# The Weighted 2-Server Problem

Marek Chrobak*†  Jiří Sgall ‡†

**Abstract**

We consider a generalization of the 2-server problem in which servers have different costs. We prove that, in uniform spaces, a version of the Work Function Algorithm is 5-competitive, and that no better ratio is possible. We also give a 5-competitive randomized, memoryless algorithm for uniform spaces, and a matching lower bound.

For arbitrary metric spaces, in contrast with the non-weighted case, we prove that there is no memoryless randomized algorithm with finite competitive ratio. We also propose a version of the problem in which a request specifies two points to be covered by the servers, and the algorithm must decide which server to move to which point. For this version, we show a 9-competitive algorithm and we prove that no better ratio is possible.

## 1 Introduction

In the *weighted $k$-server problem* we are given $k$ mobile servers in a metric space $M$, with each server $s_i$ being assigned some non-negative weight $\beta_i$. At each time step a request $r \in M$ is issued, in response to which one of the servers, say $s_i$, must move to $r$, at a cost equal to $\beta_i$ times the distance from its current location to $r$.

This is an *online problem*, in the sense that the decision as to which server to move to $r$ must be made before the next request is issued. In general, due to the incomplete information about the input request sequence, an online algorithm cannot service it in an optimal fashion. The idea behind the *competitive analysis* approach is to evaluate the performance of an online algorithm by comparing its cost to the optimal (offline) cost. More specifically, an online algorithm $\mathcal{A}$ is said to be *$R$-competitive* if the cost incurred by $\mathcal{A}$ to service each request sequence $\varrho$ is at most $R$ times the optimal service cost for $\varrho$, plus an additive constant independent of $\varrho$. The *competitive ratio of $\mathcal{A}$* is the smallest $R$ for which $\mathcal{A}$ is $R$-competitive.

The unweighted case, with all $\beta_i = 1$, has been extensively studied. The problem was originally introduced by Manasse, McGeoch and Sleator [24], who gave a 2-competitive algorithm for $k = 2$ and proved that, for any $k$, $k$ is a lower bound on the competitive ratio. For $k \geq 3$, the best known upper bound is $2k-1$, by Koutsoupias and Papadimitriou [21]. The *$k$-Server Conjecture* is that there exists a $k$-competitive algorithm that works in all metric spaces, but so far this conjecture has been proven only in a number of special cases, including uniform spaces (with all distances equal 1), trees, and spaces with at most $k+2$ points [10, 12, 22].

*Department of Computer Science, University of California, Riverside, CA 92521. Email: marek@cs.ucr.edu. Research supported by NSF grants CCR-9503498, CCR-9988360, CCR-0208856, and an NRC COBASE grant (NSF contract INT-9522667).

†Partially supported by cooperative grants ME-103/INT-9600919 from MŠMT (Czech Republic) and NSF (USA), and KONTAKT-ME476/CCR-9988360-001 from MŠMT ČR and NSF.

‡Mathematical Institute, AS CR, Žitná 25, CZ-11567 Praha 1, Czech Republic. E-mail sgall@math.cas.cz. Partially supported by Institute for Theoretical Computer Science, Prague (project LN00A056 of MŠMT ČR), grant 201/01/1195 of GA ČR, and grant A1019901 of GA AV ČR.

Very little is known about randomized algorithms for $k$ servers. For any $k \geq 2$, no algorithm with ratio less than $k$ is known for arbitrary spaces. In uniform spaces, the competitive ratio is $H_k \approx \ln k$, the $k$-th harmonic number [1, 25]. For $k = 2$, when the metric space is the line, Bartal *et al* [4] give a 1.987-competitive algorithm. Recently, a slight improvement of this bound was announced by Bein and Larmore [6]. The best known lower bound for 2 servers is $1 + e^{-1/2} \approx 1.6065$ [15]. Other lower and upper bounds for this problem can be found in [3, 8].

The weighted case of the $k$-server problem turns out to be substantially more difficult. Fiat and Ricklin [20] showed that the competitive ratio is at least $k^{\Omega(k)}$ in any space with at least $k+1$ points. They also give a doubly-exponential upper bound for uniform spaces. For $k = 2$, a more accurate lower bound was given by Koutsoupias and Taylor [23], who prove that no online algorithm can be better than 10.12-competitive, even if the underlying metric space is the line. They also prove that no memoryless randomized algorithm can be competitive for a related problem called the CNN problem. For $k = 2$ servers in uniform spaces, Feuerstein *et al* [19] gave an algorithm with competitive ratio at most 6.275. The case when the weights are nearly equal was studied in [18]. In particular, for $1 \leq \beta_2/\beta_1 \leq 2.12$ it is shown that, surprisingly, the optimal ratio is achieved by the algorithm FIFO which ignores the weights of the servers.

**Our results.** We study the case $k = 2$. In uniform spaces, we prove that the Work Function Algorithm WFA is 5-competitive, improving the upper bound from [19]. Our method is based on the pseudocost approach, that was introduced to analyze the WFA for metrical task systems and the $k$-server problem [21, 9, 14]. Then we show that that no better ratio is possible if the server weights can have arbitrary positive values. We also give a 5-competitive randomized, memoryless algorithm for uniform spaces, as well as a matching lower bound.

For arbitrary spaces, we prove that there is no memoryless randomized algorithm with finite competitive ratio, even if the underlying metric space is a real line. This strengthens the lower bound on memoryless randomized algorithms for the CNN problem from [23] in two respects: our problem is a special case of the CNN problem and, unlike in [23], we do not assume that the algorithm is invariant with respect to scaling distances. Interestingly, this result contrasts with the non-weighted case, for which memoryless algorithms exist for any $k$. For example, the harmonic algorithm is competitive for any $k$ [5, 9] and, for $k = 2$, a memoryless 2-competitive algorithm is known [11, 17].

Last, we propose a version of the problem in which a request is specified by *two points*, both of which must be covered by the servers, and the algorithm must decide which server to move to which point. For this version, we prove that a modification of the WFA is 9-competitive, and we prove that no better ratio is possible. This generalizes the result from [13] for the 2-point 1-server request problem, as well as for the closely related cow-path problem [2, 26], which are both special cases when $\beta_1 = 1$ and $\beta_2 = 0$.

A preliminary version of this paper appeared in [16].

## 2 Preliminaries

**Notation.** Throughout the paper, without loss of generality, we assume that $\beta_1 = 1$ and $\beta_2 = \beta \leq 1$. Thus $s_1$ and $s_2$ denote the expensive and the cheap server of the algorithm, respectively. Similarly, by $a_1$ and $a_2$ we denote the expensive and the cheap server of the adversary.

For any points $x, y$ in the given metric space, $xy$ denotes their distance. A metric space is called *uniform* if the distance of any two distinct points is equal to 1. For the uniform spaces, the server problem is easily seen to be equivalent to the caching (or paging) problem.

**Adversary arguments.** We sometimes view the computation as a game between two players, the algorithm and the adversary. In each round, the adversary issues a request, serves it using its servers, and then the algorithm serves the request (without knowing the position of the adversary servers).

For a given 2-server algorithm $\mathcal{A}$, by a *configuration* of $\mathcal{A}$ we mean a pair of points $(x, y)$ covered by the servers of $\mathcal{A}$, with $s_1$ at $x$ and $s_2$ at $y$. Naturally, the *adversary configuration* is defined as a set of two points occupied by $a_1$ and $a_2$. At any given time step, the current configurations of $\mathcal{A}$ and the adversary define the *state* of the computation.

There are subtle differences between different adversary models, see [7]. We use two models. The weaker, *oblivious* adversary, has to generate the whole request sequence in advance. The stronger, *adaptive online* adversary generates and serves the requests one by one, with the knowledge of the current positions of the algorithm's servers. Note that our results for randomized algorithms are strongest possible: the upper bound for uniform spaces holds against the adaptive adversary, while the lower bounds are valid for the oblivious adversary.

**Potential functions.** Our proofs are based on the amortized analysis, using potential functions. We use several variants of potential in our arguments. In its basic form, a *potential function* $\Phi$ assigns a real number to the current state. For a given move, we will use notation $\Delta cost$, $\Delta opt$ and $\Delta\Phi$ to denote the online algorithm's cost, the adversary (or optimal) cost, and the potential change in this move. To serve our purpose, $\Phi$ must satisfy the following three properties:

(pot0) $\Phi$ is bounded from below, that is $\inf \Phi > -\infty$,

(pot1) when the adversary moves then $\Delta\Phi \leq R \cdot \Delta opt$, and

(pot2) when the algorithm moves then $\Delta cost + \Delta\Phi \leq 0$.

By summing (pot1) and (pot2) over all requests, it follows that the algorithm is $R$-competitive. Intuitively, one can think of $\Phi$ as the credits that the algorithm has saved in the past and can use to pay for serving future requests.

The proofs for the algorithms that use work functions (see Section 4) follow a slightly different (but equivalent) argument. In those proofs it is convenient to include the work function values (that represent the adversary cost) in the definition of the potential function, and then we only need to prove inequality (pot2).

For randomized algorithms, property (pot2) needs to holds on average, over the random choices of the algorithm. So $\Delta cost$ and $\Delta\Phi$ represent the expected values of the cost of the algorithm and potential change.

One useful principle for designing potential functions is that of the *lazy potential.* If the adversary continues to request the position of his servers, without increasing his cost, the potential function has to provide enough credit to pay for all moves before the algorithm converges to the same server positions. One can calculate the maximum cost of the algorithm on such request sequences, and attempt to use this value as the potential function for amortized analysis. In some online problems, this method gives, indeed, a potential function that yields tight or near-tight competitive ratios. Although the potential functions in our paper are *not* lazy, they are based on an extension of the lazy-potential idea, where we also take into account request sequences on which the adversary is only allowed to move $a_2$, the cheap server. This approach leads often to rather complicated formulas. The functions we use in the paper are obtained by approximating these functions by simpler formulas that are easier to manipulate.

