The general two-server problem

Sitters, R.A.; Stougie, L.

Published: 01/01/2003

Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

- A submitted manuscript is the author's version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

Citation for published version (APA):
The general two-server problem

René Sitters and Leen Stougie

1 Department of Mathematics and Computer Science
Technische Universiteit Eindhoven
P.O.Box 513, 5600 MB Eindhoven, The Netherlands
{r.a.sitters, l.stougie}@tue.nl

2 CWI, P.O.Box 94079, 1090 GB Amsterdam, the Netherlands

Abstract. We consider the general on-line two-server problem in which at each step both servers receive a request, which is a point in a metric space. One of the servers has to be moved to its request. The special case where the requests are points on the real line is known as the CNN-problem. It has been a famous open question in on-line optimization if an algorithm with a constant competitive ratio exists for this problem. We answer this question in the affirmative sense by providing the first constant competitive algorithm for the general two-server problem on any metric space.

The basic result in this paper is a characterization of competitiveness for metrical service systems that seems much easier to use when looking for a competitive algorithm. The existence of a competitive algorithm for the general two-server problem follows rather easily from this result.

1 Introduction

In the general \( k \)-server problem we are given \( k \) servers each of which is moving in some metric space \( M_i, i = 1, \ldots, k \), starting in some prefixed point \( O_i \in M_i \). They are to serve requests \( r \in M_1 \times M_2 \times \cdots \times M_k \) which arrive one by one. A request \( r = (z_1, z_2, \ldots, z_k) \) is served by moving, for at least one \( i \), the server in space \( M_i \) to the point \( z_i \). The decision which server to move to the next request is irrevocable and has to be taken without any knowledge about future requests. The cost of moving the \( i \)-th server to \( z_i \) is equal to the distance travelled by this server from his current location to \( z_i \). The objective is to minimize the total cost to serve all given requests. We measure the performance of an on-line algorithm through competitive analysis. An on-line algorithm is \( c \)-competitive if, for any request sequence \( \sigma \), the algorithm’s cost are at most \( c \) times the cost of the optimal solution of the corresponding off-line problem plus an additive constant independent of \( \sigma \).

We can see the general \( k \)-server problem as a single server problem, moving in the metric space \( M = M_1 \times \ldots \times M_k \), and interpret the server positions and the requests in the description above as the “coordinates” of the single server and the requests. Interpreted in this way, it is a special case of a metrical service system, in which each request can be any subset of the metric space, and is served by moving the server to one of the points in the request. Metrical service systems were introduced in [5] to provide a formalism for investigating a wide variety of on-line optimization problems. A precise definition is given in Section 2. In the same section we derive the basic theorem of this paper. It provides a sufficient condition for the existence of constant competitive algorithms.
for general metrical service systems. The result on the general two-server problem then becomes a matter of verifying this condition.

The general \( k \)-server problem is a natural generalization of the well-known \emph{k-server problem} for which \( M_1 = M_2 = \ldots = M_k \) and \( z_1 = z_2 = \ldots = z_k \) at each time step. The \( k \)-server problem was introduced by Manasse, McGeoch and Sleator [12], who proved a lower bound of \( k \) on the competitive ratio of any deterministic algorithm for any metric space with at least \( k + 1 \) points and posed the well-known \( k \)-server conjecture saying that there exists a \( k \)-competitive algorithm for any metric space. The conjecture has been proved for \( k = 2 \) [12] and some special metric spaces [3][4]. For \( k \geq 3 \) the current best upper bound of \( 2k - 1 \) is given by Koutsoupias and Papadimitriou [10].

The \emph{weighted k-server problem} turns out to be much harder. In this problem a weight is assigned to each server and the total cost is the sum of the weighted distances. Fiat and Ricklin [8] prove that for any metric space there exists a set of weights such that the competitive ratio of any deterministic algorithm is at least \( k\Omega(k) \). For a uniform metric space, on which the problem is called the weighted paging problem, Feuerstein et al. [7] give a 6.275-competitive algorithm. For \( k = 2 \) Chrobak and Sgall [6] provided a 5-competitive algorithm and proved that no better competitive ratio is possible.

A weighted \( k \)-server algorithm is called competitive if the competitive ratio is independent of the weights. For a general metric space no competitive algorithm is known yet even for \( k = 2 \). It is easy to see that the general \( k \)-server problem is a generalization of the weighted \( k \)-server problem as well.

The general two-server problem in which both servers move on the real line has become well-known as the CNN-problem. Koutsoupias and Taylor [11] emphasize the importance of the CNN-problem as one of the simplest problems in a rich class of so-called sum-problems [1]. In the sum-problem each of a set of systems gets a request and only one system has to serve this request.

Koutsoupias and Taylor [11] prove a lower bound of \( 6 + \sqrt{17} \) on the competitive ratio of any deterministic on-line algorithm for the general two-server problem, through an instance of the weighted two-server problem on the real line. They also conjecture that the \emph{generalized work function algorithm} has constant competitive ratio for the general two-server problem. It seems to be a bad tradition of multiple-server problems to keep unsettled conjectures. For the general two-server problem the situation was even worse than for the \( k \)-server problem: the question if any algorithm exists with constant competitive ratio remained unanswered.

In Section 3 we answer this question affirmatively, by designing an algorithm and prove a rough upper bound of 44,800 on its competitive ratio. Our algorithm is a combination of the well-known balance algorithm and the generalized work function algorithm. The result is merely checking the condition of the general theorem for metrical service systems in Section 2, announced above.

Optimal off-line solutions of metrical service systems can easily be found by dynamic programming (see [1]), which yields a \( O(n^k) \) time algorithm for the general \( k \)-server problem. For the classical \( k \)-server problem this running time can be reduced to \( O(kn^2) \) by formulating it as a min-cost flow problem [3]. No such improvement should be expected for the general \( k \)-server problem since the problem is NP-hard as we will show in Section 4.
2 Competitiveness of metrical service systems

A metrical service system $S = (\mathbb{M}, \mathcal{R})$ is specified by a metric space $\mathbb{M}$ with distance function $d : \mathbb{M}^2 \to \mathbb{R}^+$ and a set of requests $\mathcal{R}$. Each request $r \in \mathcal{R}$ is a subset of $\mathbb{M}$. An instance of the system consists of an initial server position $O \in \mathbb{M}$ and a sequence of requests $\sigma = r_1, r_2, \ldots, r_n$. Every request $r_i$ must be served immediately and irrevocably by moving the server to a point $x_i \in r_i$, before the future requests, $r_{i+1}, r_{i+2}, \ldots$, are given. The cost of the solution is the length of the path in $\mathbb{M}$ followed by the server.

