Randomized algorithm for the k-server problem on decomposable spaces

Judit Nagy-György

Department of Mathematics, University of Szeged, Aradi vértanúk tere 1, H-6720 Szeged, Hungary, email: Nagy-Gyorgy@math.u-szeged.hu

Abstract

We study the randomized k-server problem on metric spaces consisting of widely separated subspaces. We give a method which extends existing algorithms to larger spaces with the growth rate of the competitive quotients being at most $O(\log k)$. This method yields $o(k)$-competitive algorithms solving the randomized k-server problem for some special underlying metric spaces, e.g. HSTs of "small" height (but unbounded degree). HSTs are important tools for probabilistic approximation of metric spaces.

Key words: k-server, on-line algorithms, randomized algorithms, metric spaces

1 Introduction

In the theory of designing efficient virtual memory-management algorithms, the well studied paging problem plays a central role. Even the earliest operation systems contained some heuristics to minimize the amount of copying memory pages, which is an expensive operation. A generalization of the paging problem, called the k-server problem was introduced by Manasse, McGeoch and Sleator in [16], where the first important results were also achieved. The problem can be formulated as follows. Given a metric space with k mobile servers that occupy distinct points of the space and a sequence of requests (points), each of the requests has to be served, by moving a server from its current position to the requested point. The goal is to minimize the total cost, that is the sum of the distances covered by the k servers; the optimal cost for a given sequence ϱ is denoted $\mathrm{opt}(k, \varrho)$.

An algorithm is online if it serves each request immediately when it arrives (without any prior knowledge about the future requests).

Definition 1 *An online algorithm A is c-competitive if for any initial configuration C_0 and request sequence ϱ it holds that*

$$\mathrm{cost}(A(C_0, \varrho)) \leq c \cdot \mathrm{opt}(k, \varrho) + I(C_0),$$

where I is a non-negative constant depending only on C_0.

The competitive ratio of a given online algorithm A is the infimum of the values c with A being c-competitive. The k-server conjecture (see [16]) states that there exists an algorithm A that is k-competitive for any metric space. Manasse et al. proved that k is a lower bound [16], and Koutsoupias and Papadimitriou showed $2k - 1$ is an upper bound for any metric space [14].

In the randomized online case (sometimes this model is called the oblivious adversary model [7]) the competitive ratio can be defined in terms of the expected value as follows:

Definition 2 *A randomized online algorithm R is c-competitive if for any initial configuration C_0 and request sequence ϱ we have*

$$\mathrm{E}[\mathrm{cost}(R(C_0, \varrho))] \ \leq \ c \cdot \mathrm{opt}(k, \varrho) + I(C_0),$$

where I is a non-negative constant depending only on C_0 and $\mathrm{E}[\mathrm{cost}(R(\varrho))]$ *denotes the expected value of* $\mathrm{cost}(R(C_0, \varrho))$*.*

The competitive ratio of the above randomized algorithm is defined analogously.

In the randomized version there are more problems that are still open. The randomized k-server conjecture states that there exists a randomized algorithm with a competitive ratio $\Theta(\log k)$ in any metric space. The best known lower bound is $\Omega(\log k / \log \log k)$ which follows from the results of [6] (see also [4]). A natural upper bound is the bound $2k + 1$ given for the deterministic case. By restricting our attention to metric spaces with a special structure, better bounds can be achieved: for uniform metric spaces, Fiat et al. [12] proved a lower bound $H_k = \sum_{i=0}^{k} i^{-1} \approx \log k$ (and an upper bound $2H_k$), while McGeoch and Sleator [15] showed that their algorithm PARTITION guarantees the upper bound H_k.

In this paper we also consider a restriction of the problem, namely we seek for an efficient randomized online algorithm for metric spaces that are "μ-HST spaces" [4] and defined as follows:

Definition 3 *For $\mu \geq 1$, a μ-hierarchically well-separated tree (μ-HST) is a metric space defined on the leaves of a rooted tree T. To each vertex $u \in T$ there is associated a label $\Lambda(u) \geq 0$ such that $\Lambda(u) = 0$ if and only if u is a leaf of T. The labels are such that if a vertex u is a child of a vertex v*

then $\Lambda(u) \leq \Lambda(v)/\mu$. *The distance between two leaves* $x, y \in T$ *is defined as* $\Lambda(\mathrm{lca}(x, y))$, *where* $\mathrm{lca}(x, y)$ *is the least common ancestor of* x *and* y *in* T.

The μ-HST spaces play an important role in the probabilistic embedding technique developed by Alon et al. [1] and Bartal [3]. Fakcharoenphol et al [13] proved that every metric space on n points can be α-probabilistically approximated by a set of μ-HSTs, for an arbitrary $\mu > 1$ where $\alpha = O(\mu \log n / \log \mu)$.

Seiden [17] proved the existence of an $O(\mathrm{polylog}\, k)$-competitive algorithm for $\Omega(k \log k)$-decomposable spaces, where the space can be partitioned into $O(\log k)$ uniform blocks, each having diameter 1, and where the distance of any two blocks is at least $c{\cdot}k{\cdot}\log k$. In his work he also showed that for binary HST's (where each non-leaf node has exactly two children) there exists an $O(\log^3 k)$-competitive algorithm, provided the parameter μ of the HST is sufficiently large. Very recently Coté et al. [8] designed a randomized algorithm on binary trees with competitive ratio logaritmic in the diameter of the metric (but independent of k).

