delta+2)} \cdot a_1^{l-2} + \tfrac{\delta}{(\delta+2)(\delta+3)} \cdot a_1^{l-3} + \cdots + \tfrac{\delta}{(\delta+l-2)(\delta+l-1)} a_1^1 \right] \\
& + \; g_2^1 \cdot \tfrac{1}{2} \left[\tfrac{1}{\delta+1} a_1^{l-2} + \tfrac{\delta}{(\delta+1)(\delta+2)} a_1^{l-3} + \tfrac{\delta}{(\delta+2)(\delta+3)} a_1^{l-4} + \cdots + \tfrac{\delta}{(\delta+l-3)(\delta+l-2)} a_1^1 \right] \\
& + \; g_3^1 \cdot \tfrac{1}{3} \left[\tfrac{1}{\delta+1} a_1^{l-3} + \tfrac{\delta}{(\delta+1)(\delta+2)} a_1^{l-4} + \cdots + \tfrac{\delta}{(\delta+l-4)(\delta+l-3)} a_1^1 \right] \\
& + \\
& \vdots \\
& + \; g_{l-2}^1 \cdot \tfrac{1}{l-2} \left[\tfrac{1}{\delta+1} a_1^2 + \tfrac{\delta}{(\delta+1)(\delta+2)} a_1^1 \right] \\
& + \; g_{l-1}^1 \cdot \tfrac{1}{l-1} \; \tfrac{1}{\delta+1} \cdot a_1^1 \\
& + \; g_\delta^1 \\
& + \; \tfrac{\delta}{\delta+1} g_{\delta+1}^1 + \tfrac{\delta}{\delta+2} g_{\delta+2}^1 + \cdots + \tfrac{\delta}{\delta+l-2} g_{\delta+l-2}^1 + \tfrac{\delta}{\delta+l-1} g_{\delta+l-1}^1 .
\end{aligned} \tag{A.5}$$

Using (A.1) and (A.2) formula (A.5) implies the specific equations

$$a_1^l = \tfrac{1}{2} a_1^{l-1} + \tfrac{1}{2 \cdot 3} \cdot a_1^{l-2} + \tfrac{1}{3 \cdot 4} \cdot a_1^{l-3} + \cdots + \tfrac{1}{(l-1)l} a_1^1 + 1 \tag{A.6}$$

and

$$\begin{aligned}
h_1^l = {} & \tfrac{1}{2}a_1^{l-1} + \tfrac{1}{2\cdot 3}\cdot a_1^{l-2} + \tfrac{1}{3\cdot 4}\cdot a_1^{l-3} + \cdots + \tfrac{1}{(l-1)l}a_1^1 + h_1^1 \\
& + h_2^1 \cdot \tfrac{1}{2}\left[\tfrac{1}{2}a_1^{l-2} + \tfrac{1}{2\cdot 3}a_1^{l-3} + \tfrac{1}{3\cdot 4}a_1^{l-4} + \cdots + \tfrac{1}{(l-2)(l-1)}a_1^1\right] + \tfrac{1}{2}h_2^1 \\
& + \\
& \vdots \\
& + h_{l-2}^1 \cdot \tfrac{1}{l-2}\left[\tfrac{1}{2}a_1^2 + \tfrac{1}{2\cdot 3}a_1^1\right] + h_{l-2}^1 \cdot \tfrac{1}{l-2} \\
& + h_{l-1}^1 \cdot \tfrac{1}{l-1}\ \tfrac{1}{2}\cdot a_1^1 + h_{l-1}^1 \cdot \tfrac{1}{l-1} \\
& + \tfrac{1}{l}h_l^1 \\
& = a_1^l + 2\cdot\tfrac{2}{2\cdot 3}\cdot a_1^{l-1} + 2\cdot\tfrac{3}{3\cdot 4}\cdot a_1^{l-2} + \cdots + 2\cdot\tfrac{l-1}{(l-1)l}a_1^2 + 2\cdot\tfrac{l}{l(l+1)}a_1^1.
\end{aligned}$$

Thus

$$h_1^l = a_1^l + 2\cdot\tfrac{1}{3}\cdot a_1^{l-1} + 2\cdot\tfrac{1}{4}\cdot a_1^{l-2} + \cdots + 2\cdot\tfrac{1}{l}a_1^2 + 2\cdot\tfrac{1}{l+1}a_1^1. \tag{A.7}$$

Proof of (A.5) by mathematical induction on l:
Induction basic: $g_\delta^2 = \frac{1}{\delta+1}(g_1^1 + \delta g_{\delta+1}^1) + g_\delta^1 = \frac{1}{\delta+1}g_1^1 + g_\delta^1 + \frac{\delta}{\delta+1}g_{\delta+1}^1$ corresponds to (A.5) for $l = 2$.

