Theoretical Computer Science 605 (2015) 106–118

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

R–LINE: A better randomized 2-server algorithm on the line [☆]

Lucas Bang [a], Wolfgang Bein [b,*], Lawrence L. Larmore [b]

[a] Department of Computer Science, University of California, Santa Barbara, United States
[b] Department of Computer Science, University of Nevada, Las Vegas, United States

ARTICLE INFO

Article history:
Received 7 April 2015
Received in revised form 26 August 2015
Accepted 31 August 2015
Available online 3 September 2015
Communicated by D.-Z. Du

Keywords:
Online algorithms
Randomized algorithms
Server problem
Algorithm design
T-theory
Game theory

ABSTRACT

A randomized on-line algorithm is given for the 2-server problem on the line, with competitiveness less than 1.901 against the oblivious adversary. This improves the previously best known competitiveness of $\frac{155}{78} \approx 1.987$ for the problem. The algorithm uses a new approach and defines a potential in terms of isolation indices from T-theory.

1. Introduction

In the k-server problem, there are k identical mobile servers in a metric space M. At any time, a point $r \in M$ can be "requested," and must be "served" by moving one of the k servers to the point r. The cost of that service is defined to be the distance the server is moved; for a sequence of requests the goal is to serve the requests at the least possible cost. An *online algorithm* for the server problem decides, at each request, which server to move, but does not know the sequence of future requests. We analyze an online algorithm for the server problem in terms of its *competitive ratio*, which essentially gives the ratio of its cost to the cost of an optimal (offline) algorithm which has knowledge of the entire request sequence before making any decisions. More precisely, we say that an online algorithm $\mathcal{A}$ for the server problem is *C-competitive*, if there is a constant K, such that, given any request sequence σ, $cost_{\mathcal{A}}(\sigma) \leq C \cdot cost_{\mathcal{OPT}}(\sigma) + K$, where $cost_{\mathcal{OPT}}(\sigma)$ is the minimum possible cost of any service of σ. If $\mathcal{A}$ is a randomized online algorithm, we express the inequality in terms of expected cost, *i.e.*, $E[cost_{\mathcal{A}}(\sigma)] \leq C \cdot cost_{\mathcal{OPT}}(\sigma) + K$. In the analysis of an online algorithms (cf. [14]) it is customary to think of the optimal service as performed by an *oblivious adversary*. The optimal cost is then also referred to as the "cost of the adversary," and the movement of the servers in the optimal algorithm as "adversary moves."

The server problem was first proposed by Manasse, McGeoch and Sleator [25] and the problem has been widely studied since then. They also introduced the now well-known *k-server conjecture*, which states that, for each k, there exists an online algorithm for k servers which is k-competitive for any metric space. The conjecture was immediately proved true by the

☆ A preliminary version has been presented at the 10th Workshop on Approximation and Online Algorithms.

* Corresponding author.
E-mail addresses: bang@cs.ucsb.edu (L. Bang), wolfgang.bein@unlv.edu (W. Bein), lawrence.larmore@unlv.edu (L.L. Larmore).

http://dx.doi.org/10.1016/j.tcs.2015.08.043
0304-3975/

same researchers for $k = 2$, but for larger k, the conjecture remains open, although it has been proved for a number of special cases [15,17,24,19,10].

In the randomized case, little is known. Bartal et al. [8] have an asymptotic lower bound, namely that the competitiveness of any randomized online algorithm for an arbitrary metric space is $\Omega(\log k/\log^2 \log k)$. It is conjectured that there is an $O(\log k)$ competitive algorithm for general metric spaces. A recent breakthrough is the algorithm by Bansal et al. [4], which gives a poly-logarithmic competitive algorithm for finite metric spaces. Progress has been made on some other special cases [1,5,20,26].

Surprisingly, no randomized competitive algorithm for the 2-server problem for general spaces is known to have competitiveness less than 2. The classic algorithm RANDOM SLACK [16], a very simple memoryless and trackless randomized algorithm for the 2-server problem, has had the best known competitive ratio of 2 for almost two decades now. However, the best published lower bound for competitiveness of the randomized 2-server problem is $1 + e^{-\frac{1}{2}} \approx 1.6065$ (see [18]). The barrier has been broken for a number of classes of spaces. Bein et al. [11] have shown that there is a randomized algorithm with competitive ratio of at most 1.5897 for all 3-point spaces. The competitiveness is known to be $\frac{3}{2}$ for uniform spaces, and Bein et al. [12] have given a randomized algorithm for crosspolytope spaces [13] with competitive ratio $\frac{19}{12} \approx 1.583$. Crosspolytope spaces generalize uniform spaces, as here all distances are one or two. For the 2-server problem on the line, Bartal et al. give a barely random online algorithm with competitive ratio $\frac{155}{78} \approx 1.987$ [9]; their method is to define a deterministic online algorithm for the $(6, 3)$-server problem with that competitiveness, from which three deterministic online algorithms are defined. The randomized algorithm is simply to pick one of those three at random, each with probability $\frac{1}{3}$, and then use the chosen algorithm for the entire request sequence.

One tool, which has been used implicitly in a number of online algorithms for the server problem, is the tight span construction from T-theory, a branch of discrete mathematics dealing with analysis of finite metric spaces; see the pioneering paper by Isbell [22], as well as [2,21] for introductory papers to the subject. The original motivation for the development of *T-theory*, and one of its most important application areas, is *phylogenetic analysis*, the problem of constructing a *phylogenetic tree* [3,27]. The tight span of a finite metric space provides us with a useful set of invariants of that space. We have found that these parameters are useful for applications to the server problem. In particular, potentials and behaviors of several algorithms, such as EQUIPOISE, RANDOM SLACK, BALANCE SLACK, HANDICAP, and Bartal's algorithm SLACK COVERAGE in Euclidean spaces [7], are most conveniently expressed by using these invariants of the tight span of the set of active points, i.e., points where there is a server or a request.

Our contribution Based on T-theory we give a novel randomized algorithm for the 2-server problem on the line with substantially improved competitiveness. Though our algorithm is inspired by a barely random approach our methodology is different from that of Bartal et al. [9] and makes use of a potential defined in terms of isolation indices from T-theory [2].

To describe the algorithm we define the (m, n)-*server problem*, for $m > n$, to be the variation where there are m mobile servers in the metric space, and each request must be served by at least n of them. In this paper, we give a randomized online algorithm for the $(2n, n)$-server problem on the line, for every $n \geq 3$. This implies a randomized algorithm R–LINE[n] for the 2-server problem on the line, where the distribution on configurations is supported by at most n configurations, each of which has a weight which is an integral multiple of $\frac{1}{n}$. Calculations indicate that the competitiveness of R–LINE[n] is a monotone decreasing function of n, whose limit is less than 1.901.

