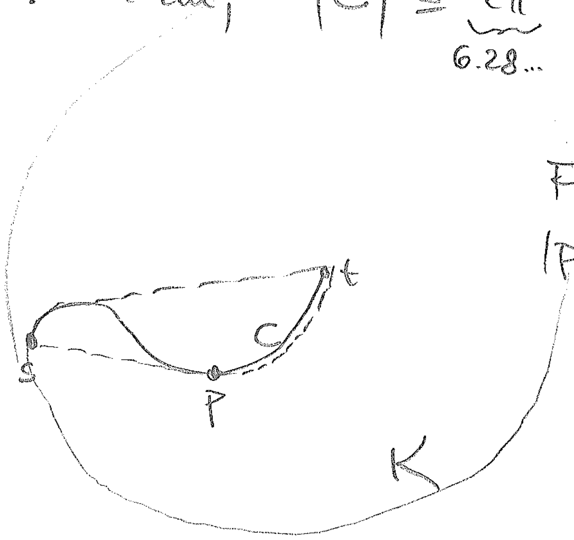
From the Main lemma, we can directly derive an upper bound to the length of self-approaching curves.

Lemma 4 Let C be an oriented self-approaching curve from s to t. Then, $|C| \le \underbrace{2\pi}_{6.28\ldots} |st|$

Proof

For each $p \in C$, we have $|pt| \le |st|$

$\Rightarrow C \subseteq$ circle K of radius $|st|$ ab



$\overset{\text{convexity}}{\Rightarrow}$ $|\partial \operatorname{conv}(C)| \le |\partial K| = 2\pi |st|$. $\boxed{\text{Lemma 4}}$

A more careful analysis results in the upper bound of 5.3331... mentioned in the Theorem. It is attained by curves like this

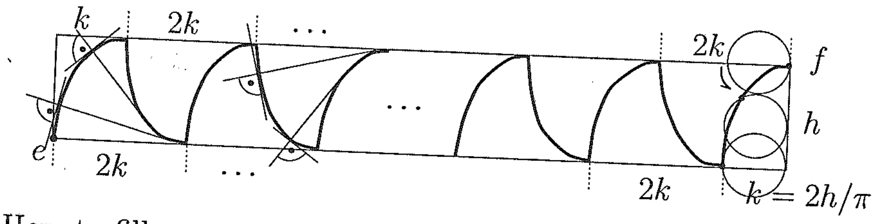
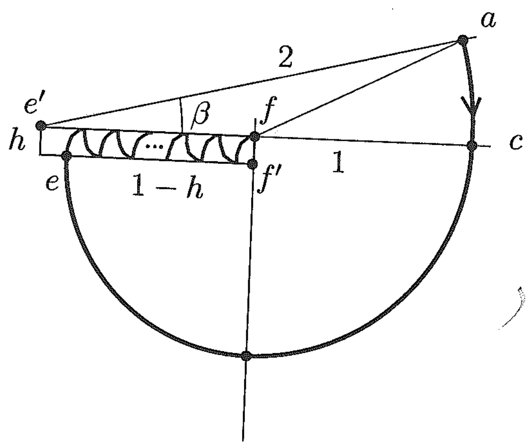


Figure 12: How to fill a rectangle of width w and height h with a self-approaching curve of length $2w$ using pieces of cycloids.



, hence tight.

Corollary There exists a 5.3331..- competitive strategy for finding the kernel of a simple polygon.

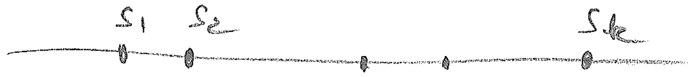
While 5.3331 - bound on self-approaching curves is tight, there are better strategies (better analysis) for kernel finding:

Lee / Chwa : $\pi + 1 = 4.141\ldots$ bound for CATS '9

Lee et al : $1 + 2\sqrt{2} = 3.828\ldots$ by different strateg

The k-servers problem

On a line: Given: k repairmen (servers) in initial positions on the line



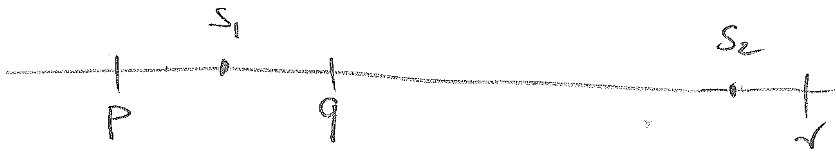
sequence of service requests at positions $p_1, p_2, p_3, \dots$

Problem Decide which server should handle request, in order to minimize total path length!