For any metrical service system we will define functions $\Psi_m(\sigma) \geq 0$ depending on the input sequence $\sigma$ and any integer $m \geq 1$. We show that if an algorithm $A$ exists for a metrical service system $S$ with the property that for some $m \geq 1$ and any sequence $\sigma$ the length of its path is at most $c$ times the length of the optimal path plus $\Psi_m(\sigma)$, then there exists an on-line algorithm $A'$ for this metrical service system producing a path of length at most $c'$ times the optimal path length, for some constant $c'$. Since $\Psi_m(\sigma) \geq 0$ the reverse of this statement is obviously always true, i.e. if $A'$ is $c'$ competitive then the length of any solution is certainly no more than $c'$ times the optimal path plus $\Psi_m(\sigma)$. If we would choose $\Psi_m(\sigma) \equiv 0$ then the statement is clearly a tautology. The statement is formalized in Theorem 1 and the usefulness of our choice of $\Psi_m(\sigma)$ follows from Section 3 in which we show how a competitive algorithm for the general two-server problem follows quite easily from Theorem 1. Roughly speaking we define $\Psi_m(\sigma)$ as the minimum sum of lengths of the paths of $m$ different adversaries, each serving the request sequence $\sigma$. What is meant by different adversaries will be formalized later. The theorem then says that if there exists an algorithm that is competitive against $m$ different adversaries then there exists an algorithm that is competitive against one adversary.

The competitive algorithm $A'$, which is called ONLINE from now on, is a combination of an algorithm $A$ with the property mentioned above and the generalized work function algorithm. The latter algorithm was introduced independently by several people and has shown to be competitive for several on-line problems. In fact we go along with Koutsoupias and Taylor [11] in conjecturing that it is competitive for the general two-server problem as well. The generalized work function algorithm bases its moves on the position of the on-line server and the values of the work function.

**Definition 1.** Given a metrical service system $S = (\mathbb{M}, \mathcal{R})$ with origin $O \in \mathbb{M}$, and request sequence $\sigma$ we define the work function $W_{O,\sigma} : \mathbb{M} \to \mathbb{R}^+$. For any point $s \in \mathbb{M}$, $W_{O,\sigma}(s)$ is the length of the shortest path that starts in $O$, ends in $s$ and serves $\sigma$. Omitting the restriction on the initial point we obtain an alternative work function $\hat{W}_{\sigma}$, i.e.; $\hat{W}_{\sigma}(s)$ is the length of the shortest path that ends in $s$ and serves $\sigma$.

We assume here that the work function is always well-defined, i.e. for any $\sigma = r_1, \ldots, r_n$ and point $s \in \mathbb{M}$ there are points $s_i \in r_i$ $(i = 1 \ldots n)$ such that $d(O, s_1) + d(s_1, s_2) + \ldots + d(s_{n-1}, s_n) + d(s_n, s) \leq d(O, t_1) + d(t_1, t_2) + \ldots + d(t_{n-1}, t_n) + d(t_n, s)$ for any set of points $t_i \in r_i$ $(i = 1 \ldots n)$. Notice that this implies $W_{O, r_1, \ldots, r_n}(s_i) = d(O, s_1) + d(s_1, s_2) + \ldots + d(s_{i-1}, s_i)$ for all $i \in \{1, \ldots, n\}$.

We will use the following properties, which are rather obvious. The work function on the empty string of requests is $W_{O,\emptyset}(s) = d(O, s)$, for all $s \in \mathbb{M}$. The work function is Lipschitz continuous: for any two points $s$ and $s'$ in $\mathbb{M}$, $|W_{O,\sigma}(s) - W_{O,\sigma}(s')| \leq d(s, s')$. It exhibits monotonicity with
respect to the request sequence for every \( s \in \mathbb{M} \): given any request sequence \( \sigma \) and any new request \( r \), \( W_{O,\sigma,r}(s) \geq W_{O,\sigma}(s) \), for all \( s \in \mathbb{M} \). Equality holds for all \( s \in r \).

Given a work function \( W_{O,\sigma} \) we say that point \( s \) is dominated by point \( t \) if \( W_{O,\sigma}(s) = W_{O,\sigma}(t) + d(s,t) \). We define the support of \( W_{O,\sigma} \) as
\[
\text{supp}(W_{O,\sigma}) = \{ t \in \mathbb{M} : t \text{ is not dominated by any other point} \}.
\]

If \( W_{O,\sigma,r} \) is a well-defined work function then \( \text{supp}(W_{O,\sigma,r}) \subseteq r \), since for any point \( s \) there is a point \( t \in r \) such that \( W_{O,\sigma,r}(s) = W_{O,\sigma,r}(t) + d(t,s) \), which implies that if \( s \in \text{supp}(W_{O,\sigma,r}) \) then \( s = t \). For more properties and a deeper analysis of work functions we refer to [2], [5].

The generalized work function algorithm is a work function based algorithm parameterized by some constant \( \lambda < 1 \). We call it \( \lambda \text{-WFA} \). For any request sequence \( \sigma \) and any new request \( r \), \( \lambda \text{-WFA} \)-moves the server from the position \( s \) it had after serving \( \sigma \) to point
\[
\lambda' = \arg\min_{t \in \mathbb{M}} W_{O,\sigma,r}(t) + \lambda d(s,t).
\]
Note that this minimum may not be well-defined if the request contains infinitely many points of the metric space. In Theorem 1 we assume that the minimum is always attained for some \( s' \in \mathbb{M} \).

As a result of any such a \( \lambda \text{-WFA} \)-move, we have for any point \( t \in \mathbb{M} \) that
\[
W_{O,\sigma,r}(s') + \lambda d(s,s') \leq W_{O,\sigma,r}(t) + \lambda d(s,t).
\]
Therefore, in particular,
\[
W_{O,\sigma,r}(s') \leq W_{O,\sigma,r}(s) - \lambda d(s,s'), \tag{1}
\]
and for any \( t \in \mathbb{M} \)
\[
W_{O,\sigma,r}(s') \leq W_{O,\sigma,r}(t) + \lambda d(s',t). \tag{2}
\]
The choice \( \lambda < 1 \) together with (2) imply that \( s' \) is not dominated by any other point, whence \( s' \in \text{supp}(W_{O,\sigma,r}) \subseteq r \). We see that if the moves of \( \lambda \text{-WFA} \) are well-defined then the choice of \( \lambda < 1 \) ensures that the point \( s' \) always serves the last request.