2. The Algorithm R–LINE

2.1. Preliminaries

Our algorithm, R–LINE, is defined to be a randomized algorithm for the $(2n, n)$-server problem, for $n \geq 3$. We make use of the following two theorems from [9]:

Theorem 2.1. *(See [9].) Given any C-competitive online algorithm for the $(2n, n)$-server problem, we can derive a randomized online algorithm for the 2-server problem that is C-competitive.*

Theorem 2.2. *(See [9].) Any optimal offline strategy for the $(2n, n)$ server problem keeps the servers in two blocks of n each, assuming that the servers are together in two blocks in the initial configuration.*

By Theorem 2.2, without loss of generality we can assume that the adversary is using an optimal 2-server algorithm, but serves with cost equal to n times the distance moved. We will use the notation s_i both to refer to the ith server and its location, when no confusion arises. We assume that $s_1 \leq s_2 \leq \ldots \leq s_{2n-1} \leq s_{2n}$. We refer to the adversary's servers as a_1 and a_2, and assume that $a_1 \leq a_2$. The algorithm thus knows the location of one of the adversary's servers, which we call the *visible* server, and which, by a slight abuse of notation, we also call r. We denote the adversary's other server by a, and refer to it as the *hidden* server, since the algorithm does not know where it is.

Our algorithm uses the notion of isolation indices, which are part of T-theory, see [2]. For $0 \le i \le 2n$ and $0 \le j \le 2$, if $1 \le i + j \le 2n + 1$, we define $\alpha_{i,j}$, the (i, j)th *isolation index* of a configuration, to be the length of the longest interval that has exactly i algorithm servers to the left and exactly j adversary servers to the left. More formally,

$$\alpha_{i,j} = \max \begin{cases} \min\left\{s_{i+1}, a_{j+1}\right\} - \max\left\{s_i, a_j\right\} \\ 0 \end{cases}$$

where we let $s_0 = a_0 = -\infty$ and $s_{2n+1} = a_3 = \infty$ by default. The following lemma, which follows directly from the previous definition, relates distances between server locations to isolation indices.

Lemma 2.3. *For* $1 \le i \le n$*:*

(a) $d(s_i, a_1) = \sum_{j=i}^{2n} \alpha_{j,0} + \sum_{j=1}^{i-1} \left(\alpha_{j,1} + \alpha_{j,2}\right)$
(b) $d(s_{n+i}, a_2) = \sum_{j=n+i}^{2n} \left(\alpha_{j,0} + \alpha_{j,1}\right) + \sum_{j=1}^{n+i-1} \alpha_{j,2}$
(c) $d(s_i, s_{n+i}) = \sum_{j=i}^{n+i-1} \left(\alpha_{j,0} + \alpha_{j,1} + \alpha_{j,2}\right)$

For each $0 \le i \le 2n$ and $0 \le j \le 2$, we define a constant $\eta_{i,j}$, the (i, j)th *isolation index coefficient*. The isolation index coefficients are used to define a suitable potential, which is used in Section 3 to prove competitiveness. We define the potential of a configuration to be

$$\phi = \sum \left\{\eta_{i,j} \cdot \alpha_{i,j} \,:\, (0 \le i \le 2n) \wedge (0 \le j \le 2) \wedge (1 \le i + j \le 2n + 1)\right\}.$$

For each given n, the competitiveness C and the isolation index coefficients $\left\{\alpha_{i,j}\right\}$ must satisfy a system of inequalities given in Section 3. We will first define R–LINE in terms of those constants, and then show that R–LINE is C-competitive if the system of inequalities is satisfied.

2.2. Algorithm description

We define a configuration of servers (R–LINE's as well as the adversary's) to be *satisfying* if at least n of R–LINE's servers are at r. We refer to a satisfying configuration as an S-configuration, and we assume that the initial configuration is an S-configuration.

Every round begins by the adversary choosing a new request point r and moving one of its two servers to r. R–LINE then moves as many of its servers as necessary to r, and the resulting configuration is once again an S-configuration. No R–LINE server will pass another R–LINE server that does not serve. In general, R–LINE deterministically moves zero or more servers to r, and then uses randomization to decide which additional servers to move. R–LINE is *lazy*, meaning that it never moves any server that does not serve the request. We now define R–LINE. Between rounds, the configuration of servers is always an S-configuration. When the adversary makes a request at a point r, R–LINE responds by making a sequence of *moves*, each consisting of the movement of one or more servers to r. Thus, during a round, R–LINE makes at most n moves. Not all configurations can arise during execution of R–LINE; in fact, we define two classes of configurations, D-configurations and R-configurations, such that every intermediate configuration of R–LINE belongs to one of those two classes. If the current configuration is a D-configuration, then R–LINE's next move is to move one or more servers deterministically to r, while if the current configuration is an R-configuration, then R–LINE's next move is to choose, using randomization, a set of servers to move to r. In this case there are always two choices – to move one or more servers from the previous request point to r, completing the round, or to move just one server from the other side, possibly not completing the round.

We now define the classes of configurations. Note that, before the current round began, there must have been n algorithm servers at the previous request point, which we call r'. Without loss of generality, $r' \ne r$.

1. **S-Configuration**: there are n algorithm servers at r.
2. **D-Configuration**: the following two conditions hold.
   (a) There are more than n algorithm servers either strictly to the left or strictly to the right of r; that is, $r > s_{n+1}$ or $r < s_n$.
   (b) If there are fewer than n algorithm servers at r', then there is no algorithm server strictly between r' and r, and furthermore, there are at least n algorithm servers at the points r' and r combined.
3. **R-Configuration**: the following two conditions hold.
   (a) There are exactly n algorithm servers on the same side of r as r', that is, either $r' = s_n < r$ or $r < r' = s_{n+1}$.
   (b) There is no algorithm server strictly between r' and r, and furthermore, there are at least n algorithm servers at the points r' and r combined.

We now give an explicit definition of R–LINE. By symmetry, we can assume, without loss of generality, that $r' < r$. The reader might also consult Fig. 1, where we illustrate R–LINE through a single round in a case where $n = 3$.

