

A 33,000-Competitive Algorithm for the General 2-Server Problem

John Lee, Jiwon Kim

December 20, 2005

Abstract

We study the general online 2-server problem and improve upon the current best competitive ratio of 100,000. Our goal for this paper is two-fold: (1) to clarify the points of the paper that introduced the 100,000-competitive algorithm, and (2) to present a variation of that algorithm, ONLINE2, that achieves a competitive ratio of less than 33,000.

1 Introduction

In the k -server problem, we are given servers s_1, \dots, s_k moving in a metric space M . Requests r_i arrive one by one, and in order to serve a request we must move one of the k servers to the point r_i . Our goal is to minimize the total cost, i.e., the total distance moved by all servers over all requests. We measure the performance of an algorithm through competitive analysis.

The problem was introduced by Manasse, McGeoch, and Sleator [3], who proved a lower bound of k on the competitive ratio for any deterministic algorithm for any metric space and conjectured that a k -competitive algorithm exists. They proved that one exists for $k = 2$, but for other k the current best upper bound is $2k - 1$ [1].

Consider the following variant of the k -server problem: each of the k servers s_i moves in its own metric space M_i . A request is a k -tuple $r = (z_1, \dots, z_k) \in M_1 \times \dots \times M_k$. In order to serve r , we must move one of the servers s_i to the corresponding point $z_i \in M_i$. Our goal is still to minimize the total distance moved by all of the servers. This problem is known as the *general 2-server* problem. The k -server problem is a special case of this where $z_1 = \dots = z_k$ for all requests and $M_1 = \dots = M_k$.

A special case of the general 2-server problem, where both space M_1 and M_2 are lines, has become known as the CNN problem [2]: if a camera crew in Manhattan taking pictures of various places, it need not move to the specific request points since it can get a clear shot looking down the street or avenue it is currently on. Koutsoupias and Taylor proved a lower

bound of $6 + \sqrt{17}$ on the competitive ratio of any deterministic algorithm for this problem, by reducing a special case of the CNN problem to an instance of the weighted 2-server problem on a line [2]. Sitters, et. al [4] gave an algorithm that achieves an upper bound of 100,000 on its competitive ratio.

The problem can be solved offline (i.e., if all requests are known beforehand) with a simple dynamic programming algorithm. In this paper, we propose a variant of the algorithm in [4] and prove that it is 33,000-competitive.

This paper is structured as follows: in Section 2, we give a sketch of the algorithm and the general idea; in Section 3 we formalize some properties of the problem; in the following sections, we give the full technical description of the algorithm and an analysis.

2 The Big Picture

We define a *tour* to be a directed path through $M_1 \times M_2$. We denote the length of a tour T by $|T|$ and say that T serves r_1, \dots, r_n if there is a sequence of points on T that serves all of the requests, in that order.

The algorithm works in phases, and the only information retained from one phase to the next is the locations of the servers at the end of the phase (which will be the locations of the servers at the beginning of the next phase). The general idea behind the competitive algorithm is that during each phase, our goal is to either:

- move at a cost comparable to the optimal tour serving the requests for that phase, or
- be much closer to the optimal tour by the end of the phase than we were at the beginning of it.

We begin the algorithm with a simple greedy strategy. Clearly, this algorithm by itself is not competitive: an adversary can force the one of the servers to alternate between two requests, while the optimal tour remains stationary and satisfies them all at zero cost. Because of this, as we are applying the greedy strategy, we stop at each step to think of what the optimal tour would do with the requests given so far.

If our greedy strategy is not much worse than the optimal tour, then we are on the way to achieving the first goal listed above, and we continue with a greedy algorithm. If, however, we notice that we made some bad decisions (i.e., there exists another tour that serves the requests seen so far at a greatly decreased cost), we will tend to move the servers to that position. If there is more than one such tour, we move the servers to the position of one of them and apply the strategy COMPETE (described in Section 4.3) until all of the short tours are relatively close together (we call such tours *neighbors*; we discuss this property in Section 3). Once this happens, we move our servers towards that position.

3 Basic Properties of the Problem

Throughout this paper, we use the notation (\hat{x}_i, \hat{y}_i) to denote the positions of our servers after serving the i th request (we let (\hat{x}_0, \hat{y}_0) be the starting positions of the servers). We denote these two servers the x -server and the y -server. We define the distance $\delta((x_1, y_1), (x_2, y_2)) = d_x(x_1, x_2) + d_y(y_1, y_2)$ (this is the “Manhattan” distance in the CNN problem). Additionally, we define a “greedy” distance function $g((x_1, y_1), (x_2, y_2)) = \min\{d_x(x_1, x_2), d_y(y_1, y_2)\}$. This is the minimum distance we need to move to serve request 2 starting from position 1, or vice versa.

Given the starting positions of the servers (\hat{x}_0, \hat{y}_0) and a request sequence r_1, \dots, r_n , let X_{ij} as the length of the tour that only uses the x -server to serve requests i through j ($1 \leq i < j \leq n$), and let X_{0j} to be the length of the tour that uses only the x -server and starts from \hat{x}_0 . We define Y_{ij} in a similar manner.

We now formalize the notion of “closeness” of tours mentioned in Section 2 and prove some properties of “close” tours. We will say that two tours T_A and T_B are *neighbors* if there exist points $x \in M_1$ and $y \in M_2$ so that both x and y are on T_A and T_B . It is important to note that the particular point (x, y) need not be on either of the tours – each tour just needs to touch x with its x -server and y with its y -server at some point in the tour, but not necessarily simultaneously.