We study decomposable spaces too, but unlike the above results our spaces consist of an arbitrary number of (not necessarily uniform) blocks with large distance between them. By slightly modifying the approach of Csaba and Lodha [9] and Bartal and Mendel [5] [1] we show that there exists a polylog k-competitive algorithm for any μ-HST that has a small depth and *arbitrary* maximum degree t, given $\mu \geq k$. Our algorithm heavily relies on the technical notion of demand (Definition 5), which plays a central role in the description and the analysis of the algorithm.

2 Notation

In [17], μ-decomposable spaces have been introduced. We consider a special case of this notion as follows:

Definition 4 *Let* $\mathcal{M}$ *be a metric space. We call* $\mathcal{M}$ uniformly μ-decomposable *for some* $\mu > 1$ *if its points can be partitioned into* $t \geq 2$ *blocks,* $B_1, \ldots, B_t$ *such that the following conditions both hold:*

(1) whenever $x, y \in \mathcal{M}$ *are belonging to different blocks, their distance is exactly* Δ*, the diameter of* $\mathcal{M}$*;*
(2) the diameter of each B_i *is at most* Δ/μ.

[1] Although the publication has been withdrawn (see `http://arxiv.org/abs/cs.DS/0406033`), the approach itself is still valuable.

For example, a μ-HST with at least two points is an uniformly μ-decomposable metric space.

For the rest of the paper we fix a uniformly μ-decomposable metric space $\mathcal{M}$ having a diameter Δ, consisting of the blocks $B_1, \dots, B_t$, with maximal diameter δ such that $\mu = \Delta/\delta$.

For a given request sequence ϱ we denote its ith member by ϱ_i, and the prefix of ϱ of length i by $\varrho_{\leq i}$. The length of the sequence is denoted $|\varrho|$.

Given a block B_s, a request sequence ϱ and an initial configuration C in B_s, let $\mathrm{cost}(A_s(C, \varrho))$ denote the cost computed by the algorithm A for the subsequence of ϱ consisting of the requests arriving to B_s. these inputs. For any number ℓ of servers, let $\mathrm{cost}(A_s(\ell, \varrho))$ stand for $\max\limits_{|C|=\ell} \mathrm{cost}(A_s(C, \varrho))$, where C runs over all the initial configurations in B_s consisting of ℓ servers. Also, let $\mathrm{opt}_s(C, \varrho)$ denote the optimal cost for the subsequence of ϱ consisting of the requests arriving to B_s, starting from configuration C and let $\mathrm{opt}_s(\ell, \varrho) = \min\limits_{|C|=\ell} \mathrm{opt}_s(C, \varrho)$. Thus, if ϱ is nonempty, $\mathrm{opt}_s(0, \varrho)$ is defined to be infinite.

Definition 5 *The* demand *of the block B_s for the request sequence ϱ is*

$$D_s(\varrho) := \min\{\ell \mid \mathrm{opt}_s(\ell, \varrho) + \ell\Delta = \min_j \{\mathrm{opt}_s(j, \varrho) + j\Delta\}\},$$

if ϱ is nonempty, otherwise it is 0.

Intuitively, $D_s(\varrho)$ denotes the least number of servers to be moved into the initially empty block B_s to achieve the optimal cost for the sequence ϱ. Observe that $D_s(\varrho)$ is finite since it is a nonnegative integer bounded by e.g. $|\varrho|$.

In the rest of the paper, the notion of demand of the blocks will play a crucial role. We now state a conjecture which would simplify the ensuing calculations, if it happened to be verified; however, we did not succeed to prove or disprove it yet.

Conjecture 6 *For any block B_s and request sequence ϱ inside B_s and index $0 < i < |\varrho|$, the difference $D_s(\varrho_{\leq i+1}) - D_s(\varrho_{\leq i})$ is either* 0 *or* 1.

A weaker, but still open question is that whether the sequence $(D_s(\varrho_{\leq i}))_{i=1}^{|\varrho|}$ is monotone for every ϱ and B_s.

We also introduce a technical notion.

Definition 7 *Suppose $\mathcal{N}$ is a metric space, A is a randomized online algorithm, f is a real function and $\mu > 0$ satisfying the following conditions:*

(1) $f(\ell)/\log \ell$ is monotone non-decreasing;

(2) for any $0 < \ell \leq \mu$ and request sequence ϱ in $\mathcal{N}$,

$$\mathrm{E}[\mathrm{cost}(A(\ell,\varrho))] \;\leq\; f(\ell)\cdot \mathrm{opt}(\ell,\varrho) + \frac{f(\ell)\cdot \ell \cdot \mathrm{diam}(\mathcal{N})}{\log \ell}. \qquad (1)$$

Then we call A an (f,μ)-efficient algorithm on $\mathcal{N}$.

Observe that if A is (f,μ)-efficient on $\mathcal{N}$, then A is $f(k)$-competitive for the k-server problem on $\mathcal{N}$ for any $0 < k < \mu$.

3 Results

Our aim is to prove the following theorem:

Theorem 8 *Suppose $\mathcal{M}$ is a uniformly μ-decomposable space and A is an (f,μ)-efficient algorithm on each block of $\mathcal{M}$. Then there exists an (f',μ)-efficient algorithm on $\mathcal{M}$, where $f'(x)$ is defined as $c\cdot f(x)\log x$ for some absolute constant $c > 0$.*

For the rest of the paper we now fix an algorithm A and a real function f such that A is an (f,μ)-efficient algorithm on each block of (the already fixed) $\mathcal{M}$. In the next subsection we define the algorithm which will be proven to be (f',μ)-efficient on $\mathcal{M}$. In the rest of the paper we suppose that $k \leq \mu$ arbitrary.

