

Prof. Dr. Sándor P. Fekete  
Michael Perk

## Online Algorithms Homework Assignment 2 May 30, 2022

Solutions are due on Monday, the 30th of May, 2022, until 15:00 in the homework cupboard. You can also hand in your solution in person before the big tutorial begins or via e-mail to [kramer@ibr.cs.tu-bs.de](mailto:kramer@ibr.cs.tu-bs.de) with CC to [mperk@ibr.cs.tu-bs.de](mailto:mperk@ibr.cs.tu-bs.de).

### Exercise 1 (The $k$ -server problem: Greedy algorithm):

In this exercise, we consider the  $k$ -server problem in the Euclidean plane. In this problem, we start with  $k$  servers at given points  $s_1, \dots, s_k \in \mathbb{R}^2$ . We are given a sequence of  $n$  requests from locations  $\sigma_j \in \mathbb{R}^2$ . Each request  $\sigma_j$  must be handled immediately by moving one of the servers from its current position  $p$  to  $\sigma_j$ . The cost for moving a server from  $p$  to  $\sigma_j$  corresponds to the Euclidean distance  $d(\sigma_j, p) = \|\sigma_j - p\|_2$ . The goal is to minimize the total distance traveled by the servers. The algorithm GREEDY serves each request using the server closest to the request. What is the competitive ratio  $c$  of GREEDY for  $k = 2$  servers? (15 pts.)

### Exercise 2 (The $k$ -server problem: Double Coverage):

In this exercise, we consider the  $k$ -server problem for the *Euclidean line*. In this problem, we start with  $k$  servers at given points  $s_1, \dots, s_k \in \mathbb{R}$ . We are given a sequence of  $n$  requests from locations  $\sigma_j \in \mathbb{R}$ . Each request  $\sigma_j$  must be handled immediately by moving one of the servers from its current position  $p$  to  $\sigma_j$ . The cost for moving a server from  $p$  to  $\sigma_j$  corresponds to the Euclidean distance  $d(\sigma_j, p) = |\sigma_j - p|$ . The goal is to minimize the total distance traveled by the servers. We want to prove that the algorithm DOUBLE COVERAGE (DC) is  $k$ -competitive. If a request falls outside the convex hull of the servers, DC serves it with the nearest server. Otherwise, the request is between two adjacent servers. In this case, DC moves both servers toward the request at equal speed until (at least) one of the server reaches it.

We denote the cost that occurs in request  $\sigma_i$  as  $c_{\text{DC}}(i)$  and  $c_{\text{OPT}}(i)$  for DC and the optimal offline algorithm. Let  $M_{\min}$  denote the minimum cost matching between the servers in OPT and DC and  $\Sigma_{\text{DC}} = \sum_{i < j} d(s_i, s_j)$  be the sum of all interpoint distances between DC's servers. Let  $\Phi(i) = k \cdot M_{\min} + \Sigma_{\text{DC}}$  be a potential function. From the previous homework we know that DC is  $k$ -competitive if

$$c_{\text{DC}}(i) + \Phi(i) - \Phi(i-1) \leq k \cdot c_{\text{OPT}}(i).$$

In this exercise we show that

$$\Phi(i) - \Phi(i-1) \leq k \cdot c_{\text{OPT}}(i) - c_{\text{DC}}(i)$$

- (a) Both the moves of DC and OPT have an influence on the change of potential  $\Phi(i) - \Phi(i - 1)$  at request  $\sigma_i$ . We split the change of potential in two steps and first consider the influence of OPT, i.e. the change of potential after OPT serves request  $\sigma_i$  and *before* DC serves the request. Show that OPT's move only changes the value of  $M_{min}$  with a difference lower or equal to  $k \cdot c_{OPT}(i)$ .
- (b) We now consider DC's move after OPT's potential change was applied. As DC behaves differently in response to some request position  $\sigma_i$ , we split it's behavior into two cases. For the first case, DC changes the position of a single server. Show that in this case  $\Phi(i) \leq \Phi(i - 1) + k \cdot c_{OPT}(i) - c_{DC}(i)$ .
- (c) In the second case, DC changes the position of two servers. Show that in this case  $\Phi(i) \leq \Phi(i - 1) + k \cdot c_{OPT}(i) - c_{DC}(i)$ , which concludes the proof.

**(10+10+10 pts.)**

**Exercise 3 (The  $k$ -server problem: DC-Tree):**

We consider a tree network in which the distance between two points on the tree is the length of the simple path between them. For a given request  $\sigma_i$  on the tree, we say that a server  $s_j$  is a *neighbor* of the request, if there is no other server on the path from  $s_j$  to  $\sigma_i$ . The DC-TREE algorithm answers a request by moving all neighboring servers at constant speed towards the request. DC-TREE is  $k$ -competitive. Let  $G$  be any  $N$ -node graph. Let  $T$  be the minimum spanning tree of  $G$ . Show that DC-TREE is  $k(N - 1)$ -competitive.

*Hint: Consider the optimal offline algorithm for  $T$  and compare it to the optimal offline algorithm for  $G$ .*

**(20 pts.)**