Induction step: If we replace g_1^l and $g_{\delta+1}^l$ in $g_\delta^{l+1} = \frac{1}{\delta+1}g_1^l + g_\delta^1 + \frac{\delta}{\delta+1}g_{\delta+1}^l$ by means of (A.5) then the following equation follows

$$\begin{aligned}
g_\delta^{l+1} = {} & \tfrac{1}{\delta+1}g_1^1\left[\tfrac{1}{2}a_1^{l-1} + \tfrac{1}{2}\cdot\tfrac{1}{3}a_1^{l-2} + \cdots + \tfrac{1}{l-1}\tfrac{1}{l}a_1^1\right] \\
& + \tfrac{1}{\delta+1}g_2^1\tfrac{1}{2}\left[\tfrac{1}{2}a_1^{l-2} + \tfrac{1}{2}\cdot\tfrac{1}{3}a_1^{l-3} + \cdots + \tfrac{1}{l-2}\tfrac{1}{l-1}a_1^1\right] \\
& + \\
& \vdots \\
& + \tfrac{1}{\delta+1}g_{l-2}^1\tfrac{1}{l-2}\left[\tfrac{1}{2}a_1^2 + \tfrac{1}{2\cdot 3}a_1^1\right] \\
& + \tfrac{1}{\delta+1}g_{l-1}^1\cdot\tfrac{1}{2}\cdot a_1^1 \\
& + \tfrac{1}{\delta+1}\left[g_1^1 + \tfrac{1}{2}g_2^1 + \tfrac{1}{3}g_3^1 + \cdots + \tfrac{1}{l}g_l^1\right] \qquad\qquad + g_\delta^1 \\
& + \tfrac{\delta}{\delta+1}g_1^1\left[\tfrac{1}{\delta+2}a_1^{l-1} + \tfrac{\delta+1}{(\delta+2)(\delta+3)}a_1^{l-2} + \cdots + \tfrac{\delta+1}{(\delta+l-1)(\delta+l)}a_1^1\right] \\
& + \tfrac{\delta}{\delta+1}g_2^1\cdot\tfrac{1}{2}\left[\tfrac{1}{\delta+2}a_1^{l-2} + \tfrac{\delta+1}{(\delta+2)(\delta+3)}a_1^{l-3} + \cdots + \tfrac{\delta+1}{(\delta+l-2)(\delta+l-1)}a_1^1\right] \\
& + \\
& \vdots \\
& + \tfrac{\delta}{\delta+1}\tfrac{1}{g_{l-2}^1}\tfrac{1}{l-2}\left[\tfrac{1}{\delta+2}a_1^2 + \tfrac{\delta+1}{(\delta+2)(\delta+3)}a_1^1\right]
\end{aligned}$$

$+\frac{\delta}{\delta+1} g^1_{l-1} \frac{1}{l-1} \cdot \frac{1}{\delta+2} \cdot a^1_1$

$+\frac{\delta}{\delta+1} g^1_{\delta+1}$

$+\frac{\delta}{\delta+1} (\frac{\delta+1}{\delta+2} g^1_{\delta+2} + \frac{\delta+1}{\delta+3} g^1_{\delta+3} + \cdots + \frac{\delta+1}{\delta+l} g^1_{\delta+l}).$

If we reorganize the sum and use such partial sums as
$(\frac{1}{\delta+1} g^1_1 \left[\frac{1}{2} a^{l-1}_1 + \frac{1}{2} \cdot \frac{1}{3} a^{l-2}_1 + \cdots + \frac{1}{l-1}\frac{1}{l} a^1_1\right]) + (\frac{1}{\delta+1} g^1_1) = \frac{1}{\delta+1} g^1_1 a^l_1,$

$(\frac{1}{\delta+1} g^1_2 \frac{1}{2} \left[\frac{1}{2} a^{l-2}_1 + \frac{1}{2} \cdot \frac{1}{3} a^{l-3}_1 + \cdots + \frac{1}{l-2}\frac{1}{l-1} a^1_1\right]) + (\frac{1}{\delta+1}\frac{1}{2} g^1_2) = \frac{1}{\delta+1} g^1_2 \frac{1}{2} a^{l-1}_1$
(and so on) then it follows that (A.5) is valid for $l + 1$.

