# HARMONIC is 3-Competitive for Two Servers

Marek Chrobak and Lawrence L. Larmore
Department of Computer Science
University of California
Riverside, CA 92521

October 4, 1996

**Abstract**

We consider the server algorithm, called HARMONIC, that always moves the servers with probabilities proportional to the inverses of distances to the request point. We show that for two servers HARMONIC is 3-competitive. Before it was only known that the competitiveness constant of this algorithm is between 3 and 6.

## 1 Introduction

The *k-server problem* is defined as follows: Let $M$ be a metric space, in which we have $k$ servers that can occupy and move between points of $M$. At each time step we are given a request point $r \in M$, and we have to move one of our servers to $r$ to "serve" the request. Our goal is to minimize the total distance traveled by our servers. The problem is that the requests must be served *on-line*, that is, without the knowledge about future requests.

Given such an on-line server algorithm $\mathcal{A}$, let $cost_{\mathcal{A}}(K, \varrho)$ be the total cost of $\mathcal{A}$ on request sequence $\varrho \in M^*$ starting from initial configuration $K \in M^k$, and $cost_{opt}(K, \varrho)$ the cost of the corresponding optimal schedule. For randomized algorithms, the expected cost is denoted by $E[cost_{\mathcal{A}}(K, \varrho)]$. Then $\mathcal{A}$ is called *c-competitive* if for each initial configuration $K$ we have

$$\sup_{\varrho \in M^*} \{ E[cost_{\mathcal{A}}(K, \varrho)] - c \cdot cost_{opt}(K, \varrho) \} < \infty.$$

In other words, for any initial configuration $K$, the cost of our algorithm $\mathcal{A}$ on each request sequence $\varrho$ is at most $c$ times the optimum cost of serving $\varrho$ from $K$, plus some additive constant independent of $\varrho$.

We briefly review the history of the problem. Manasse, McGeoch and Sleator [7] proved that there are no $k$-server $c$-competitive algorithms for $c < k$ in any metric space with at least $k + 1$ points. The problem of whether there is an on-line algorithm that achieves $c = k$ for each $k$, is open. The famous "$k$-server conjecture" from [7] states that this is indeed true. Up to now, it has been proven only for $k = 2$ in [7]. Another 2-competitive algorithm for 2 servers was given in [2]. Fiat *et al* [5] proved that for each $k$ there is a $c$-competitive algorithm, but their constant was much greater than $k$.

Let $s_i$ denote the $i$-th server. (By a slight abuse of notation, we shall use the same notation for a server as for the point in $M$ where that server is located – thus, $s_i \in M$.) To shorten our notation, we shall write $xy$ to denote the distance between any two points $x, y \in M$. In this paper we consider the following algorithm:

**Algorithm HARMONIC:** Given a request at point $r$, move $s_i$ to $r$ with probability

$$p_i = \frac{\frac{1}{s_i r}}{\frac{1}{s_1 r} + \frac{1}{s_2 r} + \ldots + \frac{1}{s_k r}},$$

Thus HARMONIC moves a server to the request with probability proportional to the inverse of its distance to the request point. Therefore the servers that are closer to the request point are more likely to serve the request. The harmonic algorithm is an example of a *memoryless* algorithm, whose decision of which server to move depends only on the current configuration of the servers. Because of its simplicity and low computational complexity, HARMONIC has attracted a lot of attention.

Raghavan and Snir [8] proved that HARMONIC has competitiveness constant between 3 and 6 for two servers. For $k = 3$ Berman *et al* [1] proved that HARMONIC is competitive. They also conjectured that for $k$ servers the constant is $k(k+1)/2$. Recently Grove [6] proved that HARMONIC is $c_k$-competitive for each $k$, but his constant $c_k$ is a very large and rapidly growing function of $k$.

In this paper we give a complete analysis of HARMONIC for $k = 2$, by proving the its competitiveness constant is equal 3. This settles the Berman-Karloff-Tardos conjecture [1] in case $k = 2$.

**Theorem 1** *HARMONIC is 3-competitive for two servers.*

For brevity, we will write $cost_{\mathcal{H}}$ to denote $cost_{\mathrm{HARMONIC}}$, the cost function of HARMONIC.