As announced before, we define an algorithm \( \text{ONLINE} \) as a combination of \( \lambda \text{-WFA} \) and some algorithm \( A \). \( \text{ONLINE} \) works in phases. The final point of a phase is the starting point of the next phase. This is the only information taken from one phase to the next phase. The phases of the algorithm induce a partition of the request sequence \( \sigma \). Let \( \sigma_j \) be the subsequence of requests served in the \( j \)-th phase of \( \text{ONLINE} \); i.e., if \( J \) is the total number of phases then \( \sigma = \sigma_1, \ldots, \sigma_J \). We denote by \( A(s,\sigma) \) and \( \text{OPT}(s,\sigma) \) the cost of the algorithm \( A \) and the cost of the optimal path starting in \( s \) and serving request sequence \( \sigma \). Thus, \( \text{OPT}(O,\sigma) = \min_{t \in \mathbb{M}} W_{O,\sigma}(t) \). In the description of a generic phase \( j \) of the algorithm we use \( \sigma^h_j \) for the subsequence of the first \( h \) requests of \( \sigma_j \). The starting point in phase \( j \) is denoted by \( O_j \). The algorithm is parameterized by \( \gamma (\gamma \geq 1) \) and \( \lambda (0 < \lambda < 1) \).

**Phase \( j \) of \( \text{ONLINE}(A,\gamma,\lambda) \):**

Given \( \sigma^h_j \), if \( A(O_j,\sigma^h_j) \leq \gamma \text{OPT}(O_j,\sigma^h_j) \), move the server according to \( A \) and continue the phase. Otherwise, \( \sigma_j = \sigma^h_j \); move the server according to \( \lambda \text{-WFA} \), based on the work function \( W_{O,\sigma,1,...,\sigma_j} \), and start a new phase.
Thus, each phase contains only one $\lambda$-WFA-move and this moves concludes the phase. We emphasize that the work function employed in the $\lambda$-WFA-move is defined over the complete input sequence, released sofar. In the sequel we refer to the moves made according to $A$ as $A$-moves.

We are now ready to define our main theorem. The idea behind it is best explained by using adversaries serving the same request sequence, but knowing the sequence in advance. Whereas our algorithm $A$ has to start from $O$, these adversaries are allowed to choose their starting point. On the other hand we restrict the paths of the adversaries by requiring that they are sufficiently different. We do so by requiring that the points where the $m$ paths end are pairwise $\sigma$-independent.

**Definition 2.** Given a sequence of requests $\sigma$ we say that the points $s_1$ and $s_2$ are $\sigma$-independent if $d(s_1, s_2) \geq \overline{W}_\sigma(s_1) + \overline{W}_\sigma(s_2)$. Otherwise, the points are called $\sigma$-dependent.

The theorem says that if $A$ is competitive against the optimal path plus $m$ such adversaries, then algorithm Online is competitive against one adversary starting also in $O$, i.e., against the length of the optimal path.

**Theorem 1.** Let $A$ be an algorithm for some metrical service system $S$, with origin $O$, on which the work function and $\lambda$-WFA are well-defined. If there exist constants $c$ and $m \geq 2$ such that for any sequence $\sigma$ and any set $\{s_1, \ldots, s_m\}$ of $m$ pairwise $\sigma$-independent points

$$A(O, \sigma) \leq c \text{Opt}(O, \sigma) + c \sum_{h=1}^{m} \overline{W}_\sigma(s_h),$$

then Online($A, \gamma, \lambda$) with $\gamma = c(1 - \lambda)(1 - 2\lambda)^{-1}$ is $(2 + 5/4c)\lambda^{-2}(1/2 - 5\lambda)^{1-m}$-competitive for $S$, for every $\lambda < 1/10$.

If algorithm $A$ satisfies (3) for $m = 1$ then $A$ itself is $2c$ competitive for $S$. Restricting $m$ to be at least 2 in the theorem allowed us to prove a slightly better ratio of Online. In Section 3 we show that for the general two-server problem the balance algorithm is competitive against the optimal path plus 3 such adversaries. A competitive algorithm follows then directly from the theorem.

As a preliminary to the proof we define functions and prove some of their properties. Given any function $f : \mathbb{X} \rightarrow \mathbb{R}$, defined on some metric space $\mathbb{X}$, define, for $i \geq 1$ and $0 < \beta < 1$, the functions $G_f^i : \mathbb{X}^i \rightarrow \mathbb{R}$ as

$$G_f^i(x_1, \ldots, x_i) = f(x_i) - \min_{h \leq i} \{ f(x_h) + \beta d(x_h, x_i) \},$$

and for $n \geq 1$ and $0 < \alpha < 1/2$ the functions $F_f^n : \mathbb{X}^n \rightarrow \mathbb{R}$ as

$$F_f^n(x_1, \ldots, x_n) = \sum_{i=1}^{n} \alpha^{i-1} G_f^i(x_1, \ldots, x_i).$$

We denote $F_f^n = \min_{x_1, \ldots, x_n \in \mathbb{X}} \{ F_f^n(x_1, \ldots, x_n) \}$.

**Lemma 1.** For any two functions, $f$ and $g$, and any two sequences, $(x_1, \ldots, x_n)$ and $(y_1, \ldots, y_n)$ such that $f(x_i) \geq g(y_i)$ for all $i \in \{1, \ldots, n\}$ we have

$$F_f^n(x_1, \ldots, x_n) - F_f^n(y_1, \ldots, y_n) \geq \sum_{i=1}^{n} \left( \alpha^{i-1} - \sum_{h=i+1}^{n} \alpha^{h-1} \right) (f(x_i) - g(y_i)) - \beta d(x_i, y_i) \sum_{h=i}^{n} \alpha^{h-1}.$$
Proof. Suppose that \( h_i < i \) is such that
\[
G_f^i(y_1, \ldots, y_i) = g(y_i) - (g(y_h) + \beta d(y_i, y_h)).
\]
Clearly,
\[
G_f^i(x_1, \ldots, x_i) = f(x_i) - \min_{h<i} \{ f(x_h) + \beta d(x_i, x_h) \} \geq f(x_i) - (f(x_h) + \beta d(x_i, x_h)).
\]
Hence, using the triangle inequality to bound \( -\beta d(x_i, x_h) \),
\[
G_f^i(x_1, \ldots, x_i) - G_g^i(y_1, \ldots, y_i) \geq f(x_i) - g(y_i) - (f(x_h) - g(y_h)) - \beta d(x_i, x_h) + \beta d(y_i, y_h)
\]
\[
\geq f(x_i) - g(y_i) - (f(x_h) - g(y_h)) - \beta d(x_i, y_i) - \beta d(x_h, y_h).
\]
Thus,
\[
F_f^n(x_1, \ldots, x_n) - F_g^n(y_1, \ldots, y_n)
\]
\[
= \sum_{i=1}^n \alpha_i^{-1} (G_f^i(x_1, \ldots, x_i) - G_g^i(y_1, \ldots, y_i))
\]
\[
\geq \sum_{i=1}^n \alpha_i^{-1} ( f(x_i) - g(y_i) - \beta d(x_i, y_i) - (f(x_h) - g(y_h)) - \beta d(x_h, y_h) )
\]
\[
\geq \sum_{i=1}^n ( \alpha_i^{-1} - \sum_{h=i+1}^n \alpha_h^{-1} ) ( f(x_i) - g(y_i) ) - \beta d(x_i, y_i) \sum_{h=i}^n \alpha_h^{-1}.
\]
\( \square \)