The following lemmas show important properties of neighboring tours. Lemma 1 implies the important property that there are at most two tours that are far apart (i.e., not neighbors) if we are considering short tours. Lemma 2 bounds the distance between points on neighboring tours, in terms of the lengths of the tours.

Lemma 1. *If the tours T_A , T_B , and T_C serve the same sequence of requests r_1, \dots, r_n and no pair out of these three tours are neighbors, then $|T_A| + |T_B| + |T_C| \geq \min\{X_{1n}, Y_{1n}\}$.*

Proof. Assume without loss of generality that the x -server of T_A shares no request with the x -server of T_B or T_C (i.e., there is no $x \in M_1$ such that x is on T_A as well as one of T_B and T_C). If the y -server of T_A serves all requests, then $|T_A| \geq Y_{1n}$, so the lemma holds. So assume request r_i is served by the x -server of T_A . This means that the i th request must be served by the y -servers of T_B and T_C . Since T_B and T_C are not neighbors, their x -servers cannot share this request. Thus, the no x -servers share any request. As a result, each request is served by at least two y -servers. It follows that for any two consecutive requests, the y -server on one of the tours serves both of them. Thus, we have that the total length of all three tours is at least Y_{1n} . \square

Corollary 1. *For any three tours T_A , T_B , and T_C serving the same sequence of n requests, if $|T_A| + |T_B| + |T_C| < \min\{X_{1n}, Y_{1n}\}$, then at least two of the tours must be neighbors.*

Lemma 2. Let (x_A, y_A) be a point on T_A and (x_B, y_B) be a point on T_B . If T_A and T_B are neighbors, then $\delta((x_A, y_A), (x_B, y_B)) \leq |T_A| + |T_B|$.

Proof. Since the tours are neighbors, let x and y be on both T_A and T_B . Then we use the definition of δ and the triangle inequality:

$$\begin{aligned} \delta((x_A, y_A), (x_B, y_B)) &= d(x_A, x_B) + d(y_A, y_B) \\ &\leq d(x_A, x) + d(x, x_B) + d(y_A, y) + d(y, y_B) \\ &= [d(x_A, x) + d(y_A, y)] + [d(x, x_B) + d(y, y_B)] \\ &\leq |T_A| + |T_B|. \end{aligned}$$

□

Corollary 2. For any two tours T_A and T_B , if $|T_A| + |T_B| < \delta((x_A, y_A), (x_B, y_B))$ for some point on T_A and T_B , then T_A and T_B are not neighbors.

4 Building Blocks

There are a number of subroutines used by ONLINE2. As discussed in Section 2, we apply different strategies and choose one carefully depending on the situation. After we receive a new request, ONLINE2 calculates the optimal offline path for all requests up to the new one and decide whether we should continue as planned or switch to something else.

As already noted, ONLINE2 runs in phases. We keep track of the distance moved by the servers over the entire phase. Let S_j^x and S_j^y be the total costs made by the x - and y -servers, respectively, after serving request r_j , and let $S_j = S_j^x + S_j^y$.

4.1 Algorithm GREEDY

The greedy strategy is simple, and blind to all previous requests. Given a starting position (\hat{x}_0, \hat{y}_0) and a request $r = (x_r, y_r)$, we move the x -server if $d(\hat{x}_0, x_r) < d(\hat{y}_0, y_r)$ and the y -server otherwise.

Lemma 3. $S_{j+1} \leq 2S_j + g((\hat{x}_0, \hat{y}_0), (x_j, y_j))$ for all $j \geq 1$

Proof. Without loss of generality, assume we moved the x -server to serve the request. By definition of GREEDY, we have $S_{j+1} = S_j + d(\hat{x}_0, x_{j+1}) \leq S_j + S_j^x + d(\hat{x}_0, x_{j+1}) \leq 2S_j + d(\hat{x}_0, x_{j+1}) = 2S_j + g((\hat{x}_0, \hat{y}_0), (x_j, y_j))$. □

4.2 Algorithm BALANCE

This algorithm is used as a subroutine to COMPETE (described in Section 4.3). It is, in a sense, similar to GREEDY, except that each time we get a request, we move the server that will have moved the least *over all requests*. As the name suggests, it tries to keep S_j^x and S_j^y close to one another. Formally, BALANCE is defined as follows:

Given a new request r_{j+1} , if $S_j^x + d(\hat{x}_j, x_{j+1}) \leq S_j^y + d(\hat{y}_j, y_{j+1})$, serve the request with the x -server. Otherwise, serve it with the y -server.

We can upper bound the cost of BALANCE in terms of the request sequence:

Lemma 4. $S_j \leq 2 \max\{S_j^x, S_j^y\} \leq 2 \min\{X_{0j}, Y_{0j}\}$.

Proof. Suppose the i th request is the last one that was served by the x -server ($i \leq j$). By definition of BALANCE, we have $S_j^x = S_i^x \leq S_{i-1}^y + d(\hat{y}_{i-1}, y_i) \leq Y_{0i} \leq Y_{0j}$. Clearly, we also have $S_j^x \leq X_{0j}$, so $S_j^x \leq \min\{X_{0j}, Y_{0j}\}$. Similarly, we can show $S_j^y \leq \min\{X_{0j}, Y_{0j}\}$. Thus, by definition of S_j , we have $S_j \leq 2 \max\{S_j^x, S_j^y\} \leq 2 \min\{X_{0j}, Y_{0j}\}$. \square

We can also upper bound the cost in terms of the total cost up to the previous step:

Lemma 5. $S_{j+1} \leq 3S_j + \min\{d(\hat{x}_0, x_{j+1}), d(\hat{y}_0, y_{j+1})\}$.