3.1 Algorithm X

The algorithm uses A as a subroutine and it works in phases. Let $\varrho^{(p)}$ denote the sequence of the pth phase. In this phase the algorithm works as follows:

Initially we mark the blocks that contain no servers.

When $\varrho_i^{(p)}$, the ith request of this phase arrives to block B_s, we compute the demand $D_s(\varrho_{\leq i}^{(p)})$ and the maximal demand

$$D_s^*(\varrho_i^{(p)}) = \max\{D_s(\varrho_{\leq j}^{(p)}) | j \leq i\}$$

for this block (note that these values do not change in the other blocks).

– If $D_s^*(\varrho_i^{(p)})$ is less than the number of servers in B_s at that moment, then the request is served by Algorithm A, with respect to the block B_s.

– If $D_s^*(\varrho_i^{(p)})$ becomes equal to the number of servers in B_s at that moment, then the request is served by Algorithm A, with respect to the block B_s and we mark the block B_s.

– If $D^*_s(\varrho^{(p)}_i)$ is greater than the number of servers in B_s at that moment, we mark the block B_s and perform the following steps until we have $D^*_s(\varrho^{(p)}_i)$ servers in that block or we cannot execute the steps (this happens when all the blocks become marked):

- Choose an unmarked block $B_{s'}$ randomly uniformly, and a server from this block also randomly. We move this chosen server to the block B_s (such a move is called a *jump*), either to the requested point, or, if there is already a server occupying that point, to a randomly chosen unoccupied point of B_s. If the number of servers in $B_{s'}$ becomes $D^*_{s'}(\varrho^{(p)}_i)$ via this move, we mark that block. In both B_s and $B_{s'}$ we restart algorithm A from the current configuration of the block.

If we cannot raise the number of servers in block B_s to $D^*_s(\varrho^{(p)}_i)$ by repeating the above steps (all the blocks became marked), then Phase $p+1$ is starting and the last request is belonging to this new phase.

Intuitively, Algorithm X consists of the following parts: the server movements inside a block are handled by the inner Algorithm A, while the "jumps" from a block to another are determined by an online matching algorithm (introduced by Csaba and Pluhár [10]); its requests are induced by the demands.

For any phase p of Algorithm X we can associate a matching problem MX. We recall from [10] that an online matching problem is defined similarly to the online k-server problem with the following two differences:

(1) Each of the servers can move only once;
(2) The number of the requests is at most k, the number of the servers.

The underlying metric space of MX is a finite uniform metric space that has the blocks B_s as points and a distance Δ between any two different points. Let $\hat{D}_s(p)$ denote the number of servers that are in the block B_s just at the end of phase p. Now in the associated matching problem we have $\hat{D}_s(p-1)$ servers originally occupying the point B_s. During phase p, if some value D^*_s increases, we make a number of requests in point B_s for the associated matching problem: we make the same number of requests that the value D^*_s has been increased with. Each of these requests have to be served by a server, moreover, one server can handle only one request (during the whole phase).

We also associate an auxiliary matching algorithm (AMA) on this structure as follows. While there exists a server in the block B_s which have not served any request yet, let this server serve the request arriving to B_s. Otherwise, D^*_s increases at some time, causing jumps. These jumps are corresponding to requests of the associated matching problem; AMA satisfies these requests by the servers that are corresponding to those involved in these jumps.

For convenience we modify the request sequence ϱ in a way that does not increase the optimal cost and does not decrease the cost of any online algorithm, hence the bounds we get for this modified sequence will hold also in the general case. The modification is defined as follows: we extend the sequence by repeatedly requesting the points of the halting configuration of a (fixed) optimal solution. We do this till $\sum_{s=1}^{t} D_s^*(\varrho_{\leq i}^{(u)})$ becomes k. Observe that the optimal cost does not change via this transformation, and any online algorithm works the same way in the original part of the sequence (hence online), so the cost computed by any online algorithm is at least the original computed cost.

In the following two subsections we will give an upper bound for the cost of Algorithm X and several lower bounds for the optimal cost in an arbitrary phase. Theorem 8 easily follows from these.

We remark that the number t of blocks do not appear in the statement of Theorem 8, which is not surprising, since in each phase, at most $2k$ blocks of $\mathcal{M}$ can be involved. This comes from the fact that each server jumps at most once during one phase (since if a server jumps into a block, that block has to be a marked one, thus the server is not allowed to jump out from that block during the same phase).

4 Upper bound

In the first step we prove an auxiliary result.

If p is not the last phase, let $\varrho^{(p)+}$ denote the request sequence we get by adding the first request of phase $p+1$ to $\varrho^{(p)}$. Now we have

$$D_s^*(\varrho^{(p)}) \leq \hat{D}_s(p) \leq D_s^*(\varrho^{(p)+}) \tag{2}$$

and in all block but at most one we have equalities there (this is the block that causes termination of the pth phase).

Denote

$$m_p := \sum_{s=1}^{t} \max\{0, \hat{D}_s(p) - \hat{D}_s(p-1)\}. \tag{3}$$

Since the auxiliary metric space is uniform, the optimal cost is $m_p \cdot \Delta$.

From Lemma 6 of [10] we immediately get the following:

Lemma 9 *The expected cost of AMA is at most* $\log k \cdot m_p \Delta$.

A lemma similar in nature to the above was also presented in [9].