Now, we will show the following properties of the sequence $(a^l)_{l=1,2,\ldots}$
(where $a^l := a^l_1$ for $l = 1, 2, \cdots$):

$$(i) \qquad (a^l)_{l=1,2,\ldots} \text{ is strictly increasing.} \tag{A.8}$$

This property can be proved by a simple mathematical induction.
Clearly $a^1 = 1 < a^2 = \frac{3}{2}$ according to (A.6).
$a^l = \frac{1}{2} a^{l-1} + \frac{1}{2} \cdot \frac{1}{3} a^{l-2} + \cdots + \frac{1}{l-1}\frac{1}{l} a^1 + 1 <$
$a^{l+1} = \frac{1}{2} a^l + \frac{1}{2} \cdot \frac{1}{3} a^{l-1} + \cdots + \frac{1}{l-1}\frac{1}{l} a^2 + \frac{1}{l}\frac{1}{l+1} a^1 + 1$
follows from $a^{l-1} < a^l, \cdots, a^2 < a^3, a^1 < a^2$.

$$(ii) \qquad \text{The sequence } (a^l - a^{l-1})_{l=1,2,\ldots} \text{ is bounded.} \tag{A.9}$$

Firstly $a^{l+1} = \frac{1}{2} a^l + \frac{1}{2} \cdot \frac{1}{3} a^{l-1} + \frac{1}{3} \cdot \frac{1}{4} a^{l-2} + \cdots + \frac{1}{l-1}\frac{1}{l} a^2 + \frac{1}{l}\frac{1}{l+1} a^1 + 1$
$= \frac{1}{2} a^l + \frac{1}{2} a^{l-1} - \frac{1}{3}(a^{l-1} - a^{l-2}) - \cdots - \frac{1}{l}(a^2 - a^1) - \frac{1}{l+1} a^1 + 1.$
Since $a^{l+1} > a^l$ it is necessary that $\frac{1}{3}(a^{l-1} - a^{l-2}) < 1$ and thus
$a^{l-1} - a^{l-2} < 3$.
For $l + 1 = 3, 4, 5, \cdots$ we get $a^2 - a^1 < 3,\ a^3 - a^2 < 3,\ a^4 - a^3 < 3, \cdots$.

$$(iii) \qquad \lim_{l\to\infty} \frac{a^l}{a^{l-1}} = 1 \tag{A.10}$$

follows from (A.8) and (A.9) in both cases that $\lim_{l\to\infty} a^l$ exists or that
$\lim_{l\to\infty} a^l = \infty$.

Finally we will show that $\lim_{l\to\infty} \frac{h^l}{a^l} = \infty$.
According to (A.7) $h^{l+1} = a^{l+1} + \frac{2}{3} a^l + \frac{2}{4} a^{l-1} + \cdots + \frac{2}{l+2} a^1$.
Thus, $\frac{h^{l+1}}{a^{l+1}} = 1 + \frac{2}{3}\frac{a^l}{a^{l+1}} + \frac{2}{4}\frac{a^{l-1}}{a^{l+1}} + \cdots + \frac{2}{l+2}\frac{a^1}{a^{l+1}}$
where

$$\frac{a^j}{a^{l+1}} = \frac{a^j}{a^{j+1}} \cdot \frac{a^{j+1}}{a^{j+2}} \cdot \cdots \cdot \frac{a^l}{a^{l+1}} \text{ (for } j < l). \tag{A.11}$$

Let $l+1-L$ be the number of terms $\frac{a^j}{a^{l+1}} \geq \frac{1}{3}$ $(l \geq j \geq L)$. We will show that $l+1-L$ tends to infinity if l tends to infinity.

Then $\frac{h^{l+1}}{a^{l+1}} \to \infty$ follows since these quotients are related to Harmonic series.

The properties (A.8) and (A.10) imply that

$\forall\, \varepsilon > 0 \;\exists\; L(\varepsilon) :\; 1 \geq \frac{a^i}{a^{i+1}} \geq 1-\varepsilon \;\forall\, i \geq L(\varepsilon)$.

Using (A.11) we obtain that $\frac{a^j}{a^{l+1}} \geq (1-\varepsilon)^{l+1-j} \;\forall\, l \geq j \geq L(\varepsilon)$.