Proof. Without loss of generality, assume that $d(\hat{x}_0, x_{j+1}) < d(\hat{y}_0, y_{j+1})$. By definition of BALANCE, we have:

$$\begin{aligned} S_{j+1} &\leq S_j + \min\{S_j^x + d(\hat{x}_j, x_{j+1}), S_j^y + d(\hat{y}_j, y_{j+1})\} \\ &\leq S_j + S_j^x + d(\hat{x}_j, x_{j+1}) \\ &\leq S_j + S_j^x + (S_j^x + d(\hat{x}_0, x_{j+1})) \\ &\leq 3S_j + d(\hat{x}_0, x_{j+1}) \end{aligned}$$

\square

4.3 Algorithm COMPETE

The COMPETE strategy is invoked when we notice that there are alternative tours that are much shorter than the one we have made up to now. We know from Lemma 1 that there are at most two that are not neighbors. ONLINE2 will move the servers to the position of one of the short tours; the goal of COMPETE is to remain competitive with the sum of the short tours that are not neighboring.

COMPETE works in phases. It depends on a parameter, (x^*, y^*) (which ONLINE2 assigns to be the positions of the servers of one of the short tours). We denote the starting positions of

the servers at the start of COMPETE to be (\hat{x}, \hat{y}) . As before, we let (\hat{x}_j, \hat{y}_j) be the positions of the servers after serving the j th request of the current phase, and define S_j^x , S_j^y , and S_j be the total cost of the current phase up to the j th request.

We define $\Delta = \frac{1}{2} \max\{d(\hat{x}_0, x^*), d(\hat{y}_0, y^*)\}$. In our description of the algorithm, we assume without loss of generality that $d(\hat{x}_0, x^*) > d(\hat{y}_0, y^*)$, so that $\Delta = \frac{1}{2}d(\hat{x}_0, x^*)$. Changing x with y in the following description gives the behavior if $d(\hat{y}_0, y^*) > d(\hat{x}_0, x^*)$. Below, we describe a single phase of COMPETE(x^*, y^*).

1. Run BALANCE, but stop when we receive a request r_j where $d(\hat{x}, x_j) \geq \Delta$.
2. Serve r_j using GREEDY.
3. If the x -server has not moved during this phase, and $j > 1$, then move the x server towards x_{j-1} by an amount S_{j-1}^y .

Consider two tours, T_1 and T_2 , where T_1 starts at the same place we do (\hat{x}, \hat{y}) and T_2 starts at (x^*, y^*) . We will show in the next lemma that if T_1 and T_2 are relatively short, then COMPETE is competitive with the sum of T_1 and T_2 .

Lemma 6. *If $|T_1| + |T_2| < \Delta$, then the cost of the tour made by COMPETE is $|T| \leq 10(|T_1| + |T_2|)$.*

Proof. First, we note that it is possible for COMPETE to terminate (i.e., the request sequence finishes) before we finish step 1. For purposes of analysis, suppose that $(x^{(1)}, y^{(1)})$ and $(x^{(2)}, y^{(2)})$ are the ending positions of T_1 and T_2 , respectively. If we add an extra request $(x^{(2)}, y^{(1)})$ at the end of the sequence, T_1 and T_2 both serve it at no extra cost. By the triangle inequality, we have $d(\hat{x}, x^{(2)}) \geq d(\hat{x}, x^*) - d(x^*, x^{(2)}) = 2\Delta - d(x^*, x^{(2)}) \geq 2\Delta - |T_2|$. Since we assumed that $|T_1| + |T_2| < \Delta$, we obtain $2\Delta - |T_2| > \Delta$. Thus, because $d(\hat{x}, x^{(2)}) > \Delta$, the condition in step 1 is met and the last request is served in step 2. Hence, we will assume through this proof that the phase does not terminate abruptly in step 1.

Consider an arbitrary phase in the algorithm with n requests. The servers are at (\hat{x}_0, \hat{y}_0) at the beginning of the phase ($(\hat{x}_0, \hat{y}_0) = (\hat{x}, \hat{y})$ for the first phase). Define C_1^x and C_1^y to be the costs of the x - and y -servers, respectively, in this phase, and let $C_1 = C_1^x + C_1^y$. We similarly define C_2^x , C_2^y , and C_2 , and let $C = C_1 + C_2$. We denote the positions of T_1 's servers after serving the j th request in this phase by (x_j^1, y_j^1) .

Before proving the lemma, we also discuss another useful property of the servers. Observe that r_n must be served by the y -server of T_1 , since otherwise, we would have $|T_1| \geq d(\hat{x}, x_n) \geq \Delta$, which contradicts our original assumption.

We also observe that T 's y -server serves the last request as well. Since r_n is the last request, it is served in step 2 with GREEDY. We show this by contradiction: if it were served by the x -server, then by definition of GREEDY we have $d(\hat{y}, y_n) \geq d(\hat{x}, x_n) \geq \Delta$. Since $|T_1| \geq \min\{d(\hat{y}, y_n), d(\hat{x}, x_n)\}$, we would get $|T_1| \geq \Delta$, which contradicts our original assumption.