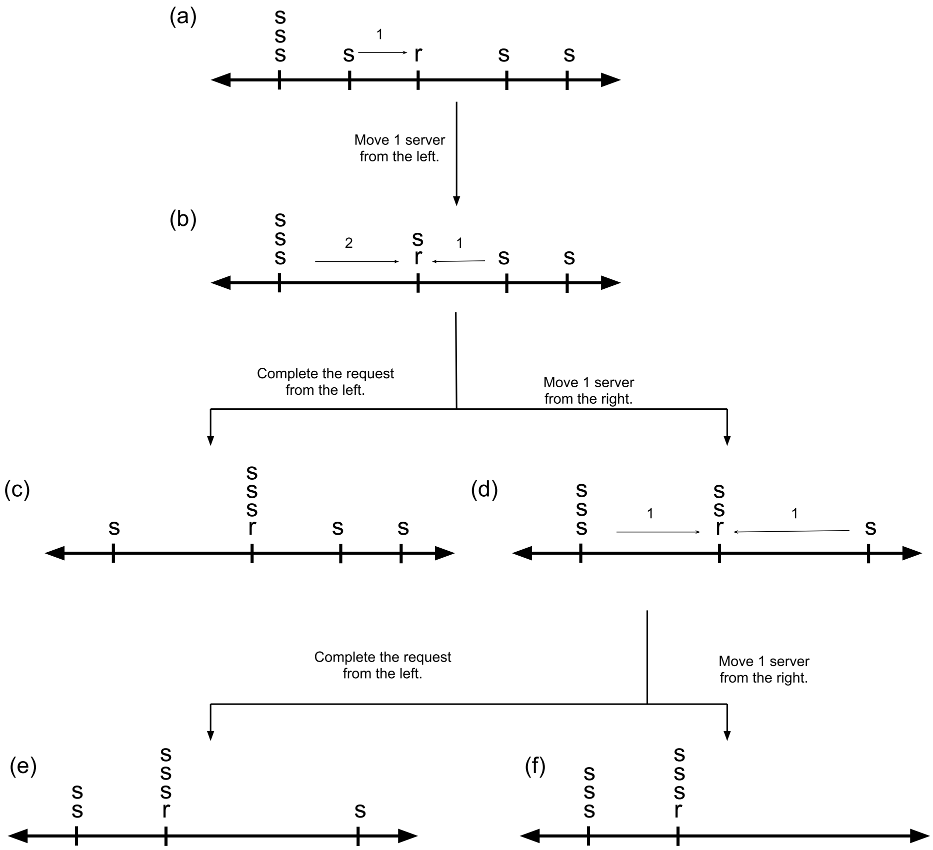

Fig. 1. (a) A D-configuration, where $n = 3$. The request is r, and there are three servers located at $r' < r$. The next move is deterministic. (b) An R-configuration. One server has moved to r from the left. The next move is randomized; either move two servers from the left or one from the right. (c) An S-configuration, after two servers have moved from the left. The round is over. (d) An R-configuration, after one server has moved from the right. The next move is randomized; either move one server from the left or one from the right. (e) An S-configuration, after one server has moved from the right. The round is over. (f) An S-configuration, after one server has moved from the left. The round is over.

1. If the current configuration is a D-configuration, then there are m algorithm servers to the left of r for some $m > n$. Move the servers $s_{n+1}, \dots, s_m$ to r. If the resulting configuration is an S-configuration, the round is over. Otherwise, the resulting configuration is an R-configuration, and proceed to the next step.
2. If the current configuration is an R-configuration, then $r' = s_n < r \le s_{n+1} < s_{2n}$. Let p be the number of algorithm servers at r. Then $s_{n+p+1} > r$. R–LINE executes one of two moves; each move is executed with a probability that is determined by solving a 2-person zero-sum game. We compute those probabilities below. The two choices of move are:
   (a) Move s_{n+p+1} to r.
   (b) Move the servers $s_{p+1} \dots s_n$ to r.
   If the resulting configuration is an S-configuration, the round is over. Otherwise, the resulting configuration is an R-configuration, and repeat this step.

For the randomized step, one of the two choices is selected by using the optimum strategy for a 2-person zero sum game, where R–LINE is the column player, and $\mathcal{A}dv$ is the row player; the choice of the row player is where to place the hidden server. As we show later, we can assume, without loss of generality, that the hidden server is located at either s_n or s_{n+p+1}. Thus, each player has exactly two strategies. Each entry of the payoff matrix is equal to $\Delta\phi + cost = \phi' - \phi + cost$, where ϕ and ϕ' are the potentials before and after the move; and $cost$ is the cost of the move, which is equal to the number of servers moved times distance moved, either $(s_{n+p+1} - r)$ or $(n - p)(r - s_n)$.

The payoff matrix is as follows:

$$G = \begin{array}{|c||c|c|}
\hline
 & \text{Move } s_{n+p+1} & \text{Move } s_{p+1} \dots s_n \\
\hline
a = s_n & \big(\eta_{n+p+1,2} - \eta_{n+p,2} + 1\big)\big(s_{n+p+1} - r\big) & \big(\eta_{p,1} - \eta_{n,1} + n - p\big)(r - s_n) \\
\hline
a = s_{n+p+1} & \big(\eta_{n+p+1,1} - \eta_{n+p,1} + 1\big)\big(s_{n+p+1} - r\big) & \big(\eta_{p,0} - \eta_{n,0} + n - p\big)(r - s_n) \\
\hline
\end{array}$$

3. Design for competitiveness

3.1. A system of inequalities

We now present a system of inequalities, which we denote by $\mathbb{S}$, which suffices for R–LINE to be C-competitive. We will prove, in Theorem 3.1, that $\mathbb{S}$ implies C-competitiveness of R–LINE.

$$\forall\, 0 \le i \le 2n: \ |\eta_{i,1} - \eta_{i,0}| \ \le \ n \cdot C \tag{1}$$

$$\forall\, 1 \le i \le n \text{ and } \forall\, 1 \le j \le 2: \ \eta_{i,j} + 1 \ \le \ \eta_{i-1,j} \tag{2}$$

$$\forall\, 1 \le i \le n \text{ and } \forall\, 1 \le j \le 2: \ \eta_{i-1,j-1} \ \le \ \eta_{i,j-1} + 1 \tag{3}$$

$$\forall\, 1 \le i \le n: \ (\eta_{i-1,1} - \eta_{i,1} + 1)(\eta_{n-i,1} - \eta_{n,1} + i) \ \le \ (\eta_{i-1,0} - \eta_{i,0} + 1)(\eta_{n-i,0} - \eta_{n,0} + i) \tag{4}$$

We note that we can assume that the isolation index coefficients satisfy a symmetry property, namely $\eta_{i,j} = \eta_{2n-i,2-j}$; furthermore, $\eta_{0,0} = \eta_{2n,2} = 0$.

Theorem 3.1. *For any assignment of values to C and $\eta_{i,j}$ for $0 \le i \le 2n$ and $0 \le j \le 2$ that satisfies the system $\mathbb{S}$,* R–LINE *is C-competitive.*

We prove Theorem 3.1 with a sequence of lemmas. We will prove that if the system of inequalities $\mathbb{S}$ is satisfied, then the following properties hold. We write $\Delta\phi = \phi' - \phi$, where ϕ is the potential before the move and ϕ' is the potential after the move.

1. For any move by the adversary, $\Delta\phi \le C \cdot cost_{\mathcal{A}dv}$. (Recall that the adversary pays n times the distance moved.)
2. For any deterministic move by R–LINE, $\Delta\phi + cost \le 0$.
3. We may assume the adversary's hidden server is at one of at most two possible locations during a given round, namely at the closest algorithm server to either the left or the right of r.
4. For any randomized move by R–LINE, $\mathrm{E}[\Delta\phi + cost] \le 0$.

We say that a move is *simple* if the move consists of moving a single server (either an algorithm or an adversary server) across an interval, and there is no other server (of either type) located strictly between the end points of that interval. We also refer to a simple move as a *step*; in general, every movement of servers is a concatenation of steps.