Lemma 2. Assume that \( F_f^n = F_f^n(x_1, \ldots, x_n) \). Any pair \( p, q \in \{1, \ldots, n\} \) satisfies
\[
|f(x_p) - f(x_q)| \leq (1 - 2\alpha)^{-1} \beta d(x_p, x_q).
\]

Proof. Assume without loss of generality that \( f(x_p) \geq f(x_q) \). Since the global minimum of \( F_f^n \) is attained in \((x_1, \ldots, x_n)\) we must have that \( F_f^n(x_1, \ldots, x_n) - F_f^n(x_1, \ldots, x_{p-1}, x_q, x_{p+1}, \ldots, x_n) \leq 0 \).

Now we apply Lemma 1 with \( g \equiv f \), \( y_p = x_q \) and \( y_i = x_i \) for \( i \neq p \).
\[
f(x_p) - f(x_q) \leq \left( \sum_{h=p}^n \alpha_h^{-1} \left( \alpha_p^{-1} - \sum_{h=p+1}^n \alpha_h^{-1} \right)^{-1} \beta d(x_p, x_q) \right.
\]
\[
\leq \left( \sum_{h=p}^n \alpha_h^{-1} \left( \alpha_p^{-1} - \sum_{h=p+1}^n \alpha_h^{-1} \right)^{-1} \beta d(x_p, x_q) \right.
\]
\[
= (\alpha_p^{-1}/(1 - \alpha))(\alpha_p^{-1} - \alpha_p/(1 - \alpha))^{-1} \beta d(x_p, x_q)
\]
\[
= (1 - 2\alpha)^{-1} \beta d(x_p, x_q).
\]
\( \square \)

Preliminary to the proof of Theorem 1 we derive an inequality which will be used several times. In the proof we choose \( \beta = (1 - 2\alpha)/5 \) implying that for any \( \alpha < 1/2 \) we have \( \alpha(1 + \beta) \leq (1 - \alpha)(1 - \beta) \) and for any \( n \geq 1 \) and any \( 1 \leq i \leq n \):
\[
\alpha^{-1} - \sum_{h=i+1}^n \alpha_h^{-1} - \beta \sum_{h=1}^n \alpha_h^{-1} = (1 - \beta) \alpha_i^{-1} - (1 + \beta) \sum_{h=i+1}^n \alpha_h^{-1}
\]
\[
= (1 - \beta) \alpha_i^{-1} - (1 + \beta)(\alpha_i - \alpha^n)/(1 - \alpha)
\]
\[
\geq (1 - \beta)(1 - \beta)/(1 - \alpha)
\]
\[
= (1 - \beta) \alpha^{n-1}.
\]
Proof of Theorem 1. For each \( j \geq 1 \) we define for \( n \geq 1 \) the functions \( G_j^i \), for \( i = 1, \ldots, n \), and \( F_j^n \), as in (4) and (5) by choosing \( f(s) = W_{O,\sigma_1,\ldots,\sigma_j-1}(s) \). We define \( \sigma_0 = \emptyset \).

From now on we let \( \beta = (1 - 2\alpha)/5 \) and \( \alpha = 1/2 - 5\lambda \), whence \( \beta = 2\lambda \). We also assume \( 0 < \lambda < 1 \) implying \( 0 < \alpha < 1/2 \).

First we show that \( F_j^n \) is a lower bound on the length of the optimal path serving \( \sigma_1, \ldots, \sigma_j-1 \) for every \( j \in \{1, \ldots, J + 1\} \) and any \( n \geq 1 \). From the definition, \( G_j^1(s) = W_{\sigma_1,\ldots,\sigma_j-1}(s) \) and \( G_j^1(s, \ldots, s) = 0 \), for all \( i > 1 \), whence \( F_j^n(s, \ldots, s) = W_{O,\sigma_1,\ldots,\sigma_j-1}(s) \) for any \( j \) and \( n \geq 1 \). In particular this implies that

\[
F_j^n(s_1, \ldots, s_n) = \min_{s \in S} F_j^n(s_1, \ldots, s_n) = \min_{s \in S} W_{O,\sigma_1,\ldots,\sigma_j-1}(s) = \text{Opt}(O, \sigma_1, \ldots, \sigma_j-1).
\]

(7)

Also, \( F_1^n(O, O, \ldots, O) = W_{O,\emptyset}(O) = 0 \), for any \( n \geq 1 \). In fact \( F_1^n = 0 \), for every \( n \geq 1 \), which follows from Lemma 1 applied to any series \( (s_1, \ldots, s_n) \):

\[
F_1^n(s_1, \ldots, s_n) - F_1^n(O, O, \ldots, O) \geq \sum_{i=1}^{n} (\alpha^{i-1} - \sum_{h=i}^{n} \alpha^{h-1}) (W_{O,\emptyset}(s_i) - W_{O,\emptyset}(O)) - \beta d(s_i, O) \sum_{h=i}^{n} \alpha^{h-1}
\]

\[
= \sum_{i=1}^{n} (\alpha^{i-1} - \sum_{h=i}^{n} \alpha^{h-1} - \beta \sum_{h=i}^{n} \alpha^{h-1}) d(s_i, O)
\]

\[
\geq (6) \sum_{i=1}^{n} (1 - \beta)\alpha^{i-1} d(s_i, O) \geq 0.
\]

Hence,

\[
F_1^n = F_1^n(O, O, \ldots, O) = 0.
\]

(8)

First we show that Theorem 1 follows from the next claim. Allowing \( \sigma_j \) to be empty, we assume, without loss of generality, that phase \( J \) ends with an \( A \)-move.

Claim. For each phase \( j \leq J - 1 \) we have \( F_{j+1}^n - F_j^n \geq \lambda \alpha^{n-1} \text{Opt}(O_j, \sigma_j) \).