Now we consider T_2 . Suppose that the x -server of T_2 serves some request r_i where $1 \leq i \leq n-1$. Then we have $|T_2| \geq d(x^*, x_i) \geq d(x^*, \hat{x}) - d(\hat{x}, x_i)$ by the triangle inequality. This quantity is equal to $2\Delta - d(\hat{x}, x_i)$. Since we assumed $i \leq n-1$, the condition under which to stop using BALANCE in step 1 was not yet satisfied, so $d(\hat{x}, x_i) < \Delta$. Thus, we have $|T_2| > 2\Delta - \Delta = \Delta$, which is a contradiction. Therefore, T_2 uses its y -server to serve *all* requests r_1 through r_{n-1} .

For the following analysis, we define the potential at the start of the phase to be $\Psi = 3d(\hat{x}_0, x_0^1)$. Thus, we have that $\Delta\Psi$ for a phase is $3d(\hat{x}_n, x_n^1) - 3d(\hat{x}_0, x_0^1)$. If we can prove that $S \leq 10C - \Delta\Psi$, then if we sum S over all phases we get $|T| \leq 10(|T_1| + |T_2|) - 3d(\hat{x}_N, x_N^1)$, where N is the last request in the last phase.

There are three cases describing the behavior of COMPETE:

- *Case 1. The y -server of T_1 serves only r_0 and r_n , and the x -server of T serves no request in this phase.* In this case, we have $C_1^y = d(y_0, y_n)$ clearly. We also know that $S_n^x = S_{n-1}^y$ by step 3 of COMPETE. The potential increases by $\Delta\Psi \leq 3(C_1^x - S_n^x)$, since the x -server of T only makes one move (during step 3) and it is towards x_n^1 . So the total cost is

$$\begin{aligned}
S &= S_n^x + S_n^y \\
&\leq S_n^x + S_n^{y-1} + d(y_{n-1}, y_n) \\
&\leq S_n^x + S_{n-1}^y + (S_{n-1}^y + d(y_0, y_n)) \\
&= S_n^x + 2S_{n-1}^y + d(y_0, y_n) \\
&= 3S_n^x + d(y_0, y_n) \\
&\leq 3C_1^x - \Delta\Psi + d(y_0, y_n) \\
&\leq 3C_1^x - C_1^y - \Delta\Psi \\
&\leq 3C - \Delta\Psi.
\end{aligned}$$

- *Case 2. The y -server of T_1 serves only r_0 and r_n , and the x -server of T serves some request r_j with $1 \leq j \leq n-1$.* Again, we have $C_1^y = d(y_0, y_n)$. We can bound the increase in potential as follows: consider the first request served by T 's x -server. At this point, the potential difference is 0, since T_1 also served that request. In the worst case, T 's x -server stops serving requests and T_1 moves farther away. Additionally, in the worst case, this particular request was r_1 . Thus, the increase in potential is upper bounded by $\Delta\Psi \leq 3(C_1^x - d(\hat{x}_0, x_1))$. We know that $S_n^x \leq d(\hat{x}_0, x_1) + C_1^x$ in this case, since T_1 uses its x -server for requests r_1 through r_{n-1} .

Suppose that the last request served by T 's y -server was r_i . Then we have that $S_{n-1}^y = S_i^y \leq S_{i-1}^x + d(x_{i-1}, x_i) \leq X_{0i} \leq X_{0n-1}$ by definition of BALANCE. Since $X_{0n-1} \leq d(\hat{x}_0, x_1) + C_1^x$, we conclude that $S_{n-1}^y \leq d(\hat{x}_0, x_1) + C_1^x$. Thus, we have

$$\begin{aligned}
S &= S_n^x + S_n^y \\
&\leq S_n^x + 2S_{n-1}^y + d(y_0, y_n) \\
&\leq 6C_1^x - \Delta\Psi + d(y_0, y_n) \\
&\leq 6C - \Delta\Psi.
\end{aligned}$$

- *Case 3.* The y -server of T_1 serves some request r_k with $1 \leq k \leq n-1$. Clearly, $C_1^y \geq d(y_0, y_k) + d(y_k, y_n)$. Since T_2 uses its y -server to serve all requests except r_n , we have $C_2^y \geq d(y_1, y_k) + d(y_k, y_{n-1})$. So we have $d(y_0, y_k) + d(y_k, y_n) + C_2^y = d(y_0, y_k) + d(y_k, y_n) + d(y_1, y_k) + d(y_k, y_{n-1}) \geq d(y_0, y_1) + d(y_{n-1}, y_n)$ by the triangle inequality. Thus, $S_n^y \leq d(y_0, y_1) + C_2^y + d(y_{n-1}, y_n) \leq C_1^y + 2C_2^y$.

Now we consider S_n^x . We know that $S_n^x = S_{n-1}^x$ since the last request is served by T 's y -server. Using Lemma 4, we have $S_{n-1}^x \leq \max\{S_{n-1}^x, S_{n-1}^y\} \leq Y_{0n-1} \leq d(y_0, y_1) + C_2^y$. The potential difference is upper bounded by $\Delta\Psi \leq 3(C_1^x + S_n^x)$. So, we get

$$\begin{aligned} S &= S_n^x + S_n^y = 4S_n^x + S_n^y - 3S_n^x \\ &\leq 5C_1^y + 10C_2^y + 3C_1^x - \Delta\Psi \\ &\leq 10C - \Delta\Psi. \end{aligned}$$

We sum over all phases to obtain $|T| \leq 10C = 10(|T_1| + |T_2|)$. This proves the result. \square

5 A Competitive Algorithm

Our competitive algorithm, ONLINE2, works in phases. It begins by applying GREEDY while keeping track of alternative short tours. If it finds two non-neighboring short tours, then we move the servers to one of them and apply COMPETE. Otherwise, if there is only one short tour or all of the short tours are neighbors, we end the phase.

