# A Lower Bound for Two-Server Balancing Algorithms*

Jon M. Kleinberg

TR 93-1350
May 1993

Department of Computer Science
Cornell University
Ithaca, NY 14853-7501

*This work was supported in part by the Sloan Fellowship of Eva Tardos.

# A Lower Bound for Two-Server Balancing Algorithms *

Jon M. Kleinberg

Department of Computer Science
Cornell University, Ithaca, NY 14853, U.S.A.

**Abstract**

We consider the class of *balancing algorithms* for two servers. Such algorithms have appeared in a number of the early papers on this problem; they are so named because they seek to "balance" the distance travelled evenly among the servers. In this paper, we show a universal lower bound on the competitive ratio of *any* balancing algorithm for two servers. The lower bound is equal to $(5+\sqrt{7})/2$ ($\sim 3.82$), and consequently shows that no optimal on-line algorithm for two servers can be expressed as a balancing algorithm.

## 1. Introduction

In the $k$-server problem, an algorithm is given a metric space $M$ and $k$ mobile robots ("servers") that can occupy points in this space. A finite request sequence $\sigma$ is presented to the algorithm, one request at a time. Each request is a point in the space $M$; the algorithm must move one of the servers to this point before seeing the next request. The goal is to minimize the total distance travelled by the servers, over the entire request sequence.

If we let $S(\sigma)$ denote the cost incurred by the algorithm on request sequence $\sigma$, and $OPT(\sigma)$ denote the minimum cost required to serve $\sigma$, then

*This work was supported in part by the Sloan Fellowship of Éva Tardos.

the algorithm is said to be $c$-competitive if for some absolute constant $\beta$, the quantity $cS(\sigma) - OPT(\sigma)$ is bounded by $\beta$ over all finite sequences $\sigma$. The infimum of the set of $c$ for which the algorithm is $c$-competitive is its *competitive ratio.*

A number of papers have proposed "balancing" algorithms for the $k$-server problem. The basic balancing algorithm works as follows: for each server $s_i$, its total distance travelled is maintained in the variable $D_i$; when a request is made at a point $r$, the algorithm sends the server which minimizes $D_i + rs_i$ (for $x, y \in M$, let $xy$ denote the distance between them). This rule was shown to be $k$-competitive for $k$ servers when the cardinality of the request space $M$ is $k + 1$ [5], and for the "weighted-cache" problem, which includes the paging problem as a special case [1].

However, the algorithm is not $c$-competitive for any $c$, even for two servers, in a general metric space $M$. Thus it was somewhat surprising that a rule minimizing the quantity $D_i + 2rs_i$ was shown to be 10-competitive for two servers [4]. A later construction showed that this algorithm was no better than 6-competitive [2].

In this note, we show a new lower bound for the class of balancing algorithms in general. Let $f : \Re^+ \to \Re$, and $B_f$ be the server algorithm which does nothing when the request point is already covered, and otherwise moves the server which minimizes $D_i + f(rs_i)$, where $D_i$ is the total distance travelled by server $i$. We will describe $B_f$ as a balancing algorithm with cost function $f$. Observe that we make no restrictions whatsoever on the nature of the function $f$.

Our main result is a lower bound of $(5+\sqrt{7})/2$ ($\sim 3.82$) on the competitive ratio of *any* such balancing algorithm for two servers. In view of the 2-competitive algorithms of [5, 3], this shows that no optimal on-line 2-server algorithm can be expressed as a decision rule $B_f$ for any $f$.

## 2. The Lower Bound Proof

We first define some notation that will be useful in what follows. For a server algorithm $S$, let $\Gamma(S)$ denote its competitive ratio. If $S$ is $c$-competitive for some $c$, we will say it is *competitive*; otherwise, we write $\Gamma(S) = \infty$. Let $A$ denote the optimal (off-line) server algorithm (whose cost on $\sigma$ is always $OPT(\sigma)$).

We will say that a property $P$ of positive real numbers holds "e.f." (everywhere but in a finite interval) if $\exists x_0 \forall x \geq x_0 P(x)$. Similarly $P$ holds "a.l." (for arbitrarily large reals) if $\forall x_0 \exists x \geq x_0 P(x)$. For any function $f : \Re^+ \rightarrow \Re$, $p \in \Re$, $p \geq 0$, say that $f \in \varphi(p)$ if $f(x) \leq px$ e.f. If for some such $p$, $f \in \varphi(p)$, we will write $\rho_f = \inf\{p : f \in \varphi(p)\}$; $\rho_f = \infty$ otherwise.