Let \( C_j \) be the cost of the \( A \)-moves in phase \( j \). By definition of algorithm ONLINE, \( C_j \leq \gamma \text{Opt}(O_j, \sigma_j) \).

Let \( O'_j \) denote the position of the server after the last \( A \)-move in phase \( j \). Clearly,

\[
W_{O,\sigma_1,\ldots,\sigma_j}(O'_j) - W_{O,\sigma_1,\ldots,\sigma_j-1}(O_j) \leq 2\text{Opt}(O_j, \sigma_j) + d(O_j, O'_j) \leq (2 + \gamma)\text{Opt}(O_j, \sigma_j).
\]

Since the last move in phase \( j \), the move from \( O'_j \) to \( O_{j+1} \), is a \( \lambda \)-WFA-move, we use (1) to obtain

\[
W_{O,\sigma_1,\ldots,\sigma_j}(O_{j+1}) \leq W_{O,\sigma_1,\ldots,\sigma_j}(O'_j) - \lambda d(O'_j, O_{j+1}).
\]

Therefore,

\[
\sum_{j=1}^{J-1} d(O'_j, O_{j+1}) \leq 1/\lambda \sum_{j=1}^{J-1} (W_{O,\sigma_1,\ldots,\sigma_j}(O'_j) - W_{O,\sigma_1,\ldots,\sigma_j}(O_{j+1}))
\]

\[
\leq 1/\lambda(2 + \gamma) \sum_{j=1}^{J-1} \text{Opt}(O_j, \sigma_j) + 1/\lambda \sum_{j=1}^{J-1} (W_{O,\sigma_1,\ldots,\sigma_j-1}(O_j) - W_{O,\sigma_1,\ldots,\sigma_j}(O_{j+1}))
\]
traversed in phase \( j \)

In the remainder of the proof we simplify notation by abbreviating \( \hat{W} \). Consider a specific phase \( j \), the starting point at the beginning of phase \( j \)

Hence,

\[
\text{ONLINE}_\sigma \leq (2/\lambda + \gamma/\lambda + \gamma) \sum_{j=1}^{J-1} \text{OPT}(O_j, \sigma_j) + C_j.
\]

Applying the claim together with (7) and (8) yields

\[
\text{ONLINE}_\sigma \leq (2/\lambda + \gamma/\lambda + \gamma) \lambda^{-1} \sum_{j=1}^{J-1} (F_{j+1}^m - F_j^m) + C_j
\]

\[
= (2/\lambda + \gamma/\lambda + \gamma) \lambda^{-1} \sum_{j=1}^{J-1} (F_j^m) + C_j
\]

\[
\leq (2/\lambda + \gamma/\lambda + \gamma) \lambda^{-1} \sum_{j=1}^{J-1} \text{OPT}(O, \sigma) + C_j.
\]

Now we bound \( C_j \). Let \( s \) be the endpoint of an optimal path starting in \( O \) and serving \( \sigma \).

\[
(1 - \lambda) d(O, O_j) \leq W_{O, \sigma_1, \ldots, \sigma_j-1}(O_j) - \lambda d(O, O_j)
\]

\[
\leq (1 - \lambda) d(O, O_j) + \lambda d(O, O_j)
\]

\[
\leq \lambda d(O, O_j)
\]

\[
\leq \text{OPT}(O, \sigma)
\]

Next we insert this bound in (9) and let \( \beta = (1 - 2\alpha)/5 \) and \( \alpha = 1/2 - 5\lambda \), whence \( \beta = 2\lambda \). Together with the choices \( \gamma = c(1 - \lambda)/(1 - 2\lambda) \) and \( 0 < \lambda < 1/10 \) and \( m \geq 2 \) this yields

\[
\text{ONLINE}(\sigma) \leq ((2 + \gamma + \gamma \lambda)\lambda^{-2} (1/2 - 5\lambda)^{1-m} + (1 + \lambda)/(1 - \lambda) + 1) \text{OPT}(O, \sigma)
\]

\[
\leq ((2 + 99/80c)\lambda^{-2} (1/2 - 5\lambda)^{1-m} + 5/2c) \text{OPT}(O, \sigma)
\]

\[
< ((2 + 5/4c)\lambda^{-2} (1/2 - 5\lambda)^{1-m}.
\]

\textbf{Proof of Claim.} Consider a specific phase \( j, 1 \leq j \leq J - 1 \). Given a point \( s \in \mathbb{M} \) we define \( s^- \) as the starting point at the beginning of phase \( j \) of the part of the optimal path, serving \( \sigma_1, \ldots, \sigma_j \) and ending in \( s \); i.e., \( W_{\sigma_1, \ldots, \sigma_j}(s) \) is the length of the part of the optimal path ending in \( s \) which is traversed in phase \( j \). By these definitions \( W_{O, \sigma_1, \ldots, \sigma_j}(s) = W_{O, \sigma_1, \ldots, \sigma_j-1}(s^-) + W_{s^-}(s) \). Clearly, \( W_{s^-}(s) \geq d(s, s^-) \). Moreover, remembering the definition of \( \hat{W} \), we have \( \hat{W}_{\sigma_j}(s) \leq W_{s^-}(s) \). In the remainder of the proof we simplify notation by abbreviating \( W_{O, \sigma_1, \ldots, \sigma_j}(s) \) to \( W_{\leq j}(s) \) and
The general two-server problem

\( W_{s_{j}}(s) \) and \( \overline{W}_{s_{j}}(s) \) to, respectively, \( W_{j}(s) \) and \( \overline{W}_{j}(s) \). Assume \( \mathcal{F}_{j+1}^{m} = \mathcal{F}_{j+1}(t_{1}, \ldots, t_{m}) \), i.e. the minimum at the end of phase \( j \) is attained in the points \( (t_{1}, \ldots, t_{m}) \). We distinguish two cases.