## 2 Preliminaries

We picture the behavior of any server algorithm $\mathcal{A}$ as a game between $\mathcal{A}$'s servers, $s_1$, $s_2$ (which we refer to as "our" servers) and an adversary's servers, $a_1$ and $a_2$.

Initially the adversary's servers are on the same positions as our servers. The adversary is allowed to create requests and serve them with its own servers trying to force the ratio $cost_{\mathcal{A}}/cost_{adv}$ be as large as possible. In the proofs we show that on each sequence of requests $\varrho$, and independently of the adversary's behavior, the inequality

$$E[cost_{\mathcal{H}}(K,\varrho)] \leq 3 \cdot cost_{adv}(K,\varrho) + \Phi(K^0),$$

always holds, where $K^0$ is the initial configuration of the servers, and function $\Phi$ is defined below. This implies that the algorithm is 3-competitive, since one possible adversary's computation will correspond to the optimal schedule on $\varrho$.

The proofs are based on the so-called potential technique, used often in the analysis of dynamic data structures. We define a function $\Phi$, called a *potential* that maps all possible states, from some state set, into the non-negative reals $\mathbf{R}^+$. (Each state will at least include the positions of the servers, ours and the adversary's.) In the initial configuration $\Phi$ depends only on the set of $k$ points on which the servers (ours and the adversary's) are located.

The game consists of a number of rounds, each consisting of one adversary move and one algorithm move. The adversary moves a single server and puts the request on the position of this server. Then one of our servers, designated by the algorithm, moves to the request point. It is not hard to see that these rules lead to no loss of generality. Our proof is based on the following lemma which can be proven by simple summation over the whole sequence of requests. From now on, for simplicity, we will omit the parameter $\varrho$ (the request sequence) in our notation, if it is understood from the context.

**Lemma 1** *Let $\mathcal{A}$ be any randomized algorithm. Let $\Phi$ be any potential taking values in $\mathbf{R}^+$. Let $\Delta\Phi$, $\Delta cost_{adv}$ and $\Delta cost_{\mathcal{A}}$ denote, respectively, the changes of the potential, the adversary's cost and our cost in the current round. Suppose that for each round*

(1) *If the adversary moves then*

$$\Delta\Phi \leq c \cdot \Delta cost_{adv},$$

(2) *and if our algorithm moves then*

$$E[\Delta\Phi] \leq -E[\Delta cost_{\mathcal{A}}].$$

*Then $E[cost_{\mathcal{A}}] \leq c \cdot cost_{adv} + \Phi_0$, where $\Phi_0$ is the initial value of the potential function. Consequently, $\mathcal{A}$ is $c$-competitive.*

## 3 The proof of 3-competitiveness

In this section we given the proof of Theorem 1.

Notice that for two servers, the probability of serving the request on $r$ with $s_i$ becomes $p_i = \frac{s_j r}{s_1 r + s_2 r}$, where $s_j$ is the other server, that is $j \neq i$.

*The potential.* The potential function we use in the proof is defined as follows. Given a configuration of all servers, ours and the adversary's, we allow the adversary to make an arbitrary sequence of lazy calls (a lazy call is a request on a position where the adversary already has a server) until our servers are matched with the adversary's servers – the potential is defined as the maximum expected cost of our algorithm, over all such sequences of lazy calls.

Notice now that the adversary has, essentially, only two possible such sequences of calls, depending on which server made the first call. It is because after one adversary's server is already covered, he can make only one lazy call on the other server. As mentioned before, we assume that at every move the adversary moves only one server and puts the request on this server. This assumption will allow us to define the potential without using the maximum, and consequently simplify the proofs.

We can assume that $a_1 \neq a_2$, since it can be easily shown to be senseless for the adversary to ever move his two servers to the same point, and, for a constant initial cost, we can assume that they are distinct from the beginning.