Finally, the behavior of $B_f$ is ambiguous when $D_i + f(rs_i) = D_j + f(rs_j)$ for $i \neq j$, and $i, j$ both minimize this expression. The standard convention here is to let the adversary break the tie; in any event, none of our constructions rely on such degeneracies.

For the remainder of this note, all server algorithms will be 2-server algorithms.

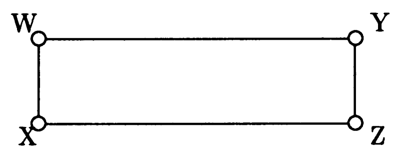


Figure 1: Metric Space $M_1$

**Lemma 1** *If $B_f$ is competitive, then $f(x) > 0$ e.f.*

*Proof.* Assume by way of contradiction that $f(x) \leq 0$ a.l. and that $B_f$ is $c$-competitive for some $c$. Choose $x_0$ so that $f(x_0) \leq 0$ and $N_0$ large enough so that $N_0 \geq cx_0$ and $N_0 > -f(x_0)$. Since $f(x) \leq 0$ a.l., we can find an $N \geq N_0$ such that $f(N) \leq 0$.

Consider metric space $M_1$, with $WX = YZ = x_0$, $WY = XZ = N$, and the servers initially on $W, X$. The first request is to the point $Y$. If $s_1$ responds, then we request point $Z$; since

$$N + f(x_0) > 0 \geq f(N),$$

server $s_2$ will respond. At this point, $B_f$ has paid $2N \geq 2cx_0$, while $A$, by using one server to cover requests to $W, X$ and the other to cover $Y, Z$, need pay only $x_0$. Moreover, the situation is now symmetric to the beginning, so we can repeat the process indefinitely, contradicting the claim the $B_f$ is $c$-competitive. Note that if $s_2$ responds to the first request, then

$$N + f(x_0) > 0 \geq f(N) \geq f(N + x_0),$$

so $s_1$ will respond to the second request, arriving at the same contradiction. ∎

Armed with this lemma, we can prove a much stronger result, generalizing the observation that the basic balancing algorithm is not competitive.

**Lemma 2** *If $B_f$ is competitive, then $\rho_f > 1$.*

*Proof.* Assume by way of contradiction that $\rho_f \leq 1$ and $B_f$ is $c$-competitive. We have two cases to consider: $f \in \varphi(1)$ and $f \notin \varphi(1)$.

If $f \in \varphi(1)$, then for some $x_0 > 0$, $0 < f(x) \leq x$ for all $x \geq x_0$. Consider metric space $M_1$, with $WX = YZ = x_0$, $WY = XZ = N = cx_0$, and the servers initially on $W, X$. Suppose we request point $Y$ and $s_1$ responds. If we now request point $Z$, then since

$$N + f(x_0) > N \geq f(N),$$

server $s_2$ will respond. Moreover, this process can be continued indefinitely. Thus $B_f$ pays $2cx_0$ in each round of this sequence, while $A$ need pay only $x_0$, as in the above arguments. The case in which $s_2$ responds to the first request is strictly analogous.

If $f \notin \varphi(1)$, then we can still find some $y > 0$ such that $0 < f(x) < (1 + \frac{1}{2c})x$ for all $x \geq y$. Moreover, since $f \notin \varphi(1)$, we can find some $x_0 > y$ such that $f(x_0) > x_0/2$. Construct metric space $M_1$ with $WX = YZ = x_0$, $WY = XZ = cx_0$, put $s_1$ on $W$, $s_2$ on $X$, and request point $Y$. If $s_1$ responds, then we request point $Z$. Since $f(x_0) > x_0/2$,

$$cx_0 + f(x_0) > (c + \frac{1}{2})x_0 = (1 + \frac{1}{2c})cx_0 > f(cx_0),$$

the last inequality following since $cx_0 > y$. Thus $s_2$ will respond. In this way, $B_f$ pays $2cx_0$ on this sequence, while $A$ need only pay $x_0$. Again, the case in which $s_2$ responds to $Y$ is analogous. ∎