$\frac{a^j}{a^{l+1}} \geq \frac{a^{L(\varepsilon)}}{a^{l+1}} \geq (1-\varepsilon)^{l+1-L(\varepsilon)} \geq \frac{1}{3}$ is valid if and only if

$l+1-L(\varepsilon) \leq \frac{-ln3}{ln(1-\varepsilon)}$.

If ε tends to 0 then $\frac{-ln3}{ln(1-\varepsilon)}$ tends to infinity

and also the number of terms $\frac{a^j}{a^{l+1}} \geq \frac{1}{3}$ $(l \geq j \geq L(\varepsilon))$. ■

B Completion of Example 2

Following configurations $C_a(l), \cdots, C_i(l)$ of the online and offline servers can occur if the corresponding requests r given by the adversary and the answers by the adversary are as below:

$C_a(l)$: ON is located on $p_i, p_j,\ i,j \in \{1,2,3\},\ d_0, d_0+d_1$.
OFF is located on $p_i, p_j,\ d_0, d_0+d_1$.
r : one server on $p_q,\ q \in \{1,2,3\} \setminus \{i,j\}$,
answer by the adversary: the offline server on $\begin{cases} p_{q-1} & \text{if } q \in \{2,3\} \\ p_3 & \text{if } q = 1 \end{cases}$ is moved.

($C_a(l)$ represents the initial configuration for $l=1, i=1$ and $j=2$.)

$C_b(l)$: ON is located on $p_i, p_j,\ i,j \in \{1,2,3\},\ d_0, d_0+d_1$.
OFF is located on $p_i, p_q,\ q \in \{1,2,3\} \setminus \{i,j\},\ d_0, d_0+d_1$.
r : one server on p_q, no server is moved by the adversary.

$C_c(l)$: ON is located on $p_1, p_2, p_3,\ d_0+d_1$.
OFF is located on $p_i, p_j,\ i,j \in \{1,2,3\},\ d_0, d_0+d_1$.
r : one server on d_0+1 and one server on $d_0+d_1+d_2$,
answer by the adversary: the offline servers on d_0, d_0+d_1 are moved.

$C_d(l)$: ON is located on $p_1, p_2, p_3,\ d_0$.
OFF is located on $p_i, p_j,\ i,j \in \{1,2,3\},\ d_0, d_0+d_1$.
r : one server on d_0+d_1, no server is moved by the adversary.

$C_e(l)$: ON is located on $p_i,\ i \in \{1,2,3\},\ d_0, d_0+d_1, d_0+d_1-d_3$.
OFF is located on $p_j, p_q,\ j,q \in \{1,2,3\} \setminus \{i\},\ d_0, d_0+d_1$.
r : one server on p_j and one server on p_q, no server is moved by the adversary.

$C_f(l)$: ON is located on $p_i,\ i \in \{1,2,3\},\ d_0, d_0+d_1, d_0+d_1-d_3$.
OFF is located on $p_i, p_j,\ j \in \{1,2,3\} \setminus \{i\},\ d_0, d_0+d_1$.
r : one server on p_j, no server is moved by the adversary.

$C_g(l)$: ON: is located on $p_i, p_j,\ i,j \in \{1,2,3\},\ d_0 + d_1, d_0 + d_1 - d_3$
OFF: is located on $p_i, p_j,\ d_0, d_0 + d_1$
r : one server on d_0, no server is moved by the adversary.

$C_h(l)$: ON is located on $p_i, p_j,\ i,j \in \{1,2,3\},\ d_0, d_0 + d_1 - d_3$.
OFF is located on $p_i, p_j,\ d_0, d_0 + d_1$.
r : one server on $d_0 + d_1$, no server is moved by the adversary.

$C_i(l)$: ON is located on $p_1, p_2, p_3,\ d_0 + d_1 - d_3$.
OFF is located on $p_i, p_j,\ i,j \in \{1,2,3\},\ d_0, d_0 + d_1$.
r: one server on d_0 and one server on $d_0 + d_1$, no server is moved by the adversary.

C Proof of the statement from Remark 1

The statement, that the Harmonic k-server algorithm applied to the k-server problem with blocking is competitive against an adaptive online adversary, can be shown analogous to the proof in (Bartal and Grove, 2000). However, blocking of servers implies other inequalities for the computation of the values of the scaling function. Then another bound of the competitive ratio follows:

In *Case 1*, see (Bartal and Grove, 5. Analysis of the Step-Change in the Potential Function, p. 6 - 8) a path $P(x) = \{X_1, \cdots, X_{l(x)}\}$ is considered. In relation to k-server problems with blocking it could be that several offline servers from the set $\{X_1, \cdots, X_{l(x)-1}\}$ lie on the same point.