(P0) If our servers match the adversary's servers, then

$$\Phi = 0.$$

(P1) If one adversary server is matched with one of our servers, say $a_1 = s_1$, then

$$\Phi = \frac{2 \cdot a_2 s_2 \cdot (2 \cdot a_1 s_2 + a_1 a_2)}{a_1 s_2 + a_1 a_2 + a_2 s_2}.$$

(P2) If no adversary server is matched with any of our servers, let $a_1$ be the adversary server which most recently moved (and thus the adversary is about to put the request on $a_1$). Then

$$\Phi = \frac{2}{a_1 s_1 + a_1 s_2} \left[ a_1 s_1 \cdot a_1 s_2 \; + \frac{a_1 s_2 \cdot a_2 s_2 \cdot (2 \cdot a_1 s_2 + a_1 a_2)}{a_1 s_2 + a_1 a_2 + a_2 s_2} + \frac{a_1 s_1 \cdot a_2 s_1 \cdot (2 \cdot a_1 s_1 + a_1 a_2)}{a_1 s_1 + a_1 a_2 + a_2 s_1} \right].$$

We note that (P0) is the special case of (P1) where $s_2 = a_2$, and that (P1) is the special case of (P2) where $s_1 = a_1$. Note also that if the indices 1 and 2 in (P2) are exchanged, and then $s_1$ is set to $a_1$, the formula again simplifies to (P1).

These observations give the following:

**Lemma 2** *$\Phi$ is a continuous function of the server positions.*

Note that (P2) is symmetric under exchange of $s_1$ and $s_2$, though not under exchange of $a_1$ and $a_2$.

*Our server moves.* Now we will analyze the change of the potential when our server serves the request. With elementary calculations one can readily verify that the formulas for $\Phi$ above are indeed equal to the expected cost of our servers on the sequence of lazy calls, when the first call is on $a_2$ in case (P1), and on $a_1$ in case (P2). Then the following lemma follows easily from the definition of the potential, by straightforward calculations.

**Lemma 3** *When HARMONIC serves the request, then we have*

$$E[\Delta\Phi] \leq -E[\Delta cost_{\mathcal{H}}]$$

*The adversary moves.* Now we will prove that the first hypothesis of Lemma 1 is also true. This part is much harder and requires rather tedious calculations.

**Lemma 4** *If the adversary moves, then* $\Delta\Phi \leq 3 \cdot \Delta cost_{adv}$.

*Proof:* We first note that, between rounds, at least one adversary server matches one of our servers. Let $r$ be the request point, let $\Phi$ be the potential before the move, and $\Phi_i$ the potential after the move if $a_i$ moves. We need only show that $3 \cdot a_i r - \Phi_i + \Phi \geq 0$ for $i = 1, 2$.

To simplify the notation, we write $x = a_1 = s_1$, $y = s_2$, and $z = a_2$, referring to the positions before the move. By continuity of the potential, it suffices to consider only the case where $x, y, z, r$ are distinct.

We now write $\Phi$ and $\Phi_i$ in terms of $x, y, z, r$

$$\begin{aligned}
\Phi &= \frac{2 \cdot yz \cdot (2 \cdot xy + xz)}{xy + xz + yz} \\
\Phi_1 &= \frac{2}{xr + yr}\left[xr \cdot yr + \frac{yr \cdot yz \cdot (2 \cdot yr + zr)}{yr + zr + yz} + \frac{xr \cdot xz \cdot (2 \cdot xr + zr)}{xz + xr + zr}\right] \\
\Phi_2 &= \frac{2 \cdot yr \cdot (2 \cdot xy + xr)}{xy + xr + yr}
\end{aligned}$$

The lemma will follow immediately from Lemma 1 and claims A, B below. First, we state the following rule that will be used several times in the calculations: Let $\alpha, \beta, \gamma > 0$. Then

$$\alpha \leq \beta \quad \text{iff} \quad \frac{\alpha}{\beta} \leq \frac{\alpha + \gamma}{\beta + \gamma}. \tag{R}$$

**Claim A:** $\Phi_1 - \Phi \leq 3 \cdot xr$.

We will now prove Claim A. Since $xr + zr + xz \geq 2xz$ we have:

$$\frac{xz \cdot (4 \cdot xr + 2 \cdot zr)}{xr + zr + xz} = xz + \frac{xz \cdot (3 \cdot xr + zr - xz)}{xr + zr + xz} \leq \frac{1}{2} \cdot xz + \frac{3}{2} \cdot xr + \frac{1}{2} \cdot zr. \tag{1}$$