Lemma 3.2. *If $\mathbb{S}$ holds, then Property 1 holds.*

Proof. By the symmetry of the $\eta_{i,j}$, inequality (1) implies that $|\eta_{i,j} - \eta_{i,j-1}| \le n \cdot C$ for $j = 1, 2$. Without loss of generality the move is simple, since every move which is not simple is the concatenation of steps. Without loss of generality, the adversary server a_j moves to the right, from x to y, where $x < y$. Since the move is simple, $s_i \le x$ and $y \le s_{i+1}$ for some $0 \le i \le 2n$,. (Recall the default values $s_0 = -\infty$ and $s_{2n+1} = \infty$.) Thus, $\alpha_{i,j}$ decreases by $y - x$ and $\alpha_{i,j-1}$ increases by $y - x$. The cost to the adversary of this move is $n(y - x)$. By definition of the potential, $\Delta\phi = \big(\eta_{i,j} - \eta_{i,j-1}\big)(y - x) \le n \cdot C \cdot (y - x) \le C \cdot cost_{\mathcal{A}dv}$. □

Lemma 3.3. *If $\mathbb{S}$ holds, then Property 2 holds.*

Proof. For convenience, we assume that $r < r' = s_{n+1}$. There are exactly m algorithm servers to the right of r, for some $m > n$. Servers $s_{2n-m+1} \ldots s_n$ move to r. The move is the concatenation of steps, and it suffices to show that $\Delta\phi \ge cost_{\text{R–LINE}}$ for each of those steps.

Fix one step. During the step, s_i moves from x to y, where $y < x$, for some $2n - m + 1 \le i \le n$. The algorithm cost of the step is $x - y$. Pick the maximum j such that $a_j \le y$. Since $r \le y$, j is either 1 or 2. The move causes $\alpha_{i,j}$ to decrease by $x - y$ and $\alpha_{i-1,j}$ to increase by the same amount. By inequality (1), and the definition of the potential: $\Delta\phi + cost_{\text{R–LINE}} = (x - y)\big(\eta_{i,j} - \eta_{i-1,j} + 1\big) \le 0$. □

Lemma 3.4. *If $1 \le i \le 2n$ and $j = 1, 2$, then $\eta_{i,j} + \eta_{i-1,j-1} \le \eta_{i,j-1} + \eta_{i-1,j}$.*

Proof. Suppose $i \le n$. Then $1 + \eta_{i,j} \le \eta_{i-1,j}$ by (2), while $-1 + \eta_{i-1,j-1} \le \eta_{i,j-1}$ by (3). Adding the two inequalities, we obtain the result.

If $i > n$, then $\eta_{2n-i+1,3-j} + \eta_{2n-i,2-j} \le \eta_{2n-i+1,2-j} + \eta_{2n-i,3-j}$ by the previous case. By symmetry, we are done. □

Lemma 3.5. *If $\mathbb{S}$ holds, then Property 3 holds.*

Proof. Since a could be any point on the line, the payoff matrix of the game has infinitely many rows. We need to prove that just two of those rows, namely $a = s_n$ and $a = s_{n+p+1}$, dominate the others.

By batching the row strategies, we illustrate the $\infty \times 2$ payoff matrix below.

		Move s_{n+p+1}	Move $s_{p+1} \dots s_n$
I	$a \le s_n$	$(\eta_{n+p+1,2} - \eta_{n+p,2} + 1)(s_{n+p+1} - r)$	$(\eta_{p,1} - \eta_{n,1} + n - p)(r - s_n)$
II	$s_n \le a \le r$	$(\eta_{n+p+1,2} - \eta_{n+p,2} + 1)(s_{n+p+1} - r)$	$(\eta_{p,1} - \eta_{n,1} + n - p)(r - a)$ $+$ $(\eta_{p,0} - \eta_{n,0} + n - p)(a - s_n)$
III	$r \le a \le s_{n+p+1}$	$(\eta_{n+p+1,2} - \eta_{n+p,2} + 1)(s_{n+p+1} - a)$ $+$ $(\eta_{n+p+1,1} - \eta_{n+p,1} + 1)(a - r)$	$(\eta_{p,0} - \eta_{n,0} + n - p)(r - s_n)$
IV	$a \ge s_{n+p+1}$	$(\eta_{n+p+1,1} - \eta_{n+p,1} + 1)(s_{n+p+1} - r)$	$(\eta_{p,0} - \eta_{n,0} + n - p)(r - s_n)$

The row strategy $a = s_n$ trivially dominates all row strategies in Batch I. It also dominates all row strategies in Batch II, because

$$\begin{aligned}
\eta_{p,1} - \eta_{n,1} &= \sum_{i=p+1}^{n} (\eta_{i-1,1} - \eta_{i,1}) \\
&\ge \sum_{i=p+1}^{n} (\eta_{i-1,0} - \eta_{i,0}) \quad \text{by Lemma 3.4} \\
&= \eta_{p,0} - \eta_{n,0}.
\end{aligned}$$

The row strategy $a = s_{n+p+1}$ trivially dominates all row stages in Batch IV. It also dominates all row strategies in Batch III, because $\eta_{n+p+1,1} - \eta_{n+p,1} \ge \eta_{n+p+1,2} - \eta_{n+p,2}$, which we can similarly prove using Lemma 3.4. □

We make use of a standard game theory lemma taken from [6]. To this end we remind the reader that a *saddle point* of a zero-sum game is defined to be an entry $a_{i,j}$ of the payoff matrix that is both a maximum of its row and a minimum of its column. If a game has a saddle point $a_{i,j}$, then the value the game is the value of the saddle point, and it is optimum for the row player to always play the ith row, and for the column player to always play the jth column.

Lemma 3.6. *(See [6].) Suppose* $A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$ *is the payoff matrix for a 2-person zero sum game G, and there is no saddle point. Then*

$$v(G) = \frac{\det A}{a_{11} - a_{12} - a_{21} + a_{22}}.$$

Furthermore, the optimum strategy for the row player is:

Play row 1 with probability $\frac{a_{22} - a_{21}}{a_{11} - a_{12} - a_{21} + a_{22}}$
Play row 2 with probability $\frac{a_{11} - a_{12}}{a_{11} - a_{12} - a_{21} + a_{22}}$

While the optimum strategy for the column player is:

Play column 1 with probability $\frac{a_{22} - a_{12}}{a_{11} - a_{12} - a_{21} + a_{22}}$
Play column 2 with probability $\frac{a_{11} - a_{21}}{a_{11} - a_{12} - a_{21} + a_{22}}$

Lemma 3.7. *If $\mathbb{S}$ holds, then Property 4 holds.*

Proof. Consider the 2×2 payoff matrix G of Section 2.2. By $\mathbb{S}$, the upper left and lower right entries of G are negative, while the upper right and lower left entries are positive. By Lemma 3.6, the value of our game is

$$\frac{\det(G)}{(\eta_{n+p+1,2} + \eta_{n+p+1} - \eta_{n+p,2} - \eta_{n+p+1,1}) \cdot (s_{n+p+1} - r) + (\eta_{p,0} + \eta_{n,1} - \eta_{n,0} - \eta_{p,1}) \cdot (r - s_n)}.$$

The numerator is non-negative by the inequalities of $\mathbb{S}$ labeled (4). The denominator is negative, which we can prove by combining inequalities of $\mathbb{S}$ labeled (2) and (3). Thus, $E[\Delta\phi + cost_{\text{R-LINE}}] = v(G) \le 0$ as claimed. $\square$

Theorem 3.1 follows immediately from Lemmas 3.2, 3.3, 3.5, and 3.7.

