Technische Universität Braunschweig

Online Algorithms - Tutorial 02

Summer term 2022, 16. May 2022

The k-server problem

The k-server problem

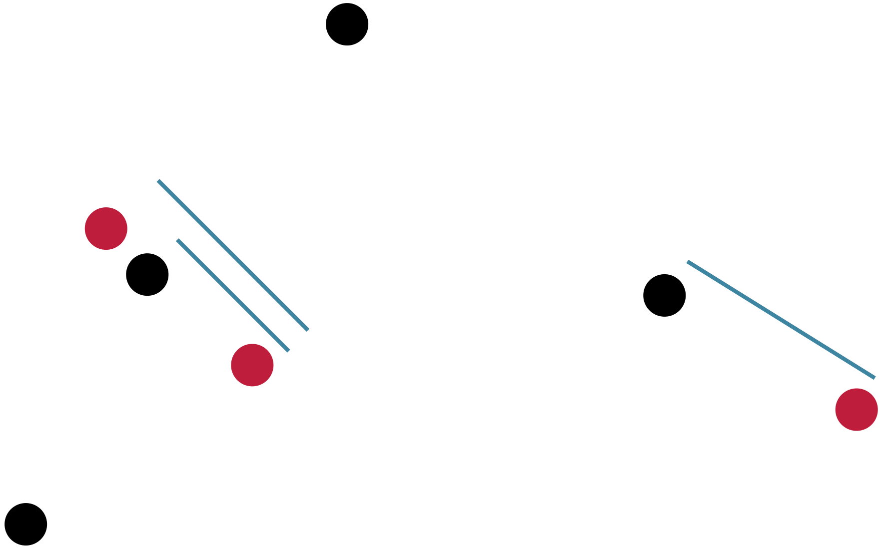
- Suppose you have a possibly infinite metric space M
- The algorithm controls k servers that can move (change their position) inside M

- You are given a sequence of requests which correspond to points in M
- To **serve** a request, one server has to move itself to the requested position

- Goal: minimize the total distance travelled by all servers

Formal definition: See board.

$\mathbb{R}^2$ and $k = 4$

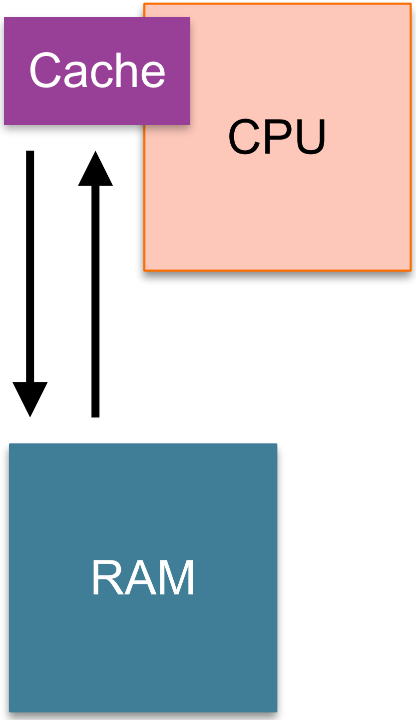


Total cost:

The k-server problem: Connection to paging

Paging

- Small cache size k and large no. of items
- At each point in time, an item is requested
- If not in cache: replace item from cache with new item (cost 1)



The k-server problem generalizes to paging

Example: See board.

The k-server problem: Deterministic Lower Bound

<u>Theorem 2.1:</u> Let M be any metric space with at least $k+1$ points. No deterministic online algorithm can achieve a competitive ratio better than k

Proof: See board.

1. Best value of <u>any</u> online algorithm
   - Question: If you were an evil adversary, what would you do?

2. Best offline solution
   - Consider $\binom{k}{k-1}$ offline algorithms
   - Show that for every request, only one of the algorithms does move a server
   - Argue the competitive ratio with the average performance

The k-server conjecture

Conjecture 2.2: Any metric space allows for a deterministic k-competitive, k-server algorithm.

On the k-Server Conjecture

ELIAS KOUTSOUPIAS

University of California at Los Angeles, Los Angeles, California

AND

CHRISTOS H. PAPADIMITRIOU

University of California at San Diego, La Jolla, California

Abstract. We prove that the *work function algorithm* for the k-server problem has a competitive ratio at most $2k - 1$. Manasse et al. [1988] conjectured that the competitive ratio for the k-server problem is exactly k (it is trivially at least k); previously the best-known upper bound was exponential in k. Our proof involves three crucial ingredients: A *quasiconvexity property* of work functions, a *duality lemma* that uses quasiconvexity to characterize the configuration that achieve maximum increase of the work function, and a *potential function* that exploits the duality lemma.

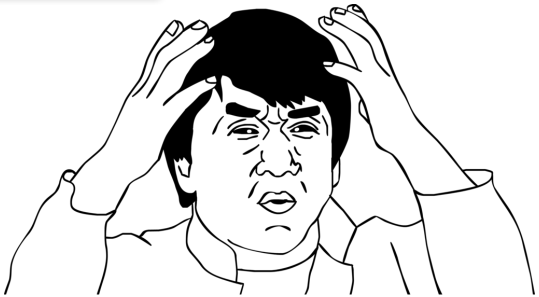
Shown for $2k - 1$!

The k-server problem: Double coverage algorithm

- Consider the real number line with Euclidean metric

Algorithm DC

- If the request falls outside the $conv(S)$
  - Serve with the nearest server
- Otherwise the request is between two adjacent servers
  - Move both servers towards the request at equal speeds until (at least) one server reaches it



The k-server problem: Double coverage algorithm

- Consider the real number line with Euclidean metric

Algorithm DC

- If the request falls outside the $conv(S)$
  - Serve with the nearest server
- Otherwise the request is between two adjacent servers
  - Move both servers towards the request at equal speeds until (at least) one server reaches it

Theorem 2.3: DC is k-competitive.

Proof: Homework :)

The k-server problem: Double coverage algorithm for trees

- Generalization of DC on trees (DC = line tree)
- A server s_i is neighboring any point σ_i iff there is no other server on the shortest path between the points
- When two servers are on the same spot only one of them is the neighbor

Algorithm DC-Tree

- For a request σ_i all servers neighboring the request are moving in a constant speed toward σ_i

Theorem 2.4: DC-Tree is k-competitive.

The k-server problem: Applications of DC-Tree

- **Finite graphs**: Consider any spanning tree — $(n-1)k$-competitive

- **Paging**: see board
- **Weighted Paging** (any page p has some nonnegative weight $w(p)$): see board — k-competitive

- Possible generalization into general metric spaces?