Next we prove the following inequalities:

$$\frac{xz \cdot (4 \cdot xr + 2 \cdot zr)}{xz + xr + zr} - \frac{yz \cdot (4 \cdot xy + 2 \cdot xz - 4 \cdot yr)}{xy + xz + yz} - 3 \cdot xr - yr \leq 0 \tag{2}$$

$$\frac{4 \cdot yr + 2 \cdot zr}{yr + zr + yz} - \frac{4 \cdot xy + 2 \cdot xz + 4 \cdot xr}{xy + xz + yz} \leq 0 \tag{3}$$

*Proof of inequality* (2): Let $L_2$ denote the left side of (2). Then, by (1) we have:

$$\begin{aligned}
L_2 &\le -\frac{3}{2}\cdot xr + \frac{1}{2}\cdot zr + \frac{1}{2}\cdot xz - yr - \left[yz + \frac{yz\cdot(3\cdot xy + xz - yz - 4\cdot yr)}{xy+xz+yz}\right] \\
&= -\frac{1}{2}\cdot(3\cdot xr + 2\cdot yr + 2\cdot yz - zr - xz) - \frac{yz\cdot(3\cdot xy + xz - yz - 4\cdot yr)}{xy+xz+yz} \\
&\le -\frac{yz}{xy+xz+yz}\cdot(3\cdot xr + 2\cdot yr + 2\cdot yz - zr - xz) - \frac{yz\cdot(3\cdot xy + xz - yz - 4\cdot yr)}{xy+xz+yz} \\
&= -\frac{yz}{xy+xz+yz}\left(2\cdot(xr+xy-yr) + (xr+xy+yz-zr)\right) \ \le\ 0.
\end{aligned}$$

In the second step we used the fact that $3\cdot xr + 2\cdot yr + 2\cdot yz - zr - xz \ge 0$.

*Proof of inequality* (3): Let $L_3$ denote the left side of inequality (3). We have two cases.

*Case 1:* $xy + xz \ge yr + zr$. Then

$$\begin{aligned}
L_3 &= \frac{2\cdot yr - 2\cdot yz}{yr+zr+yz} - \frac{2\cdot xy - 2\cdot yz + 4\cdot xr}{xy+xz+yz} \\
&= \frac{2\cdot yz\cdot(yr+zr-xy-xz)}{(xy+xz+yz)(yr+zr+yz)} + \frac{2\cdot yr}{yr+zr+yz} - \frac{2\cdot xy + 4\cdot xr}{xy+xz+yz} \\
&\le 0 + \frac{2\cdot yr + xy + xz - yr - zr}{yr+zr+yz+xy+xz-yr-zr} - \frac{2\cdot xy + 4\cdot xr}{xy+xz+yz} \\
&= \frac{(yr-xy-xr) + (xz-zr-xr) - 2\cdot xr}{xy+xz+yz} \ \le\ 0.
\end{aligned}$$

*Case 2:* $xy + xz \le yr + zr$. Notice that $0 \le yr + zr - xy - xz < yr + zr + yz \le 4\cdot yr + 2\cdot zr$. Thus, using inequality (R) we derive

$$\begin{aligned}
L_3 &\le \frac{4\cdot yr + 2\cdot zr - (yr+zr-xy-xz)}{yr+zr+yz-(yr+zr-xy-xz)} - \frac{4\cdot xy + 2\cdot xz + 4\cdot xr}{xy+xz+yz} \\
&= \frac{3\cdot(yr-xy-xr) + (zr-xz-xr)}{xy+xz+yz} \ \le\ 0.
\end{aligned}$$

This completes the proof of inequalities (2) and (3). Now we are ready to prove Claim A:

$$\begin{aligned}
\Phi_1 - \Phi &= \frac{xr}{xr+yr}\left[2\cdot yr + \frac{xz\cdot(4\cdot xr + 2\cdot zr)}{xz+xr+zr} - \frac{yz\cdot(4\cdot xy + 2\cdot xz - 4\cdot yr)}{xy+xz+yz}\right] \\
&\qquad + \frac{yz\cdot yr}{xr+yr}\left[\frac{4\cdot yr + 2\cdot zr}{yr+zr+yz} - \frac{4\cdot xy + 2\cdot xz + 4\cdot xr}{xy+xz+yz}\right] \\
&\le \frac{xr}{xr+yr}\left[2\cdot yr + 3\cdot xr + yr\right] + \frac{yz\cdot yr}{xr+yr}\cdot 0 \ =\ 3\cdot xr.
\end{aligned}$$

by (2) and (3). This completes the proof of Claim A.