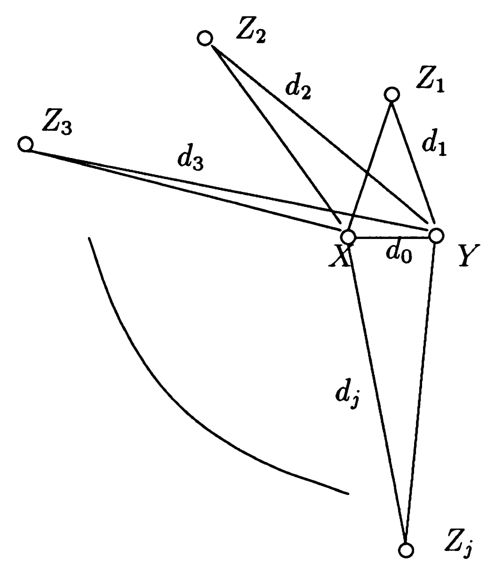


Figure 2: Metric Space $M_2$

**Lemma 3** *If $\rho_f = p > 1$, then $\Gamma(B_f) \geq p/2 + 3/2$.*

*Proof.* We show that for each $\beta > 0$, there exist arbitrarily long request sequences $\sigma$ for which $B_f(\sigma) > (p/2 + 3/2 - \beta)A(\sigma)$; this will establish the lemma. We construct a countably infinite metric $M_2$ as follows. The "core" of the space is a short segment $XY$ of length $d_0$, chosen so that $pd_0/2 < f(d_0) < 2pd_0$. The sequence $\sigma$ is built in phases; in phase $j$, $j = 1, 2, \ldots$, all the requests are to $X$, $Y$, and a point $Z_j$ at distance $d_j$ from both $X$ and $Y$. Let $\sigma_j$ denote the sequence up through the end of phase $j$.

Let $D_j$ denote $\sum_{i=0}^{j-1} d_j$, and $D'_j$ denote the sum of $D_j$ and the total distance travelled by both on-line servers in all previous phases. We fix some very small $\alpha > 0$, to be determined below, and since $\rho_f = p$, we can choose $d_j$ such that $\alpha D'_j < d_j$, and $f(d_j) > (p - \beta/2)d_j$. Phase $j$ begins with the two on-line servers sitting on $X$ and $Y$. First the point $Z_j$ is requested; then the points $X$ and $Y$ are requested repeatedly until the on-line server at $Z_j$ returns to the core $XY$; at this point the phase comes to an end.

In this phase, one adversary server moves out to $Z_j$ and immediately returns on the next request, for a cost of $2d_j$. Meanwhile, the on-line server covering $X$ and $Y$ has built up a distance of no more than $D'_j$ from all previous phases, so it must move more than

$$(p+1)d_j - 2p\alpha d_j - (\beta/2)d_j$$

before $d_j + f(d_j)$ could possibly exceed its total distance plus $f(d_0) < 2pd_0 \leq 2p\alpha d_j$, and the server at $Z_j$ is selected to return. Thus, the on-line servers move a distance of at least

$$(p+3)d_j - 2p\alpha d_j - (\beta/2)d_j$$

in this phase, while the adversary servers move no more than $2d_j + 2\alpha d_j$ in *all* phases up to this one. By choosing $\alpha$ small enough, we can ensure that

$$\frac{B_f(\sigma_j)}{A(\sigma_j)} \geq \frac{(p+3) - 2p\alpha - (\beta/2)}{2+2\alpha} \geq p/2 + 3/2 - \beta.$$

Since we can continue this construction for an arbitrary number of phases, the result follows. ∎

It is clear that a very similar construction involving metric space $M_2$ shows that if $\rho_f = \infty$, then $B_f$ is not competitive. The final lemma is based on a construction in [2].

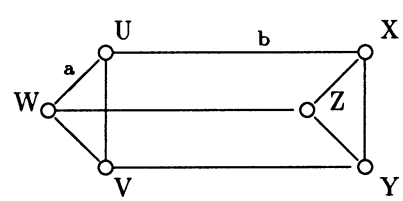


Figure 3: Metric Space $M_3$

**Lemma 4** *If* $\rho_f = p > 1$, *then* $\Gamma(B_f) \geq 3p/(p-1)$.

*Proof.* In the style of the previous proof, we show that for each $\beta > 0$, $\Gamma(B_f) > 3p/(p-1)-\beta$. Choose positive $\delta < \beta/2$ and $\varepsilon < [2(p-1)^2/(4p-1)]\delta$. Now take $N$ large enough so that $f(x) < (p+\varepsilon)x$ for all $x > N$ and, since $\rho_f > p - \varepsilon$, we can find $a > N$ such that $f(a) > (p-\varepsilon)a$.