## 3 Randomized Memoryless Algorithms

In this section we consider randomized memoryless algorithms. Our model of a memoryless randomized algorithm is this: A *memoryless algorithm* is simply a function that receives on input the distances from each server to the request point $r$ and the distance between the servers, and determines, for each $i$, the probability that $r$ is served with $s_i$. The algorithm only moves one server, and only to the request point. This is a natural requirement since, in certain spaces, it may be possible to encode memory states by perturbing the other server position.

We should point out that this model is somewhat more restricted than the definition found commonly in the literature, where an algorithm is allowed to depend on the server configuration and the location of the request. In other words, in addition to distances, the algorithm also depends on the point identities. It would be of some interest to extend the results presented in this section to this more general model.

It is quite easy to see that, without loss of generality, we can assume that when a request is on a position of one server, then the other server does not move. Otherwise the algorithm cannot be competitive: Consider any configuration $(x, y)$ such that upon a request on $x$ or $y$ has a non-zero probability of moving. Then the same is true for configuration $(y, x)$ since the algorithm is memoryless. Now take a sequence of alternating requests on $x$ and $y$. The cost of the algorithm is unbounded, while the optimal cost is a constant.

We present the following results in this section. First we give a 5-competitive algorithm for uniform spaces and prove that it is optimal. The lower bound holds even for the weaker oblivious adversary, while the upper bound is valid against the stronger adaptive online adversary. For general spaces, we prove that no memoryless algorithm can achieve a finite competitive ratio, even if the underlying metric spaces is the line.

Both lower bounds are based on the following observation. Consider three points $a$, $b$ and $c$, where $ab = ac$. Suppose that initially $s_1$ is at point $a$ and $s_2$ at $c$, and the adversary alternates requests to $b$ and $c$. Then the probability that $s_1$ ends at $c$ is smaller than the probability that $s_1$ ends at $b$. The reason is that as long as $s_1$ stays on $a$, the situation remains identical from a viewpoint of a memoryless algorithm. The first request is on $b$, so $s_1$ is more likely to move to $b$ than to $c$. Once $s_1$ moves, both $b$ and $c$ are covered, therefore $s_1$ never moves again.

### 3.1 An Upper Bound for Uniform Spaces

On average, our algorithm moves the expensive server after paying approximately $\frac{3}{2}$ for the moves of the cheap server. More precisely, it is defined as follows:

**Algorithm** Rand**:** If the request is on $s_1$ or $s_2$, do nothing. Otherwise serve the request by $s_1$ with probability $p = 2\beta/(3 + \beta)$ and by $s_2$ otherwise.

**Theorem 3.1** *For each fixed $0 \leq \beta \leq 1$, Algorithm* Rand *for the weighted 2-server problem is $(5-\beta)$-competitive in uniform spaces against an adaptive online adversary.*

*Proof:* Consider the algorithm described above. The proof is by amortized analysis. Define a potential function $\Phi$:

$$\Phi(s_1, s_2, a_1, a_2) = \begin{cases} 0 & \text{if } a_1 = s_1 \text{ and } a_2 = s_2, \\ \beta(5-\beta) & \text{if } a_1 = s_1 \text{ and } a_2 \neq s_2, \\ 5-\beta & \text{if } a_1 \neq s_1 \text{ and } a_2 = s_2, \text{ and} \\ 3(5-\beta)/(3-\beta) & \text{if } a_1 \neq s_1 \text{ and } a_2 \neq s_2. \end{cases}$$

Note that the potential increases as RAND's configuration diverges from the adversary's configurations: $0 \leq \beta(5-\beta) \leq 5-\beta \leq 3(5-\beta)/(3-\beta)$.

When the adversary moves $a_1$ and $a_1 \neq s_1$, or the adversary moves $a_2$ and $a_2 \neq s_2$, then $\Delta\Phi \leq 0$, so (pot1) holds trivially. If the initial state is $a_1 = s_1$ and $a_2 = s_2$ then $\Delta\Phi = (5-\beta)\Delta opt$, so (pot1) holds again. The remaining two cases are: either (i) $a_1 = s_1$, $a_2 \neq s_2$ and the adversary moves $a_1$, or (ii) $a_1 \neq s_1$, $a_2 = s_2$ and the adversary moves $a_2$. In both cases the sum of the old potential and $(5-\beta)\Delta opt$ is equal to $(5-\beta)(1+\beta)$, while the new potential is $3(5-\beta)/(3-\beta)$. The condition (pot1) holds since $1+\beta \geq 3/(3-\beta)$.

We now claim that condition (pot2) holds when RAND moves after the adversary requests one of his server positions. If the adversary requests a point occupied by $s_1$ or $s_2$, (pot2) holds trivially. Otherwise, $\Delta cost = p + (1-p)\beta = \beta(5-\beta)/(3+\beta)$. Now we distinguish four cases.

Case 1: $a_1 = s_1$ and the request is on $a_2 \neq s_2$. The potential is $\beta(5-\beta)$, with probability $p$ it changes to $3(5-\beta)/(3-\beta)$ and with probability $1-p$ to 0. Thus

$$\begin{aligned}\Delta cost + \Delta\Phi &= \frac{\beta(5-\beta)}{3+\beta} + p\frac{3(5-\beta)}{3-\beta} - \beta(5-\beta)\\ &= -\frac{\beta^2(5-\beta)(1-\beta)}{(3+\beta)(3-\beta)}\\ &\leq 0.\end{aligned}$$

Case 2: $a_2 = s_2$ and the request is on $a_1 \neq s_1$. The potential is $5-\beta$, with probability $p$ it changes to 0 and with probability $1-p$ to $3(5-\beta)/(3-\beta)$. Thus

$$\begin{aligned}\Delta cost + \Delta\Phi &= \frac{\beta(5-\beta)}{3+\beta} + (1-p)\frac{3(5-\beta)}{3-\beta} - (5-\beta)\\ &= 0.\end{aligned}$$

Case 3: $a_1 \neq s_1$ and the request is on $a_2 \neq s_2$. The potential is $3(5-\beta)/(3-\beta)$, with probability $p$ it does not change, and with probability $1-p$ it changes to $5-\beta$. Thus

$$\begin{aligned}\Delta cost + \Delta\Phi &= \frac{\beta(5-\beta)}{3+\beta} + (1-p)\left((5-\beta) - \frac{3(5-\beta)}{3-\beta}\right)\\ &= 0.\end{aligned}$$

Case 4: $a_2 \neq s_2$ and the request is on $a_1 \neq s_1$. The potential is $3(5-\beta)/(3-\beta)$, with probability $p$ it changes to $\beta(5-\beta)$ and with probability $1-p$ it does not change. Thus

$$\begin{aligned}\Delta cost + \Delta\Phi &= \frac{\beta(5-\beta)}{3+\beta} + p\left(\beta(5-\beta) - \frac{3(5-\beta)}{3-\beta}\right)\\ &= -\frac{\beta(5-\beta)(1-\beta)(3-2\beta)}{(3+\beta)(3-\beta)}\\ &\leq 0.\end{aligned}$$

Summarizing, we conclude that the potential function satisfies the conditions (pot1) and (pot2). The theorem follows from (pot1) and (pot2) by amortization. □

### 3.2 A Lower Bound for Uniform Spaces

**Theorem 3.2** *Let $\mathcal{A}$ be a memoryless randomized algorithm for the weighted 2-server problem in a uniform space with at least three points. Suppose that $\beta$ can take arbitrary positive values. Then the competitive ratio of $\mathcal{A}$ against an oblivious adversary is at least 5.*

*Proof:* Let $a$, $b$, and $c$ denote three distinct points in a uniform metric space. $\mathcal{A}$ has only one parameter, which is the probability $p$ that a request unoccupied by a server is served by $s_1$. For $p = 0, 1$ it is easy to see that $\mathcal{A}$ cannot be competitive, so we can assume that $p \neq 0, 1$. At the beginning, we assume that the expensive servers $s_1$ and $a_1$ are at $a$ and the cheap servers $s_2$ and $a_2$ at $b$. We also assume that $\beta \to 0$, and all $O$-notation is relative to this. We prove a lower bound of $5 - o(1)$, the bound of 5 follows by taking $\beta$ sufficiently small.

First consider what happens if we repeat requests $b$ and $c$ infinitely long. Eventually, $\mathcal{A}$ moves $s_1$ to $b$ or $c$ and $s_2$ to the other of these two points. The expected cost of the algorithm is $1 + \beta/p - \beta$, since it will take on average $1/p - 1$ of moves of $s_2$. We choose $k$ large enough, say $k = \lceil \log \beta / \log(1-p) \rceil$, so that after the sequence $(cb)^k$, the probability that $s_1$ remains at $a$ is $o(1)$ and the expected cost of $\mathcal{A}$ is $1 + \beta/p - o(1)$. Since the first (non-trivial) request is at $c$, after requests $(cb)^k$, the probability that $(s_1, s_2) = (c, b)$ is at least $\frac{1}{2} - o(1)$. (See the observation before Section 3.1.)

Consider a sequence of requests $((cb)^k(ab)^k)^l$ for $l = \omega(1)$. Let us call a subsequence of requests $(cb)^k$ or $(ab)^k$ a *phase*. By the previous paragraph, until $\mathcal{A}$ moves $s_1$ to $b$, it pays $1 + \beta/p - o(1)$ for each phase, and it takes on average at least $2 - o(1)$ phases to move $s_1$ to $b$. Thus, starting with $(s_1, s_2) = (a, b)$, the total cost of $\mathcal{A}$ on $((cb)^k(ab)^k)^l$ is $2(1 + \beta/p)(1 - o(1))$.