3.2. The Coppersmith Doyle Raghavan Snir potential

Coppersmith et al. [19] define a potential Φ_{CDRS}, which we call the CDRS potential, for the k-server problem on an arbitrary metric space (M, d). Restricting this definition to the special case that $M = \mathbb{R}$ and $k = 2$, we have:

$$\begin{aligned}\Phi_{CDRS} &= 2 \cdot d(s_1, a_1) + 2 \cdot d(s_2, a_2) + d(s_1, s_2)\\ &= 2 \cdot |s_1 - a_1| + 2 \cdot |s_2 - a_2| + (s_2 - s_1)\end{aligned}$$

Using Lemma 2.3 we can write Φ_{CDRS} in terms of T-theory.

Lemma 3.8. *For 2-server problem on the line, $\Phi_{CDRS} = \sum \zeta_{i,j} \cdot \alpha_{i,j}$ where $\zeta_{1,0} = 3$, $\zeta_{2,0} = 4$, $\zeta_{0,1} = 2$, $\zeta_{1,1} = 1$, $\zeta_{2,1} = 2$, $\zeta_{0,2} = 4$, $\zeta_{1,2} = 3$.*

Proof. Lemma 2.3 for $n = 1$ implies:

$$\begin{aligned}d(s_1, a_1) &= \alpha_{1,0} + \alpha_{2,0} + \alpha_{0,1} + \alpha_{0,2}\\ d(s_2, a_2) &= \alpha_{1,2} + \alpha_{2,0} + \alpha_{2,1} + \alpha_{0,2}\\ d(s_1, s_2) &= \alpha_{1,0} + \alpha_{1,1} + \alpha_{1,2}\end{aligned}$$

Substituting these values, we obtain:

$$\begin{aligned}\Phi_{CDRS} &= 2d(s_1, a_1) + 2d(s_2, a_2) + d(s_1, s_2)\\ &= 2\big(\alpha_{1,0} + \alpha_{2,0} + \alpha_{0,1} + \alpha_{0,2}\big) + 2\big(\alpha_{1,2} + \alpha_{2,0} + \alpha_{2,1} + \alpha_{0,2}\big) + \alpha_{1,0} + \alpha_{1,1} + \alpha_{1,2}\\ &= 3\alpha_{1,0} + 4\alpha_{2,0} + 2\alpha_{0,1} + \alpha_{1,1} + 2\alpha_{2,1} + 4\alpha_{0,2} + 3\alpha_{1,2} \qquad \square\end{aligned}$$

Next we rewrite the CDRS potential for the $(2n, n)$ server problem in terms of T-theory. Let $s_1, s_2, \ldots, s_{2n}$ be the algorithm servers and a_1, a_2 the adversary servers. Such a configuration γ of servers is represented by ordered $2n+2$-tuple, consisting of points s_i for $1 \le i \le n$, a_1, and a_2, where $s_1 \le \cdots \le s_{2n}$ and $a_1 \le a_2$. We decompose γ into configurations γ_i, which consist of the 4-tuple s_i, s_{n+i}, a_1, a_2. Then $\Phi_{CDRS}(\gamma_i)$ is defined as above, and $\Phi_{CDRS}(\gamma) := \frac{1}{n}\sum_{i=1}^{n} \Phi_{CDRS}(\gamma_i)$; hence

Remark 3.9.

$$\begin{aligned}\Phi_{CDRS}(\gamma) &= \frac{1}{n}\sum_{i=1}^{n} \Phi_{CDRS}(\gamma_i)\\ &= \frac{1}{n}\sum_{i=1}^{n}\Big(2 \cdot d(s_i, a_1) + 2 \cdot d(s_{n+i}, a_2) + d(s_i, s_{n+i})\Big)\\ &= \frac{1}{n}\sum_{i=1}^{n}\Big(2 \cdot |s_i - a_1| + 2 \cdot |s_{n+i} - a_2| + s_{n+i} - s_i\Big)\end{aligned}$$

Lemma 3.10. *For the $(2n, n)$-server problem on the line, $\Phi_{CDRS} = \frac{1}{n}\sum \zeta_{i,j} \cdot \alpha_{i,j}$ where*

$$\begin{aligned}\zeta_{i,0} = \zeta_{2n-i,2} &= 3i\\ \zeta_{i,1} = \zeta_{2n-i,1} &= (2n - i)\\ \zeta_{i,2} = \zeta_{2n-i,0} &= (4n - i)\end{aligned}$$

for all $0 \le i \le n$.

Proof. Use Remark 3.9 to write Φ_{CDRS} in terms of distances between servers, then replace each distance by a summation of isolation indices, using the formulas given by Lemma 2.3. Finally, gather the terms, and observe that the coefficient of each $\alpha_{i,j}$ is correct. $\square$

The potential Φ_{CDRS} implies 2-competitiveness:

Remark 3.11. With the values of $\zeta_{i,j}$ taken in Φ_{CDRS} the algorithm R–LINE is a 2-competitive algorithm for the $(2n, n)$-server problem on the line.

Proof. The reader may verify inequalities (1), (2), (3) and (4) of Section 3.1. The Remark then follows immediately from Theorem 3.1. □

3.3. Improving the CDRS potential

We need to find a solution to the system $\mathbb{S}$ (inequalities (1), (2), (3) and (4) of Section 3.1) for which C is smaller than 2, preferably as small as possible. To this end we tweak the CDRS potential by defining values $\eta_{i,j} := \zeta_{i,j} - \delta_{i,j}$. Furthermore we make the following assumptions:

1. Inequalities (1) are tight for $i = 0$ and for $\forall\ n \le i \le 2n$.
2. Inequalities (2) are tight for $\forall\ 1 \le i \le n$ and $\forall 1 \le j \le 2$.
3. Inequalities (3) are tight for $\forall\ 1 \le i \le n$ when $j = 2$.
4. Inequalities (4) are tight for $\forall\ 1 \le i \le n - 1$.

Lemma 3.12. *Assumptions 1 through 4 imply that $\delta_{i,j}$ is entirely determined by parameters $\delta, \delta_1, \dots, \delta_{n-1}$. Specifically,*

$$\eta_{i,j} = \begin{cases} \zeta_{i,j} - \delta & \text{for all } 0 \le i \le 2n \text{ when } j = 1 \\ \zeta_{i,j} - 2\delta & \text{for all } n \le i \le 2n \text{ when } j = 0 \text{ and for all } 0 \le i \le n \text{ when } j = 2 \\ \zeta_{i,j} - \delta_i & \text{for all } 1 \le i \le n-1 \text{ when } j = 0 \\ \zeta_{2n-i,j} - \delta_i & \text{for all } 1 \le i \le n-1 \text{ when } j = 2 \end{cases}$$

Furthermore, $C = 2 - \frac{1}{n}\delta$.