Lemma 10 *The expected cost of Algorithm X in the pth phase is at most*

$$f(k)\left(\sum_{s=1}^{t}\mathrm{opt}_s(D_s(\varrho^{(p)+}),\varrho^{(p)})+\left(\sum_{s=1}^{t}D_s(\varrho^{(p)+})-k\right)\Delta\right)$$

$$+\Big(f(k)\log k+f(k)+\log k\Big)m_p\Delta+\frac{f(k)}{\log k}\Delta.$$

PROOF. Consider the pth phase of an execution of Algorithm X on the request sequence ϱ. We fix a possible associated execution τ of AMA (which satisfies the request sequence induced by ϱ); let $\mathcal{E}_\tau$ denote the event that the execution of AMA is this τ. We will give an upper bound to the overall expected cost of Algorithm X during phase p assuming τ. After that, we get the expected cost appearing in Lemma 10 as a weighted sum.

Let B_s be a block in which some request arrives during this phase. For the sake of convenience we will omit the subscript s when it is clear from the context. While the block B_s is unmarked, only jump-outs can happen from this block (in phase p); let d^- be the number of these jump-outs. After B_s has been marked, only jump-ins happen into this block; let d^+ be the number of these jump-ins and let $d = d^- + d^+$ denote the total number of jumps involving B_s during phase p.

Also, for any $1 \le i \le d^-$ let r_i be the index of the request in $\varrho^{(p)}$ which causes the ith jump-out from B_s, and for any $1 \le i \le d^+$ let r_{d^-+i} be the index of the request which causes the ith jump-in to B_s.

Denote $\sigma_i = \varrho^{(p)}_{r_i}\dots\varrho^{(p)}_{r_{i+1}-1}$, where $\varrho^{(p)}_{r_0}$ is the first request of the phase and $\varrho^{(p)}_{r_{d+1}-1}$ is the last request of the phase. (In other words, σ_i is the ith maximal segment of $\varrho^{(p)}$ between two jumps. Table 1 shows an illustration.)

It is clear that the number of servers inside B_s does not change between two jumps; for each $0 \le i \le d$, let k_i denote the number of servers inside B_s during σ_i. Finally, let $\ell_i = D_s(\varrho^{(p)}_{<r_{i+1}})$ (the demand of B_s for the sequence $\varrho^{(p)}_{<r_{i+1}}$). Observe that $\ell_i \le k$ for each i, moreover $D_s(\varrho^{(p)}_{\le r_i})$ is exactly k_i, when $i > d^-$, and is strictly less than k_i, when $i < d^-$.

jump		outs		outs		ins		ins	
	σ_0	$\downarrow$	σ_1	$\downarrow$	σ_2	$\downarrow$	σ_3	$\downarrow$	σ_4
$\varrho^{(p)}_j$:	$\overbrace{\dots\varrho^{(p)}_{r_1-1}}$		$\overbrace{\varrho^{(p)}_{r_1}\dots\varrho^{(p)}_{r_2-1}}$		$\overbrace{\varrho^{(p)}_{r_2}\dots\varrho^{(p)}_{r_3-1}}$		$\overbrace{\varrho^{(p)}_{r_3}\dots\varrho^{(p)}_{r_4-1}}$		$\overbrace{\varrho^{(p)}_{r_4}\dots}$

Table 1
Partitioning of a phase. Here $d^- = d^+ = 2$.

A jump-in to the block satisfies the last request, hence there is no server movement inside the block during a jump. The expected cost of non-jump movements in this block (this is called the *inner cost*) is, applying (1), at most

$$\begin{aligned}\sum_{i=0}^{d} \mathrm{E}[A_s(k_i,\sigma_i)|\,\mathcal{E}_\tau] \;&\le\; \sum_{i=0}^{d}\Big(f(k_i)\mathrm{opt}_s(k_i,\sigma_i) + \frac{k_i\cdot f(k_i)}{\log k_i}\delta\Big)\\ &\le\; f(k)\sum_{i=0}^{d}\mathrm{opt}_s(k_i,\sigma_i) + \delta\sum_{i=0}^{d}\frac{k_i\cdot f(k_i)}{\log k_i}. \qquad (4)\end{aligned}$$

Recall that $\mathcal{E}_\tau$ is the random event that τ is the associated run of AMA.

We bound the right hand side of (4) piecewise. In the first step we bound the inner costs till the $(d^- - 1)$th jump (which is still a jump-out):

$$\sum_{i=0}^{d^- -1}\mathrm{opt}_s(k_i,\sigma_i) \le \sum_{i=0}^{d^- -1}\mathrm{opt}_s(k_{d^-},\sigma_i) \le \mathrm{opt}_s(k_{d^-},\varrho^{(p)}_{\le r_{d^-}}). \qquad (5)$$

From the last jump-out till the last jump-in:

$$\begin{aligned}\sum_{i=d^-}^{d-1}\mathrm{opt}_s(k_i,\sigma_i) &\le \sum_{i=d^-}^{d-1}\mathrm{opt}_s(\ell_i,\sigma_i)\\ &= \sum_{i=d^-}^{d-1}\Big(\mathrm{opt}_s(\ell_i,\sigma_i) + \mathrm{opt}_s(\ell_i,\varrho^{(p)}_{\le r_i}) - \mathrm{opt}_s(\ell_i,\varrho^{(p)}_{\le r_i})\Big)\\ &\le \sum_{i=d^-}^{d-1}\Big(\mathrm{opt}_s(\ell_i,\varrho^{(p)}_{<r_{i+1}}) - \mathrm{opt}_s(\ell_i,\varrho^{(p)}_{\le r_i})\Big)\\ &\le \sum_{i=d^-}^{d-1}\Big(\Big(\mathrm{opt}_s(k_{i+1},\varrho^{(p)}_{<r_{i+1}}) + (k_{i+1}-\ell_i)\Delta\Big)\\ &\qquad\qquad - \Big(\mathrm{opt}_s(k_i,\varrho^{(p)}_{\le r_i}) + (k_i-\ell_i)\Delta\Big)\Big) \qquad (6)\\ &\le \sum_{i=d^-}^{d-1}\Big(\mathrm{opt}_s(k_{i+1},\varrho^{(p)}_{\le r_{i+1}}) - \mathrm{opt}_s(k_i,\varrho^{(p)}_{\le r_i}) + (k_{i+1}-k_i)\Delta\Big)\\ &= \mathrm{opt}_s(k_d,\varrho^{(p)}_{\le r_d}) - \mathrm{opt}_s(k_{d^-},\varrho^{(p)}_{\le r_{d^-}}) + (k_d - k_{d^-})\Delta. \qquad (7)\end{aligned}$$

Inequality (6) follows from Definition 5, since the demand of B_s for $\varrho^{(p)}_{<r_{i+1}}$ is ℓ_i and the demand of B_s for $\varrho^{(p)}_{\le r_i}$ is k_i.

Since $k_d \ge D_s(\varrho^{(p)})$, analogously we get

$$\begin{aligned}
\mathrm{opt}_s(k_d,\sigma_d) &\le \mathrm{opt}_s(D_s(\varrho^{(p)}),\sigma_d)\\
&\le \mathrm{opt}_s(D_s(\varrho^{(p)}),\varrho^{(p)}) - \mathrm{opt}_s(D_s(\varrho^{(p)}),\varrho^{(p)}_{\le r_d})\\
&\le \mathrm{opt}_s(D_s(\varrho^{(p)+}),\varrho^{(p)}) + (D_s(\varrho^{(p)+}) - D_s(\varrho^{(p)}))\Delta\\
&\quad - \mathrm{opt}_s(k_d,\varrho^{(p)}_{\le r_d}) - (k_d - D_s(\varrho^{(p)}))\Delta \qquad (8)\\
&= \mathrm{opt}_s(D_s(\varrho^{(p)+}),\varrho^{(p)}) - \mathrm{opt}_s(k_d,\varrho^{(p)}_{\le r_d})\\
&\quad +(D_s(\varrho^{(p)+}) - k_d)\Delta. \qquad (9)
\end{aligned}$$

Again, (8) follows from Definition 5, since the demand of B_s for $\varrho^{(p)}$ is $D_s(\varrho^{(p)})$ and the demand of B_s for $\varrho^{(p)+}_{\le r_d}$ is k_d. Note that if the first request of the $p+1$th phase arrives to block B_s, then $D_s(\varrho^{(p)}) < D_(\varrho^{(p)+})$, otherwise the two demands are equal.

Summing up the right hand sides of (5), (7) and (9) we get

$$\sum_{i=0}^{d} \mathrm{opt}_s(k_i,\sigma_i) \le \mathrm{opt}_s(D_s(\varrho^{(p)+}),\varrho^{(p)}) + (D_s(\varrho^{(p)+}) - k_{d^-})\Delta, \qquad (10)$$

and substituting this to the right hand side of (4) we get that the expected inner cost in B_s is at most

$$f(k)\left(\mathrm{opt}_s(D_s(\varrho^{(p)+}),\varrho^{(p)}) + (D_s(\varrho^{(p)+}) - k_{d^-})\Delta\right) + \sum_{i=0}^{d} \frac{k_i \cdot f(k_i)}{\log k_i}\delta. \qquad (11)$$

On the other hand,

$$D_s(\varrho^{(p)+}) - k_{d^-} = (D_S(\varrho^{(p)+}) - \hat{D}_s(p)) + (\hat{D}_s(p) - k_{d^-}), \qquad (12)$$

where we know that $(\hat{D}_s(p) - k_{d^-})$ is the number of jump-ins into this block.

From Definition 7, $\sum_{i=0}^{d} \frac{k_i \cdot f(k_i)}{\log k_i}\delta \le \frac{f(k)}{\log k}\delta \sum_{i=0}^{d} k_i$ (since $k_i \le k$). Recall that by definition k_i denotes the number of servers in B_s, after the ith jump involving block s. Summing these values for every block and for every jump, we get $(|\tau|+1)k$ as an upper bound, where $|\tau|$ is the total number of jumps. Hence, the sum of the expressions of the form $\frac{k_i \cdot f(k_i)}{\log k_i}\delta$ can be bounded by

$$(|\tau|+1)\frac{f(k)}{\log k}k\delta. \qquad (13)$$

We also remark that

$$|\tau| \le k, \qquad (14)$$

since any server can jump at most once: after a server jumps into a block, the block has to be marked, thus no server can jump out from that given block in this phase.

