Algorithms Group **Summer 2020**
Department of Computer Science - IBR
TU Braunschweig

Prof. Dr. Sándor P. Fekete
Phillip Keldenich

Online Algorithms
Exercise 4
June 17, 2020

Hand in your solutions as PDF file until July 1, 2020, 11:30 AM via e-mail to `v.sack@tu-bs.de`, with CC to `keldenich@ibr.cs.tu-bs.de`. If you cannot turn your solution into a PDF file (for example by writing it in LaTeX or Word), you can also submit photographs or scans. In that case, be careful to keep the file size acceptable (about 3 MB per page) by using appropriate compression and resolution; however, make sure that your solutions are still readable.

Exercise 1 (Cow Path Problem):
Suppose that you are a hungry cow on a path. You know that at some distance $D \geq 1$ along the path there is a pasture with tasty grass; however, you do not know the distance D or the direction in which you have to go. In order to reach the pasture, you start by traveling into one of the two directions for one unit of distance. If you do not find your food, you return to the start and then travel two units of distance into the other direction. If you still have not found your food, you return to the start and then travel four units of distance into the first direction, and so on. You repeat this procedure, doubling the distance each time you return back to the start, until you finally reach your goal.

a) Prove that this strategy is 9-competitive, i.e., that you never have to travel more than $9D$ units of distance until you reach the goal. Moreover, prove that this is tight, i.e., that there is no constant $c < 9$ such that this strategy is c-competitive.

b) Prove that the restriction $D \geq 1$ is vital, i.e., that without it, there is no c-competitive strategy for any constant c.

(15+5 pts.)

Exercise 2 (Online Coloring):
In this exercise, we consider the problem of Graph Coloring in an online scenario. Our input sequence consists of the vertices $v_1, \dots, v_n$ of an undirected graph

$$G = (\{v_1, \dots, v_n\}, E).$$

Together with each vertex v_i, we are given the list E_i of all edges connecting v_i to previously given vertices $v_1, \dots, v_{i-1}$. Edges from v_i to vertices v_j with $j > i$ are only revealed once vertex v_j is given. The number of edges and vertices in the graph is not known in advance.

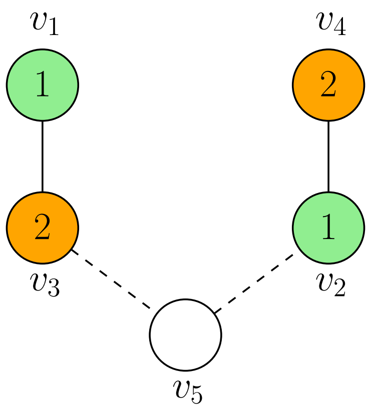

Figure 1: A snapshot from the execution of an online graph coloring algorithm. The algorithm has colored the vertices $v_1, \dots, v_4$ using colors 1 and 2 and is now given a new vertex v_5 connected to v_2 and v_3. It cannot use color 1 or 2 for v_5 and therefore has to introduce a new color for v_5.

When we are given v_i, we have to choose a color $c(v_i) \in \mathbb{N}$ for v_i in such a way that no vertex adjacent to v_i has color $c(v_i)$. As usual, this choice is final; we cannot change the color later on. We want to minimize the number of colors used. For an example, see Figure 1.
We want to show that there is no deterministic c-competitive algorithm for this problem for any constant c. For any constant number $2 \leq k \in \mathbb{N}$ of colors and any deterministic online algorithm $\mathcal{A}$, devise a strategy for an adversary that satisfies the following requirements.

- The strategy always produces a forest $T_{\mathcal{A},k}$.
- The online algorithm $\mathcal{A}$ uses at least k colors on $T_{\mathcal{A},k}$.

Offline, every forest can be colored with 2 colors; therefore, this implies that $\mathcal{A}$'s competitive ratio is at least ${}^{k}\!/_{2}$. **(20 pts.)**