The adversary strategy depends on $p$. If $p \leq \frac{2}{3}\beta$ then the request sequence is the above sequence $((cb)^k(ab)^k)^l$ for some $l$ satisfying $l = o(1/\beta)$ and $l = \omega(1)$. The adversary serves the sequence by moving $a_1$ to $b$ and then moving $a_2$ between $c$ and $a$ $2l$ times, with total cost $1 + 2l\beta = 1 + o(1)$. Thus the ratio is at least

$$\frac{2(1 + \beta/p)(1 - o(1))}{1 + o(1)} \quad \geq \quad 5 - o(1).$$

If $p \geq \frac{2}{3}\beta$ then the request sequence is $(cb)^m((ca)^k(ba)^k)^l$, where $m$ and $l$ are chosen such that $m = o(1/p)$ (and thus $m = o(1/\beta)$), $l = o(m)$, $l = \omega(1)$. The adversary does not move $a_1$ and serves the requests at $b$ and $c$ with $a_2$, at cost $2(l + m)\beta = 2m\beta(1 + o(1))$. The probability that $s_1$ does not move during the first $2m$ steps is

$$\begin{aligned}
q &= (1-p)^{2m} \\
&= 1 - 2mp + \sum_{j=2}^{2m} \binom{2m}{j} (-p)^j \\
&= 1 - 2mp + \Theta(m^2 p^2) \\
&= 1 - 2mp(1 - o(1)).
\end{aligned}$$

The expected cost of $s_2$ during the first $2m$ steps is $(1 - q)(\beta/p - \beta)$ (the cost would be $\beta/p - \beta$ for an infinite sequence; with probability $q$ we stop after $2m$ steps and save $\beta/p - \beta$, which is the expected cost starting after $2m$ steps conditioned on the fact that we have at least $2m$ steps). If $s_1$ moves, the additional cost is $1 + 2(1 + \beta/p)(1 - o(1))$. So the total expected cost of $\mathcal{A}$ is at least $(1 - q) \cdot 3(1 + \beta/p)(1 - o(1))$ and the ratio is at least

$$\begin{aligned}
\frac{2mp \cdot 3(1 + \frac{\beta}{p})(1 - o(1))}{2m\beta(1 + o(1))} &= 3\left(\frac{p}{\beta} + 1\right)(1 - o(1)) \\
&\geq 5 - o(1).
\end{aligned}$$

We have shown that for any choice of $p$ there is a request sequence that forces the expected cost of $\mathcal{A}$ to be at least $5 - o(1)$ times the optimal cost. Thus the proof is complete. □

### 3.3 A Lower Bound for the Line

**Theorem 3.3** *Suppose that $\beta$ can take arbitrary positive values. Then there is no competitive randomized memoryless algorithm for the weighted 2-server problem on the real line.*

*Proof:* Suppose that $\mathcal{A}$ is a memoryless randomized algorithm on the real line. First observe that we may assume that the algorithm never moves a server if the current request is already served. If not, i.e., for some distance $d$ there is a non-zero probability that upon a request to one of the server location the algorithm moves the other server to the request as well, alternate requests on the points 0 and $d$; then the cost of the algorithm is unbounded while the cost of the adversary is a constant and thus the algorithm is not competitive.

Let $k > 1$ be an arbitrarily large integer. We will show that the competitive ratio of $\mathcal{A}$ is at least $k/2$. We choose $\beta = 2^{-k}$. At the beginning, assume that $s_2$ and $a_2$ are at 0 and $s_1$ and $a_1$ are at 1. We have $k$ phases. In each phase $i = 1, 2, \ldots, k$, we alternate requests to points $2^i$ and 0, for a total of $k2^k$ requests per phase, starting at $2^i$ and ending at 0. The optimal algorithm moves $a_1$ to 0 and serves all other requests with $a_2$. The total cost is $1 + 2^k\beta = 2$.

Let $C_i$ be expected cost of $\mathcal{A}$ in phases $i+1, \ldots, k$, assuming that after $i$ phases $s_2$ is at 0 and $s_1$ is at $2^i$. We claim that

$$C_i \;\geq\; (k-i)2^i. \tag{1}$$

The theorem follows from (1), since $\mathcal{A}$ pays at least $C_0 \geq k$, and thus the competitive ratio is at least $k/2$.

The proof of (1) is by backward induction. By definition, $C_k = 0$. For the inductive step, we assume that $C_{i+1} \geq (k-i-1)2^{i+1}$ and show that (1) holds.

To estimate $C_i$, we analyze phase $i+1$ assuming that $s_2$ starts at 0 and $s_1$ at $2^i$. $\mathcal{A}$ can serve this phase in one of three ways:

(a) $s_1$ stays at $2^i$ and all requests are served with $s_2$. Then the cost is at least $k2^k \cdot 2^{i+1}\beta = 2k \cdot 2^i > (k-i)2^i$ in phase $i$ alone.

(b) $s_1$ moves to $2^{i+1}$ at some step. Then no server moves in the rest of this phase and the cost of this and all following phases is at least $2^i + C_{i+1}$.

(c) $s_1$ moves to 0 at some step. Then no server moves in the rest of this phase and the cost of this phase is at least $2^i$.

If $s_1$ moves, then (by the observation before Section 3.1) case (b) is more likely to occur than (c), so the expected total cost in this case is at least $2^i + \frac{1}{2}C_{i+1} \geq 2^i + \frac{1}{2}(k-i-1)2^{i+1} = (k-i)2^i$. Thus, independently of whether $s_1$ stays at $2^i$ or moves, the cost of $\mathcal{A}$ is at least $(k-i)2^i$, and (1) follows. □

Note that, on the line, and for a fixed value of $\beta$, we can easily achieve an upper bound of $2/\beta$ by ignoring the weights and running the resistive-harmonic algorithm for two servers [17, 11, 9]. This algorithm is memoryless and has competitive ratio 2 on the line. The optimal costs of the weighted and non-weighted versions differ by a factor at most $\beta$, and an upper bound of $2/\beta$ on the competitive ratio follows. On the other hand, if we choose $\beta = 2^{-k}$ in the above proof, the calculations yield the lower bound of $\frac{1}{2}\log(1/\beta)$. Thus, for fixed values of $\beta$, there is still substantial gap between the lower and upper bounds.

# 4 Deterministic Algorithms in Uniform Spaces

Now we focus on deterministic algorithms. We start by introducing work functions. Then, we give a 5-competitive algorithm for the weighted 2-server problem in uniform spaces and prove that the ratio 5 is optimal. In the last subsection we study a simplified version of the weighted 2-server problem; for this version we give a 9-competitive algorithm for an arbitrary metric space.

## 4.1 Work Functions

Recall that a *configuration* is a pair $(x, y)$, where $x, y$ are the locations of the expensive and cheap servers, respectively. For a request sequence $\sigma$, the *work function* $\omega_\sigma$ associates with each configuration $(x, y)$ the minimum cost $\omega_\sigma(x, y)$ of serving all requests in $\sigma$ in such a way that the final configuration is $(x, y)$.

We can allow the algorithms to move its servers to some new positions even in-between the requests. By the triangle inequality, this will not change the overall minimum cost, and it is convenient because it allows us to define $\omega_\sigma$ on configurations that do not contain the last request.

Each work function $\omega_\sigma$ satisfies the following Lipschitz condition: for any points $x, y, u, v \in M$,

$$\omega_\sigma(u, v) \quad \leq \quad \omega_\sigma(x, y) + xu + \beta \cdot yv. \tag{2}$$

Inequality (2) is quite obvious: one possible way to service $\sigma$ ending at $(u, v)$ is to serve $\sigma$ ending at $(x, y)$ in an optimal fashion, and then to move from $(x, y)$ to $(u, v)$ at cost $xu + \beta \cdot yv$.

We can compute $\omega_\sigma$ by dynamic programming as follows. Initially, by definition, $\omega_\epsilon(x, y) = x_0x + \beta \cdot y_0y$, where $(x_0, y_0)$ is the initial configuration. Consider a request sequence $\sigma r$, i.e., $\sigma$ followed by a request at $r$. Then

$$\omega_{\sigma r}(x, y) \quad = \quad \min \left\{ \begin{array}{c} \omega_\sigma(r, y) + rx \\ \omega_\sigma(x, r) + \beta \cdot ry \end{array} \right\} \; = \; \min \left\{ \begin{array}{c} \omega_{\sigma r}(r, y) + rx \\ \omega_{\sigma r}(x, r) + \beta \cdot ry \end{array} \right\} \tag{3}$$

The second equality follows trivially from the definition of work functions, since whenever $\sigma$ is served and one of the servers ends at $r$, also $\sigma r$ is served. To justify the first equality in (3), suppose that we reach $(x, y)$ in some optimal way serving $\sigma r$. In the previous step, before serving $r$, we were in some configuration $(x', y')$ and we moved our servers from $(x', y')$ to $(x, y)$ in such a way that one of them visited $r$. If we postpone the move of a server to $r$ to be the last one, (2) and the triangle inequality implies that one of the optimal choices of $(x', y')$ is either $(r, y)$ or $(x, r)$, and (3) follows.

## 4.2 Work Functions in Uniform Spaces

First, we rewrite the definition of work functions, adapting it to the special case of uniform spaces. If $(x_0, y_0)$ is the initial configuration, we define the initial work function $\omega_\epsilon$ by $\omega_\epsilon(x_0, y_0) = 0$, $\omega_\epsilon(y_0, x_0) = 1$, and for $x, y \neq x_0, y_0$, $\omega_\epsilon(x_0, y) = \beta$, $\omega_\epsilon(x, y_0) = 1$, and $\omega_\epsilon(x, y) = 1 + \beta$. Note that the choice $\omega_\epsilon(y_0, x_0) = 1$ (instead of $1 + \beta$) deviates slightly from the definition in the previous section. It is motivated by Lemma 4.1 below. (Of course, the values of the initial work function do not affect the competitive ratio, as they are absorbed by the constant term.)