At any point during the algorithm, we end the phase immediately if we receive a request that requires a very large move in comparison to previous moves. That request becomes the first request in the next phase. We call this the *exception rule*.

In the description of the algorithm below and the following analysis, we consider an arbitrary phase of ONLINE2. We denote v as the positions of the servers at the end of this phase and v_{-1} to be the positions at the beginning of this phase (which were the positions at the end of the previous phase). The requests in this phase are r_1, \dots, r_n . We use S_j to denote the cost of serving the requests r_1 through r_j in the GREEDY step (step 1) of the ONLINE2. Let Z_j be the cost of COMPETE alone up to the j th request. We also let A_j be the total cost of ONLINE2 up to the j th request.

For the rest of the proof, we rely on two constants. We let $\eta = 1/100$, and $R = 0.35$. While these choices are somewhat arbitrary, we describe in Section 5.1 a number of constraints on these two values; tuning these parameters may affect the competitive ratio slightly.

Exception rule: If we ever receive a request r_j with $j \geq 2$ for which $g(v_{-1}, r_j) \geq 2A_{j-1}$, move back to v_{-1} and terminate the phase (i.e., r_j is the first request in the next phase).

1. Apply GREEDY. After we serve a request r_j , try to find a tour T_A such that $|T_A| < \eta S_j$ and $\delta(v_{-1}, v_A) \leq R S_j$. If such a tour is found, return to v_{-1} and

continue to the next step; otherwise, continue applying GREEDY. We will denote that last request served in this step by r_{k_1} .

2. Try to find two tours T_B and T_C that serve r_1, \dots, r_{k_1} , and satisfy $\max\{|T_B|, |T_C|\} < \eta S_{k_1-1}$ and $\delta(v_B, v_C) \geq 16\eta S_{k_1-1}$. If we cannot find such tours, set $k_2 = k_1$ and go on to the next step. Assume without loss of generality that $d(v_B, v_{-1}) < d(v_C, v_{-1})$. Move the servers to v_B and apply COMPETE(v_C). Z_j is the total distance moved by COMPETE in this phase so far. We stop using COMPETE when $Z_j \geq S_{k_1-1}$, and then move the servers back to v_{-1} . We will denote the last request served in this step by COMPETE as r_{k_2} .
3. Let T_D be the optimal tour that serves r_1, \dots, r_{k_2} . If $|T_D| < \eta S_{k_1-1}$, then we move the servers from v_{-1} towards v_D by a distance RS_{k_1-1} and end the phase. We point out that it does not matter *how* we move this distance: we can move just one of the servers, or both the x - and y -servers by that total distance.

5.1 Competitive Analysis

There are two types of phases of ONLINE2. We denote the phases where the exception rule is invoked to be type I. All other phases are type II. Observe that the last phase, which we call the N th phase, is of type II, since any phase of type I leaves an unserved request and is thus followed by at least one more phase.

With the following two lemmas, we bound the cost of phases of type II. We consider an arbitrary type II phase with n requests. To simplify notation, we let S be S_{k_1-1} (only the GREEDY steps contribute to this cost). We let $|P|$ be the cost of the tour for the entire phase.

Lemma 7. *If $(\frac{1}{2} - 2\eta) > R$, then for any phase of type II, we have $\min\{X_{1k_1}, Y_{1k_1}\} \geq \frac{1}{2}(\frac{1}{2} - R - \eta)S$.*

Proof. We assume without loss of generality that $X_{1k_1} \leq Y_{1k_1}$. Consider requests r_1 through r_{k_1} and the cost S_{k_1} . By definition of GREEDY, we know that $S_{k_1} \leq 2 \min\{X_{0k_1}, Y_{0k_1}\}$. The tour T_A cannot only serve y -requests, because in that case we have $\delta(v_{-1}, v_A) \geq d(\hat{y}_0, y_1) - |T_A| = Y_{0k_1} - Y_{1k_1} - |T_A| \geq Y_{0k_1} - 2|T_A| \geq (\frac{1}{2} - 2\eta)S_{k_1} > RS_{k_1}$. However, this is a contradiction, since by definition of T_A we made $\delta(v_{-1}, v_A) \leq RS_{k_1}$.

Assume that r_j was the last request served by the x -server of T_A . Then we have $RS_{k_1} \geq \delta(v_{-1}, v_A) \geq d(\hat{x}_0, x_j) - |T_A| \geq X_{0k_1} - 2X_{1k_1} - |T_A| \geq (\frac{1}{2} - \eta)S_{k_1} - 2X_{1k_1}$. Solving for X_{1k_1} , we obtain $X_{1k_1} \geq \frac{1}{2}(\frac{1}{2} - R - \eta)S_{k_1} \geq \frac{1}{2}(\frac{1}{2} - R - \eta)S$. \square

We emphasize that in order to prove this, we relied on the assumption that $(\frac{1}{2} - 2\eta) > R$. This is satisfied by our choice of η and R . If this were not satisfied, then it might be possible for T_A to serve only y -requests, making the proof of Lemma 7 invalid.

Next, we bound the cost of the entire phase in terms of S .