Consider metric space $M_3$. Triangles $UVW$ and $XYZ$ have sides of length $a$; the three edges crossing between the triangles have length $b = ((2p+1)/(2p-2) - \delta)a$. Put server $s_1$ on $U$ and $s_2$ on $V$. We place the adversary servers on $W$ and $Z$. Also, set $D_1 = a/2$ and $D_2 = 0$ (this can be easily accomplished by starting one of the on-line servers from a seventh point in the metric space).

The adversary's strategy is as follows. It requests $Z$, which will be served by $s_2$. It then requests $Y$, which will be served by $s_1$, thanks to the way we have defined the distances. At this point, $D_2$ exceeds $D_1$ by $a/2$, and the situation is symmetric to the beginning. The optimal algorithm can use one server to cover each triangle and hence pays $a$; $B_f$ pays $2b + a$, for a ratio of

$$\frac{2b+a}{a} = 1 + \frac{2p+1}{p-1} - 2\delta > \frac{3p}{p-1} - \beta.$$

Thus we need only verify that $s_1$ will serve the request at $Y$. This follows because

$$D_1 + f(b) = a/2 + f((\frac{2p+1}{2p-2} - \delta)a) \leq a\frac{2p^2+2p-1+(2p+1)\varepsilon-(2p^2-2p)\delta}{2(p-1)},$$

which is less than or equal to

$$D_2 + f(a) = a + b + f(a) \geq a\frac{2p^2+2p-1-(2p-2)(\delta+\varepsilon)}{2(p-1)}.$$

∎

**Theorem 1** *For all $f$, $\Gamma(B_f) \geq (5+\sqrt{7})/2$.*

*Proof.* If $\rho_f \leq 1$ or $\rho_f = \infty$, then $B_f$ is not competitive. In the case where $1 < \rho_f < \infty$, Lemmas 3 and 4 imply that

$$\Gamma(B_f) \geq \max\{\frac{3\rho_f}{\rho_f - 1}, \frac{\rho_f + 3}{2}\}.$$

This expression is minimized at $\rho_f = 2 + \sqrt{7}$, with a value of $(5+\sqrt{7})/2$. ∎

As noted above, we have the following interesting corollary.

**Corollary 1** *No optimal on-line 2-server algorithm can be expressed as $B_f$ for any function $f$.*

## 3. Conclusion

We have shown a non-trivial lower bound on the competitive ratio of any 2-server balancing algorithm. It would be interesting to generalize this result to show a lower bound of $ck$, for some $c > 1$, on the competitive ratio of any $k$-server balancing algorithm. Probably more interesting, however, would be to give a non-trivial lower bound on the competitive ratio of any server algorithm from some computationally limited class. For example, does there exist a 2-competitive 2-server algorithm which uses only constant time or space per request?

Finally, it is worth noting that the lower bound given here is less than the best known upper bound of 4 on the competitive ratio of a 2-server algorithm using constant time and space [2]. It would be interesting to determine more precisely the best possible competitive ratio that can be achieved by a 2-server balancing algorithm.

# References


[1] M. Chrobak, H. Karloff, T. Payne, S. Vishwanathan, "New Results on Server Problems," *SIAM J. Discrete Math.*, 4(1991), pp. 172–181.

[2] M. Chrobak, L. Larmore, "On Fast Algorithms for Two Servers," *J. Algorithms*, 12(1991), pp. 607–614.

[3] M. Chrobak, L. Larmore, "A New Approach to the Server Problem," *SIAM J. Discrete Math.*, 4(1991), pp. 323–328.

[4] S. Irani, R. Rubinfeld, "A Competitive 2-Server Algorithm," *Information Processing Letters*, 39(1991), pp. 85–91.

[5] M. Manasse, L. McGeoch, D. Sleator, "Competitive Algorithms for Server Problems," *J. Algorithms*, 11(1990), pp. 208–230.