For non-empty requests, we proceed inductively. Let $q$ be the last request in $\sigma$. If $r = q$, then

$\omega_{\sigma r} = \omega_\sigma$. For $r \neq q$ we have

$$\omega_{\sigma r}(x,y) \quad = \quad \begin{cases} \omega_\sigma(x,y) & \text{if } r \in \{x,y\} \\ \\ \min \left\{ \begin{array}{c} \omega_\sigma(x,r) + \beta \\ \omega_\sigma(r,y) + 1 \end{array} \right\} & \text{otherwise} \end{cases} \tag{4}$$

**Notation.** Throughout the rest of this section, let $\omega = \omega_\varrho$ be an arbitrary work function. If $\varrho$ is non-empty, $r$ denotes the last request in $\varrho$ and $\sigma$ denotes the request sequence without the last request, i.e., $\varrho = \sigma r$. Furthermore $\mu = \omega_\sigma$ denotes the previous work function and $q$ denotes the last request in $\sigma$, if $\sigma$ is non-empty. Without loss of generality, we can assume that $r \neq q$. (Otherwise the work function does not change and the algorithm does not move, so we can remove the duplicate request.)

The paragraph above leaves $r$, $q$ and $\mu$ undefined for the initial steps. This does not really matter for the competitive analysis, as the cost of the first step can be absorbed by the constant term. However, noting that the initial work function does not change if $x_0$ or $y_0$ is requested, we can extend the definition as follows. If $\varrho = \epsilon$, we take $r = x_0$, $q = y_0$, and $\mu = \omega = \omega_\epsilon$. If $\varrho$ has only one request $r$, $\mu = \omega_\epsilon$ and we take $q = x_0$. Now all the analysis below is correct also for the initial state and the first request.

**Auxiliary lemmas.** In the uniform space, the Lipschitz condition says that $\omega(x,y) \leq \omega(x,v) + \beta$, $\omega(x,y) \leq \omega(u,y) + 1$, and $\omega(x,y) \leq \omega(u,v) + 1 + \beta$. It turns out that for some situations a tighter inequality can be shown, with $\beta$ eliminated from the last inequality.

**Lemma 4.1** *Let $x \neq y$ and $u \neq v$.*

(a) *If $r \notin \{u,v\}$ then $\omega(x,y) \leq \omega(u,v) + 1$, and*

(b) $\omega(x,y) \leq \omega(y,x) + 1$.

*Proof:* (a) By (4) and the Lipschitz condition, $\omega(u,v) + 1 = \min\{\omega(u,r) + 1 + \beta, \omega(r,v) + 2\} \geq \omega(x,y)$.

(b) The proof is by induction on the number of requests. It holds for the empty request sequence by the definition of the initial work function. Otherwise let $\omega = \omega_{\sigma r}$. If $r \notin \{x,y\}$, then the claim follows by (a). If $r \in \{x,y\}$, then we have $\omega(x,y) = \mu(x,y)$ and $\omega(y,x) = \mu(y,x)$, so the lemma follows by induction from (b) for $\mu$. $\Box$

If equality holds in the Lipschitz condition, it means that the offline optimal algorithm always prefers the configuration with the smaller work function over the other one involved in the condition. For the weighted 2-server problem in uniform spaces, Lemma 4.1(b) implies that, interestingly, we can never tell with certainty which point is occupied by the adversary server $s_1$ and which by $s_2$. This explains the slight modification of the initial work function $\omega_\epsilon$.

Next we want to describe possible work function values $\omega(x,\cdot)$, for any fixed $x \neq r$. For $x = q$ we can give a complete description: $\omega(q,r)$ is the minimal value and all the other values are equal to $\omega(q,r) + \beta$. This property does not hold for $x \neq q$, and for an arbitrary $x$ we only prove a weaker statement that all these values are at least $\omega(x,r)$, with the possible exception of $\omega(x,q)$.

For $\beta \leq \frac{1}{2}$, a much stronger characterization can be shown (proof not included), namely that for every $x, y \neq \{q,r\}$, $\omega(x,y) = \min\{\omega(x,r), \omega(x,q)\} + \beta$. Even though this property does not hold for arbitrary $\beta$, we think it provides valuable insight that helps to understand the idea of our analysis.

**Lemma 4.2** *Let $x, y \notin \{q,r\}$ and $x \neq y$. Then*

(a) $\omega(q,y) = \omega(q,r)+\beta$, *and*
(b) $\omega(x,y) \geq \omega(x,r)$.

*Proof:* (a) We have $\omega(r,y)+1 = \mu(r,y)+1 = \min\{\mu(r,q)+1+\beta, \mu(q,y)+2\} \geq \mu(q,r)+\beta = \omega(q,r)+\beta$, by applying Lemma 4.1(b) and the Lipschitz condition to the terms in the minimum. So $\omega(q,y) = \min\{\omega(r,y)+1, \omega(q,r)+\beta\} = \omega(q,r)+\beta$.

(b) By Lemma 4.1(a) applied to $\mu$ and using $q \notin \{y,r\}$, we obtain $\omega(r,y)+1 = \mu(r,y)+1 \geq \mu(x,r) = \omega(x,r)$. Thus $\omega(x,y) = \min\{\omega(x,r)+\beta, \omega(r,y)+1\} \geq \omega(x,r)$. □

The next lemma states that for points $x \notin \{q,r\}$, the order of the previous work function values $\mu(x,q)$ fully determines the order of values $\omega(x,y)$ (for each fixed $y$), as well as the order of the *increases* of the work function on $(x,q)$, when $r$ is requested. Thanks to this lemma, in the analysis of our algorithm we can focus on the point $x$ that minimizes $\mu(x,q)$.

**Lemma 4.3** *Let $x, x' \notin \{q,r\}$ be such that $\mu(x,q) \leq \mu(x',q)$. Then*
(a) $\omega(x,y) \leq \omega(x',y)$ *for all* $y \notin \{x,x'\}$, *and*
(b) $\omega(x,q) - \mu(x,q) \geq \omega(x',q) - \mu(x',q)$.

*Proof:* (a) For the special case $y=r$, we have $\omega(x,r) = \mu(x,r) = \min\{\mu(x,q)+\beta, \mu(q,r)+1\} \leq \min\{\mu(x',q)+\beta, \mu(q,r)+1\} = \mu(x',r) = \omega(x',r)$. For $y \neq r$, using the previous case, $\omega(x,y) = \min\{\omega(x,r)+\beta, \omega(r,y)+1\} \leq \min\{\omega(x',r)+\beta, \omega(r,y)+1\} = \omega(x',y)$.

(b) Denoting $W = \min\{\mu(q,r)+\beta, \mu(r,q)\}+1$, and, applying (4) to $\omega$ and then to $\mu$, we have $\omega(x,q) - \mu(x,q) = \min\{\mu(x,q)+2\beta, \mu(q,r)+1+\beta, \mu(r,q)+1\} - \mu(x,q) = \min\{2\beta, W-\mu(x,q)\}$, and the same equality holds for $x'$ instead of $x$. Therefore $\omega(x,q)-\mu(x,q) = \min\{2\beta, W-\mu(x,q)\} \geq \min\{2\beta, W-\mu(x',q)\} = \omega(x',q)-\mu(x',q)$. □

**An auxiliary function.** For each work function $\omega$, we define

$$\bar{\omega}(x) \quad = \quad \min_{y \neq y'} \{\omega(x,y)+\omega(x,y')\},$$

where $y, y' \neq x$. Thus $\bar{\omega}(x)$ is simply the sum of the two smallest values of $\omega(x,\cdot)$. Note that, by the Lipschitz condition, for any $y \neq x$ we have $2\omega(x,y) = \bar{\omega}(x) + O(\beta)$. Functions $\bar{\omega}$ are of little interest of their own, but they will be useful in the analysis of our algorithm.

The following lemmas provide a characterization of $\bar{\omega}(x)$ for $x \neq r$. For $x = q$, we give a complete description.

For $x \notin \{q,r\}$, the situation is more complicated. In the case of $\beta \leq \frac{1}{2}$ that we mentioned earlier, we always have $\bar{\omega}(x) = \omega(x,r)+\omega(x,q)$. In the general case, $\omega(x,r)$ always contributes to $\bar{\omega}(x)$, but in certain situations it may happen that $\omega(x,q)$ does not. However, we show that the deviation from the regular situation can be bounded, and, in an important case when $\bar{\omega}(x)$ is relatively small, such bad situations cannot occur at all.

**Lemma 4.4** *Let $x, y \notin \{q,r\}$. Then*
(a) $\bar{\omega}(q) = 2\omega(q,r)+\beta = 2\mu(q,r)+\beta = 2\omega(q,y)-\beta$, *and*
(b) $\bar{\omega}(x) = \omega(x,r) + \min_{p \notin \{x,r\}} \omega(x,p)$.

*Proof:* The lemma follows directly from Lemma 4.2. □

**Lemma 4.5** *Let $x \notin \{q,r\}$. If $\omega(x,p) < \omega(x,q)$, for some $p \notin \{q,r,x\}$, then*
(a) $\omega(x,p) = \omega(r,p)+1 = \mu(q,p)+2$, *and*
(b) $\omega(x,q) - \omega(x,p) \leq \bar{\omega}(q) - \bar{\mu}(q) - 1$.

*Proof:* (a) We have $\omega(x,p) < \omega(x,q) \le \omega(x,r) + \beta$. Thus $\omega(x,p) = \omega(r,p) + 1 < \omega(x,q) \le \omega(r,q) + 1$ and $\mu(r,p) = \omega(r,p) < \omega(r,q) = \mu(r,q)$. This in turn implies $\mu(r,p) = \min\{\mu(q,p) + 1, \mu(r,q) + \beta\} = \mu(q,p) + 1$ and $\omega(x,p) = \mu(r,p) + 1 = \mu(q,p) + 2$.