Lemma 8. *If $6\eta < (\frac{1}{2} - 2\eta)$, then for any phase of type II, $|P| \leq (52 + 33R + 40\eta)S$.*

Proof. By Lemma 3, we have $S_{k_1} \leq 2S + g((\hat{x}_0, \hat{y}_0), (x_{k_1}, y_{k_1}))$. Since this is a phase of type II, we assume the exception rule was not invoked, i.e., $g(v_{-1}, r_{k_1}) < 2A_{j-1} = 2S$. Thus, we have $S_{k_1} \leq 2S + 2S = 4S$. Note that after serving r_{k_1} , we move back to v_{-1} . Thus, the total cost of step 1 is at most $2S_{k_1} \leq 8S$.

The cost of step 3 is at most RS , by definition of ONLINE2. Thus, if COMPETE was not applied in step 2, the total cost for this phase is given by $|P| \leq (8 + R)S$, which proves the result for this case.

Assume COMPETE was applied in step 2. We move to v_B , so we will first bound this cost. By definition of ONLINE2, we know that $|T_B| + |T_C| < 2\eta S$. By Corollary 2 and the fact that $\delta(v_B, v_C) \geq 16\eta S$, we can conclude that T_B and T_C are not neighbors.

Additionally, we have $|T_A| + |T_B| + |T_C| \leq \eta S_{k_1} + \eta S + \eta S \leq 4\eta S + \eta S + \eta S = 6\eta S < (\frac{1}{2} - 2\eta)S$. We can now apply Lemma 6: $|T_A| + |T_B| + |T_C| < \min\{X_{1k_1}, Y_{1k_1}\}$. By Corollary 1, some pair out of these three tours must be neighbors. Since we already know that T_B and T_C are not neighbors, it follows that T_A is neighbors T_B or T_C .

Suppose T_A neighbors T_B . By the triangle inequality, $\delta(v_{-1}, v_B) \leq \delta(v_{-1}, v_A) + \delta(v_A, v_B)$. Using Lemma 2, this quantity is at most $\delta(v_{-1}, v_A) + |T_A| + |T_B|$. Similarly, if T_A neighbors T_C , then $\delta(v_{-1}, v_C) \leq \delta(v_{-1}, v_A) + |T_A| + |T_C|$. In step 2, we assumed $\delta(v_{-1}, v_B) < \delta(v_{-1}, v_C)$, so we define an upper bound, BS of $\delta(v_{-1}, v_B)$ as follows:

$$\begin{aligned} \delta(v_{-1}, v_B) &\leq \delta(v_{-1}, v_A) + |T_A| + \min\{|T_B|, |T_C|\} \\ &\leq RS_{k_1} + \eta S_{k_1} + \eta S \\ &\leq (4R + 5\eta)S \\ &:= BS \end{aligned}$$

We now bound the cost Z_{k_2} of COMPETE. To do this, we first bound A_{k_2-1} , the total cost of ONLINE2 for this phase except for the last request. Since COMPETE terminates when Z_j exceeds S ($j = k_2$), we sum the costs of step 1, the cost of moving from v_{-1} to v_B , and the COMPETE steps to get $A_{k_2-1} \leq 8S + BS + S = (9 + B)S$.

We distinguish two cases for the final request r_{k_2} served by COMPETE. First, we suppose that the last request was served by the BALANCE step in a phase of COMPETE. Let the starting position of that particular COMPETE phase (not to be confused with the starting position of the ONLINE2 phase v_{-1}) be the point (x, y) . Applying Lemma 5, we define an upper bound

CS on Z_{k_2} :

$$\begin{aligned}
Z_{k_2} &< 3S + \min\{d(x, x_{k_2}), d(y, y_{k_2})\} \\
&\leq 3S + \delta((x, y), (\hat{x}_0, \hat{y}_0)) + \min\{d(\hat{x}_0, x_{k_2}), d(\hat{y}_0, y_{k_2})\} \\
&\leq 3S + \delta((x, y), (\hat{x}_{k_1}, \hat{y}_{k_1})) + \delta((\hat{x}_{k_1}, \hat{y}_{k_1}), (\hat{x}_0, \hat{y}_0)) + \min\{d(\hat{x}_0, x_{k_2}), d(\hat{y}_0, y_{k_2})\} \\
&\leq 3S + BS + S + \min\{d(\hat{x}_0, x_{k_2}), d(\hat{y}_0, y_{k_2})\} \\
&\leq 3S + BS + S + 2(B+9)S \\
&:= CS
\end{aligned}$$

We used the triangle inequality for the third expression, and the exception rule with A_{k_2-1} for the fifth expression.

The other case is when r_{k_2} is served in the GREEDY step of COMPETE. By definition of GREEDY, the cost of serving that request is $Z_{k_2-1} + g((\hat{x}_{k_1}, \hat{y}_{k_1}), (x_{k_2}, y_{k_2}))$. The next step of COMPETE, also by definition, is at most Z_{k_2-1} . Summing over all of the COMPETE steps,

$$\begin{aligned}
Z_{k_2} &\leq 3Z_{k_2-1} + g((\hat{x}_{k_1}, \hat{y}_{k_1}), (x_{k_2}, y_{k_2})) \\
&\leq 3S + \delta((\hat{x}_{k_1}, \hat{y}_{k_1}), (\hat{x}_0, \hat{y}_0)) + \min\{d(\hat{x}_0, x_{k_2}), d(\hat{y}_0, y_{k_2})\} \\
&< CS
\end{aligned}$$

We can now bound the total cost of a type II phase. Step 1 costs $8S$ and step 3 costs RS . In step 2, we move from $v_{-1}tov_B$, then move a distance Z_{k_2} , then move back to v_{-1} . The cost of this is $2(BS + CS)$. So our final expression for $|P|$ is $8S + 2(BS + CS) + RS = (52 + 33R + 40\eta)S$. \square

For the parameters $R = 0.35$ and $\eta = 1/100$, this cost is given by 63.95. Note that this choice of R and η do indeed satisfy the condition of this lemma.