Now we bound the cost of the jumps. Let T be the set of the potential associated runs of AMA and let η be the random variable $\mathcal{E}_\tau \mapsto |\tau|$, for each $\tau \in T$. Applying Lemma 9 we get that the expected value of the total number of jumps is

$$\mathrm{E}[\eta] = \sum_{\tau \in T} \Pr(\mathcal{E}_\tau)|\tau| \leq \log k \cdot m_p \tag{15}$$

Summing up the results (11), (12), (13) and (15) for all the blocks we get the following bound for the expected cost of Algorithm X:

$$\begin{aligned}
&\sum_{s=1}^{t} \sum_{i=0}^{d_s^- + d_s^+} \mathrm{E}[A_s(k_i, \sigma_i) + \eta\Delta] && (16)\\
&= \sum_{\tau \in T} \left(\Pr(\mathcal{E}_\tau) \sum_{s=1}^{t} \sum_{i=0}^{d_s^- + d_s^+} \mathrm{E}[A_s(k_i, \sigma_i) | \, \mathcal{E}_\tau] \right) + \mathrm{E}[\eta]\Delta \\
&\leq \sum_{\tau \in T} \Pr(\mathcal{E}_\tau) \Bigg(f(k) \Bigg(\sum_{s=1}^{t} \mathrm{opt}_s(D_s(\varrho^{(p)+}), \varrho^{(p)+}) \\
&\quad + \sum_{s=1}^{t} \Big(D_s(\varrho^{(p)+}) - \hat{D}_s(p) \Big)\Delta + |\tau|\Delta \Bigg) + (|\tau| + 1) \frac{f(k)}{\log k} k\delta \Bigg) \\
&\quad + \log k \cdot m_p \cdot \Delta \\
&\leq \sum_{\tau \in T} \Pr(\mathcal{E}_\tau) f(k) \left(\sum_{s=1}^{t} \mathrm{opt}_s(D_s(\varrho^{(p)+}), \varrho^{(p)+}) + \sum_{s=1}^{t} D_s(\varrho^{(p)+})\Delta - k\Delta \right) && (17)\\
&\quad + \sum_{\tau \in T} \Pr(\mathcal{E}_\tau)|\tau| \left(f(k)\Delta + \frac{f(k)}{\log k} k\delta \right) + \frac{f(k)}{\log k} k\delta + \log k \cdot m_p \cdot \Delta && (18)\\
&\leq f(k) \left(\sum_{s=1}^{t} \mathrm{opt}_s(D_s(\varrho^{(p)+}), \varrho^{(p)+}) + \sum_{s=1}^{t} D_s(\varrho^{(p)+})\Delta - k\Delta + \log k \cdot m_p\Delta \right) \\
&\quad + \left(\frac{f(k)}{\log k} m_p \log k + \frac{f(k)}{\log k} + m_p \log k \right) \Delta,
\end{aligned}$$

if we apply $\sum_{\tau \in T} \Pr(\mathcal{E}_\tau) = 1$ in (17) and $k\delta \leq \Delta$ in (18). □

5 Analyzing the optimal cost

Consider an optimal solution of the k-server problem. Let $C_s^*(\varrho)$ be the maximal number of servers in B_s of this optimal solution during ϱ and let $C_s(\varrho)$ be the number of servers in B_s of the optimal solution at the end ϱ. We modify ϱ as follows: we extend each phase (except the last one) with a copy of the first request of the next phase, and consider $\varrho^{(p)+}$ instead of $\varrho^{(p)}$. In this section we bound the optimal cost for this modified sequence. It is obvious that the optimal cost of these sequences is the same.

Observe that for any s and p, $C^*_s(\varrho^{(p)+}) \geq C_s(\varrho^{(p-1)+}))$, since each (modified) phase p begins with the last configuration of phase $p-1$. Then, $\sum_{s=1}^t (C^*_s(\varrho^{(p)+}) - C_s(\varrho^{(p-1)+}))$ is clearly a lower bound for the number of jumps of the optimal solution during (the modified) phase p. Thus, the cost of the optimal solution during $\varrho^{(p)+}$ (which has $C_s(\varrho^{(p-1)+}$, $s = 1, \dots, t$ as the initial configuration) can be bounded by

$$\mathrm{opt}(k, \varrho^{(p)+}) \geq \sum_{s=1}^{t} (C^*_s(\varrho^{(p)+}) - C_s(\varrho^{(p-1)+}))\Delta + \sum_{s=1}^{t} \mathrm{opt}_s(C^*_s(\varrho^{(p)+}), \varrho^{(p)+}), \quad (19)$$

i.e., Δ times a lower bound for the number of jumps, plus a lower bound for the inner cost, where we treat each block as if we had the maximal number of servers during the whole phase.

Lemma 11

$$\mathrm{opt}(k, \varrho^{(p)+}) \geq \sum_{s=1}^{t} \mathrm{opt}_s(D_s(\varrho^{(p)+}), \varrho^{(p)+}) + \left(\sum_{s=1}^{t} D_s(\varrho^{(p)+}) - k \right) \Delta.$$

PROOF. From Definition 5 we have

$$\begin{aligned} &\sum_{s=1}^{t} \left(\mathrm{opt}_s(C^*_s(\varrho^{(p)+}), \varrho^{(p)+}) + (C^*_s(\varrho^{(p)+}) - C_s(\varrho^{(p-1)+}))\Delta \right) \\ &\geq \sum_{s=1}^{t} \left(\mathrm{opt}_s(D_s(\varrho^{(p)+}), \varrho^{(p)+}) + (D_s(\varrho^{(p)+}) - C_s(\varrho^{(p-1)+}))\Delta \right). \end{aligned}$$

Since $\sum_{s=1}^t C_s(\varrho^{(p-1)+}) = k$, the statement follows by (19). $\square$

Proposition 12 *For any s and p,*

$$\begin{aligned} &\mathrm{opt}_s(C^*_s(\varrho^{(p)+}), \varrho^{(p)+}) + (C^*_s(\varrho^{(p)+}) - C_s(\varrho^{(p-1)+}))\Delta \\ &\geq \max\{0, D^*_s(\varrho^{(p)+}) - C_s(\varrho^{(p-1)+})\}\Delta. \end{aligned}$$