Let l' be the number of blocked offline servers from the set $\{X_1, \cdots, X_{l(x)-1}\}$ $(l' \leq l(x) - 1)$ and l'' be the number of the remaining blocked offline servers then $k' = k - l' - l''$ is the number of non-blocked offline (and online) servers.

By reason of blocked servers we must replace the last inequality $j\ f(j) \geq (k-j)\ f(j+1) + k$ on page 10 in (Y. Bartal and E. Grove), which corresponds to $j\ \hat{f}_j \geq (k-j)\ \hat{f}_{j+1} + k$ using our symbols, by

$(j - l')\ \hat{f}_j \geq (k' + l' - j)\ \hat{f}_{j+1} + k' = (k - l'' - j)\ \hat{f}_{j+1} + k - l' - l''$.

Since $\hat{f}_j$ is increasing if l'' is decreasing we must consider the above inequality for $l'' = 0$ in order to compute a bound of the competitive ratio.

$(j - l')\ \hat{f}_j \geq (k-j)\ \hat{f}_{j+1} + k - l'\ (l' \leq j - 1)$ is equivalent to

$\hat{f}_j \geq \frac{k-l'}{j-l'}\ (\hat{f}_{j+1} + 1) - \hat{f}_{j+1}$. If $l' = j - 1$ then $\frac{k-l'}{j-l'} = k - j + 1$ is the largest possible factor.

We set $\hat{f}_k = 1$ and then the other values of the scaling function can be successively calculated by means of the inequalities $\hat{f}_j \geq (k-j)(\hat{f}_{j+1} + 1) + 1, j = k-1, \cdots, 1$. (Obviously, the values are greater than those in (Bartal and Grove).) Finally, $C(k) = k\ \hat{f}_1 + (k-1)\ \hat{f}_2$ follows as in (Bartal and Grove, p. 9 and 10) by means of property (3).

D Computations relating to Example 3

If we want to compute absorbing probabilities $P_C(s'^{(i)}|s, r)$ we need for the corresponding Markov chains besides the states $s =: s^{(0)}$ and the absorbing states $s'^{(1)}, s'^{(2)}, \cdots, s'^{(6)}$ also the states

$s^{(1)} = (1,0,1,1,1,0)^T, s^{(2)} = (1,0,1,1,0,1)^T, s^{(3)} = (1,0,1,0,1,1)^T,$
$s^{(4)} = (1,0,0,1,1,1)^T, s^{(5)} = (0,1,1,1,1,0)^T, s^{(6)} = (0,1,1,1,0,1)^T,$
$s^{(7)} = (0,1,1,0,1,1)^T, s^{(8)} = (0,1,0,1,1,1)^T$, which are transient states. [2]

For example $(s^{(0)}, s^{(2)}, s^{(6)}, s^{(2)}, s^{(6)}, s'^{(4)})$ is a realization of the time-homogeneous Markov chain with the absorbing state $s'^{(4)}$, where the corresponding surrogate request sequence $(\bar{r}^j)_{j=1,2,\cdots}$ is
$((1,0,0,0,0,0)^T, (0,1,0,0,0,0)^T, (1,0,0,0,0,0)^T, (0,1,0,0,0,0)^T, (1,0,0,0,0,0)^T)$
as described in Section 3.4

If we want to compute the absorbing probabilities $P_C(s'^{(i)}|s,r)$ we need the (one-step) transition probabilities. At first we give the matrix of transition probabilities from the transient states $s^{(0)}, s^{(1)}, \cdots, s^{(8)}$ into these transient states:

$$B = \begin{pmatrix}
0 & \frac{1/d_{61}}{N_f} & \frac{1/d_{51}}{N_f} & \frac{1/d_{41}}{N_f} & \frac{1/d_{31}}{N_f} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \frac{1/d_{12}}{N_f^4} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \frac{1/d_{12}}{N_f^3} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1/d_{12}}{N_f^2} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1/d_{12}}{N_f^1} \\
0 & \frac{1/d_{12}}{N_f^8} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{1/d_{12}}{N_f^7} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \frac{1/d_{12}}{N_f^6} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \frac{1/d_{12}}{N_f^5} & 0 & 0 & 0 & 0
\end{pmatrix}$$

where $d_{ij} := d(p_i, p_j)$, $N_f = 1/d_{61} + 1/d_{51} + 1/d_{41} + 1/d_{31}$,