Proof. Set $\delta := \delta_{0,1}$ and let

$$\delta_i := \delta_{i,0} \text{ for } 1 \le i \le n.$$

By symmetry

$$\delta_{2n-i,2} = \delta_i \text{ for } 1 \le i \le n.$$

Using Assumption 2, with $j = 1$, we derive

$$\delta = \delta_{0,1} = \delta_{1,1} = \cdots = \delta_{n,1}$$

and again symmetry implies

$$\delta = \delta_{n,1} = \delta_{1,1} = \cdots = \delta_{2n,1}.$$

Assumption 1, for $i = n$ yields

$$\delta_{n,0} = 2\delta.$$

Applying Assumption 1 for $i = n, \dots, 2n$, yields

$$\delta_{n,0} = \delta_{n+1,0} = \cdots = \delta_{2n-1,0} = \delta_{2n,0} = 2\delta.$$

Symmetry implies

$$\delta_{0,2} = \delta_{1,2} = \cdots = \delta_{n-1,2} = \delta_{n,2} = 2\delta.$$

Finally, Assumption 1, for $i = 0$, implies $C = 2 - \frac{1}{n}\delta$. □

3.4. An improved CDRS potential in the case $n = 3$

For $n = 3$ the competitiveness of R–LINE can be calculated in closed form. Table 1 shows the construction of Lemma 3.12 in this case. Furthermore, the system $\mathbb{S}$ reduces to the system $\mathbb{S}'$ below. The variables of $\mathbb{S}'$ are $\eta_{i,j}$ for $0 \le i \le 6$ and $0 \le j \le 2$, and C, where we fix $\eta_{0,0} = 0$, and we assume symmetry, *i.e.*, $\eta_{i,j} = \eta_{6-i,2-j}$.

Table 1
The "tweaked" CDRS potential for $n = 3$ in terms of δ, δ_1 and δ_2.

$\eta_{i,j} =$

	0	1	2
0	0	$6-\delta$	$12-2\delta$
1	$3-\delta_1$	$5-\delta$	$11-2\delta$
2	$6-\delta_2$	$4-\delta$	$10-2\delta$
3	$9-2\delta$	$3-\delta$	$9-2\delta$
4	$10-2\delta$	$4-\delta$	$6-\delta_2$
5	$11-2\delta$	$5-\delta$	$3-\delta_1$
6	$12-2\delta$	$6-\delta$	0

System $\mathbb{S}'$:

$$\begin{aligned}
\forall\, 0 \le i \le 6 \text{ and } \forall\, 0 \le j \le 1 :\ \left|\eta_{i,0} - \eta_{i,1}\right| &\le 3C \\
\forall\, 0 \le i \le 3 \text{ and } \forall\, 1 \le j \le 2 :\ \eta_{i,j} + 1 &\le \eta_{i-1,j} \\
\forall\, 0 \le i \le 3 \text{ and } \forall\, 0 \le j \le 1 :\ \eta_{i-1,j} &\le \eta_{i,j} + 1 \\
\forall\, 1 \le i \le 3 : (\eta_{i-1,1} - \eta_{i,1} + 1)(\eta_{3-i,1} - \eta_{3,1} + i) &\le (\eta_{i-1,0} - \eta_{i,0} + 1)(\eta_{3-i,0} - \eta_{3,0} + i)
\end{aligned}$$

The values of the $\eta_{i,j}$, obtained from the first three inequalities of $\mathbb{S}'$, assuming Assumptions 1, 2, and 3, are given in Table 1. We now expand the last inequality, changing the first two inequalities to equalities by applying Assumption 4, obtaining the system $\mathbb{A}$ given below.

System $\mathbb{A}$:

$$\begin{aligned}
\big(\eta_{0,1} - \eta_{1,1} + 1\big)\big(\eta_{2,1} - \eta_{3,1} + 1\big) &= \big(\eta_{0,0} - \eta_{1,0} + 1\big)\big(\eta_{2,0} - \eta_{3,0} + 1\big) \\
\big(\eta_{1,1} - \eta_{2,1} + 1\big)\big(\eta_{1,1} - \eta_{3,1} + 2\big) &= \big(\eta_{1,0} - \eta_{2,0} + 1\big)\big(\eta_{1,0} - \eta_{3,0} + 2\big) \\
\big(\eta_{2,1} - \eta_{3,1} + 1\big)\big(\eta_{0,1} - \eta_{3,1} + 3\big) &\le \big(\eta_{2,0} - \eta_{3,0} + 1\big)\big(\eta_{0,0} - \eta_{3,0} + 3\big)
\end{aligned}$$

We now rewrite the $\eta_{i,j}$ in terms of the variables δ, δ_1, δ_2, and C. Recall that $\eta_{0,0} = 0$.

$$\begin{aligned}
(6 - \delta - 5 + \delta + 1)(4 - \delta - 3 + \delta + 1) &= (-3 + \delta_1 + 1)(6 - \delta_2 - 9 + 2\delta + 1) \\
(5 - \delta - 4 + \delta + 1)(5 - \delta - 3 + \delta + 2) &= (3 - \delta_1 - 6 + \delta_2 + 1)(3 - \delta_1 - 9 + 2\delta + 2) \\
(4 - \delta - 3 + \delta + 1)(6 - \delta - 3 + \delta + 3) &\le (6 - \delta_2 - 9 + 2\delta + 1)(-9 + 2\delta + 3)
\end{aligned}$$

Simplifying, and reversing left and right for the sake of appearance, we obtain

System $\mathbb{B}$:

$$\begin{aligned}
(2 - \delta_1)(2 - 2\delta + \delta_2) &= 4 \\
(2 - \delta_2 + \delta_1)(4 - 2\delta + \delta_1) &= 8 \\
(2 - 2\delta + \delta_2)(6 - 2\delta) &\ge 12
\end{aligned}$$

The competitiveness of our algorithm will be $C = 2 - \frac{\delta}{3}$, and thus we wish to find a solution to $\mathbb{B}$ which maximizes δ. In that calculation, we will ignore the last inequality since it is slack, although our solution must satisfy it. We replace the first two equations of $\mathbb{B}$ by an equivalent system involving the variables x, y, and δ, by using the substitution:

$$\begin{aligned}
\delta_1 &= 2x + 2\delta - 4 \\
\delta_2 &= 2y + 2\delta - 2
\end{aligned}$$

We then obtain the system $\mathbb{D}$:

$$\begin{aligned}
y(3 - x - \delta) &= 1 \\
x(x - y) &= 2
\end{aligned}$$

The set of solutions of $\mathbb{D}$ is 1-dimensional, and we can express both y and δ as functions of x:

$$y = x - \frac{2}{x} = \frac{x^2-2}{x}$$
$$\delta = 3 - x - \frac{1}{y} = 3 - x - \frac{x}{x^2-2}$$

The maximum value of δ will be achieved at the point that the derivative of δ with respect to x is zero.