PROOF. Let $\varrho^{(p)*}$ be the subsequence of $\varrho^{(p)}$ which we get by omitting each request that arrives to a block B_s after the demand of that block reaches $D^*_s(\varrho^{(p)+})$ (note that $\varrho^{(p)*}$ is not neccessarily a prefix of $\varrho^{(p)}$). Now we have two cases: first, if $D^*_s(\varrho^{(p)+}) > C_s(\varrho^{(p-1)+})$, then by Definition 5

$$\begin{aligned}
&\text{opt}_s(C_s^*(\varrho^{(p)+}), \varrho^{(p)+}) + (C_s^*(\varrho^{(p)+}) - C_s(\varrho^{(p-1)+}))\Delta \\
\geq &\,\text{opt}_s(C_s^*(\varrho^{(p)+}), \varrho^{(p)*}) + (C_s^*(\varrho^{(p)+}) - C_s(\varrho^{(p-1)+}))\Delta \\
\geq &\,(\text{opt}_s(D_s^*(\varrho^{(p)+}), \varrho^{(p)*}) + \left(D_s^*(\varrho^{(p)+}) - C_s(\varrho^{(p-1)+}))\right)\Delta \\
\geq &\,\max\{0, D_s^*(\varrho^{(p)+}) - C_s(\varrho^{(p-1)+}))\}\Delta.
\end{aligned}$$

Otherwise it holds that

$$\max\{0, D_s^*(\varrho^{(p)+}) - C_s(\varrho^{(p-1)+}))\} = 0,$$

and also obviously

$$\text{opt}_s(C_s^*(\varrho^{(p)+}), \varrho^{(p)+}) + (C_s^*(\varrho^{(p)+}) - C_s(\varrho^{(p-1)+}))\Delta \geq 0.$$

□

Lemma 13 *The optimal cost is at least*

$$\frac{1}{6}\sum_{p>1} m_p \cdot \Delta.$$

PROOF. Since $\sum_{s=1}^{t} \hat{D}_s(p) = \sum_{s=1}^{t} C_s(\varrho^{(p-1)+}) = k$, it holds that

$$\begin{aligned}
&\sum_{s=1}^{t} \max\{0, \hat{D}_s(p) - C_s(\varrho^{(p-1)+})\}\Delta \\
= &\sum_{s=1}^{t} \frac{1}{2}|\hat{D}_s(p) - C_s(\varrho^{(p-1)+})|\Delta. \qquad (20)
\end{aligned}$$

Summing up the cost of the jumps performed by the optimal solution we get

$$\sum_{s=1}^{t} |C_s(\varrho^{(p)+}) - C_s(\varrho^{(p-1)+})|\Delta \leq 2 \cdot \text{opt}(k, \varrho^{(p)+}). \qquad (21)$$

Note that the factor of 2 comes from the fact that each jump appears twice on the left hand side. Applying to (19) the statement of Proposition 12 and (20), using $D_s^*(\varrho^{(p)+}) \geq \hat{D}_s(p)$ we get

$$\begin{aligned}
&2 \cdot \text{opt}(k, \varrho^{(p)+}) \\
\geq &\sum_{s=1}^{t} \left(|\hat{D}_s(p) - C_s(\varrho^{(p-1)+})|\right)\Delta \qquad (22)
\end{aligned}$$

Now summing (22) and (21) and applying the triangle inequality we get

$$\begin{aligned}
&4 \cdot \mathrm{opt}(k, \varrho^{(p)+}) \\
&\geq \sum_{s=1}^{t} \Big(|\hat{D}_s(p) - C_s(\varrho^{(p-1)+}))| + |C_s(\varrho^{(p)+}) - C_s(\varrho^{(p-1)+})| \Big) \Delta \\
&\geq \sum_{s=1}^{t} |\hat{D}_s(p) - C_s(\varrho^{(p)+}))| \Delta. \qquad (23)
\end{aligned}$$

Also, from summing (22) and (23), the latter relativized to phase $p-1$, and applying again the triangle inequality,

$$\begin{aligned}
&2 \cdot \mathrm{opt}(k, \varrho^{(p)+}) + 4 \cdot \mathrm{opt}(k, \varrho^{(p-1)+}) \\
&\geq \sum_{s=1}^{t} \Big(|\hat{D}_s(p) - C_s(\varrho^{(p-1)+}))| + |\hat{D}_s(p-1) - C_s(\varrho^{(p-1)+}))| \Big) \Delta \\
&\geq \sum_{s=1}^{t} |\hat{D}_s(p) - \hat{D}_s(p-1)| \Delta = m_p \cdot \Delta, \qquad (24)
\end{aligned}$$

and the statement follows. □

6 Proof of Theorem 8

Now we are able to prove the theorem about competitiveness of Algorithm X.