$N_f^1 = 1/d_{12} + 1/d_{62} + 1/d_{52} + 1/d_{42},\ N_f^5 = 1/d_{21} + 1/d_{61} + 1/d_{51} + 1/d_{41},$

$N_f^2 = 1/d_{12} + 1/d_{62} + 1/d_{52} + 1/d_{32},\ N_f^6 = 1/d_{21} + 1/d_{61} + 1/d_{51} + 1/d_{31},$

$N_f^3 = 1/d_{12} + 1/d_{62} + 1/d_{42} + 1/d_{32}, N_f^7 = 1/d_{21} + 1/d_{61} + 1/d_{41} + 1/d_{31},$

$N_f^4 = 1/d_{12} + 1/d_{52} + 1/d_{42} + 1/d_{32}, N_f^8 = 1/d_{21} + 1/d_{51} + 1/d_{41} + 1/d_{31}.$

$\bar{B}$ is the matrix of transition probabilities from the transient states into the absorbing states:

[2] For this example the more detailed representation $(s^{(l)}, \bar{r}^{(\cdot)})$, where $\bar{r}^{(1)} = \bar{r}^j = (1,0,0,0,0,0)^T$ for $j = 1,3,\cdots$ and $\bar{r}^{(2)} = \bar{r}^j = (0,1,0,0,0,0)^T$ for $j = 2,4,\cdots$ of the transient states is not necessary since these assignments are unique for each $s^{(l)}$: $(s^{(0)}, \bar{r}^{(1)})$ at the beginning, $(s^{(1)}, \bar{r}^{(2)})$, $(s^{(2)}, \bar{r}^{(2)})$ and so on.

$$\bar{B} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1/d_{52}}{N_f^4} & \frac{1/d_{42}}{N_f^4} & \frac{1/d_{32}}{N_f^4} & 0 & 0 & 0 \\ \frac{1/d_{62}}{N_f^3} & 0 & 0 & \frac{1/d_{42}}{N_f^3} & \frac{1/d_{32}}{N_f^3} & 0 \\ 0 & \frac{1/d_{62}}{N_f^2} & 0 & \frac{1/d_{52}}{N_f^2} & 0 & \frac{1/d_{32}}{N_f^2} \\ 0 & 0 & \frac{1/d_{62}}{N_f^1} & 0 & \frac{1/d_{52}}{N_f^1} & \frac{1/d_{42}}{N_f^1} \\ \frac{1/d_{51}}{N_f^8} & \frac{1/d_{41}}{N_f^8} & \frac{1/d_{31}}{N_f^8} & 0 & 0 & 0 \\ \frac{1/d_{61}}{N_f^7} & 0 & 0 & \frac{1/d_{41}}{N_f^7} & \frac{1/d_{31}}{N_f^7} & 0 \\ 0 & \frac{1/d_{61}}{N_f^6} & 0 & \frac{1/d_{51}}{N_f^6} & 0 & \frac{1/d_{31}}{N_f^6} \\ 0 & 0 & \frac{1/d_{61}}{N_f^5} & 0 & \frac{1/d_{51}}{N_f^5} & \frac{1/d_{41}}{N_f^5} \end{pmatrix} .$$

Finally, the absorbing probabilities can be computed by the following linear systems, see e.g. (Langrock and Jahn, Theorem 6.6).

$$u^{(j)} = B\, u^{(j)} + \bar{B}^{(j)}$$

with variables $u_i^{(j)}$ $(i = 0, \cdots, 8)$ and where $\bar{B}^{(j)}$ is the j-th column of matrix $\bar{B}$. The solution value of $u_i^{(j)}$ is the absorbing probability of state $s'^{(j)}$ $(j \in \{1, \cdots, 6\})$, if the initial state of the corresponding Markov chain is $s^{(i)}$ $(i \in \{0, 1, \cdots, 8\})$. Thus $u_0^{(j)} = P_C(s'^{(j)}|s, r)$.

Author's information

R. Hildenbrandt received a PhD degree as well a doctor habilitus degree from Ilmenau Technical University and works now as a Privatdozent in the Optimization Department on stochastic dynamic programming and online optimization.

Received January, 2016 , revised May 4, 2016, accepted June 13, 2016