(b) Using (a) and Lemma 4.1(b), $\omega(x,q) - \omega(x,p) \le (\omega(q,x) + 1) - (\mu(q,p) + 2) = (\omega(q,x) + \omega(q,r)) - (\mu(q,r) + \mu(q,p)) - 1 \le \bar{\omega}(q) - \bar{\mu}(q) - 1$. □

**Lemma 4.6** *Let $u \notin \{q,r\}$ be such that $\bar{\omega}(u) \le \bar{\omega}(q)$. Then*

(a) $\bar{\omega}(u) = \omega(u,r) + \omega(u,q)$, *and*

(b) $\mu(u,r) = \mu(u,q) + \beta$.

*Proof:* If (a) is false, then, by Lemma 4.4(b), for some $p \notin \{q,r\}$, $\omega(u,p) < \omega(u,q) \le \omega(u,r) + 1$, and $\bar{\omega}(u) = \omega(u,r) + \omega(u,p) > 2\omega(u,p) - 1 = 2\mu(q,p) + 3 \ge 2\mu(q,r) + 1 \ge \bar{\omega}(q)$, using Lemma 4.5, the Lipschitz condition, and Lemma 4.4(a) in the last three steps. This contradicts the assumption of the lemma.

If (b) is false, then $\mu(u,r) = \mu(q,r) + 1 < \mu(u,q) + \beta \le \mu(u,q) + 1$. Thus, using (a), $\bar{\omega}(u) = \omega(u,r) + \omega(u,q) \ge \mu(u,r) + \mu(u,q) > 2\mu(u,r) - 1 = 2\omega(q,r) + 1 \ge \bar{\omega}(q)$, contradicting the assumption of the lemma. □

### 4.3 The Work-Function Algorithm for Uniform Spaces

In this section we show that the Work-Function Algorithm (WFA), known to achieve optimal competitive ratios for several online problems (see [9, 14]), also gives an optimal ratio for the weighted 2-server problem in uniform spaces, for $\beta \to 0$.

**Algorithm WFA:** Suppose the current configuration, after a request sequence $\sigma$, is $(u,v)$, and $r$ is the new request. If $r \in \{u,v\}$, do not move. Otherwise, if $\omega_{\sigma r}(u,r) + \beta \le \omega_{\sigma r}(r,v) + 1$ then move $s_2$, else move $s_1$.

**Example.** Before analyzing the algorithm, it is instructive to look at the (tight) example of small $\beta$ and a space with three points $a$, $b$, and $c$. The adversary always requests the point not occupied by any of the WFA's servers. Suppose that the expensive server starts at $a$. We estimate all costs by ignoring additive $O(\beta)$ terms. Then $\omega(a,x) = 0$, $\omega(b,x) = \omega(c,x) = 1$. The adversary keeps requesting points $b$ and $c$, until $\omega(a,x) = 2$, when the WFA moves the expensive server to $b$ (or $c$, which is symmetric). The cost of WFA in this phase is 3. Next, the adversary alternates requests at $a$ and $c$, until $\omega(b,x) = 2$. In this second phase, the WFA pays 2. Overall, WFA pays 5, the final state is isomorphic to the initial state, and the global increase of $\omega$ (under this isomorphism) is 1.

**Metrical Service Systems.** Recall that a *metrical service system* (also called a *forcing task system*) [9, 14] is a pair $S = (M, \mathcal{R})$, where $M$ is a metric space and $\mathcal{R} \subseteq 2^M$ is a set of admissible requests. We have one server, and on a request $X \in \mathcal{R}$ the server needs to move to some point in $X$.

We now note that a weighted 2-server problem in any metric space $M$ can be expressed as a metrical service system: Introduce a metric space $M' = M \times (\beta M)$, where in $\beta M$ all distances are multiplied by $\beta$, and we use the $L^1$ metric in the Cartesian product. Then a request $r$ can be represented as a MSS request $(\{r\} \times (\beta M)) \cup (M \times \{r\})$.

**Pseudocost.** Given a work function $\omega_\sigma$ and a request $r$, define the *pseudocost* of $r$ on $\omega_\sigma$ [9, 14] as

$$\nabla_r(\omega_\sigma) \quad = \quad \max_{x,y} \{\omega_{\sigma r}(x,y) - \omega_\sigma(x,y)\},$$

that is, the maximum increase of the work function on request $r$. If $\varrho = r_1 r_2 \ldots r_n$ is a sequence of requests, then the *total pseudocost* of $\varrho$ is:

$$\nabla_\varrho \;=\; \sum_{i=1}^{n} \nabla_{r_i}(\omega_{r_1 r_2 \ldots r_{i-1}}).$$

Using the relationship between the cost of WFA and the pseudocost [14] in metrical service systems, we have the following:

**Lemma 4.7** *Suppose that there exists $c$ such that for each request sequence $\varrho$, $\nabla_\varrho \le 6 \cdot \min(\omega_\varrho) + c$. Then* WFA *is 5-competitive.*

The proof of this lemma is by showing that the pseudocost of any given sequence can be written as a sum of WFA's cost and the optimal cost (up to a constant additive term) for this sequence. Thus the competitive ratio of WFA is one less than that of the pseudocost. See [14] for the details.

The next lemma characterizes and estimates the pseudocost.

**Lemma 4.8** *Choose $x \notin \{q, r\}$ that minimizes $\mu(x, q)$ and $y \notin \{q, r\}$ that minimizes $\mu(q, y)$. Then*

$$\nabla_r(\mu) \;=\; \min \left\{ \begin{array}{c} \omega(x,q) - \mu(x,q) \\ \omega(q,y) - \mu(q,y) \end{array} \right\} \;\le\; 2\beta.$$

*Proof:* By the Lipschitz condition, for all distinct points $u, v$, we have $\omega(u,v) \le \omega(u,r) + \beta = \mu(u,r) + \beta \le \mu(u,v) + 2\beta$. (Informally, we can always serve the new request by the cheap server and then return back to $v$ at cost $2\beta$.) This proves the inequality in the lemma.

Choose two different points $u, v$ such that $\nabla_r(\mu) = \omega(u,v) - \mu(u,v)$. Since $\omega(u,r) = \mu(u,r)$ and $\omega(r,v) = \mu(r,v)$, we can assume $r \notin \{u, v\}$.

Suppose now that $q \notin \{u, v\}$ as well. Then $\mu(u,v) = \min\{\mu(u,q) + \beta, \mu(q,v) + 1\}$. If $\mu(u,v) = \mu(u,q) + \beta$, then $\omega(u,v) - \mu(u,v) = \omega(u,v) - \mu(u,q) - \beta \le \omega(u,q) - \mu(u,q)$. If $\mu(u,v) = \mu(q,v) + 1$ then $\omega(u,v) - \mu(u,v) = \omega(u,v) - \mu(q,v) - 1 \le \omega(q,v) - \mu(u,q)$. We conclude that $\nabla_r(\mu) = \omega(u,v) - \mu(u,v)$ for some $u, v$ such that $q \in \{u, v\}$ and $r \notin \{u, v\}$.

If $v = q$, then Lemma 4.3 and the choice of $x$ imply that $\omega(u,q) - \mu(u,q) \le \omega(x,q) - \mu(x,q)$. If $u = q$ then $\omega(q,v) = \omega(q,y)$ by Lemma 4.2(a); together with the choice of $y$ this implies $\omega(q,v) - \mu(q,v) \le \omega(q,y) - \mu(q,y)$. The lemma follows, as by the the previous paragraph all the cases are covered. $\Box$

We are now ready to prove the following theorem.

**Theorem 4.9** WFA *is a 5-competitive algorithm for the weighted 2-server problem in uniform spaces.*

*Proof:* Given a work function $\omega = \omega_{\sigma r}$, and three distinct points $u, v, t$, define

$$\Psi_{uvt}(\omega) \;=\; \bar{\omega}(u) + \bar{\omega}(v) + \bar{\omega}(t), \quad \text{and} \tag{5}$$

$$\Psi(\omega) \;=\; \min_{u,v,t} \Psi_{uvt}(\omega). \tag{6}$$

We use the function $\Psi(\omega)$ in our amortized analysis. $\Psi(\omega)$ is not a potential function in the sense described in Section 2; the actual potential function is $-\Psi(\omega)$. We use the negated form to simplify calculations.

We claim that for any request sequence $\sigma$ and for any request $r$,

$$\nabla_r(\omega_\sigma) \;\le\; \Psi(\omega_{\sigma r}) - \Psi(\omega_\sigma). \tag{7}$$

If (7) holds then, by amortizing over the whole request sequence $\varrho$, we get $\nabla_\varrho \leq \Psi(\omega_\varrho) = 6 \cdot \min(\omega_\varrho) + \Theta(1)$, where the constant term absorbs the differences of the contributing terms from the minimum of the work function, which are bounded by the Lipschitz condition. The theorem then follows from Lemma 4.7. Thus it is sufficient to prove (7).

Before we get to the general case, note that (7) is trivial if $M$ has only three points, as $\Psi$ is always the sum of all the six values of the work function, and its increase is the sum of increases of all the values (none of them can decrease, of course).