PROOF. [Theorem 8] The first term in the right hand of the formula in Lemma 10 can be bounded by $f(k)\mathrm{opt}(k, \varrho^{(p)+})$ by Lemma 11. Furthermore if $p = 1$ we can write k instead of $m_1 \log k$ by (14), otherwise applying 13 we get that $\sum_p m_p \Delta$ can be bounded by $\frac{\Delta \cdot k}{\log k} + 6 \cdot \mathrm{opt}(k, \varrho)$. Summing up

$$\begin{aligned}
\mathrm{E}(\mathrm{cost}(\mathrm{X}(\varrho))) \leq\, & f(k)\mathrm{opt}(k, \varrho) \\
& + \left(\frac{\Delta \cdot k}{\log k} + 6 \cdot \mathrm{opt}(k, \varrho) \right) \Big(f(k) \log k + f(k) + \log k \Big) \\
& + \frac{f(k)}{\log k} \Delta, \\
=\, & \mathrm{opt}(k, \varrho) \Big(6 f(k) \log k + 7 f(k) + 6 \log k \Big) \\
& + \frac{f(k) \log k + f(k) + \log k + f(k)/\log k}{\log k} \cdot k \cdot \Delta
\end{aligned}$$

hence Algorithm X is (f', μ)-efficient on $\mathcal{M}$ with $f'(k) = O(f(k) \log k)$. □

7 Conclusions

Starting from the PARTITION algorithm [15] and iterating Theorem 8 we get the following result:

Corollary 14 *There exists a $(c_1 \log k)^h$-competitve randomized online algorithm on any μ-HST of height h (here $\mu \geq k$), where c_1 is a constant. Consequently, when $h < \frac{\log k}{\log c_1 + \log\log k}$, this algorithm is $o(k)$-competitive.*

In [11] a model has been investigated, where one does not have a fixed number of servers but they can be bought. The expression $\min_\ell\{\mathrm{opt}_s(\ell, \varrho) + \ell\Delta\}$ can be seen as the optimal cost in a model where one has to buy the servers, for a cost of Δ each. This problem on uniform spaces was studied in [11]. In this case $D_s(\varrho)$ is the number of servers bought in an optimal solution. Considering the sequence ϱ_i, the behavior of the associated sequence $D_s(\varrho_i)$ is not well understood at the moment, see Conjecture 6: the proofs would be substantially simpler, if this conjecture happened to be verified.

Another interesting question is that whether the $\log k$ factor in the competitive ratio per level of the HST is unavoidable, or an overall competitive ratio of $\Theta(\log k)$ holds for any HST. It is a bit more natural to require $\Delta \geq \delta M$ to hold, where M is the size of the greatest block. If additionally $M < k$ holds, we maybe get a better competitive ratio. It may be an another genuine advance to combine this this approach with other [8] and [17] to obtain improved randomized algorithms for the k-server problem.

8 Acknowledgement

The author wish to thank Béla Csaba for his guidance and suggestions, furthermore Péter Hajnal, Csanád Imreh and András Pluhár for their valuable remarks, as well as the two anonymous referees for their careful and detailed reviews. I am also thankful to one of them for calling my attention to the very recent result [8]. This research is partially supported by OTKA Grant K76099.

References

[1] N. Alon, R. M. Karp, D. Peleg, D. West, A graph-theoretic game and its application to the k-server problem, SIAM Journal on Computing, 1995 24(1): 78–100.

[2] Y. Bartal, Probabilistic approximation of metric space and its algorithmic application, in: Proceedings of 37th Annual IEEE Symposium on Foundations of Computer Science, October 1996, pp. 184–193.

[3] Y. Bartal, On approximating arbitrary metrics by tree metrics, in: Proceedings of 30th Annual ACM Symposium on Theory of Computing, May 1998, pp. 161–168.

[4] Y. Bartal, B. Bollobás, M. Mendel, Ramsey-type theorems for metric spaces with applications to online problems, Journal of Computer and System Sciences, 2006 72(5):890–921.

[5] Y. Bartal, M. Mendel, Randomized k-server algorithms for growth-rate bounded graphs, Journal of Algorithms 55 (2005), no. 2, 192–202.

[6] Y. Bartal, N. Linial, M. Mendel, A. Naor, On metric Ramsey-type phenomena, Annals of Mathematics, 2005, 162(2):643–709.

[7] S. Ben-David, A. Borodin, R. Karp, G. Tardos, A. Wigderson, On the power of randomization in on-line algorithms, Algorithmica 11 (1994) 2–14.

[8] A. Coté, A. Meyerson, L. Poplawski, Randomized k-server on hierarchical binary trees, available at `http://www.cs.ucla.edu/~awm/papers/kserver.pdf`

[9] B. Csaba, S. Lodha, A randomized on-line algorithm for the k-server problem on a line, Random Structures and Algorithms 29 (2006) no. 1, 82–104.

[10] B. Csaba, A. Pluhár, A randomized algorithm for on-line weighted bipartite matching problem, Journal of Scheduling 11 (2008) 449–455.

[11] J. Csirik, Cs. Imreh, J. Noga, S. S. Seiden, G. Woeginger, Buying a constant competitive ratio for paging, in: Proceedings of ESA 2001, LNCS 2161, 2001, pp. 98–108.

[12] A. Fiat, R. M. Karp, M. Luby, L. McGeoch, D. Sleator, N. E. Young, Competitive paging algorithms, Journal of Algorithms 12 (1991) no. 4, 685–699.

[13] J. Fakcharoenphol, S. Rao, K. Talwar, A tight bound on approximating arbitrary metrics by tree metrics, J. Comput. System Sci. 69 (2004), no. 3, 485–497.

[14] E. Koutsoupias, C. Papadimitriou, On the k-server conjecture, Journal of the ACM 42 (1995), no. 5, 971–983.

[15] L. McGeoch, D. Sleator, A strongly competitive randomized paging algorithm, Algorithmica 6 (1991), no. 6, 816–825.

[16] M. Manasse, L. McGeoch, D. Sleator, Competitive algorithms for server problems, Journal of Algorithms 11 (1990), no. 2, 208–230.

[17] S. S. Seiden, A general decomposition theorem for the k-server problem, Information and Computation 174(2) (2002) 193–202.