Remark Cab or maintenance company would perhaps prefer to minimize customer waiting time, instead of the overall gas bill.

Natural idea: Greedy algorithm: always move nearest server to request location

Example $k=2$



request sequence $r(pq)^\infty$:

Greedy: moves s_2 to r ; stays there
oscillates s_1 between p and q

OPT : moves s_2 to r
s_1 to p
s_2 to q and stays there

Lemma Even for $k=2$ servers on the line, the Greedy strategy is not competitive.

Question: Is there a competitive strategy?

Lower bound for very general setting:
(M, d) a metric space, i.e.,

$d : M \times M \longrightarrow \mathbb{R}_{\geq 0}$ metric, satisfying

$\forall x,y,z \in M$: $d(x,y) = 0 \iff x = y$
$d(x,y) = d(y,x)$
$d(x,y) \leq d(x,z) + d(z,y)$

Theorem Let (M,d) be a metric space, $k \in \mathbb{N}$, $|M$
Then, no strategy for the k-Server problem in (M,d) can b
better than k-competitive.

Proof Let A be a k-server algorithm, $p_1, p_2, \ldots, p_k \in M$
the initial server locations.
Add $(k+1)$-th point p_0.
"Bad" request sequence δ for A: $(x_1, x_2, \ldots x_{m+1}) =$
request always the position where no server
is located (the hole) $x_1 = p_0$, ...

Then, strategy A always moves server from $\underbrace{x_{t+1}}_{\text{next hole}}$ to $\underbrace{x_t}_{\text{prev. hole}}$

$$\Rightarrow \quad cost_A(\delta) = \sum_{t=1}^{m} d(x_{t+1}, x_t).$$

Define algorithms B_i, $1 \leq i \leq k$ as follows

initially, B_i covers all positions but p_i
On request $x_t = p_i$, B_i moves server from x_{t-1} to x_t

Claim Let $S_i :=$ server positions of B_i at time t
Then, $\forall i \neq j$: $S_i \neq S_j$

Proof by induction on t. $t = 0$: by definition.
Assume request $x_t \in S_i \cap S_j$: no move necessary, claim still holds
Assume $x_t \in S_j, \notin S_i$: B_i moves server from x_{t-1} to x_t

$\Rightarrow \quad x_{t-1} \notin S_i'$ but
$x_{t-1} \in S_j = S_j'$ because of previous requ

c

$$\Rightarrow \quad s_i' \neq s_j' \qquad \boxed{\text{Claim}}$$

Claim $\Rightarrow$ at request of x_t, at most one B_i has to move server (from x_{t-1} to x_t)

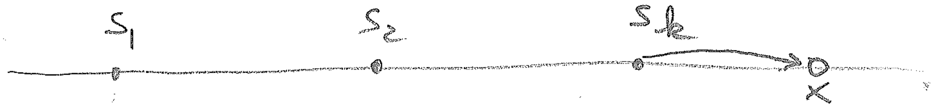
$$\Rightarrow \quad \sum_{i=1}^{k} \operatorname{cost}_{B_i}(\sigma) = \sum_{i=2}^{m+1} d(x_{t-1}, x_t) = \operatorname{cost}_A(\sigma)$$

$$\Rightarrow \quad \text{ex } i : \quad \operatorname{cost}_{B_i}(\sigma) \le \frac{1}{k} \operatorname{cost}_A(\sigma) \qquad \boxed{\text{Theorem}}$$

<u>Question</u> Can lower bound of k be attained?

Answer not known for general metric spaces.
Will look at two special cases.

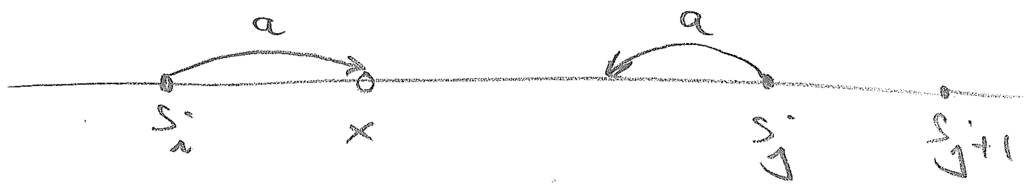
<u>k server on the line</u> Strategy DC (= double coverage



server posit

on request x left of s_1 or right of s_k:
move s_1 (resp., s_k) to x (same as Greedy