$$\frac{d\delta}{dx} = -1 - \frac{\left(x^2-2\right) - 2x^2}{\left(x^2-2\right)^2} = 0$$

which yields

$$x = \pm\sqrt{\frac{5 \pm \sqrt{17}}{2}}$$

Of the four choices of x, the only one that fits the constraints is the largest:

$$x = \sqrt{\frac{5 + \sqrt{17}}{2}}$$

Routine calculations yield all constants:

$$\delta = 3 - \frac{\sqrt{71+17\sqrt{17}}}{4} \approx 0.030437788626282103$$
$$\delta_1 = 2 - \frac{\sqrt{7+\sqrt{17}}}{2} \approx 0.332433987392278141$$
$$\delta_2 = 4 - \frac{\sqrt{79-7\sqrt{17}}}{2} \approx 0.459581217543735487$$
$$C = 1 + \frac{\sqrt{71+17\sqrt{17}}}{12} \approx 1.989854070457905966$$

As it turns out, the optimum values of δ, δ_1, δ_2, as well as the competitiveness C, all lie in the splitting field of the fourth degree polynomial $x^4 - 5x^2 + 2$.

The analytic methods used to find the above constants do not easily generalize and so we utilize approximation methods to determine the values of the constants for larger values of n. It is worth noting that Bartal et al. provided an algorithm for the $(6,3)$-server problem in [9] with competitiveness $\frac{155}{78} \approx 1.9871795$ which is better than the result shown here. However in the next section we show that by using larger values of n we achieve a better upper bound on the competitiveness of the 2-server problem.

3.5. Calculating an improved CDRS potential in general

In this section we describe a method to calculate an improved Coppersmith Doyle Raghavan Snir potential for general n. In order to more easily compute the values of the coefficients, we make a change of variables. Define $\zeta_{i,0} - \delta_i = \eta_{i,0}$ for all $0 \le i \le n$. Thus, $\delta_n = 2\delta$. Let $\epsilon_i = \delta_i - 2\delta$. Thus $\epsilon_0 = -2\delta$. Our problem is now to find a solution to the following system of equations which maximizes δ.

For all $0 < i < n : (2i + \epsilon_{n-i})(2 - \epsilon_i + \epsilon_{i-1}) = 4i$

and

Verify that $(2n - 2\delta)(2 + \epsilon_{n-1}) \ge 4n$.

We approximate the value of C numerically, using a program to find a solution to $\mathbb{S}$. Our program computes a function f, where $\delta = f(\epsilon_{\lfloor n/2 \rfloor})$. To find the maximum value of δ, we assume that f is bimodal,[1] that is, there is some $x^* > 0$ for which $f(x)$ is maximum, and that $f(x)$ is monotone increasing for $0 < x < x^*$ and monotone decreasing for $x > x^*$. We then use a divide and conquer algorithm similar to binary search to find $f(x^*)$. Our program is as follows:

[1] However, the validity of the program does not depend on the bimodality of f.

1. Guess $\epsilon_{\lfloor n/2 \rfloor}$, using our search algorithm.
2. If n is odd, then solve the following equation for $\epsilon_{(n+1)/2}$:

$$(n+1+\epsilon_{(n-1)/2})(2-\epsilon_{(n+1)/2}+\epsilon_{(n-1)/2}).$$

3. For all $0 < i < \lfloor \frac{n}{2} \rfloor$ in decreasing order:
   (a) Solve the following equation for ϵ_i:

$$(2(i+1)+\epsilon_{n-i-1})(2-\epsilon_{i+1}+\epsilon_i) = 4(i+1).$$

   (b) Solve the following equation for ϵ_{n-i}:

$$(2(n-i)+\epsilon_i)(2-\epsilon_{n-i}+\epsilon_{n-i-1}) = 4(n-i).$$

4. Solve the following equation for δ:

$$(2+\epsilon_{n-1})(2-\epsilon_1-2\delta) = 4.$$

5. Verify the following inequality:

$$(2n-2\delta)(2+\epsilon_{n-1}) \geq 4n.$$

6. If our search interval is small enough, proceed to the last step. Otherwise, return to step 1.
7. $C \leftarrow \frac{2n-\delta}{n}$.

Our calculations show that $C \approx 1.90098671$ for $n = 2000$. In the next section we show a limiting competitiveness of

$$\lim_{n\to\infty} c \approx 1.9007617.$$

3.6. The continuous case

We now consider the problem in the limit. In particular, for any fixed $0 \leq t \leq 1$, let

$$h(t) = \lim_{n\to\infty} \epsilon_{n,[t\cdot n]}/n$$

where $[x]$ denotes the nearest integer to x.

In the limiting case, the differential-difference equation then becomes

$$(2t + h(1-t)) \cdot (2 - h'(t)) = 4t$$

for $0 \leq t \leq 1$.

We can substitute variables to make the equation look more symmetric:

$$(x+1+f(-x)) \cdot (1-f'(x)) = x+1$$

for $-1 \leq x \leq 1$. Let

$$g(x) = f(x) - x + 1.$$

Then

$$g(-x) = f(-x) + x + 1 \text{ and } g'(x) = f'(x) - 1.$$

Substitution yields

$$g(-x) \cdot g'(x) = -x - 1 \text{ from which we obtain } g(x) \cdot g'(-x) = x - 1.$$

Let $F(x) = g(x)g(-x)$, then

$$F'(x) = g'(x)g(-x) - g(x)g'(-x) = -2x. \qquad \text{(i)}$$

Taking the anti-derivative, we obtain

$$F(x) = -x^2 + D.$$

In the range of interest, $F(x)$ must be positive, hence $D = K^2$ for some $K > 1$. We write

$$F(x) = -x^2 + K^2,$$

from which we obtain

$$\frac{g'(x)}{g(x)} = \frac{g'(x)\cdot g(-x)}{g(x)\cdot g(-x)} = \frac{-x-1}{F(x)} = \frac{-x-1}{K^2-x^2} = \frac{A}{K+x} + \frac{B}{K-x} \qquad \text{(ii)}$$

for some constants A and B, where $(A+B)K = -1$ and $A - B = 1$. The solution is that $A = \frac{K-1}{2K}$ and $B = -\frac{K+1}{2K}$. Note that the left side of (ii) is the derivative of $\ln g(x)$. Taking the anti-derivative of both ends of (ii), we obtain

$$\ln g(x) = A \ln(K + x) + B \ln(K - x) + \ln |E|$$

for some constant E. Hence

$$g(x) = E \cdot (K + x)^{\frac{K-1}{2K}} \cdot (K - x)^{\frac{K+1}{2K}}.$$

Substituting back in (i), we can show that $E = \pm 1$. In the range of interest, $g(x)$ cannot be negative. We thus obtain the one-parameter family

$$f(x) = (K + x)^{\frac{K-1}{2K}} \cdot (K - x)^{\frac{K+1}{2K}} + x - 1.$$

The resulting competitiveness is

$$C = 2 + \frac{f(-1)}{2}.$$

The minimum value of C is obtained by finding that K for which $\ln(K + 1) - \ln(K - 1) = 2K$. By numeric computation, we obtain $K \approx 1.199678640258$, and thus $C \approx 1.90076169687385$.