**Case 1.** The points \( t_{1}, \ldots, t_{m} \) are pairwise \( \sigma_{j} \)-independent. Condition (3) together with the definition of \( \text{ONLINE} \) implies for any \( j \leq J - 1 \) that \( \gamma \text{OPT}(O_{j}, \sigma_{j}) \leq C_{j} \leq c \text{OPT}(O_{j}, \sigma_{j}) + c \sum_{i=1}^{m} \overline{W}_{j}(t_{i}) \), whence \( \sum_{i=1}^{m} W_{j}(t_{i}) \geq \sum_{i=1}^{m} \overline{W}_{j}(t_{i}) \geq (\gamma/c - 1)\text{OPT}(O_{j}, \sigma_{j}) \). We apply Lemma 1 with \( f \equiv W_{s_{j}} \) and with \( g \equiv W_{s_{j-1}} \).

\[
\mathcal{F}_{j+1}^{m} - \mathcal{F}_{j}^{m} \geq \sum_{i=1}^{m} (((\alpha^{i-1} - \sum_{h=i+1}^{m} \alpha^{h-1})(W_{s_{j}}(t_{i}) - W_{s_{j-1}}(t_{i}^-)) - \beta d(t_{i}, t_{i}^-) \sum_{h=i}^{m} \alpha^{h-1}) \geq \sum_{i=1}^{m} ((\alpha^{i-1} - \sum_{h=i+1}^{m} \alpha^{h-1} - \beta \sum_{h=i}^{m} \alpha^{h-1})W_{j}(t_{i}) \geq (6) \alpha^{m-1}(1 - \beta) W_{j}(t_{i}) \geq \alpha^{m-1}(1 - \beta)(\gamma/c - 1)\text{OPT}(O_{j}, \sigma_{j}) = \alpha^{m-1}\lambda\text{OPT}(O_{j}, \sigma_{j}).
\]

**Case 2.** There are points \( t_{p} \) and \( t_{q} \) \( \{p \neq q \in \{1, \ldots, m\}\} \) that are \( \sigma_{j} \)-dependent, i.e. \( d(t_{p}, t_{q}) < \overline{W}_{j}(t_{p}) + \overline{W}_{j}(t_{q}) \leq W_{j}(t_{p}) + W_{j}(t_{q}) \). Assume without loss of generality that \( W_{j}(t_{p}) \geq W_{j}(t_{q}) \). Define the series \( (u_{1}, \ldots, u_{m}) \) by \( u_{i} = t_{i}^- \) for \( i \neq q \) and \( u_{q} = t_{p}^- \). Lemma 1 implies

\[
\mathcal{F}_{j+1}^{m}(t_{1}, \ldots, t_{m}) - \mathcal{F}_{j}^{m}(u_{1}, \ldots, u_{m}) \geq \sum_{i=1}^{m} H_{i},
\]

with for any \( i \neq q \),

\[
H_{i} = (\alpha^{i-1} - \sum_{h=i+1}^{m} \alpha^{h-1} - \beta \sum_{h=i}^{m} \alpha^{h-1})W_{j}(t_{i}) \geq (6) \alpha^{m-1}(1 - \beta)W_{j}(t_{i}) \geq 0
\]

and

\[
H_{q} = (\alpha^{q-1} - \sum_{h=q+1}^{m} \alpha^{h-1}) (W_{s_{j}}(t_{q}) - W_{s_{j-1}}(t_{p}^-)) - \beta d(t_{q}, t_{p}^-) \sum_{h=q}^{m} \alpha^{h-1}.
\]

To see that \( H_{q} \geq 0 \), notice that by Lemma 2

\[
W_{s_{j}}(t_{q}) - W_{s_{j-1}}(t_{p}^-) = (W_{s_{j}}(t_{p}) - W_{s_{j-1}}(t_{p}^-)) + (W_{s_{j}}(t_{q}) - W_{s_{j}}(t_{p})) \geq W_{j}(t_{p}) - (1 - 2\alpha)^{-1} \beta d(t_{p}, t_{q}) > W_{j}(t_{p}) - 2(1 - 2\alpha)^{-1} \beta W_{j}(t_{p}).
\]

Also notice that

\[
d(t_{q}, t_{p}^-) \leq d(t_{q}, t_{p}) + d(t_{p}, t_{p}^-) \leq 2W_{j}(t_{p}) + W_{j}(t_{p}) = 3W_{j}(t_{p}).
\]

Hence, since we have chosen \( \beta = (1 - 2\alpha)/5 \),

\[
H_{q} / W_{j}(t_{p}) \geq (\alpha^{q-1} - \sum_{h=q+1}^{m} \alpha^{h-1})(1 - 2(1 - 2\alpha)^{-1}) \beta - 3 \beta \sum_{h=q}^{m} \alpha^{h-1}
\]
Next we define series (two cases in defining \( G \)) thus, \( u_0 = t_p^- \) and \( u_b = t_p^+ \), with \( a < b \). By definition
\[
G_j^m(u_1, \ldots, u_m) \geq W_{\leq j-1}(u_b) - \{ W_{\leq j-1}(u_a) + \beta d(u_a, u_b) \} = 0. \tag{12}
\]
Next we define series \((v_1, \ldots, v_{m-1})\) of \( m-1 \) points with \( v_i = u_i \) for \( i = 1, \ldots, b-1 \). We distinguish two cases in defining \( v_b, \ldots, v_m \). If \( \sum_{i=b+1}^m G_j^i(u_1, \ldots, u_i) \leq 0 \) then \( v_i = u_i+1 \) for \( i = b, \ldots, m-1 \). Thus, \( G_j(v_1, \ldots, v_i) = G_{j+1}(u_1, \ldots, u_i, u_i+1) \) for \( i = b, \ldots, m-1 \), and, using (12), we have
\[
F_j^m(u_1, \ldots, u_m) - F_j^{m-1}(v_1, \ldots, v_{m-1}) \geq (\alpha - 1) \sum_{i=b+1}^m \alpha^{-i} G_j^i(u_1, \ldots, u_i) \geq 0.
\]
If \( \sum_{i=b+1}^m G_j^i(u_1, \ldots, u_i) \geq 0 \) then define \( v_i = \text{argmin} \{ W_{\leq j-1}(u_i) 1 \leq h \leq b-1 \} \) for \( i = b, \ldots, m-1 \), implying that \( G_j(v_1, \ldots, v_i) = 0 \) for \( i = b, \ldots, m-1 \). Again
\[
F_j^m(u_1, \ldots, u_m) - F_j^{m-1}(v_1, \ldots, v_{m-1}) = \sum_{i=b}^m \alpha^{-i} G_j^i(u_1, \ldots, u_i) - \sum_{i=b}^{m-1} \alpha^{-i} G_j^i(v_1, \ldots, v_i) \geq 0.
\]
Now choose \( v_m = O_j \). Then
\[
G_j^m(v_1, \ldots, v_m) = W_{\leq j-1}(O_j) - \min_{1 \leq m-1} \{ W_{\leq j-1}(v_i) + \beta d(v_i, O_j) \}.
\]
Assume that the minimum in this expression is attained for the point \( v_i = t_p^- \). (Notice that the multiset \( \{ v_1, \ldots, v_{m-1} \} \) is a subset of the multiset \( \{ t_1^- \ldots, t_m^- \} \).) Again we apply (2) to obtain,
\[
W_{\leq j-1}(t_p^-) \geq W_{\leq j-1}(O_j) - \lambda d(O_j, t_p^-). \quad \text{Another observation is } \text{OPT}(O_j, \sigma_j) \leq d(O_j, t_p^-) + W_j(t_p^-). \quad \text{Together they yield,}
\]
\[
F_j^{m-1}(v_1, \ldots, v_{m-1}) - F_j^m(v_1, \ldots, v_m) = -\alpha^{-m} G_j^m(v_1, \ldots, v_m) = -\alpha^{-m} (W_{\leq j-1}(O_j) - W_{\leq j-1}(t_p^-)) - \lambda d(t_p^-, O_j) \geq \alpha^{-m} (\beta - \lambda) d(t_p^-, O_j) \geq \alpha^{-m} (\beta - \lambda) \text{OPT}(O_j, \sigma_j) - W_j(t_p^-). \tag{13}
\]
Hence,
\[
\begin{align*}
F_j^{m+1} & = F_j^m(t_1, \ldots, t_m) \\
& \geq F_j^m(u_1, \ldots, u_m) + \sum_i H_i \\
& \geq F_j^{m-1}(v_1, \ldots, v_{m-1}) + H_l \\
& \geq F_j^m(v_1, \ldots, v_m) + \alpha^{-m} (\beta - \lambda) \text{OPT}(O_j, \sigma_j) - W_j(t_p^-) + H_l \\
& \geq F_j^m + \alpha^{-m} \lambda (\text{OPT}(O_j, \sigma_j) - W_j(t_p)) + H_l \\
& \geq F_j^m + \alpha^{-m} (1 - \beta - \lambda) W_j(t_p) + \alpha^{-m} \lambda \text{OPT}(O_j, \sigma_j) \\
& \geq F_j^m + \alpha^{-m} \lambda \text{OPT}(O_j, \sigma_j).
\end{align*}
\]
3 The general two-server problem