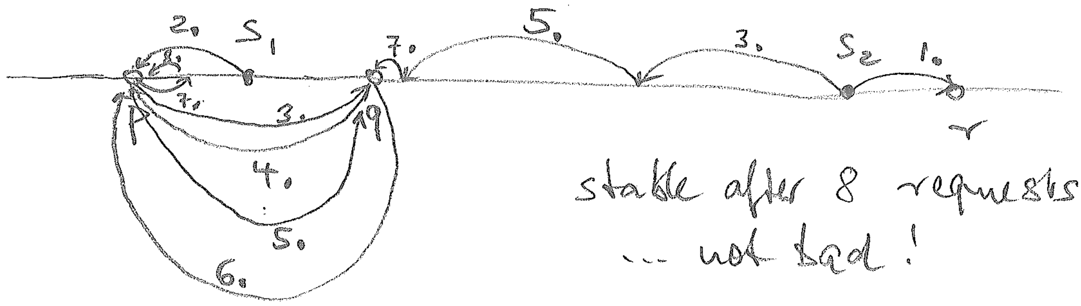
on request x between s_i, s_j
move closer one of s_i, s_j to x
and the other one towards x by same distance



(looks strange, at first glance)

d

old example: request sequence $r(79)^{\infty}$



stable after 8 requests
... not bad!

<u>Theorem</u> Strategy DC is k-competitive for k servers on the line.

<u>Proof</u> Uses a general technique, interesting in its own right: potential function ($\hat{=}$ bank account)
idea: actions have immediate consequences but also (structural) long term effects.

More precisely,

let Z_{ALG}, Z_{OPT} the state sets of ALG and OPT and

$$\phi : Z_{ALG} \times Z_{OPT} \longrightarrow \mathbb{R}$$

a potential function satisfying the following axioms.

Let $\delta = e_1, e_2, \ldots, e_n$ denote sequence of moves of ALG and OPT (not necessarily in turns)

(i) if e_i move of ALG : $\quad ALG(e_i) \le \phi_{i-1} - \phi_i$

(ii) if e_i move of OPT : $\quad OPT(e_i) \ge \dfrac{\phi_i - \phi_{i-1}}{c}$

(iii) $\forall i$: $\phi_i \ge u$ for some constant u.

<u>Lemma</u> Under these assumptions, ALG is c-competitive.

e

Proof Let $A = \{ i \mid e_i \text{ is move of ALG} \}$

By the axioms,

$$ALG(\delta) \leq \sum_{i \in A} (\phi_{i-1} - \phi_i)$$

$$c\ OPT(\delta) \geq \sum_{i \in A^C} (\phi_i - \phi_{i-1})$$

$\Rightarrow$ it is sufficient to show the following: there exists additive constant V such that

$$\sum_{i \in A} (\phi_{i-1} - \phi_i) \overset{!}{\leq} \sum_{i \in A^C} (\phi_i - \phi_{i-1}) + V$$

$$\Leftrightarrow \underbrace{\sum_{i \in A} \phi_{i-1} + \sum_{i \in A^C} \phi_{i-1}}_{\sum_{i=1}^{n} \phi_{i-1}} \overset{!}{\leq} \underbrace{\sum_{i \in A} \phi_i + \sum_{i \in A^C} \phi_i}_{\sum_{i=1}^{n} \phi_i} + V$$

$$\Leftrightarrow \phi_0 \overset{!}{\leq} \phi_n + V$$

can be fulfilled by $V := \phi_0 - u$, since $\phi_n \geq u$

Lemma

Big question: Which potential function ϕ to use in proving DC is k-kompetitive?

Def: $\phi := k \cdot M_{min} + \Sigma_{DC}$, where

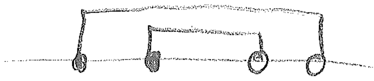
M_{min} := cost of minimum matching between server positions of ALG = DC and OPT

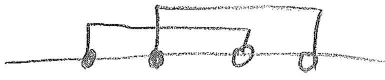
Σ_{DC} := sum of all distances between server positions of DC

f minimum matching: • : server of DC
∘ : server of OPT

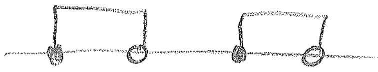
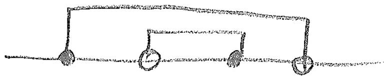
connect pairs •, ∘ such that total length of connections is minimum