In the following lemma, we denote the optimal tour in this phase as P^* . We emphasize that the starting position of P^* is not necessarily v_{-1} ; P^* is simply the section of the globally optimal tour that serves the same requests as we do in a particular phase. We let v_{-1}^* and v^* denote the starting and ending positions of P^* , respectively, for this phase. Also, we introduce a potential function $\Phi = c_2\delta(v, v^*)$ and define the potential at the end of the last phase to be zero. The potential at the beginning of a phase is given by Φ_{-1} , and the potential at the end of it is Φ , which is the same as the potential at the start of the next phase. We will specify the value of c_2 later in the proof.

Lemma 9. *If $1/10 \geq 4\eta$, then for each phase of type II, $2|P| < 32410|P^*| - \Phi + \Phi_{-1}$.*

Proof. For now, we let c_1 and c_2 be tunable parameters. Define $F = c_1|P^*| + c_2(\delta(v_{-1}, v_{-1}^*) - \delta(v, v^*)) = c_1|P^*| - \Delta\Phi$. If we can prove that $F > 2|P|$ for suitable values of c_1 and c_2 , the lemma follows. We distinguish three cases, and specify constraints on c_1 and c_2 along the way.

- *Case 1.* $|P^*| \geq \eta S$. Then we have $\delta(v_{-1}, v_{-1}^*) - \delta(v, v^*) \geq -\delta(v, v_{-1}) - \delta(v^*, v_{-1}^*) \geq -RS - |P^*|$. We use this to obtain $F \geq c_1|P^*| - c_2(RS + |P^*|) = (c_1 - c_2)|P^*| - c_2RS \geq$

$(c_1 - c_2)\eta S - c_2RS$. If this quantity is greater than $2(52 + 33R + 40\eta)S$, then it follows from Lemma 8 that $F > 2|P|$.

Constraint 1. $(c_1 - c_2)\eta - c_2R > 2(52 + 33R + 40\eta)$

- *Case 2.* $|P^*| < \eta S$ and COMPETE was not applied. Since we only incurred a cost in steps 1 and 3 of ONLINE2 in this case, we have $|P| \leq (8 + R)S$. By definition of step 1 of ONLINE2, if there is a tour of length less than ηS , when we receive request r_{k_1-1} , then the distance from v_{-1} to that tour's endpoint must be greater than RS , because otherwise we would have made that tour our T_A and terminated step 1 earlier. Hence, $\delta(v_{-1}, v_D) > RS$. Because we did not use COMPETE, no two short tours are more than a distance $16\eta S$ apart. In particular, $\delta(v_D, v^*) < 16\eta S$. By the triangle inequality, we have $\delta(v_{-1}, v^*) \geq \delta(v_{-1}, v_D) - \delta(v_D, v^*) > \delta(v_{-1}, v_D) - 16\eta S$.

By the definition of step 3 of ONLINE2 and the fact that we did not use COMPETE, the only difference between v_{-1} and v is that v is a distance RS closer to v_D , i.e., $\delta(v_{-1}, v_D) - RS = \delta(v, v_D)$. Thus,

$$\begin{aligned} \delta(v, v^*) &\leq \delta(v, v_D) + \delta(v_D, v^*) \\ &= \delta(v_{-1}, v_D) - RS + \delta(v_D, v^*) \end{aligned}$$

We can now bound the potential difference. By the triangle inequality,

$$\begin{aligned} \delta(v_{-1}, v_{-1}^*) - \delta(v, v^*) &\geq \delta(v_{-1}, v^*) - \delta(v, v^*) - \delta(v_{-1}^*, v^*) \\ &> [\delta(v_{-1}, v_D) - 16\eta S] - [\delta(v_{-1}, v_D) - RS + \delta(v_D, v^*)] - |P^*| \\ &= -32\eta S + RS - |P^*| \end{aligned}$$

So the inequality we need to satisfy is $F > c_1|P^*| + c_2(-32\eta S + RS - |P^*|) = (c_1 - c_2)|P^*| + c_2(RS - 32\eta S)$. If we require $c_1 > c_2$, then we require $F > c_2(RS - 32\eta S) > 2(8 + R)S$.

Constraint 2. $c_1 > c_2$.

Constraint 3. $c_2(R - 32\eta) > 2(8 + R)$.

- *Case 3.* $|P^*| < \eta S$ and COMPETE was applied. Define T'_B to be a tour of minimum length serving $r_{k_1+1}, \dots, r_{k_2}$ and starting at v_B . We define T'_C in a similar manner, starting from v_C . We use Lemma 6, where T'_B corresponds to T_1 and T'_C corresponds to T_2 . In this case, $\Delta = \frac{1}{2} \max\{d(x_B, x_C), d(y_B, y_C)\} \geq \frac{1}{2}(\delta(v_B, v_C)/2) \geq 4\eta S$ by definition of COMPETE and ONLINE2. Thus, by Lemma 6, $|T'_B| + |T'_C| \geq \min\{\Delta, S/10\} \geq 4\eta S$. For this inequality, we used the lemma's assumption that $1/10 \geq 4\eta$.