**Claim B:** $\Phi_2 - \Phi \le 3\cdot zr$.

Now we will prove Claim B. Using rule (R) and the triangle inequality we derive the following inequalities:

$$\begin{aligned}
\Phi &= \frac{yz\cdot(4\cdot xy+2\cdot xz)}{xy+xz+yz} \geq \frac{yz\cdot(4\cdot xy+2\cdot xz+xr+zr-xz)}{xy+xz+yz+xr+zr-xz}\\
&= \frac{yz\cdot(4\cdot xy+xz+xr+zr)}{xy+yz+xr+zr} \qquad (4)\\
2\Phi_2 &\leq \frac{(2\cdot yr+zr+yz-yr)(4\cdot xy+2\cdot xr)}{xy+xr+yr+zr+yz-yr}\\
&\leq \frac{(yr+zr+yz)(4\cdot xy+xr+xz+zr)}{xy+xr+zr+yz} \qquad (5)
\end{aligned}$$

Using inequalities (4), (5) and the triangle inequality we obtain

$$\begin{aligned}
2(\Phi_2-\Phi) &= \frac{(4\cdot xy+xz+xr+zr)(yr+zr+yz-2\cdot yz)}{xy+xr+zr+yz}\\
&\leq \frac{(3\cdot xy+3\cdot xr+3\cdot zr+yz)(2\cdot zr)}{xy+xr+zr+yz} \leq 6\cdot zr.
\end{aligned}$$

This completes the proof of Claim B, and Lemma 4. □

Lemmas 1, 4 and 3 imply Theorem 1, completing the proof.

That Theorem 1 is sharp follows from the work in [8]. For completeness we show the sketch of the lower bound proof: Initially the servers positions are $s_1 = a_1 = x$ and $s_2 = a_2 = y$, with $xy = 1$. Now $a_1$ makes a number of calls on points that are not more than $\epsilon$ away from $x$, until $s_2$ moves. Then the adversary places a lazy call on $a_2$ and repeats the process. By simple calculation, when $\epsilon$ converges to zero, the ratio $cost_{\mathcal{H}}/cost_{adv}$ converges to 3.

## References


[1] P. Berman, H. Karloff, G. Tardos, A competitive algorithm for three servers, Proc. First Annual ACM-SIAM Symposium on Discrete Algorithms (1990) 291-300.

[2] M. Chrobak, L. Larmore, A new approach to the server problem, SIAM J. Computing, to appear.

[3] M. Chrobak, L. Larmore, Fast algorithms for two servers, J. Algorithms, to appear.

[4] D. Coppersmith, P. G. Doyle, P. Raghavan, M. Snir, Random walks on weighted graphs and applications to on-line algorithms, Proc. 22nd Annual ACM Symp. on Theory of Computing (1990) 369-378.

[5] A. Fiat, Y. Rabani, Y. Ravid, Competitive $k$-server algorithms, Proc. 31st Annual Symp. on Foundations of Computer Science (1990) 454-463.

[6] E. Grove, The harmonic on-line $k$-server algorithm is competitive, a manuscript.

[7] M. Manasse, L. A. McGeoch, D. Sleator, Competitive algorithms for server problems, Tech. Rep. CMU-CS-88-197, Carnegie-Mellon University, also Proc. 20th ACM STOC (1988) 322-333, and J. Algorithms 11 (1990) 208-230.

[8] P. Raghavan, M. Snir, Memory versus randomization in on-line algorithms, Proc. ICALP'88, Lecture Notes in Computer Science, vol. 372, pp. 687-703, Springer-Verlag, New York, 1988.