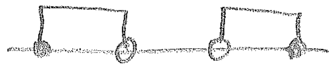
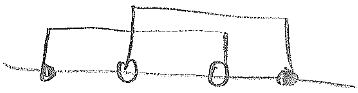
Examples





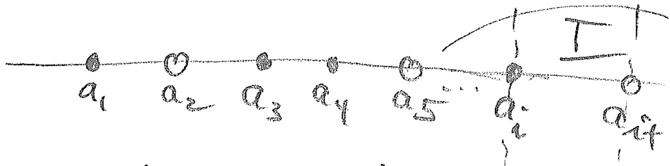
both minimum matchings

 is shorter than 

 is shorter than 

Let $a_1, a_2, \ldots, a_{2k}$ be the enumeration of all server positions of DC and OPT, in left-to-right order

for $1 \le i \le 2n$:



$\delta_i := |\ \#\bullet - \#\circ$ with index $\le i\ |$

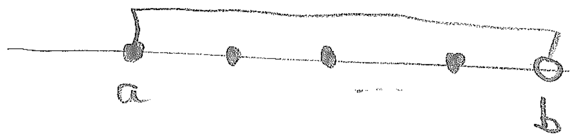
Lemma M_{min} = cost of minimum matching

$$= \sum_{i=1}^{2n-1} \delta_i \cdot |a_{i+1} - a_i|$$

Proof: "$\ge$" at least δ_i edges must cross over inte[rval] I to partners of indices $\ge i+1$

"$\le$" Construct minimum matching inductively, check that formula is preserved:

Pick leftmost position $a = a_1$; assume wlog $a = $ •
let $b :=$ leftmost ∘ position
connect a, b, and delete both



[Lemma]

Now we check if $\phi = k \cdot M_{min} + \Sigma_{DC}$ fulfills axioms (i), (ii), (iii) of potential functions. (12

(assume: first OPT moves, then

(iii) $\phi \geq 0$ obvious

(ii) suppose OPT moves server by distance d in move e_i

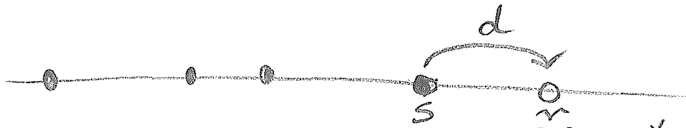
$\Rightarrow$ Σ_{DC} does not change

M_{min} grows by $\leq d$

$$\Rightarrow \phi_i \leq \phi_{i-1} + kd = \phi_{i-1} + k \cdot OPT(e_i)$$

(i) suppose DC moves only one server by distance in move e_i

$\Rightarrow$



request; already occupied by OPT-Serv

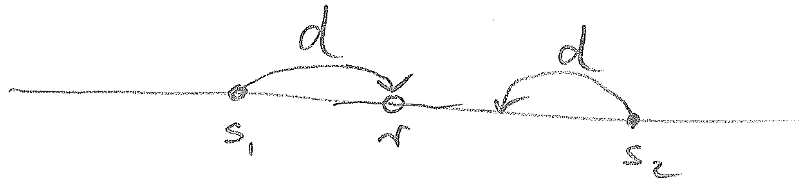
$\Rightarrow$ Σ_{DC} grows by $(k-1)\,d$

M_{min} decreases by at least d, because in old minimum matching, s must have been matched to r ($\rightarrow$ proof of previous lemma)

$$\Rightarrow \phi = k \cdot M_{min} + \Sigma_{DC} \text{ decreases by at leas}$$

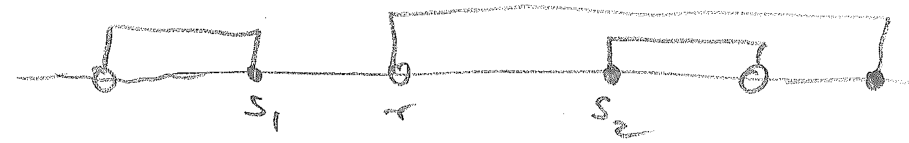
$$\phi_i \leq \phi_{i-1} - d = \phi_{i-1} - DC(e_i)$$