Let $u \notin \{q,r\}$ be a minimum of $\bar{\omega}$ excluding $q$ and $r$, i.e., $\bar{\omega}(u) \leq \bar{\omega}(z)$ for all $z \notin \{q,r\}$. By the choice of $u$, we have $\Psi(\omega) = \Psi_{uvt}(\omega)$, for two distinct points $v, t \neq u$. Furthermore, we can choose $v$ and $t$ so that if $\bar{\omega}(q) \leq \max\{\bar{\omega}(u), \bar{\omega}(v), \bar{\omega}(t)\}$ then $v = q$, i.e., in case of a tie, we prefer $q$ over other points, and we choose the notation so that $v = q$ (note that $u \neq q$ by the definition of $u$ and $v$ and $t$ are symmetric). We fix these points $v$ and $t$ for the rest of the proof. To prove (7), it is sufficient to find three distinct points $u', v', t'$ such that

$$\nabla_r(\mu) \;\leq\; \Psi_{uvt}(\omega) - \Psi_{u'v't'}(\mu). \tag{8}$$

The rest of the proof is devoted to the proof of (8).

Choose $x \notin \{q,r\}$ that minimizes $\mu(x,q)$ and $y \notin \{q,r\}$ that minimizes $\mu(q,y)$. Using Lemma 4.8, $\nabla_r(\mu) = \min\{\omega(x,q) - \mu(x,q), \omega(q,y) - \mu(q,y)\}$. We distinguish two cases corresponding to the terms that realize this minimum.

Case 1: $\nabla_r(\mu) = \omega(x,q) - \mu(x,q)$. By the choice of $x$, $\mu(x,q) \leq \mu(u,q)$. We can assume that $v \neq x$, by the symmetry of $v$ and $t$ (note that this applies even if $v = q$). Define $t' = u$ if $t = x$ and $t' = t$ otherwise. Thus $x$, $v$ and $t'$ are distinct points, and, by the choice of $u$, in both cases

$$\bar{\omega}(t) \;\;\geq\;\; \bar{\omega}(t') \;\geq\; \bar{\mu}(t'). \tag{9}$$

Using Lemma 4.3(a) and $\omega(x,r) = \mu(x,r)$, we have

$$\nabla_r(\mu) \;=\; [\omega(x,r) + \omega(x,q)] - [\mu(x,r) + \mu(x,q)] \;\leq\; \omega(u,r) + \omega(u,q) - \bar{\mu}(x). \tag{10}$$

Case 1.1: $\bar{\omega}(u) = \omega(u,r) + \omega(u,q)$. Using (10) and (9) we get

$$\begin{aligned}
\nabla_r(\mu) \;&\leq\; \omega(u,r) + \omega(u,q) - \bar{\mu}(x).\\
&=\; \bar{\omega}(u) - \bar{\mu}(x)\\
&=\; \Psi_{uvt}(\omega) - [\bar{\mu}(x) + \bar{\omega}(v) + \bar{\omega}(t)]\\
&\leq\; \Psi_{uvt}(\omega) - \Psi_{xvt'}(\mu).
\end{aligned}$$

Case 1.2: $\bar{\omega}(u) = \omega(u,r) + \omega(u,p) < \omega(u,r) + \omega(u,q)$, for some $p \notin \{q,r\}$. By Lemma 4.6(a), we have $\bar{\omega}(q) < \bar{\omega}(u)$. Thus $q = v$, by the choice of $v$.

Therefore, using (10), Lemma 4.5(b) (that can be applied because $\omega(u,p) < \omega(u,q)$), and (9) we get

$$\begin{aligned}
\nabla_r(\mu) \;&\leq\; \omega(u,r) + \omega(u,q) - \bar{\mu}(x)\\
&=\; \bar{\omega}(u) + \omega(u,q) - \omega(u,p) - \bar{\mu}(x)\\
&\leq\; \bar{\omega}(u) + \bar{\omega}(q) - \bar{\mu}(q) - \bar{\mu}(x)\\
&=\; \Psi_{uqt} - [\bar{\mu}(x) + \bar{\mu}(q) + \bar{\omega}(t)]\\
&\leq\; \Psi_{uqt}(\omega) - \Psi_{xqt'}(\mu).
\end{aligned}$$

Note that the calculations in this case are not tight; we have a slack of 1 in the application of Lemma 4.5(b) in line 3.

Case 2: $\nabla_r(\mu) = \omega(q,y) - \mu(q,y)$. We can assume that $v \neq r$, by the symmetry of $v$ and $t$ (note that this again applies even if $v = q$).

We have $\bar{\mu}(q) \leq \mu(q,y) + \mu(q,r) = \mu(q,y) + \omega(q,r)$, and thus, using Lemma 4.4(a),

$$\nabla_r(\mu) \;=\; [\omega(q,r) + \omega(q,y)] - [\mu(q,r) + \mu(q,y)] \;\leq\; \bar{\omega}(q) - \bar{\mu}(q). \tag{11}$$

Case 2.1: $\bar{\omega}(q) \leq \max\{\bar{\omega}(v), \bar{\omega}(t)\}$. Thus $q = v$, by the choice of $v$. Then, using (11), we get

$$\begin{aligned} \nabla_r(\mu) &\leq \bar{\omega}(q) - \bar{\mu}(q) \\ &\leq \Psi_{uqt}(\omega) - \Psi_{uqt}(\mu). \end{aligned}$$

Case 2.2: $\bar{\omega}(q) > \max\{\bar{\omega}(v), \bar{\omega}(t)\}$. This implies that $u$, $v$, $q$ and $t$ are four distinct points. Lemma 4.6 implies that for any $w \in \{u, v\}$,

$$\mu(w,r) = \mu(w,q) + \beta \qquad \text{and} \tag{12}$$

$$\bar{\omega}(w) = \omega(w,q) + \omega(w,r). \tag{13}$$

Using (12), $\bar{\mu}(w) \leq \mu(w,r) + \mu(w,q) \leq 2\mu(w,r) - \beta$, which, together with (13) yields

$$\bar{\omega}(w) - \bar{\mu}(w) \;\geq\; [\omega(w,r) + \omega(w,q)] - [2\mu(w,r) - \beta] \;=\; \omega(w,q) - \omega(w,r) + \beta. \tag{14}$$

We have now two more subcases.

Case 2.2.1: $\omega(w,q) = \omega(w,r) + \beta$ for some $w \in \{u, v\}$. Then, by (14), no matter whether $w = u$ or $w = v$, we have

$$\begin{aligned} \nabla_r(\mu) &\leq 2\beta \\ &= \omega(w,q) - \omega(w,r) + \beta \\ &\leq \bar{\omega}(w) - \bar{\mu}(w) \\ &\leq \Psi_{uvt}(\omega) - \Psi_{uvt}(\mu). \end{aligned}$$

Case 2.2.2: $\omega(u,q) = \omega(v,q) = \omega(r,q) + 1$. By Lemma 4.1(b), $\omega(q,r) \leq \omega(u,q) = \omega(v,q)$. Also, $\bar{\omega}(u) \leq \bar{\omega}(v)$ and $\omega(u,q) = \omega(v,q)$ together imply $\omega(u,r) \leq \omega(v,r)$. Thus, using (14) for $w = u$ and (13) for $w = v$ we obtain

$$\begin{aligned} \nabla_r(\mu) &\leq \bar{\omega}(q) - \bar{\mu}(q) \\ &= 2\omega(q,r) + \beta - \bar{\mu}(q) \\ &\leq \omega(u,q) + \omega(v,q) + \beta - \bar{\mu}(q) \\ &\leq [\omega(u,q) - \omega(u,r) + \beta] + [\omega(v,r) + \omega(v,q)] - \bar{\mu}(q) \\ &\leq \bar{\omega}(u) + \bar{\omega}(v) - \bar{\mu}(u) - \bar{\mu}(q) \\ &\leq \Psi_{uvt}(\omega) - \Psi_{uqt}(\mu). \end{aligned}$$

We now have examined all cases, completing the proof of (8) and the theorem. □

### 4.4 A Lower Bound for Uniform Spaces

**Theorem 4.10** *Let $\mathcal{A}$ be a deterministic online algorithm for the weighted 2-server problem in a uniform space with at least three points. Then the competitive ratio of $\mathcal{A}$ is at least 5.*

*Proof:* Let $a, b, c$ be three distinct points. Let the initial configuration be $(a, b)$, that is, the expensive server is at $a$ and the cheap server at $b$. The adversary strategy is to always requests the point in $\{a, b, c\}$ not occupied by $s_1$, $s_2$. We divide the request sequence into *phases*, where each phase starts right after the previous phase ends (except the first phase that starts on first request), and it ends on the first request served by $\mathcal{A}$ with $s_1$. The adversary will end the request sequence only after the last phase is complete. Let $k$ be the number of phases and $\ell_i$ be the number of requests served by $s_2$ in phase $i = 1, \ldots, k$. Thus the algorithm pays $\ell_i\beta + 1$ in phase $i$.

Define $A_i$ to be the minimum cost of serving phases $1, 2, \ldots, i$ and ending in a configuration with the expensive server at $a$ (see Section 4.1.) In other words, $A_i = \min_{x\in\{b,c\}} \omega(a, x)$, where $\omega$ is the work function at the end of phase $i$. We define analogous quantities $B_i$ for $b$ and $C_i$ for $c$. For $i = 0$, we have $A_0 = 0$, $B_0 = 1$ and $C_0 = 1$.

For each $i = 0, \ldots, k$, let

$$\phi_i \quad = \quad 2(A_i + B_i + C_i) - \min\{A_i, B_i, C_i\}.$$

Note that $\phi_i$ equals approximately (up to an additive constant) five times the optimal cost. The idea of the proof is to show that it is also an approximate lower bound on the cost of the algorithm. More specifically, we claim that for every phase $i$, we have

$$\phi_i - \phi_{i-1} \quad \leq \quad \ell_i\beta + 1 + 5\beta. \tag{15}$$

This implies the theorem as follows. Let $cost_{\mathcal{A}}$ be the total cost of $\mathcal{A}$. The total optimal cost is $\min \omega_k$. By summing over all $k$ phases, (15) implies that

$$\begin{aligned} cost_{\mathcal{A}} + 5k\beta \quad &\geq \quad \phi_k - \phi_0 \\ &\geq \quad 5 \min \omega_k - 4. \end{aligned}$$

Since $\mathcal{A}$ pays at least 1 in each phase, we have $cost_{\mathcal{A}} \geq k$, and the theorem follows by taking $\beta$ arbitrarily small.