In the general two-server problem we are given a server, to whom we will address as the $x$-server, moving in a metric space $X$, starting from point $x_0 \in X$ and a server, the $y$-server, moving in a metric space $Y$, starting in $y_0 \in Y$. Requests $(x, y) \in X \times Y$ are presented on-line one by one and are served by moving one of the servers to the corresponding point in its metric space. The objective is to minimize the sum of the distances travelled by the two servers. This problem can easily be modelled as a metrical service system: There is one server moving in the product space $X \times Y$ and any pair $(x, y) \in X \times Y$ defines a request $r = \{(x, y)\} \cup \{(x, y)\}$. For any two points $(x_1, y_1)$ and $(x_2, y_2)$ in $X \times Y$ we define $d((x_1, y_1), (x_2, y_2)) = d_x(x_1, x_2) + d_y(y_1, y_2)$, where $d_x$ and $d_y$ are the distance function of the metric spaces $X$ and $Y$.

**Lemma 3.** The work function and $\lambda$-WFA are well-defined for the general two-server problem.

*Proof.* Let $\sigma = (x_0, y_0), \ldots, (x_n, y_n)$ be a request sequence for some general two-server problem, where we assume, without loss of generality, that the first request is given in the origin $O = (x_0, y_0)$. Consider an arbitrary path serving $\sigma$ and ending in some point $(x, y)$. If at some request both servers move on this path then the move of the server that does not serve the request can be postponed to the next request at no extra cost. Hence, there exists a path ending in $(x, y)$ of at most the same length on which only one server is moved at each request, possibly with the exception of the last request. Since the number of paths that move only one server with each request is $2^n$ there exists a path ending in $(x, y)$ that has minimal length, whence the work function is well-defined. More precisely, the endpoint of any such path is in the set $S = \{x_n\} \times \{y_0, \ldots, y_{n-1}\} \cup \{x_0, \ldots, x_{n-1}\} \times \{y_n\}$. Hence $W_{O, \sigma}(x, y) = W_{O, \sigma}(s) + d(s, (x, y))$ for some $s \in S$ implying supp($W_{O, \sigma}$) $\subseteq S$. Since $S$ has only a finite number $(2n - 2)$ of elements the generalized work function algorithm $\lambda$-WFA is well-defined.

Thus, a part of the conditions of Theorem 1 is satisfied. We are still to define an algorithm $A$ that satisfies condition (3). As such an algorithm we have chosen the simple BALANCE algorithm, which keeps track of the costs made by the $x$- and $y$-server and tries to balance their costs. The BALANCE algorithm is not competitive for our problem as it is known not to be competitive even for the classical two-server problem. However, we will show that it satisfies condition (3) with $c = 4$ and $m = 3$.

We define BALANCE starting in $(x_0, y_0)$ and serving the request sequence $\sigma = (x_1, y_1), (x_2, y_2), \ldots$. Let $S^x_j$ and $S^y_j$ be the total costs made by, respectively, the $x$- and the $y$-server after the $j$-th request is served and let $S_j := S^x_j + S^y_j$. We denote the positions of the servers after serving request $(x_j, y_j)$ by $(\hat{x}_j, y_j)$.

**BALANCE**

If $S^x_j + d(\hat{x}_j, x_{j+1}) \leq S^y_j + d(\hat{y}_j, y_{j+1})$, then move the $x$-server to request $x_{j+1}$. Else move the $y$-server to request $y_{j+1}$.

The following lemmas give an upper bound on the cost of BALANCE. We denote by $P^x_{ij}$, $(0 \leq i < j)$, the length of the path $x_i, x_{i+1}, \ldots, x_j$. We denote $P^y_{ij}$ $(0 \leq i < j)$ in a similar way.
Lemma 4. If Balance is applied to the request sequence \((x_1, y_1), \ldots, (x_j, y_j)\) starting from \((x_0, y_0)\), then \(S_j \leq 2 \min\{P^x_{0j}, P^y_{0j}\}\) \(\forall j \geq 0\).

Proof. Clearly, \(S^x_j \leq P^x_{0j}\) and \(S^y_j \leq P^y_{0j}\). Let request \((x_i, y_i)\), be the last request served by the \(x\)-server. Then, by definition, \(S^x_i = S^x_{i-1} + d(y_{i-1}, y_i) \leq P^y_{0i} \leq P^y_{0j}\). Hence, \(S^x_j \leq \min\{P^x_{0j}, P^y_{0j}\}\). Similarly it is shown that \(S^y_j \leq \min\{P^x_{0j}, P^y_{0j}\}\). \(\square\)

Let \(\text{Opt}(O, \sigma)\) denote the cost of an optimal path serving the sequence \(\sigma\) and starting from \((x_0, y_0)\).

