Recall: online algorithm – handles sequence of requests as they come.

Alg. $\mathcal{A}$ is <u>c-competitive</u> if

$\mathcal{A}(\sigma) \le c \cdot OPT(\sigma) + b$ for min problem

sequence; the opt. offline solution; constant

$\mathcal{A}(\sigma) \ge c \cdot OPT - b$ for max. problem

<u>k-Server Problem</u> – k servers to service requests in metric space of points $p_1 \cdots p_n$

request for p_i
- if a server is at p_i fine
- else move a server from its location, say p_j, to p_i at cost $d(p_j, p_i)$

distance.

Goal: serve requests in given order and min. total distance

e.g. plumbing repair trucks.

The offline k-server problem can be solved in poly. time via dynamic programming.

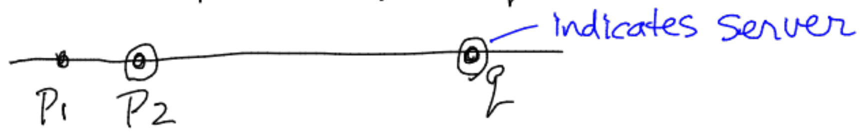
Paging is special case. — the "points" are the pages in slow memory, "serving" a request = putting the page in the cache, distances are all 1.

Local greedy algorithm
- to meet next request, say at p, move server from q to minimize $d(p,q)$ – i.e. use closest server

Claim. this is not c-competitive for any c.

Anna Lubiw
Univ. Waterloo

Pf. Consider point p_1, p_2, q on a line



2 servers initially at p_2 and at q

request sequence: $p_1\ p_2\ p_1\ p_2 \cdots$

greedy moves one server between p_1 and p_2

cost = length of sequence

OPT moves server from q to p_1 — cost is constant $d(q, p_1)$

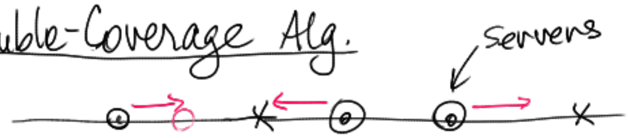
Ex. what does LRU do for k-server? Find a bad example

Conjecture (open since 1988) There is a k-competitive alg. for k-server problem.

Two cases with k-competitive algs: 1. paging (LRU)
2. the following

For points on a line, there is a k-competitive algorithm

Double-Coverage Alg.



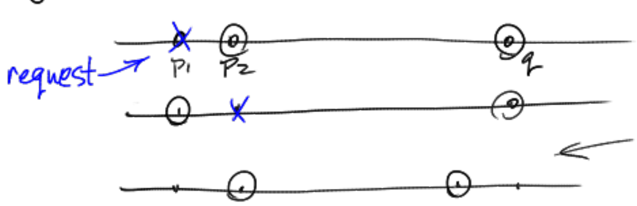
case i - if request is to right [left] of all servers, move closest server

case ii - if request is between 2 servers, move both until one reaches the request. (the other will stop at a non-request point)

(if multiple servers at same point, break ties arbitrarily).

Thm. This is k-competitive.

e.g. on above bad example



← Note that q starts moving left.

eventually q will reach p_2 and then all requests are free.

Anna Lubiw
Univ. Waterloo

Pf of theorem. We need to compare ALG to OPT

will use an amortized analysis with potential fn Φ.

Think of ALG having k servers and OPT having k servers

Think of: OPT moves, then ALG moves

Φ will depend on the difference

Properties we will ensure.

Note: there is no reason for OPT to move more than 1 server.

1. - $\Phi_i \geq 0$
2. - if move of OPT costs s_i then incr. of Φ is $\leq k \cdot s_i$
3. - if move of ALG costs t_i then incr of Φ is $\leq -t_i$ (i.e. Φ decreases by $\geq t_i$)

Then $\Phi_{i+1} - \Phi_i \leq k \cdot s_i - t_i$

Taking sum:

$$\Phi_f - \Phi_0 \leq k \cdot \underbrace{\Sigma s_i}_{\text{OPT}} - \underbrace{\Sigma t_i}_{\text{cost of ALG}}$$

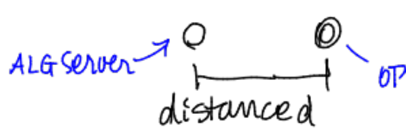
$$\text{ALG} \leq k\,\text{OPT} + \Phi_0 - \Phi_f \leq k\,\text{OPT} + \Phi_0$$

How do we define Φ?

$$\Phi = k \cdot M + D$$

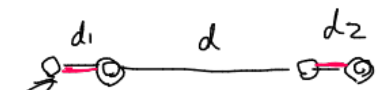
M = min matching between ALG's servers and OPT's servers

e.g. 1 server each



$M = d$

2 servers each



this red matching has weight $d_1 + d_2$

other matching: $d_1 + 2d + d_2$. So $M = d_1 + d_2$.

D = sum of all $\binom{k}{2}$ distances between pairs of ALG's servers.

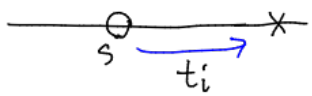
Proving the properties:

1. $\Phi_i \geq 0$ ✓
2. Suppose OPT moves a server s_i then D unchanged and M incr. by $\leq s_i$

3. Suppose ALG move costs t_i

2 cases:

case (i) ALG moved rightmost (or leftmost) server.

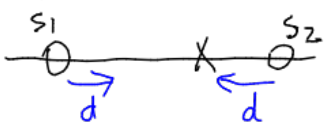


OPT already put a server there, so M decreases by $\geq t_i$ – and we count this k times

D goes up by $(k-1)t_i$ since server s moves away from others

Φ decreases by $\geq t_i$

case (ii) ALG moved two servers cost $t_i = 2d$



M – one match decreases by d, one match might go up by d – M doesn't increase

D – $d(s_1, s_2)$ decreases by 2d

for any other s, say to right, $d(s, s_1)$ increases by d
$d(s, s_2)$ decreases by d
so $\sum_r d(s,r)$ is constant.

$\therefore$ Φ decreases by $\geq t_i$

Ex. Find an example to show competitive ratio can be k.

For the general k-server problem:

Best so far: competitive ratio $2k-1$ [Koutsoupias & Papadimitriou '94]
combines local greedy with "retrospective alg." that goes to the state the opt. alg. would be in based on requests so far.

For randomized algs, even more open
lower bound $\Omega(\log n)$ (as for paging)
and upper bound $2k-1$ (deterministic)
progress in 2011 $\tilde{O}(\log^3 n \log^2 k)$ Bansal et al.