So it is sufficient to prove (15). Consider a fixed phase $i$. Since the formula for $\phi_i$ is symmetric, we can rename the points $a$, $b$, and $c$ so that at the beginning of phase $i$ server $s_1$ is at $a$ and that $B_i \leq C_i$. During this phase, the adversary alternates requests to $b$ and $c$, for the total of $\ell_i + 1$ requests.

If the adversary started with $s_1$ at $b$, he can move $s_2$ to $c$ at the beginning of the phase, if necessary, and server all other requests at no cost, ending with $s_1$ at $b$. So $B_i \leq B_{i-1} + \beta$, Similarly, $C_i \leq C_{i-1} + \beta$. By a similar argument, considering the cases when the adversary starts at $a$ or $b$, we have $A_i \leq \min\{A_{i-1} + \ell_i\beta + \beta, B_{i-1} + 1 + \beta\}$. Putting it all together, we have

$$\begin{aligned} \phi_i - \phi_{i-1} \quad &\leq \quad (A_i + \max\{A_i, B_i\} + B_i + 2C_i) - (A_{i-1} + 2B_{i-1} + 2C_{i-1}) \\ &= \quad (A_i - A_{i-1}) + \max\{A_i - B_{i-1}, B_i - B_{i-1}\} + (B_i - B_{i-1}) + 2(C_i - C_{i-1}) \\ &\leq \quad (\ell_i\beta + \beta) + (1 + \beta) + 3\beta \\ &= \quad \ell_i\beta + 1 + 5\beta. \end{aligned}$$

This completes the proof of (15) and the theorem. $\square$

## 5 The Weighted 2-Point Request Problem

In this section we study the modification of the weighted 2-server problem in which each request is specified by two points, say $\{r, s\}$. In response to this request, the algorithm must move one server to $r$ and the other to $s$. The decision to be made is which server to move to which point.

This can be viewed as a subproblem of the weighted 2-server problem: Replace the 2-point request $\{r, s\}$ by a long sequence $(rs)^*$. Any competitive 2-server algorithm eventually moves his servers to $r$ and $s$. In this way any $R$-competitive weighted 2-server algorithm yields a $R$-competitive algorithm for the weighted 2-point request problem.

On the other hand, in the limit for $\beta \to 0$, we obtain the 2-point request problem with one server studied in [13], which in turn contains the closely related cow-path problem [2, 26]. This yields a lower bound of 9. We prove a matching upper bound of 9 for the algorithm $\text{WFA}_3$ which at each step minimizes the cost of the move plus three times the optimal cost of the new configuration.

Work functions extend naturally to the 2-point request problem. However, since we require that both points in the request are covered, the work function after requesting $\{r, s\}$ has only two relevant values on the two possible configurations: $(r, s)$ and $(s, r)$.

**Algorithm** $\text{WFA}_3$**:** Let $(x, y)$ be the current configuration, $\{r, s\}$ the new request, and $\omega'$ the new work function. If $xr + \beta \cdot ys + 3\omega'(r, s) \le xs + \beta \cdot yr + 3\omega'(s, r)$, then move to $(r, s)$, otherwise move to $(s, r)$.

**Theorem 5.1** $\text{WFA}_3$ *is $R$-competitive for the weighted 2-point request problem in any metric space, where $R = (9 - 3\beta)/(1 + \beta)$.*

*Proof:* Let $(x, y)$ be the current configuration and let $\omega_x = \omega(x, y)$ and $\omega_y = \omega(y, x)$ denote the work function values when the expensive server is at $x$ and $y$, respectively. Also, write $d = xy$.

First, we claim that $\text{WFA}_3$ satisfies the following invariant:

$$3(\omega_x - \omega_y) \quad \le \quad (1 + \beta)d \tag{16}$$

To prove (16), suppose that the previous configuration of $\text{WFA}_3$ was $(u, v)$. By the choice of $\text{WFA}_3$ and the triangle inequality, we have $3(\omega_x - \omega_y) \le uy + \beta \cdot vx - ux - \beta \cdot vy \le (1 + \beta)d$.

The proof of the theorem is by amortized analysis. We use the following potential function:

$$\Phi \quad = \quad \left[ 2d + \frac{6}{1+\beta}(\omega_x - \omega_y) \right]^+ - R\omega_x$$

where $[\alpha]^+ = \max(\alpha, 0)$.

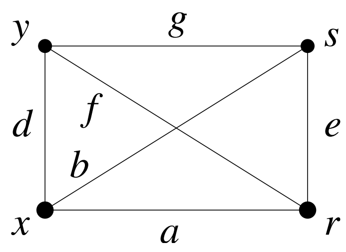
Consider now one move, on request $\{r, s\}$. Let $(r, s)$, $\omega'_r$, and $\omega'_s$ be the new configuration and work function values, after serving the request $\{r, s\}$. Recall that $d = xy$. Denote the other distances as follows: $e = rs$, $a = xr$, $b = xs$, $f = yr$, and $g = ys$ (see the figure below).

The cost of the algorithm for serving the new request is $\Delta cost = a + \beta g$, and the new work function is

$$\omega'_r \;=\; \min \left\{ \begin{array}{c} \omega_x + a + \beta g \\ \omega_y + f + \beta b \end{array} \right\} \qquad\qquad \omega'_s \;=\; \min \left\{ \begin{array}{c} \omega_x + b + \beta f \\ \omega_y + g + \beta a \end{array} \right\}$$

Since in the current step $\text{WFA}_3$ moved to $(r, s)$ and not $(s, r)$, we have

$$a + \beta g + 3\omega'_r \quad \le \quad b + \beta f + 3\omega'_s. \tag{17}$$




Denote by $\Phi'$ the potential value after the move, and $\Delta\Phi = \Phi' - \Phi$. We claim that

$$\Delta cost + \Delta\Phi \quad \leq \quad 0. \tag{18}$$

To prove (18), we distinguish several cases according to the possible values of $\Phi'$, $\omega'_r$, and $\omega'_s$. The derivations below use inequalities (16) (17), $R \geq 1$, and the triangle inequality.
Case 1: $(1+\beta)e + 3(\omega'_r - \omega'_s) \leq 0$. Then $\Phi' = -R\omega'_r$. We have two sub-cases.

Case 1.1: $\omega'_r = \omega_x + a + \beta g$. Then

$$\Delta cost + \Delta\Phi \;\leq\; a + \beta g - R(\omega_x + a + \beta g) + R\omega_x \;\leq\; 0$$

Case 1.2: $\omega'_r = \omega_y + f + \beta b$. Then, using (16), we have

$$\begin{aligned}
\Delta cost + \Delta\Phi &\leq a + \beta g - R(\omega_y + f + \beta b) - \left[2d + \frac{6}{1+\beta}(\omega_x - \omega_y) - R\omega_x\right] \\
&= a + \beta g - Rf - R\beta b - 2d + \frac{3-3\beta}{1+\beta}(\omega_x - \omega_y) \\
&\leq a + \beta g - f - \beta b - 2d + (1-\beta)d \\
&= (a - d - f) + \beta(g - d - b) \;\leq\; 0
\end{aligned}$$

Case 2: $(1+\beta)e + 3(\omega'_r - \omega'_s) \geq 0$. We have four sub-cases.

Case 2.1: $\omega'_r = \omega_x + a + \beta g$ and $\omega'_s = \omega_x + b + \beta f$. From (17), we have $a + \beta g \leq b + \beta f$. Thus

$$\begin{aligned}
\Delta cost + \Delta\Phi &\leq a + \beta g + \left[2e + \frac{6}{1+\beta}(a + \beta g - b - \beta f) - R(\omega_x + a + \beta g)\right] + R\omega_x \\
&= \frac{2}{1+\beta}\big[(e - b - a) + \beta(e - f - g) + 2\beta(a + \beta g) - 2(b + \beta f)\big] \;\leq\; 0
\end{aligned}$$

Case 2.2: $\omega'_r = \omega_x + a + \beta g$ and $\omega'_s = \omega_y + g + \beta a$. Then

$$\begin{aligned}
\Delta cost + \Delta\Phi &\leq a + \beta g + \left[2e + \frac{6}{1+\beta}(\omega_x + a + \beta g - \omega_y - g - \beta a) - R(\omega_x + a + \beta g)\right] \\
&\quad - \left[2d + \frac{6}{1+\beta}(\omega_x - \omega_y) - R\omega_x\right] \\
&= 2(e - d - a - g) - 4(1-\beta)g \;\leq\; 0
\end{aligned}$$

Case 2.3: $\omega'_r = \omega_y + f + \beta b$ and $\omega'_s = \omega_y + g + \beta a$. By substituting, and using inequalities (16) and $\beta \le 1$, we have

$$\begin{aligned}
\Delta cost + \Delta\Phi &\le a + \beta g + \left[2e + \frac{6}{1+\beta}(\omega_y + f + \beta b - \omega_y - g - \beta a) - R(\omega_y + f + \beta b)\right] \\
&\quad - \left[2d + \frac{6}{1+\beta}(\omega_x - \omega_y) - R\omega_x\right] \\
&= \frac{1-5\beta}{1+\beta}a - \frac{6-\beta-\beta^2}{1+\beta}g + 2e + \left[\frac{3-3\beta}{1+\beta}(\omega_x - \omega_y) - 2d\right] - \frac{3-3\beta}{1+\beta}(f + \beta b) \\
&\le \frac{1-5\beta}{1+\beta}a - 2g + 2e - (1+\beta)d - \frac{3-3\beta}{1+\beta}f.
\end{aligned}$$