Lemma 5. If Balance is applied to the request sequence \(\sigma = (x_1, y_1), \ldots, (x_j, y_j)\) starting from \((x_0, y_0)\), then \(S_j \leq 2\text{Opt}(O, \sigma) + 4 \min\{P^x_{1j}, P^y_{1j}\}\).

Proof. If the optimal path uses only one server then \(\text{Opt}(O, \sigma) = \min\{P^x_{0j}, P^y_{0j}\} \geq S_j/2\) and the lemma follows immediately from Lemma 4. So assume the optimal path uses both servers and assume without loss of generality \(P^x_{1j} \leq P^y_{1j}\). Since the \(x\)-server of the optimal path serves at least one request \(\text{Opt}(O, \sigma) \geq P^y_{01} - P^x_{1j}\). Again using Lemma 4 yields \(S_j \leq 2P^y_{01} + 2P^x_{1j} \leq 2\text{Opt}(O, \sigma) + 4P^x_{1j} = 2\text{Opt}(O, \sigma) + 4 \min\{P^x_{1j}, P^y_{1j}\}\). \(\square\)

Lemma 6. If \(s_1, s_2\) and \(s_3\) are pairwise \(\sigma\)-independent points with respect to the request sequence \(\sigma = (x_1, y_1), \ldots, (x_j, y_j)\), then \(\overline{W}_\sigma(s_1) + \overline{W}_\sigma(s_2) + \overline{W}_\sigma(s_3) \geq \min\{P^x_{1j}, P^y_{1j}\}\).

Proof. Denote by \(T_i\) \((i = 1, 2, 3)\) the path of length \(\overline{W}_\sigma(s_i)\) that serves \(\sigma\) and ends in \(s_i\). Since \(\overline{W}_\sigma(s_1) + \overline{W}_\sigma(s_2) \leq d(s_1, s_2)\) it is impossible that there are two requests such that one request is served by the \(x\)-servers of \(T_1\) and \(T_2\) and the other is served by the \(y\)-servers of \(T_1\) and \(T_2\). The same holds of course for the other two pairs \(T_1, T_3\) and \(T_2, T_3\). So assume without loss of generality that the \(x\)-server of \(T_1\) shares no request with the \(x\)-server of \(T_2\) and shares no request with the \(x\)-server of \(T_3\). If the \(y\)-server of \(T_1\) serves all requests then the lemma obviously holds. So assume request \((x_i, y_i)\) is served by the \(x\)-server of \(T_1\). Then \(T_2\) and \(T_3\) must serve point \(y_i\). But this means that the \(x\)-servers of \(T_2\) and \(T_3\) do not share a request. Hence, the three \(x\)-servers share no request. Thus, each request is served by at least two \(y\)-servers and for each two consecutive requests there is a \(y\)-server that serves both requests. \(\square\)

Combining Lemma 5, Lemma 6 and Theorem 1 yields the next result.

Corollary 1. Algorithm \textsc{Online(Balance, }\gamma, \lambda)\text{ with }\gamma = 4(1 - \lambda)/(1 - 2\lambda)\text{ is }7/(\lambda(1/2 - 5\lambda))^2\text{-competitive for any instance of the general two-server problem, for any }0 < \lambda < 1/10. \(\square\)

The ratio is minimized by \(\lambda = 1/20\), yielding a competitive ratio of 44,800.

4 Postlude

Unfortunately, Theorem 1 does not provide a competitive algorithm for the general \(k\)-server problem for \(k \geq 3\) just as easily as for \(k = 2\). The question whether a competitive algorithm exists for \(k \geq 3\) remains unresolved. We believe that the generalized work function is competitive for the general \(k\)-server problem for any \(k \geq 1\) and \(\lambda < 1\).
It is not hard to prove that the generalized work function algorithm for the general two-server problem satisfies Lemma 5 up to a constant factor. This does not imply that the generalized work function algorithm is competitive for the general two-server problem, since the work function in this part of the Online algorithm is defined for the requests within the phase. This is different from the work function used in the last move of Online in a phase, which is defined for the complete sequence. It remains open to prove competitiveness of the generalized work function algorithm. We notice that our proof does not use any information about the work function that is specific for the general two-server problem. To make this step it seems essential to have a better understanding of the structure of the work function of the general two-server problem.

A problem to which Theorem 1 applies directly is the $k$-client problem. In this metrical service system the set of possible requests consists of all $k$-element subsets of the metric space. Burley [2] showed that the generalized work function algorithm is $O(k^2)$-competitive when $\lambda$ is chosen appropriately for each $k$. Our proof has many similarities with the proof of Burley. Notice that if we choose $m = k + 1$ in Theorem 1 the condition of this theorem is satisfied for any algorithm $A$ since for any sequence $\sigma$ at most $k$ pairwise $\sigma$-independent points exist. Therefore we can restrict the phases in Online to the $\lambda$-Wfa-moves. This direct application of the theorem only shows that $\lambda$-Wfa is $O(k^2 2^k)$-competitive for appropriate $\lambda$. However, adjusting the proof of Theorem 1 to the $k$-client problem gives exactly Burley’s proof.

Condition (3) of Theorem 1 is imposed on an algorithm $A$. It would be more interesting to give a (non-trivial) sufficient condition for a metrical service system to have a finite competitive ratio. Even more interesting would it be to know under what condition on the system $\lambda$-Wfa is competitive.

We conclude with the NP-hardness proof of the off-line version of the general $k$-server problem. The proof is straightforward from the exact 3-cover problem. We first remind the reader to the definition of the latter problem. For NP-Completeness of this problem we refer to [9].

**Exact 3-cover.**
An instance is given by a set $Z$ with $|Z| = 3q$ and a collection $C$ of 3-element subsets of $Z$. The question is whether $C$ contains an exact cover for $Z$, i.e. a subcollection $C' \subseteq C$ such that every element of $Z$ occurs in exactly one member of $C'$.

**Theorem 2.** The general $k$-server problem is NP-hard.

**Proof.** Take any instance $I$ of the exact 3-cover problem (with the notation as defined above) we define an instance $I'$ of the general $k$-server problem. For each 3-element subset we define a metric space with its server. The $|C|$ metric spaces are identical and consist of an origin, one point at distance 1 from the origin, and one point at distance $q + 1$ from the origin. For each element of $Z$ we define one request, i.e. a $|C|$-tuple of points, one in each of the $|C|$ metric spaces. In each metric space the three requests that correspond to the 3-element subset are given at the point at distance 1. The remaining $3q - 3$ requests are given in the point at distance $q + 1$. Instance $I'$ has a solution with value of at most $q$ if and only if $I$ has an exact cover for $Z$. □
References