Suppose DC moves two servers by distance d in move e_i

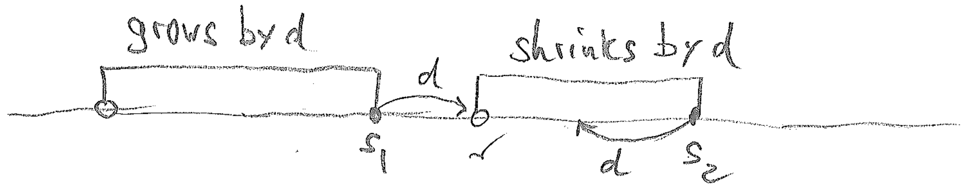


in old minimum matching, one of s_1, s_2 must have been matched to r

(otherwise



not minimal



$\Rightarrow$ M_{min} does not grow

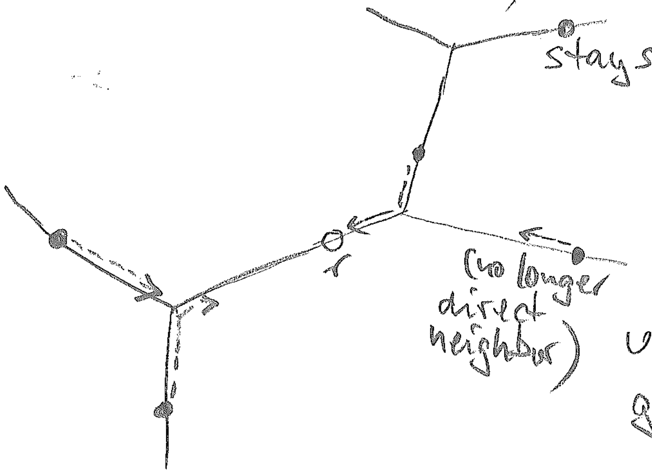
in Σ_{DC}: $|s_1 - s_2|$ shrinks by $2d$

all $|s_1 - s| + |s - s_2|$ remain unchanged

$\Rightarrow$ $k \cdot M_{min} + \Sigma_{DC}$ decreases by at least $2d$.

$\boxed{\text{Theorem}}$

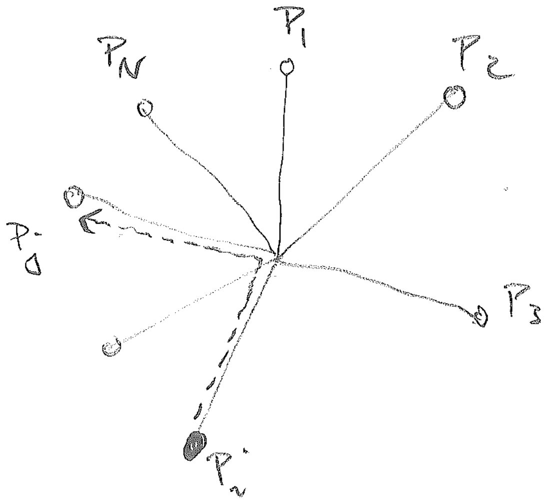
Algorithm DC (and its analysis) can be generalized to trees:



Only the direct neighbors of r move towards r until first server gets there

<u>Theorem</u> DC is k-kompetitive in trees.

Example: k servers on a star



interpretation

on request of P_j, server from P_i moves there

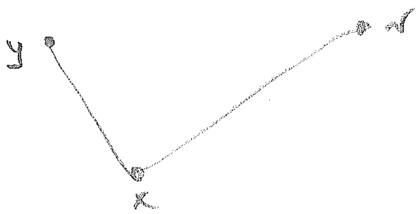
server locations
$\hat{=}$ memory pages stored in cache of size k

on request for page j, page i gets evicted from cache

algorithm DC $\iff$ "flush when full" must be k-competitive!

Another interesting special case: 2 servers in $\mathbb{R}^2$

Definition



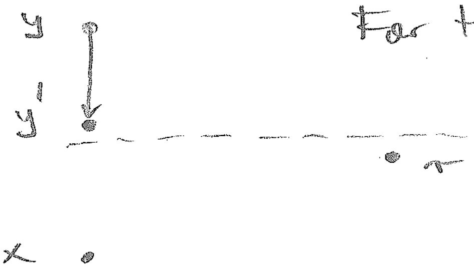
$$\text{slack}(x,y,r) := |xy| + |xr| - |yr| \geq 0 \text{ by } \triangle$$

Algorithm SC (= slack coverage)

Suppose servers positioned at x and y, new request at r comes in, where $|xr| \leq |yr|$ (*)

Then, move
server at y by $\frac{1}{2}\text{slack}(x,y,r)$ towards x
server at x to r