The rest of the derivation depends on the value of $\beta$. If $\beta \le \frac{1}{5}$, then $1 + \beta \ge 1 \ge \frac{1-5\beta}{1+\beta}$ and

$$\Delta cost + \Delta\Phi \le \frac{1-5\beta}{1+\beta}(a - d - f) + 2(e - f - g) \le 0$$

If $\frac{1}{5} \le \beta \le 1$, then $1 + \beta \ge \frac{5\beta-1}{1+\beta}$ and

$$\Delta cost + \Delta\Phi \le \frac{5\beta-1}{1+\beta}(e - a - d - g) + \frac{3-3\beta}{1+\beta}(e - f - g) \le 0$$

Case 2.4: $\omega'_r = \omega_y + f + \beta b$ and $\omega'_s = \omega_x + b + \beta f$. Then using (17), we have

$$\begin{aligned}
\Delta cost + \Delta\Phi &\le a + \beta g + \left[2e + \frac{6}{1+\beta}(\omega'_r - \omega'_s) - R\omega'_r\right] - \left[2d + \frac{6}{1+\beta}(\omega_x - \omega_y) - R\omega_x\right] \\
&= a + \beta g + 2e - 2d + 3(\omega'_r - \omega'_s) - \frac{6}{1+\beta}(\omega'_r - \omega_y) - \frac{3-3\beta}{1+\beta}(\omega'_s - \omega_x) \\
&\le a + \beta g + 2e - 2d + (b + \beta f - a - \beta g) - \frac{6}{1+\beta}(f + \beta b) - \frac{3-3\beta}{1+\beta}(b + \beta f) \\
&= 2(e - d - b - f) - 4(1-\beta)f \le 0
\end{aligned}$$

We have now examined all cases, completing the proof of (18).

The $R$-competitiveness of $\text{W}\small{\text{FA}}_3$ follows from (18) by amortization, as follows.

Denote the initial configuration by $(x_0, y_0)$. The initial work function is $\omega_0(x_0, y_0) = 0$ and $\omega_0(y_0, x_0) = (1+\beta)d_0$, where $d_0 = x_0y_0$. Substituting it into the definition of the potential, we obtain that the initial potential is $\Phi_0 = [-4d_0]^+ = 0$.

Let the final work function after a request sequence $\varrho$ be $\mu$ and the configuration of $\text{W}\small{\text{FA}}_3$ be $\{u, v\}$ and denote $h = uv$. Since $\Phi_0 = 0$, by adding up all inequalities (18) over all requests, we obtain that that total cost of $\text{W}\small{\text{FA}}_3$ is

$$cost(\varrho) \le R\mu_u - \left[2h + \frac{6}{1+\beta}(\mu_u - \mu_v)\right]^+$$

Trivially, we have $cost(\varrho) \leq R\mu_u$. Substituting $R = (9-3\beta)/(1+\beta)$ and Using (16), we also have

$$\begin{aligned}
cost(\varrho) &\leq \frac{9-3\beta}{1+\beta}\mu_u - 2h - \frac{6}{1+\beta}(\mu_u - \mu_v) \\
&= \frac{3(1-\beta)}{1+\beta}\mu_u - 2h + \frac{6}{1+\beta}\mu_v \\
&\leq \frac{3(1-\beta)}{1+\beta}\mu_v + (1-\beta)h - 2h + \frac{6}{1+\beta}\mu_v \\
&= \frac{9-3\beta}{1+\beta}\mu_v - (1+\beta)h \\
&\leq R\mu_v.
\end{aligned}$$

Thus $cost(\varrho) \leq R \cdot \min(\mu_u, \mu_v)$, completing the proof. □

# 6 Final Comments

Many open problems remain. For example, no competitive algorithm for the weighted 2-server problem in arbitrary spaces is known, and the lower bound of 10.12 from [23] can probably be improved. This lower bound shows that the optimal competitive ratio for the weighted 2-point request problem is strictly smaller than for the general weighted 2-server problem. Another research direction would be to determine the competitive ratio for uniform spaces and arbitrary number of servers $k \geq 2$.

The weighted 2-server problem is related to the CNN problem from [23]. In this problem, we have one server in the plane. Each request is a point $(x, y)$, and to serve this request we need to move the server to some point with x-coordinate $x$ or with y-coordinate $y$. The special case when the requests are restricted to some line in the plane is equivalent to a weighted 2-server problem on the line. Very recently, René Sitters, Leen Stougie and Willem E. de Paepe have announced a constant competitive algorithm for the CNN problem [27]. Their algorithm has a huge competitive ratio; getting tighter bounds remains an attractive open problem.

In our paper we focused on the asymptotic competitive ratios, with $\beta \to 0$. It would be of some interest to determine the tight competitive ratios for any fixed $\beta$. For example, in the weighted 2-point request problem from Section 5, for $\beta = 1$ we get $R = 3$, although the best ratio is 1 (since both servers have the same weight, at each step we move according to the minimum matching). We conjecture that the optimal algorithm is $\textsc{Wfa}_\gamma$ for some $\gamma$ that converges to 3 with $\beta \to 0$.

**Acknowledgements.** We would like to thank the anonymous referees for pointing out some mistakes and for insightful comments that helped us improve the presentation.

# References

[1] D. Achlioptas, M. Chrobak, and J. Noga. Competitive analysis of randomized paging algorithms. *Theoretical Computer Science*, 234:203–218, 2000.

[2] R. Baeza-Yates, J. Culberson, and G. Rawlins. Searching in the plane. *Information and Computation*, 106:234–252, 1993.

[3] Y. Bartal, A. Blum, C. Burch, and A. Tomkins. A polylog(n)-competitive algorithm for metrical task systems. In *Proc. 29th Symp. Theory of Computing*, pages 711–719, ACM, 1997.

[4] Y. Bartal, M. Chrobak, and L. L. Larmore. A randomized algorithm for two servers on the line. *Information and Computation*, 158:53–69, 2000.

[5] Y. Bartal and E. Grove. The harmonic $k$-server algorithm is competitive. *Journal of the ACM*, 47(1):1–15, 2000.

[6] W. Bein and L. L. Larmore. Knowledge state algorithms. Unpublished manuscript, 2002.

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

[8] A. Blum, H. Karloff, Y. Rabani, and M. Saks. A decomposition theorem and lower bounds for randomized server problems. In *Proc. 33rd Symp. Foundations of Computer Science*, pages 197–207, IEEE, 1992.

[9] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis.* Cambridge University Press, 1998.

[10] M. Chrobak, H. Karloff, T. H. Payne, and S. Vishwanathan. New results on server problems. *SIAM Journal on Discrete Mathematics*, 4:172–181, 1991.

[11] M. Chrobak and L. L. Larmore. On fast algorithms for two servers. *Journal of Algorithms*, 12:607–614, 1991.

[12] M. Chrobak and L. L. Larmore. An optimal online algorithm for $k$ servers on trees. *SIAM Journal on Computing*, 20:144–148, 1991.

[13] M. Chrobak and L. L. Larmore. The server problem and on-line games. In *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, volume 7, pages 11–64, 1992.

[14] M. Chrobak and L. L. Larmore. Metrical task systems, the server problem, and the work function algorithm. In A. Fiat and G. J. Woeginger, editors, *Online Algorithms: The State of the Art*, pages 74–94. Springer-Verlag, 1998.

[15] M. Chrobak, L. L. Larmore, C. Lund, and N. Reingold. A better lower bound on the competitive ratio of the randomized 2-server problem. *Information Processing Letters*, 63:79–83, 1997.

[16] M. Chrobak and J. Sgall. The weighted 2-server problem. In *17th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 1770 of *Lecture Notes in Computer Science*, pages 593–604. Springer-Verlag, 2000.

[17] D. Coppersmith, P. G. Doyle, P. Raghavan, and M. Snir. Random walks on weighted graphs and applications to on-line algorithms. *Journal of the ACM*, 40:421–453, 1993.

[18] L. Epstein, C. Imreh, and R. van Stee. More on weighted servers or FIFO is better than LRU. In *Proc. 27th Symp. on Mathematical Foundations of Computer Science (MFCS'02)*, volume 2420 of *Lecture Notes in Computer Science*, pages 257–268, 2002.

[19] E. Feuerstein, S. S. Seiden, and A. S. de Loma. The related server problem. Unpublished manuscript, 1999.

[20] A. Fiat and M. Ricklin. Competitive algorithms for the weighted server problem. *Theoretical Computer Science*, 130:85–99, 1994.

[21] E. Koutsoupias and C. Papadimitriou. On the $k$-server conjecture. *Journal of the ACM*, 42:971–983, 1995.

[22] E. Koutsoupias and C. Papadimitriou. The 2-evader problem. *Information Processing Letters*, 57:249–252, 1996.

[23] E. Koutsoupias and D. Taylor. The CNN problem and other $k$-server variants. In *The 17th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 1770 of *Lecture Notes in Computer Science*, pages 581–592. Springer-Verlag, 2000.

[24] M. Manasse, L. A. McGeoch, and D. Sleator. Competitive algorithms for server problems. *Journal of Algorithms*, 11:208–230, 1990.

[25] L. McGeoch and D. Sleator. A strongly competitive randomized paging algorithm. *Algorithmica*, 6(6):816–825, 1991.

[26] C. H. Papadimitriou and M. Yannakakis. Shortest paths without a map. *Theoretical Computer Science*, 84:127–150, 1991.

[27] R. Sitters, L. Stougie, and W. E. de Paepe. A competitive algorithm for the general 2-server problem. In *Proc. 29th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 2719 of *Lecture Notes in Computer Science*, pages 624–636. Springer-Verlag, 2003.