4. Conclusions and future work

Though not claimed in this paper, our preliminary investigation indicates that R–LINE – unlike the Bartal et al. algorithm – generalizes to trees. We also suspect that R–LINE generalizes naturally to all split decomposable spaces, including the Manhattan plane.[2]

Our real goal is to obtain a “better than 2” result for general spaces. R–LINE[n] uses a potential defined in terms of isolation indices. If a metric space is split-decomposable, then isolation indices should be sufficient, allowing us to (hopefully) generalize R–LINE[n] to those cases.

However, when we generalize to an arbitrary metric spaces, we discover that a configuration might not be described in terms of isolation indices. Isolation indices are a special case of a more general set of invariants, *coherency indices* [23]. The potential needed for the general case will make use of coherency indices that are not isolation indices. For $n = 3$, the potential is defined using the tight span of a set of five points: the last request point, where we have three servers, as well as the locations of our other three servers and the adversary’s hidden server. Metrics on five points are well-understood; there are three generic cases. Thus, we conjecture that we can extend R–LINE[3] to all spaces, obtaining (hopefully) a competitiveness of $1 + \frac{\sqrt{71+17\sqrt{17}}}{12} \approx 1.9898540704579059966$.

For $n = 4$, the potential is defined for a set of six points. Sturmfels and Yu [28] have determined that there are 339 generic cases of six point metric spaces. We suspect that any result for $n \geq 4$ will require machine computation.

Theorem 2.1 extends to $k \geq 3$ for the line and the circle, though not for general metric spaces. This fact prompts the obvious question: can we use techniques similar to those presented in this paper, to find better competitiveness for the randomized k-server problem on the line or the circle?

Acknowledgements

The work of author Wolfgang Bein was supported by National Science Foundation grant IIA 1427584 and by sabbatical support from the University of Nevada, Las Vegas.

References

[1] Anna Adamaszek, Artur Czumaj, Matthias Englert, Harald Räcke, An O(log k)-competitive algorithm for generalized caching, in: Proceedings of the Twenty-Third Annual ACM–SIAM Symposium on Discrete Algorithms, SODA 2012, SIAM, 2012, pp. 1681–1689.
[2] Hans-Jürgen Bandelt, Andreas Dress, A canonical decomposition theory for metrics on a finite set, Adv. Math. 92 (1) (1992) 47–105.
[3] Hans-Jürgen Bandelt, Andreas Dress, Split decomposition: a new and useful approach to phylogenetic analysis of distance data, Mol. Phylogenet. Evol. 1 (3) (1992) 242–252.
[4] Nikhil Bansal, Niv Buchbinder, Aleksander Madry, Joseph Naor, A polylogarithmic-competitive algorithm for the k-server problem, in: Proceedings of the 52nd Annual Symposium on Foundations of Computer Science, FOCS 2015, IEEE Computer Society, 2011, 10 pages.
[5] Nikhil Bansal, Niv Buchbinder, Joseph Naor, A primal-dual randomized algorithm for weighted paging, J. ACM 59 (4) (2012) 19, 24 pages.
[6] Emmanuel N. Barron, Game Theory: An Introduction, John Wiley & Sons, New Jersey, 2008.
[7] Yair Bartal, A fast memoryless 2-server algorithm in Euclidean spaces, unpublished manuscript, 1994.
[8] Yair Bartal, Béla Bollobás, Manor Mendel, Ramsey-type theorems for metric spaces with applications to online problems, J. Comput. System Sci. 72 (5) (2006) 890–921.
[9] Yair Bartal, Marek Chrobak, Lawrence L. Larmore, A randomized algorithm for two servers on the line, Inform. Comput. 158 (1) (2000) 53–69.

[2] An infinite metric space is said to be split decomposable if all finite subspaces are split decomposable.

[10] Wolfgang Bein, Marek Chrobak, Lawrence L. Larmore, The 3-server problem in the plane, Theoret. Comput. Sci. 287 (1) (2002) 387–391.
[11] Wolfgang Bein, Kazuo Iwama, Jun Kawahara, Randomized competitive analysis for two-server problems, Algorithms 1 (1) (2008) 30–42.
[12] Wolfgang Bein, Kazuo Iwama, Jun Kawahara, Lawrence L. Larmore, James A. Oravec, A randomized algorithm for two servers in cross polytope spaces, Theoret. Comput. Sci. 412 (2) (2011) 563–572.
[13] Wolfgang Bein, Lawrence L. Larmore, John Noga, Rüdiger Reischuk, Knowledge state algorithms, Algorithmica 60 (3) (2011) 653–678.
[14] Allan Borodin, Ran El-Yaniv, Online Computation and Competitive Analysis, Cambridge University Press, 1998.
[15] Marek Chrobak, Howard Karloff, Tom H. Payne, Sundar Vishwanathan, New results on server problems, SIAM J. Discrete Math. 4 (2) (1991) 172–181.
[16] Chrobak Marek, Lawrence L. Larmore, On fast algorithms for two servers, J. Algorithms 12 (4) (1991) 607–614.
[17] Chrobak Marek, Lawrence L. Larmore, An optimal online algorithm for k servers on trees, SIAM J. Comput. 20 (1) (1991) 144–148.
[18] Marek Chrobak, Lawrence L. Larmore, Carsten Lund, Nick Reingold, A better lower bound on the competitive ratio of the randomized 2-server problem, Inform. Process. Lett. 63 (2) (1997) 79–83.
[19] Don Coppersmith, Peter G. Doyle, Prabhakar Raghavan, Marc Snir, Random walks on weighted graphs and applications to on-line algorithms, J. ACM 40 (3) (1993) 421–453.
[20] Béla Csaba, Sachin Lodha, A randomized on-line algorithm for the k-server problem on a line, Random Structures Algorithms 29 (1) (2006) 82–104.
[21] Andreas W.M. Dress, Vincent Moulton, Werner Terhalle, T-theory: an overview, European J. Combin. 17 (2–3) (1996) 161–175.
[22] John R. Isbell, Six theorems about metric spaces, Comment. Math. Helv. 39 (1) (1964) 65–74.
[23] Jacobus Koolen, Vincent Moulton, Udo Tönges, The coherency index, Discrete Math. 192 (103) (1998) 205–222.
[24] Elias Koutsoupias, Christos Papadimitriou, Beyond competitive analysis, SIAM J. Comput. 30 (1) (2000) 300–317.
[25] Mark Manasse, Lyle A. McGeoch, Daniel Sleator, Competitive algorithms for server problems, J. Algorithms 11 (1990) 208–230.
[26] Judit Nagy-György, Randomized algorithm for the k-server problem on decomposable spaces, J. Discrete Algorithms 7 (4) (2009) 411–419.
[27] Charles Semple, Mike Steel, Phylogenetics, Oxford University Press, 2003.
[28] Bernd Sturmfels, Josephine Yu, Classification of six-point metrics, Electron. J. Combin. 11 (1) (2004), 16 pages.