We now define \hat{T}_B and \hat{T}_C as arbitrary tours that serve r_1, \dots, r_{k_2} and neighbor T_B and T_C , respectively. Since T_B and \hat{T}_B are neighbors, there is some point $x \in M_1$ and some point $y \in M_2$ that both tours go to. So we can construct an extension of T_B which serves $r_{k_1+1}, \dots, r_{k_2}$ (since T_B by itself only serves r_1, \dots, r_{k_1}) by moving the servers to (x, y) and serving the remaining requests the same way as T_B does. By optimality of T'_B as an

extension to T_B , we obtain $|T'_B| \leq |T_B| + |\hat{T}_B|$. We similarly obtain $|T'_C| \leq |T_C| + |\hat{T}_C|$. Putting these together, we have

$$\begin{aligned} |\hat{T}_B| + |\hat{T}_C| &\geq |T'_B| + |T'_C| - |T_B| - |T_C| \\ &\geq 4\eta S - \eta S - \eta S \\ &= 2\eta S. \end{aligned}$$

Consider an arbitrary tour T that serves r_1, \dots, r_{k_2} with $|T| < \eta S$. Then $|T_B| + |T_C| + |T| < 3\eta S$. Since we assume Lemma 8 holds for our choice of η , $3\eta S < 6\eta S \leq \min\{X_{1k_1}, Y_{1k_1}\}$ by Lemma 7. Hence, by Corollary 1, some pair of those tours are neighbors. Since T_B and T_C are not neighbors (by Corollary 2), T must be neighboring either T_B or T_C . It follows that any such tour, (in particular, P^* and T_D) neighbors T_B or any such tour neighbors T_C .

Since T_D and P^* neighbor the same tour, we can bound the distance between the endpoints v_D and v^* . By the triangle inequality and Lemma 2, $\delta(v_D, v^*) \leq |T_D| + \max\{|T_B|, |T_C|\} + |P^*| < 2\eta S + |P^*|$. Finally, we can bound the change in potential:

$$\begin{aligned} \delta(v_{-1}, v_{-1}^*) - \delta(v, v^*) &\geq \delta(v_{-1}, v^*) - \delta(v, v^*) - \delta(v_{-1}^*, v^*) \\ &\geq (\delta(v_{-1}, v_D) - \delta(v, v_D)) - 2\delta(v_D, v^*) - |P^*| \\ &> RS - 2(2\eta S + |P^*|) - |P^*| \end{aligned}$$

So our constraint is $F > c_1|P^*| + c_2(RS - 4\eta S - 3|P^*|) = (c_1 - 3c_2)|P^*| + c_2(RS - 4\eta S)$. If we assume $c_1 > 3c_2$, then $F > c_2(R - 4\eta)S$. Again, we want $F > 2|P|$.

Constraint 4. $c_1 > 3c_2$.

Constraint 5. $c_2(R - 4\eta) > 2(52 + 33R + 40\eta)$.

Our expression for F indicates that c_1 will turn out to be the competitive ratio of ONLINE2. Observe that all of the constraints we put on c_1 and c_2 are linear. As a result, we can put the constraints into an LP solver and minimize the value of c_1 . Using the values $R = 0.35$ and $\eta = 1/100$, we get values $c_1 = 32410$ and $c_2 = 545$. \square

With this lemma, we can easily prove competitiveness.

Theorem 1. ONLINE2 is 32410-competitive.

Proof. Consider a phase j of type I. By the exception rule, any phase of type I is followed by another phase in which the total cost is at least twice the cost of that type I phase. Since the last phase is type II, the cost over all phases of type I is at most $\frac{1}{2} + \frac{1}{4} + \dots = 1$ times the

cost of all type II phases. Thus,

$$\begin{aligned}
\sum_{j \in I \cup II} |P_j| &\leq 2 \sum_{j \in II} |P_j| \\
&\leq \sum_{j \in II} 32410 |P_j^*| - \Phi_j + \Phi_{j-1} \\
&\leq \sum_{j \in I \cup II} 32410 |P_j^*| - \Phi_j + \Phi_{j-1} \\
&= \sum_{j \in I \cup II} 32410 |P_j^*|
\end{aligned}$$

To obtain the third inequality, we used the fact that a type I phase has the servers end the phase in the same position as it started, and thus the only change in potential is from the movement of the optimal servers. The contribution to the sum is given by $32410 |P_j^*| - \Phi_j + \Phi_{j-1} \geq 32410 - 545 |P_j^*| \geq 0$. \square

References

- [1] E. Koutsoupias and C. Papadimitriou, *On the k -server conjecture*, Journal of the ACM **42** (1995), 971–983.
- [2] E. Koutsoupias and D. Taylor, *The cnn problem and other k -server variants*, The 17th Annual Symposium on Theoretical Aspects of Computer Science 2000 (STACS), LNCS 1770 (2000), 581–592.
- [3] L. A. McGeoch M. Manasse and D. Sleator, *Competitive algorithms for server problems*, Journal of Algorithms **11** (1990), 208–230.
- [4] L. Stougie R. Sitters and W. de Paepe, *A competitive algorithm for the general 2-server problem*, In Proc. 30th International Colloquium on Automata, Languages and Programming (ICALP), volume 2719 of Lecture Notes in Computer Science. (2003), 624–636.