Lemma

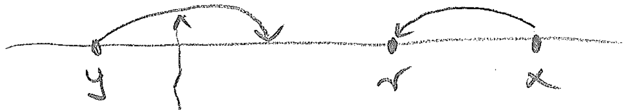


For the new position, y', of y-server:
$|y'r| < |yr|$

Proof: $\frac{1}{2}\text{slack}(x,y,r) = \frac{1}{2}(|xy| + |xr| - |yr|) \underset{(*)}{\leq} \frac{1}{2}|xy|$ $\boxed{\text{Lemma}}$

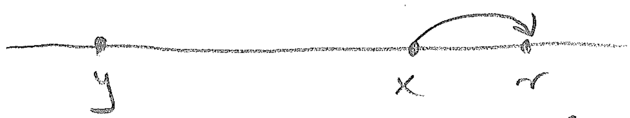
j What does SC to points on the line ?

Case 1



$$\tfrac{1}{2}\mathrm{slack}(x,y,r) = \tfrac{1}{2}(|xy| + |xr| - |yr|) = |xr|$$

Case 2



$$\tfrac{1}{2}\mathrm{slack}(x,y,r) = \tfrac{1}{2}(|xy| + |xr| - |yr|) = 0$$

$\Rightarrow$ SC = DC on the line !

Theorem For two servers in the plane, SC is 3-competitive

Proof Want to apply potential function approach with
$\phi := 3\,M_{min} + 2|xy| \geq 0.$ (like before)

Need to prove 2 properties :

(i) when OPT moves distance D :
$\Delta\phi = \phi_{i+1} - \phi_i \leq 3D$

(ii) When SC moves distance D :
$\Delta\phi \leq -D$

first OPT then SC

Proof of (i) OPT moves server a distance D

$\Rightarrow$ $|xy|$ does not change, M_{min} grows by $\leq D$

proof of (ii) assume $|xr| \le |yr|$

and SC moves server from y by $\frac{1}{2}$slack(x,y,r) to y'
server from x to r

$\Rightarrow D = |xr| + \frac{1}{2}\text{slack}(x,y,r)$ total distance moved by SC

Clear: $\Delta|xy| = |ry'| - |xy| \underset{\text{Lemma}}{\le} |ry| - |xy|$

to be determined: ΔM_{min}

Let s_1, s_2 be the OPT-servers, $s_1 = r$ after serving request

Case 1 Before SC moves, $\left.\begin{array}{l} x \Leftrightarrow s_1 = r \\ y \Leftrightarrow s_2 \end{array}\right\}$ in M_{min}

$\Rightarrow$ M_{min} decreases by at least $|xr|$
increases by at most $\frac{1}{2}$slack(x,y,r)
as SC moves

$$\Rightarrow \Delta\phi \le 3\Big(\underbrace{\tfrac{1}{2}\text{slack}(x,y,r)}_{\frac{1}{2}(|xy| + |xr| - |yr|)} - |xr|\Big) + 2\big(|ry| - |xy|\big)$$

$$= -|xr| - \tfrac{1}{2}|xy| - \tfrac{1}{2}|xr| + \tfrac{1}{2}|yr| = -D$$

Case 2 Before SC moves, $\begin{array}{l} x \Leftrightarrow s_2 \\ y \Leftrightarrow s_1 = r \end{array}$ in M_{min}

$\Rightarrow$ afterwards $\begin{array}{l} x \Leftrightarrow s_1 = r \\ y' \Leftrightarrow s_2 \end{array}$ (identical)

$\Rightarrow \Delta M_{min} \le |y's_2| - |xs_2| - |yr|$

$$\underset{\triangle}{\leq} \; |y'x| - |yr| \underset{\text{Def } y'}{=} |yx| - \tfrac{1}{2}\,\text{slack}(x,y,r) - |yr|$$

$$\Rightarrow \Delta\phi \leq 3\left(|yx| - \tfrac{1}{2}|xy| - \tfrac{1}{2}|xr| + \tfrac{1}{2}|yr| - |yr|\right) + \underbrace{2|ry| - 2|xy|}_{2\Delta|xy|}$$

$$= -|xr| - \tfrac{1}{2}|xy| - \tfrac{1}{2}|xr| + \tfrac{1}{2}|yr| = -D.$$

$\boxed{\text{Theorem}}$

Best result known for general k, general metric spaces:

<u>Theorem</u> (Koutsoupias, Papadimitriou '95)
There exists a $(2k-1)$-competitive algorithm for the k server problem in general metric spaces.

<u>k-Server-Problem</u> Is there a k-competitive algorithm?